Lesson 11 – Physically-based rendering
Image-based lighting

PV227 – GPU Rendering

Jiří Chmelík, Jan Čejka
Fakulta informatiky Masarykovy univerzity

5. 12. 2018

Physically-based rendering (PBR)

- Physically-based rendering: Theory (cont.)
  - Light & Lights
  - BRDF
  - Sensors (cameras, eyes)
- Image-based lighting

Light – quantities and units

- Quantities and units
  - Radiant energy
  - Radiant flux
  - Irradiance
  - Intensity
  - Radiance
- Different equations use different quantities
- Convertible between each other

Light – quantities and units (cont.)

- Radiant energy (Q)
  - "Energy of one photon"
  - Joule: J
- Radiant flux, radiant power (Φ)
  - "Energy per second"
  - dQ/dt
  - Watt: $W = J/s$
  - Great to describe the power of lights like light bulb, area lights, ...

Light – quantities and units (cont.)

- Irradiance (E)
  - "Flux through area"
  - $d\Phi / dA$
  - Watt per square meter: W / m^2
  - Drops with the square of the distance
  - Great to describe the power of strong distant lights like the sun
- Intensity (I)
  - "Flux through a cone of directions"
  - $d\Phi / d\omega$
  - Watt per steradian: W / sr
  - Does not drop with the distance

Light – quantities and units (cont.)

- Radiance (L)
  - "Flux through a cone of directions from an area" or "Flux through an area from a cone of directions"
  - $d^2\Phi / dA_{proj} d\omega$
  - Watt per square meter: $W / m^2 sr$
  - This is what sensors measure

BRDF

Bidirectional Reflectance Distribution Function

- Describes the relation between the incoming and outcoming light

$$f(\vec{l}, \vec{v}) = \frac{dL_o(\vec{v})}{dE(\vec{l})}$$

- Surface is illuminated from direction $\vec{l}$ with irradiance $dE(\vec{l})$
- It is reflected in various directions
- $dL_o(\vec{v})$ is the outcoming radiance in direction $\vec{v}$

Properties of BRDF

- For non-area lights:

$$f(\vec{l}, \vec{v}) = \frac{L_o(\vec{v})}{E_L \cos(\theta_i)}$$

- Energy conservation (for each incoming direction $\vec{l}$:

$$\int_\Omega f(\vec{l}, \vec{v}) \cos\theta_o d\omega_o < 1$$

BRDF – Examples

- BRDF of diffuse light:

$$f(\vec{l}, \vec{v}) = \frac{C_{diff}}{\pi}$$

- Note: this is what we use in shaders:

    *dif = max(0.0, dot(N, L)) * Cdiff;*

BRDF – Examples

- BRDF in the Cook-Torrance paper

the surface roughness. The next two sections consider the directional and wave length dependence of the reflectance model.

DIRECTIONAL DISTRIBUTION OF THE REFLECTED LIGHT

The ambient and diffuse components reflect light equally in all directions. Thus R_a and R_d do not depend on the location of the observer. On the other hand, the specular component reflects more light in some directions than in others, so that R_s does depend on the location of the observer.

The angular spread of the specular component can be described by assuming that the surface consists of microfacets, each of which reflects specularly [23]. Only facets whose normal is in the direction **H** contribute to the specular component of reflection from **L** to **V**. The specular component is

$$R_s = \frac{F}{\pi} \frac{DG}{(\mathbf{N}\cdot\mathbf{L})(\mathbf{N}\cdot\mathbf{V})}.$$

The Fresnel term F describes how light is reflected from each smooth microfacet. It is a function of incidence angle and wavelength and is discussed in the next section. The geometrical attenuation factor G accounts for the shadowing and

BRDF – Examples

- BRDF in TriAce (presented at SIGGRAPH 2010 course)

