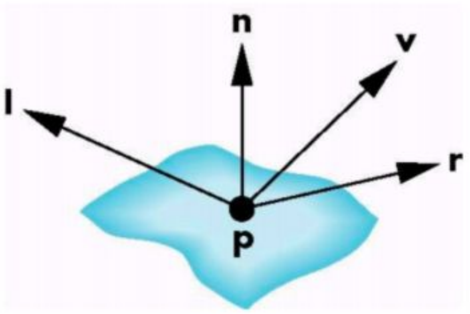
ICG 2024

HW3

2024/11/26

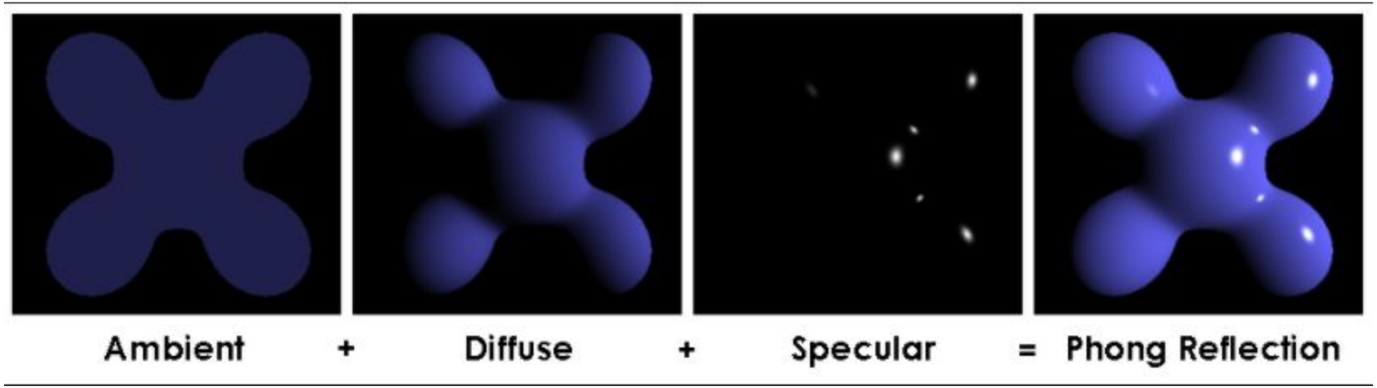
Phong Reflection Model

- Ambient = $L_{ambient} \times K_{ambient}$
- Diffuse = $L_{diffuse} \times K_{diffuse} \times (L \cdot N)$
- Specular = $L_{specular} \times K_{specular} \times (V \cdot R)^{a}$



$$I = I_{ambient} + I_{diffuse} + I_{specular}$$
$$= k_a I_a + k_d I_d (\mathrm{l} \cdot \mathrm{n}) + k_s I_s (\mathrm{v} \cdot \mathrm{r})^{\alpha}$$

-

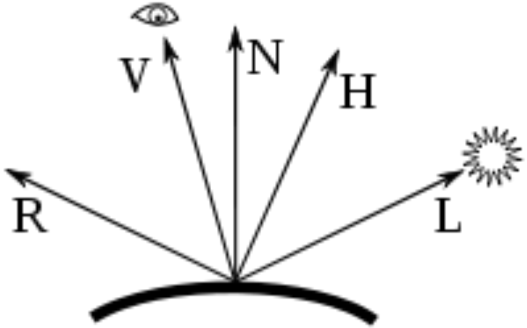


Bling-Phong Shading

- Calculate the halfway vector between viewer and light vector.

