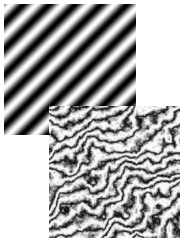
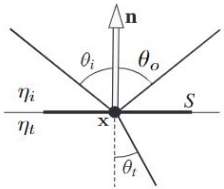
Rendering Marble

Zahra Ghavasieh

Project Plan





Skeleton and Environment Set up

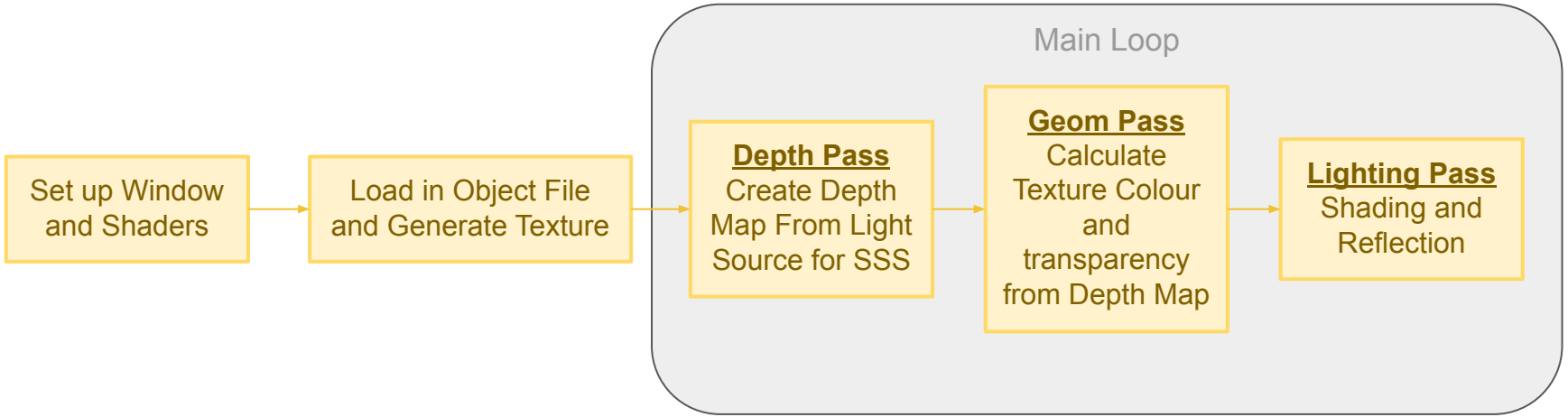
Procedural Texture Generation

Subsurface Scattering Approximation

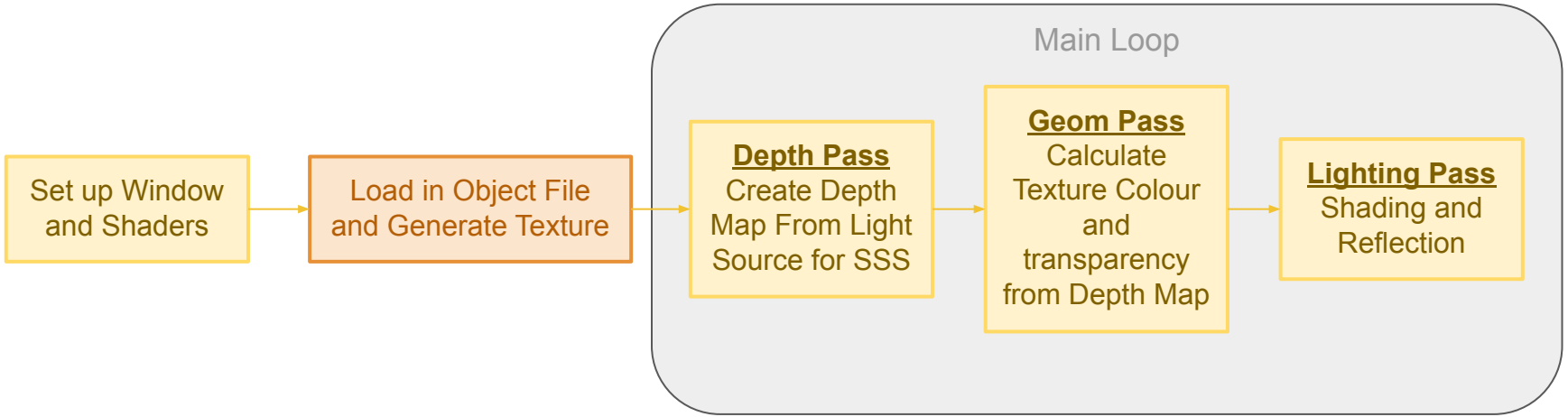
Reflection Approximation

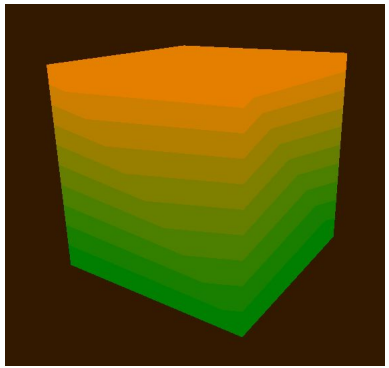
Realistic Marble Model

Pipeline


Main Loop
Set up Window and Shaders
Load in Object File and Generate Texture
Depth Pass
Create Depth Map From Light Source for SSS
Geom Pass
Calculate Texture Colour and transparency from Depth Map
Lighting Pass
Shading and Reflection