2. Customized Blinn-Phong model

The following equation is our BRDF model based on the Blinn-Phong model[2]:

$$\rho = \frac{R_d}{\pi}\left((1 - F_{diff}(f_0)\right) + \frac{(n+2)}{4\pi(2 - 2^{-\frac{n}{2}})} \cdot \frac{F_{spec}(f_0)(N \cdot H)^n}{\max(N \cdot L, N \cdot E)}. \quad (4)$$

$F_{diff}(f_0)$ and $F_{spec}(f_0)$ are Fresnel functions. We used Schlick's approximation for them:

$$F_{diff}(f_0) = f_0 + (1 - f_0)(1 - N \cdot L)^5, \quad (5)$$

$$F_{spec}(f_0) = f_0 + (1 - f_0)(1 - E \cdot H)^5. \quad (6)$$

The BRDF shown in Equation 4 basically follows the laws of energy conservation. However, the function

BRDF – Examples

- BRDF in Frostbite (presented at SIGGRAPH 2014 course)

3.1.2 Material models

In the context of this standard material model, a surface response f is often decomposed into two different terms: a low angular frequency signal called "diffuse" (f_d) and a low to high angular frequency part called "specular" (f_r), see Figure 5. An interface separates two media: the air and the matter. Surfaces made of a flat interface can easily be represented by the Fresnel law [Wikd] for both dielectric and conductor surfaces. When the interface is irregular, see Figure 6, the literature shows that microfacet based models [CT82] are well adapted to characterize the light interaction for these types of surfaces. A microfacet model is described by Equation 1, for more details about the derivations see [Hei14]:

$$f_{d/r}(\mathbf{v}) = \frac{1}{|\mathbf{n}\cdot\mathbf{v}||\mathbf{n}\cdot\mathbf{l}|}\int_{\Omega} f_m(\mathbf{v},\mathbf{l},\mathbf{m})\ G(\mathbf{v},\mathbf{l},\mathbf{m})\ D(\mathbf{m},\alpha)\ \langle\mathbf{v}\cdot\mathbf{m}\rangle\langle\mathbf{l}\cdot\mathbf{m}\rangle\ \mathrm{d}\mathbf{m} \tag{1}$$

The term D models the microfacet distribution (i.e. the NDF, Normal Distribution Function). The G term models the occlusion (shadow-masking) of the microfacets. This formulation is valid for both the diffuse term f_d and the specular term f_r. The difference between these two terms lies in the microfacet BRDF f_m. For the specular term, f_m is a perfect mirror and thus is modeled with the Fresnel F law, which leads to the well-known following formulation:

$$f_r(\mathbf{v}) = \frac{F(\mathbf{v},\mathbf{h},f_0,f_{90})\ G(\mathbf{v},\mathbf{l},\mathbf{h})\ D(\mathbf{h},\alpha)}{4\langle\mathbf{n}\cdot\mathbf{v}\rangle\langle\mathbf{n}\cdot\mathbf{l}\rangle} \tag{2}$$

The term D plays an important role in the appearance of surfaces, as shown by Figure 6. The literature, [Wal+; Bur12] has recently pointed out that "long-tailed" NDFs, like the GGX distribution, are good

Sensors

- Many small sensors, each measure irradiance (flux through an area) over time
- System of lences and aperatures, which define the cone
  - Lences in camera or eye, aperature of a camera, pupil in an eye
  - So instead of irradiance, the system measures radiance
  - Remember Depth-of-field techniques
- The result is the energy
- Conversion to the output signal (logarithmic etc.)
  - Linear color space (RGB) vs. non-linear spaces (sRGB)
  - Remember HDR, gamma correction