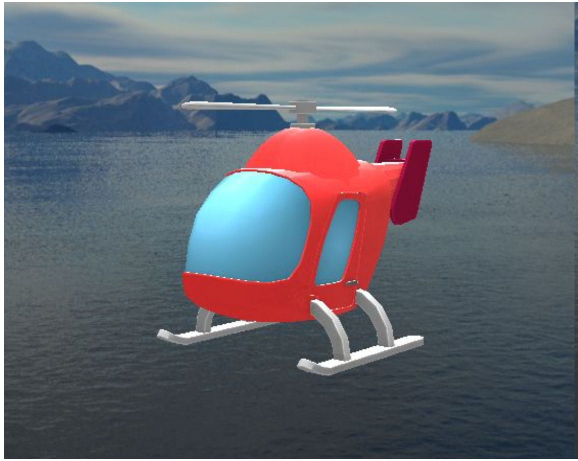
$$H = \frac{L + V}{||L + V||}$$

- Use $(N \cdot H)$ *to replace* $(R \cdot V)$, Specular = $L_{specular} \times K_{specular} \times (N \cdot H)^{a}$

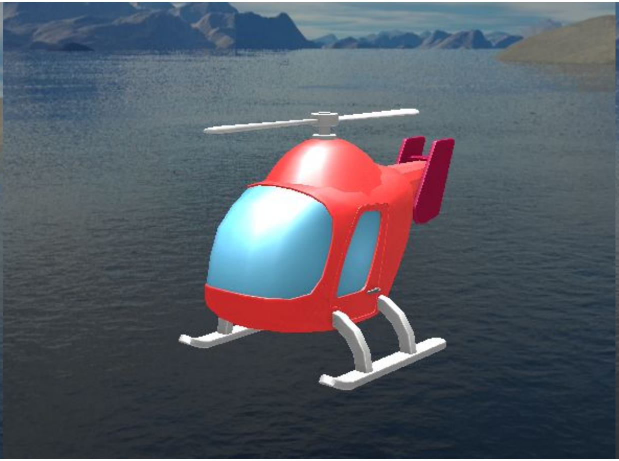


Gouraud Shading

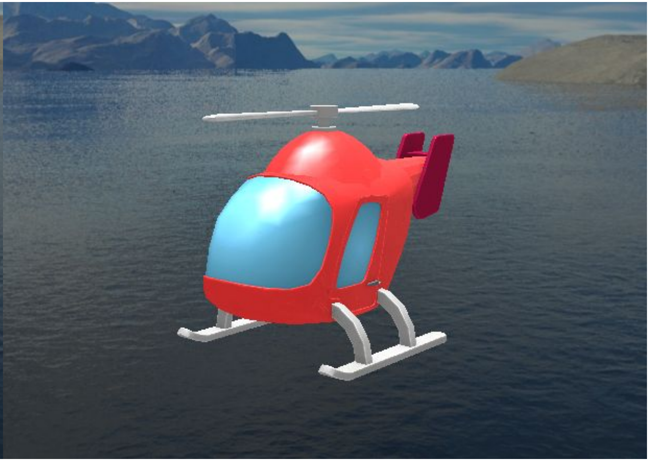
- Implement the Phong lighting model at each vertex.
- Define normals at each vertex and use them to calculate lighting.



Phong



Bling-Phong



Gouraud

Reflection & Refraction

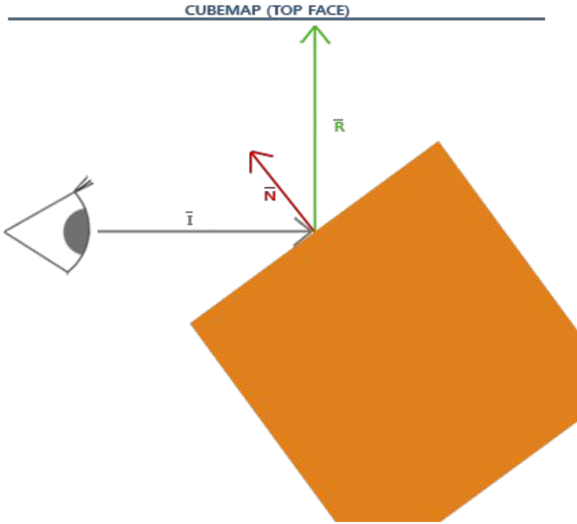
- Reflection
  - $R = I - 2 \cdot (I \cdot N)N$
- Refraction
  - According to Snell's law.