Image-based lighting

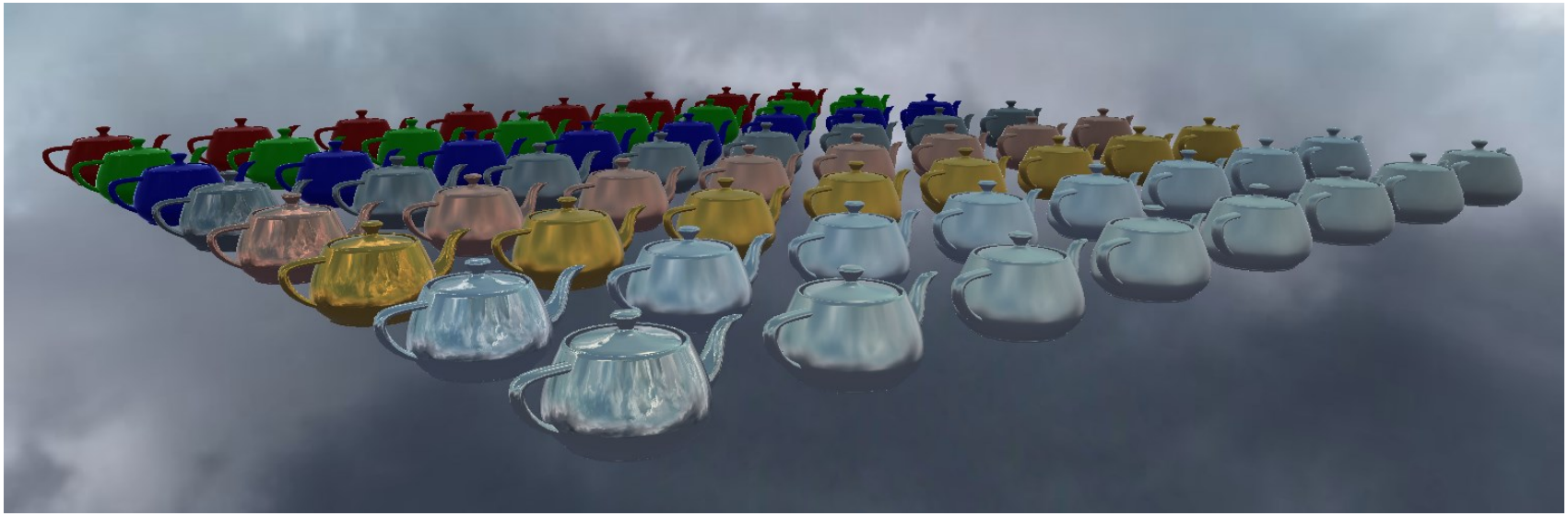


Image-based lighting

Image-based lighting

- Use the light from a texture
  - Environment textures, light probes
  - Usually HDR cubemap textures
- Evaluate the integral using the BRDF to obtain the final color
  - Sampling the directions
    - ★ Uniform sampling
    - ★ Non-uniform importance sampling
  - Precomputation

Task: Implement image-based lighting

- Based on *Real Shading in Unreal Engine 4* (presented at SIGGRAPH 2013 Course)
- With some changes, we use:
  - Uniform sampling for diffuse light
  - Importance sampling for specular light
  - Cook-Torrance based material
    - ⋆ Fresnel as at the previous lecture
    - ⋆ Geometry attenuation as at the previous lecture
    - ⋆ Microfacet distribution is not important (according to the paper)

Legend to the following equations

- $\vec{N}$, $\vec{T}$, $\vec{B}$ are surface normal, tangent, and bitangent
- $\vec{L}$ is direction to the light, $\vec{V}$ is direction to the viewer
- $\vec{H}$ is half-vector, vector between the light and the viewer
- All dot products are non-negative, e.g.: $\max(0, \vec{N} \cdot \vec{L})$
  - For better result, clamp them to be non-zero, e.g. not less than 0.001, to avoid divisions by zero
- All vectors are normalized
- $Fresnel(\vec{V} \cdot \vec{H}) = F_0 + (1 - F_0)(1 - \vec{V} \cdot \vec{H})^5$
- Geom. atten. $G = min(1, \frac{2 \cdot (\vec{N} \cdot \vec{H}) \cdot (\vec{N} \cdot \vec{V})}{(\vec{V} \cdot \vec{H})}, \frac{2 \cdot (\vec{N} \cdot \vec{H}) \cdot (\vec{N} \cdot \vec{L})}{(\vec{V} \cdot \vec{H})})$

Uniform sampling for diffuse lighting

Output: Random direction $\vec{r}$ on a hemisphere (in the direction of z)
Input: Two random numbers $R.x$ and $R.y$, uniformly distributed in $(0, 1)$

begin

$\phi \leftarrow 2\pi \cdot R.x$

$\cos(\theta) \leftarrow R.y$

$\sin(\theta) \leftarrow \sqrt{1 - \cos^2(\theta)}$

$\vec{r}.x \leftarrow \sin(\theta)\cos(\phi)$

$\vec{r}.y \leftarrow \sin(\theta)\sin(\phi)$

$\vec{r}.z \leftarrow \cos(\theta)$

return $\vec{r}$

end

Computation of diffuse lighting

Output: Diffuse color *color*
begin
  $color \leftarrow (0, 0, 0)$
  forall diffuse samples *i* **do**
    $R.x, R.y \leftarrow$ *i*-th pair of random numbers
    $\vec{r} \leftarrow$ random direction from $R.x, R.y$
    $\vec{L} \leftarrow \vec{r}.x \cdot \vec{T} + \vec{r}.y \cdot \vec{B} + \vec{r}.z \cdot \vec{N}$
    $light \leftarrow SampleCubeTexture(\vec{L})/\#samples$
    $color \leftarrow color + C_{diff} \cdot (\vec{N} \cdot \vec{L}) \cdot light$
  end
  return *color*
end

Non-uniform importance sampling for specular lighting

Output: Random direction $\vec{r}$ on a hemisphere (in the direction of z)
Input: Two random numbers $R.x$ and $R.y$, uniformly distributed in $(0, 1)$, roughness m
begin