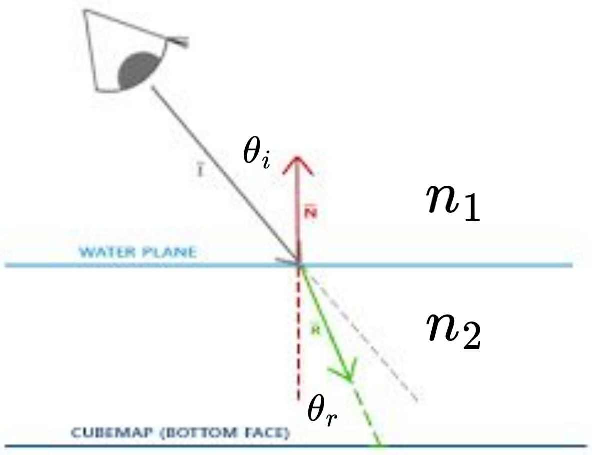
$$n_1 sin\theta_i = n_2 sin\theta_r, \rightarrow \eta \sin\theta_i = \sin\theta_r, \eta = \frac{n_1}{n_2}$$

$$\vdots$$

$$T = \eta I - (\eta(I \cdot N) + \sqrt{\underbrace{1 - \eta^2 \left(1 - (I \cdot N)^2\right)}_{\text{K}}})N$$

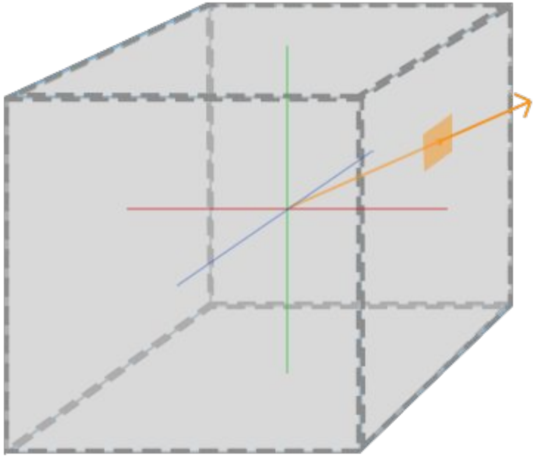
if K < 0, then set T to (0, 0, 0)





CubeMap

- A cubemap is a texture that contains 6 individual 2D textures that each form one side of a cube: a textured cube.
- Sampling a texture value from the cube map with an orange direction vector looks a bit like this:



CubeMap

- How make the cube texture appear far far away even if it is small.
  - Consider the homogenous coordinates

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \Rightarrow \quad \mathbf{q} = \begin{bmatrix} x \\ y \\ z \\ z/d \end{bmatrix}$$

  - Perspective division is performed after the vertex shader has run, dividing the gl_Position's xyz coordinates by its w component.

Metallic Shading

- ❖ Using reflection to sample the environment color.
- ❖ Lambertian Reflection (diffuse)

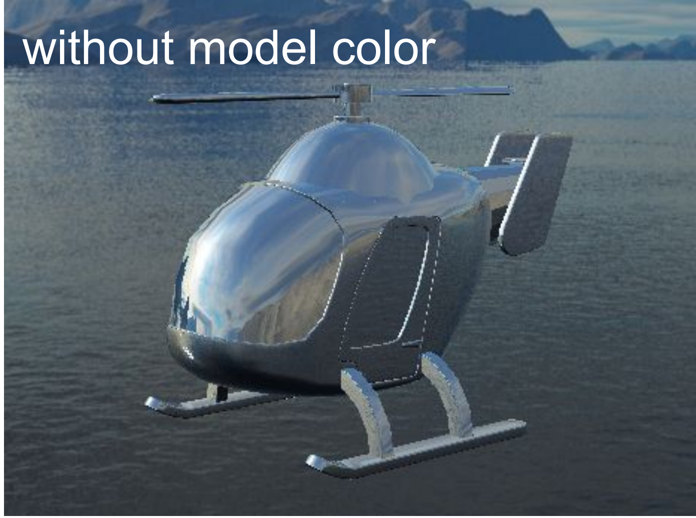
$$B_d = \max\left(\left(L \cdot N\right) I_l, 0\right)$$