$\phi \leftarrow 2\pi \cdot R.x$

$\cos(\theta) \leftarrow \sqrt{\frac{1-R.y}{1+(m^2-1)R.y}}$

$\sin(\theta) \leftarrow \sqrt{1 - \cos^2(\theta)}$

$\vec{r}.x \leftarrow \sin(\theta)\cos(\phi)$

$\vec{r}.y \leftarrow \sin(\theta)\sin(\phi)$

$\vec{r}.z \leftarrow \cos(\theta)$

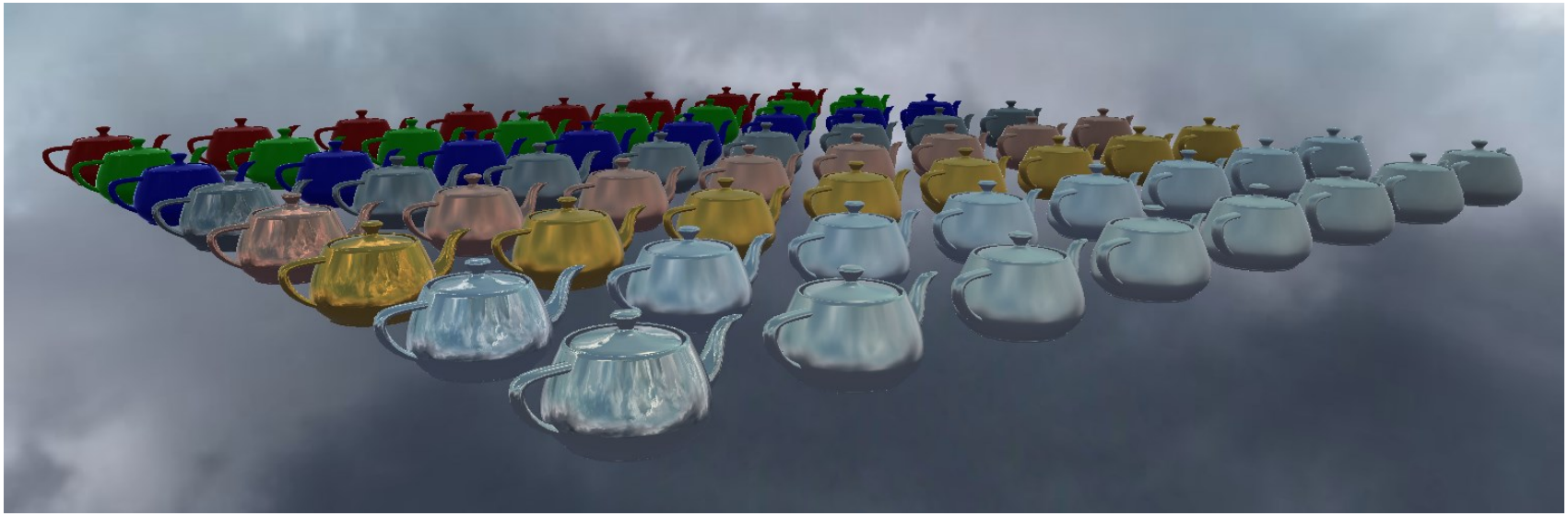
return $\vec{r}$

end

Computation of specular lighting

Output: Specular color *color*
begin
  $color \leftarrow (0, 0, 0)$
  forall specular samples *i* **do**
    $R.x, R.y \leftarrow$ another i-th pair of random numbers
    $\vec{r} \leftarrow$ random direction from $R.x, R.y$
    $\vec{H} \leftarrow \vec{r}.x \cdot \vec{T} + \vec{r}.y \cdot \vec{B} + \vec{r}.z \cdot \vec{N}$
    $\vec{L} \leftarrow reflect(-\vec{V}, \vec{H})$
    $light \leftarrow SampleCubeTexture(\vec{L})/\#samples$
    $F \leftarrow Fresnel(\ldots)$
    $G \leftarrow GeometricAttenuation(\ldots)$
    $color \leftarrow color + F \cdot G \cdot (\vec{V} \cdot \vec{H})/((\vec{N} \cdot \vec{H}) \cdot (\vec{N} \cdot \vec{V})) \cdot light$
  end
  return *color*
end