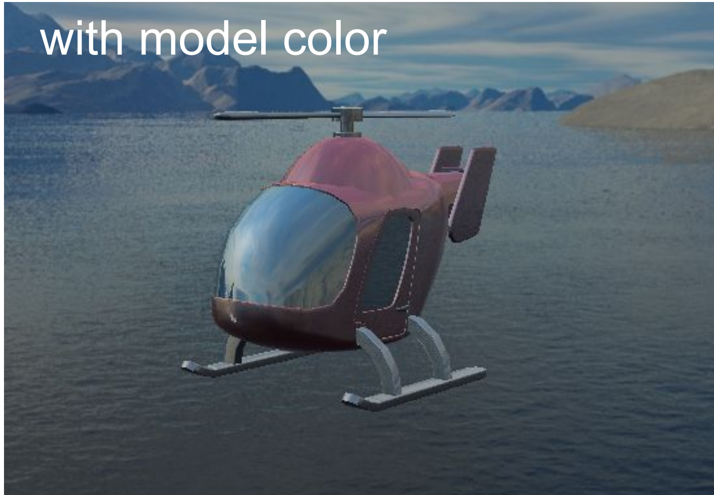
where L is the input light vector,
N is the sruface normal
I_l is the light intensity

$$B = B_d + \text{bias}$$

- ❖ Mix the reflect and original color using the a predefined ratio alpha.

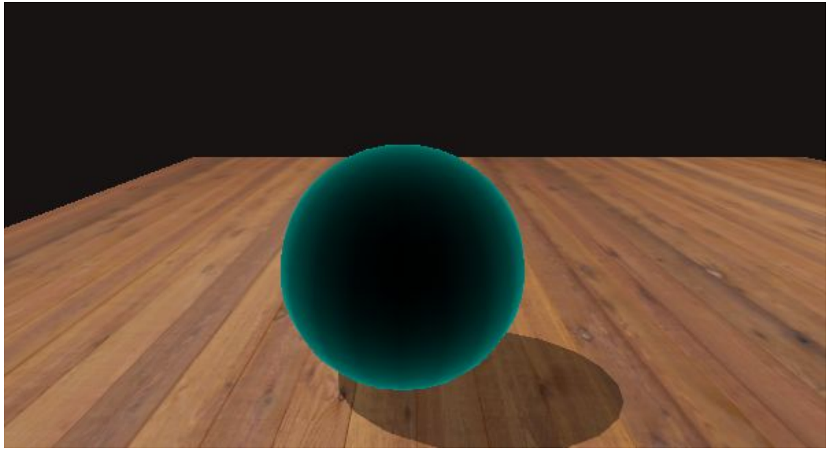
$$C_{\text{final}} = \alpha \times B \times C_{\text{model color}} + (1 - \alpha)\, C_{\text{reflect}}$$



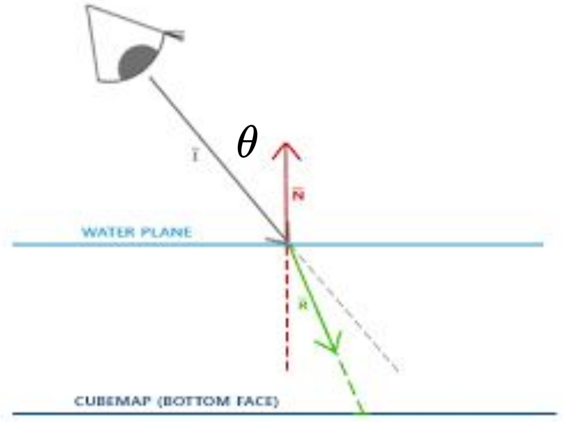


Fresnel Effect

- In general, when light reaches an interface between two materials, some light reflects off the surface at the interface, and some refracts through the surface. This phenomenon is known as the Fresnel effect.



Fresnel Effect (cont)



- Schlick Approximation

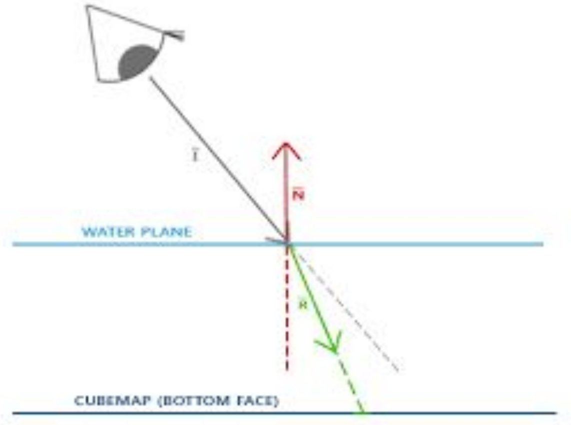
$$R_\theta = R_0 + (1 - R_0)(1 + I \cdot N)^5$$

$$R_0 = \left(\frac{n_1 - n_2}{n_1 + n_2}\right)^2, \text{ where } n \text{ is the refractive index}$$

$$C_{\text{final}} = R_\theta \, C_{\text{reflect}} + (1 - R_\theta) \, C_{\text{refract}}, \text{ C represents the color.}$$

- For more details checkout:
  - https://en.wikipedia.org/wiki/Schlick%27s_approximation

Fresnel Effect (cont)



- Empricial Approximation

$$R_\theta = \max\left(0, \min\left(1, \text{bias} + \text{scale} \times (1 + I \cdot N)^{power}\right)\right)$$

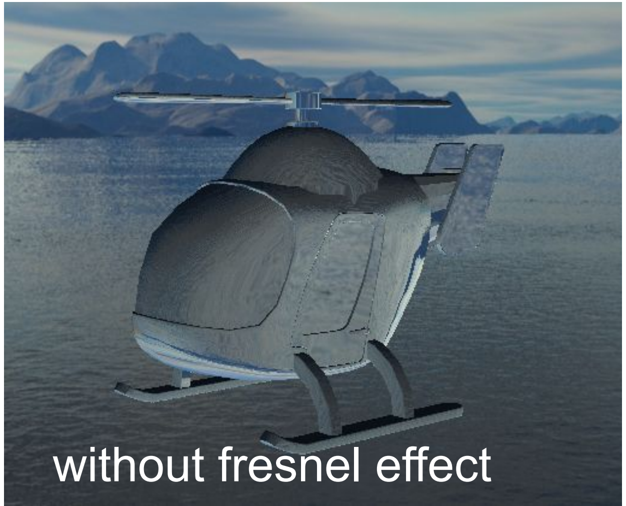
$$C_\text{final} = R_\theta \, C_\text{reflect} + (1 - R_\theta) \, C_\text{refract}, \text{ C represents the color.}$$

- For more details check out:
  - https://developer.download.nvidia.com/CgTutorial/cg_tutorial_chapter07.html

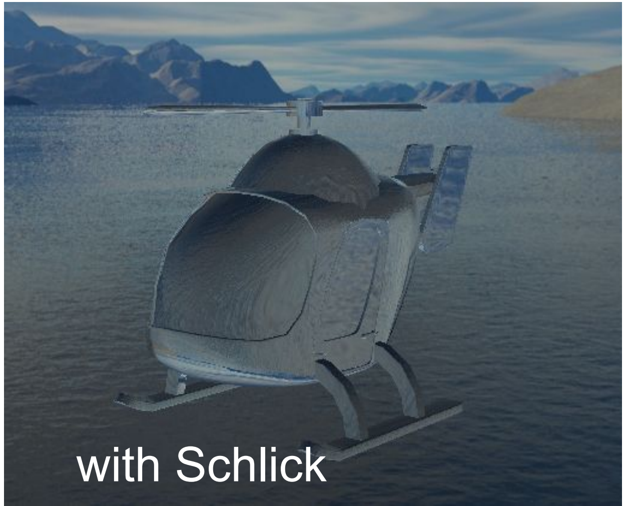
Glass Shading

- Using refraction to sample the environment color.
- Using the above mentioned methods (Schlick, Empricial) to determine the ratio of reflection and refraction.