Task: Test scene



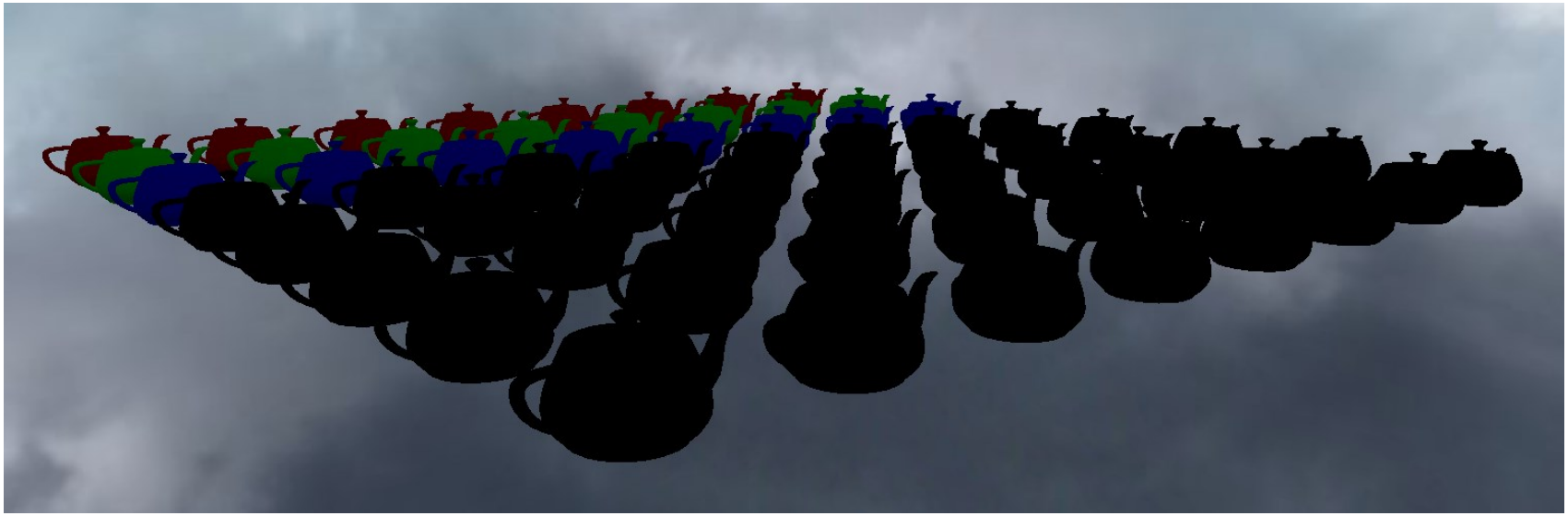
Test scene

- Materials: red/green/blue plastics, iron, copper, gold, alluminium, silver
- Roughness: 0.0, 0.05, 0.1, 0.15, 0.2, 0.3, 0.4, 0.5

Task: IBL with diffuse lighting

- **Task 1:** Implement diffuse lighting
  - Fragment shader *object_fragment.glsl*
  - Try higher number of samples
  - Try sampling higher mipmap-levels of cube map texture

Task: IBL with diffuse lighting

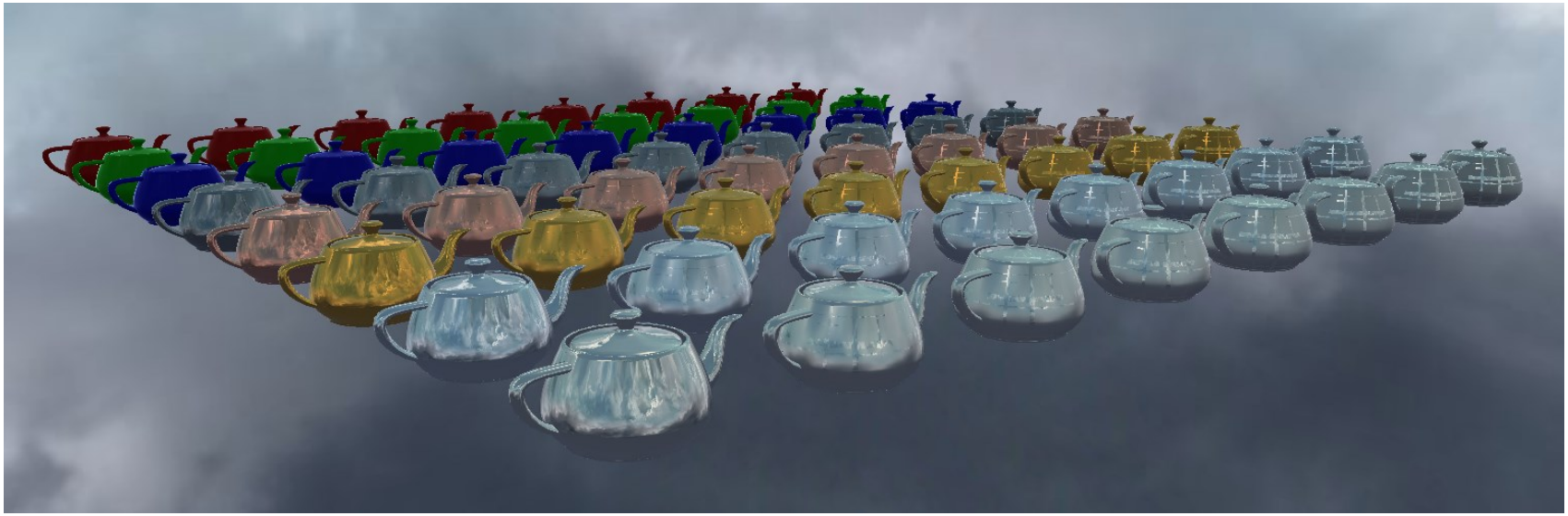


Result, metals have zero diffuse light

Task: IBL with specular lighting

- **Task 2:** Implement specular lighting
  - Try higher number of samples
  - Try sampling higher mipmap-levels of cube map texture
  - Try using a mask texture to change the roughness