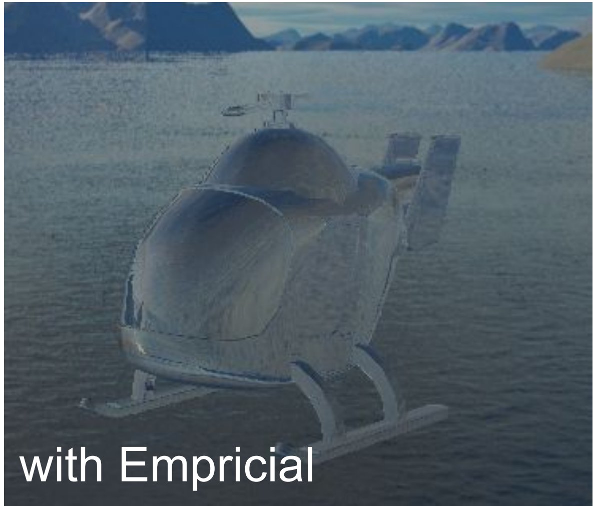
$$C_{\text{final}} = R_\theta \, C_{\text{reflect}} + (1 - R_\theta) \, C_{\text{refract}}.$$



without fresnel effect



with Schlick



with Empricial

Homework 3 - Introduction

Story

- ## The Awakening Above the Lake

The helicopter hummed softly, descending toward Earth. Fatigue overcame the pilot, and as his hands slipped from the controls, darkness claimed him.

He awoke to the helicopter hovering motionless above a mirror-like lake. The console blinked with five glowing buttons, each pulsing rhythmically. Hesitant yet curious, he pressed the first one.

A resonant hum filled the cabin as the helicopter transformed. Its surface turned smooth and glossy, reflecting sunlight with brilliance. Another button softened its glow, creating a flowing, ethereal sheen. The next press rendered it metallic, gleaming with cold, intricate reflections. Then, a shimmering glass form emerged, its edges glowing with iridescent light, blending with the lake's tranquil beauty.

Beneath him, the lake shimmered, its surface alive with vibrant colors that mirrored the helicopter's evolving form.

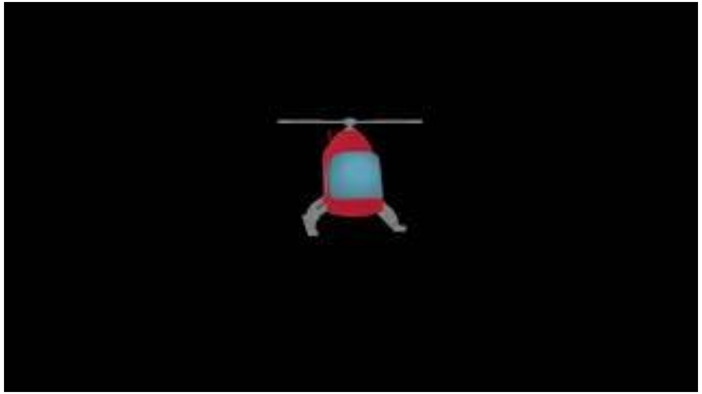
"What… is this?" he whispered, awe and unease in his voice.

Pressing the final button, the helicopter became a radiant fusion of all its forms. It surged forward, skimming the lake and soaring skyward, leaving trails of light in its wake. No longer just a machine, it was alive—a partner guiding him into the unknown.

Demo

- https://youtu.be/enTJ6zVwDII

What you can use



Default view

https://youtu.be/4W6NUD1Yr0g

Keyboard control

Key 1: Switch to Bling-Phong shading

Key 2: Switch to Gouraud shading

Key 3: Switch to Matellic shading

Key 4: Switch to Glass_schlick shading

Key 5: Switch to Glass_empricial shading

KeyW: Zoom in

Key S: Zoom out

Key D: Rotate camera to the right.

Key A: Rotate camera to the left.

Some functions you can use:

shader_program_t.set_uniform_value(name, type)

shader_program_t.use() // use the selected shader program

shader_program_t.release() // release the selected program

Requirements

- TODO1: Load parameters
  - Set and load the needed values into your shader (material and lighting coefficient for example).
  - You can use the functions in the shader_program_t class to implement this part.
  - You can use the default settings or use your own setting as long as it looks good :)

Settings	Position	ambient	diffuse	specular
Light	(0, 1000, 0)	(1.0, 1.0, 1.0)	(1.0, 1.0, 1.0)	(1.0, 1.0, 1.0)

Settings	gloss	ambient	diffuse	specular
Material	10.5	(1.0, 1.0, 1.0)	(1.0, 1.0, 1.0)	(0.7, 0.7, 0.7)

Requirements

- TODO2: Bling-Phong Shader
  - Implement the bling-phong.vert & bling-phong.frag

- TODO3: Gouraud Shader
  - Implement the gouraud.vert & gouraud.frag

- TODO4: Environment Cubemap
  - Implement the cubemap.vert and cubemap.frag.
  - Draw the environment cube map.

Requirements

- TODO5: Metallic Shader
  - Set the hyperparameters for metallic shader. (bias = 0.2, alpha = 0.4, light intensity=1)
  - Implement reflect to sample the color from environment.
  - Mix the model texture with the reflected color using the equation in p.8.

- TODO6: Glass Shader
  - Set the hyperparameters for glass shader. (AIR_coeff = 1, GLASS_coeff = 1.52)
  - Implement refract to sample the color from the environment
  - Mix the color of the reflection and refraction using Schlick Approximation.
  - Mix the color of the reflection and refraction using Empricial Approximation.

Setting for empricial	Scale	Power	Bias
value	0.7	2	0.2

Score

1. Correctly setup the shader program (5%)
   1. Load the needed uniform values into the shader program.
2. Implement Bling-Phong shading via shader (15%)
   1. Each incorrect implementation (-3%)
3. Implement Gouraud shading via shader (15%)
   1. Each incorrect implementation (-3%)
4. Implement the cubemap environment (15%)
   1. Implement the cubemap.vert & cubemap.frag (5%)
   2. Draw the cubemap environment (10%)

Score

5. Implement metallic effect via shader (15%)
    a. Implement the reflect on your own. (5%)
    b. Mix the environment and model color base on the lighting and a predefined alpha(10%)
6. Implement glass effect via shader (15%)
    a. . Implement the refract on your own. (5%)
    b. Mix the environment and model color base using the Schlick Approximation. (5%)
    c. Mix the environment and model color base using the Empricial Approximation. (5%)
7. Report (20%)
    a. Brief discription of your implementation of TODO 2~6 (4% each)
8. Window name (-3%)

Homework 3 - Submission

- Deadline: 2024/12/10 23:59:59
  - 10% penalty for each week late
  - Final score = original score * 0.9 for less than a week late (12/11 ~ 12/17)
  - Final score = original score * 0.8 for one week late (12/18 ~ 12/24)
  - So on…
- Zip your main.cpp, shaders folder, report.pdf and upload the zip file to E3
- Zip name : studentID_HW3.zip (-5% for incorrect file name/format)
  - e.g.

```
12345678_HW3.zip
        ├─ src/
        │       ├─shaders /
        │       ├─ main.cpp
        │       └─ any additional files you need.
        └─ report.pdf
```

Reference

- https://learnopengl.com/
- https://www.glfw.org/documentation
- https://jcgt.org/published/0003/04/03/
- https://developer.download.nvidia.com/CgTutorial/cg_tutorial_chapter07.html