Task: IBL with specular lighting

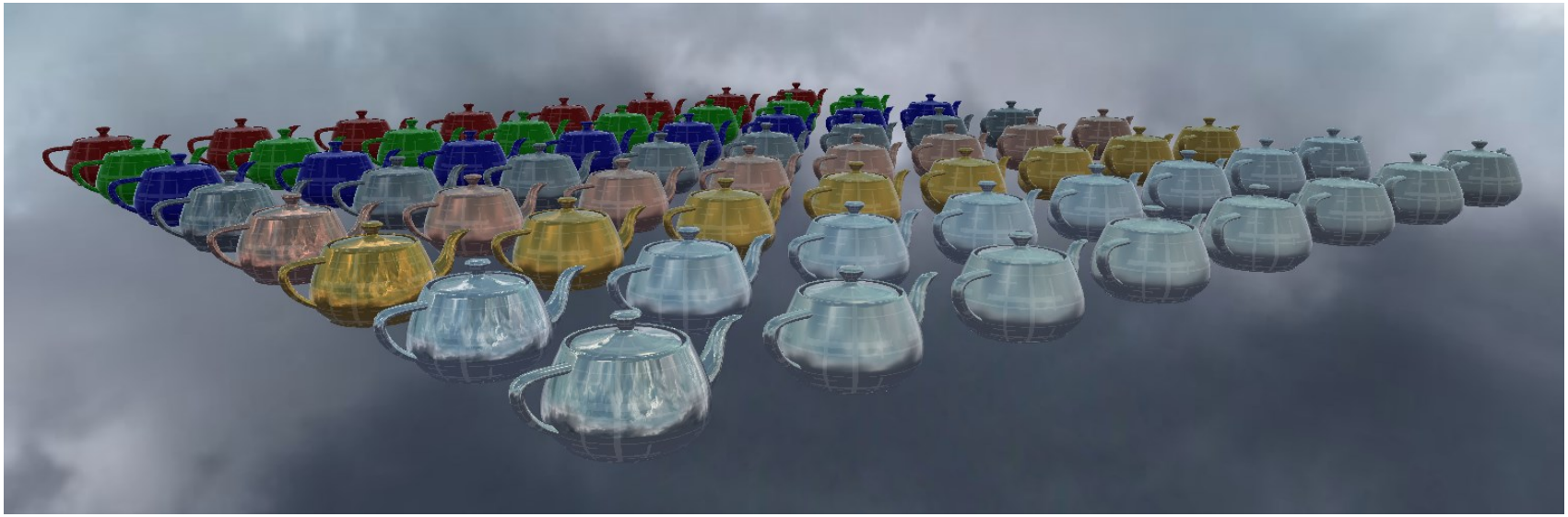


Result, with masked roughness

Task: Layered material

- **Task 3:** Create a thin shiny layer
  - The layer is completely transparent (except for the perfect reflection)
  - Set its base Fresnel reflectance to 0.04 (it is a dielectric material)
  - Try using a mask texture to create parts of semitransparent white areas.

Task: Layered material



Result, with masked semitransparent areas

Things we used

- Depth-prepass
  - Some graphic cards reorder evaluation of fragment shaders and evaluation of the depth test (when safe)
    - ⋆ Depth test is first, skipping FS when the fragment is hidden
  - Sometimes, it is benefical to render the whole scene very simply into depth buffer first, and then into the color buffer
    - ⋆ Each fragment is evaluated only once
    - ⋆ Rendering the objects from the closest also helps
- Early depth tests
  - Fragment shader: *layout (early_fragment_tests) in;*
  - Forces the above behaviour
  - Stencil test is also performed before running the fragment shader
  - Do not use this when you change the fragment depth or when you discard the fragment