Pipeline

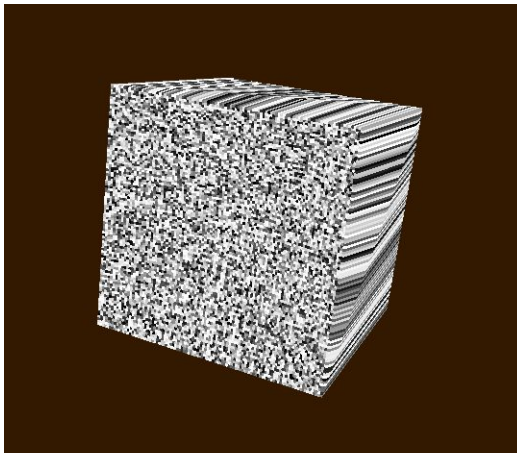

Main Loop
Set up Window and Shaders
Load in Object File and Generate Texture
Depth Pass
Create Depth Map From Light Source for SSS
Geom Pass
Calculate Texture Colour and transparency from Depth Map
Lighting Pass
Shading and Reflection



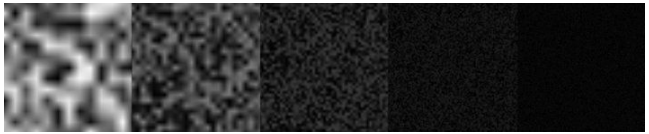
Texture Mapping

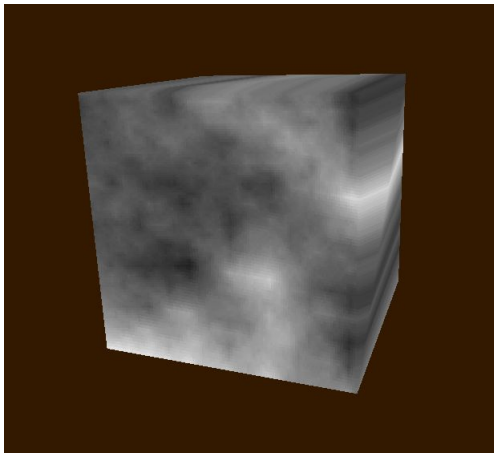
```
//Procedurally Generated Texture
for (float s = 0; s < texturewidth; s++) {
        for (float t = 0; t < textureheight; t++) {
                generateTexture(s,t);
        }
}

// Texture Mapping
for (auto vertex : vertices) {
        textureCoords.push_back(vec2(vertex.x, vertex.z));
}
```

Texture Generation: Perlin Noise



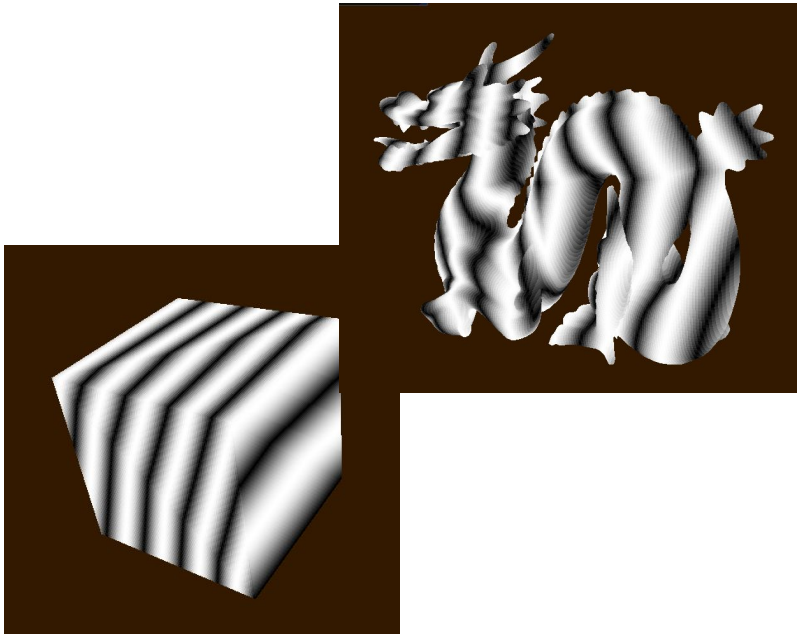
White Noise





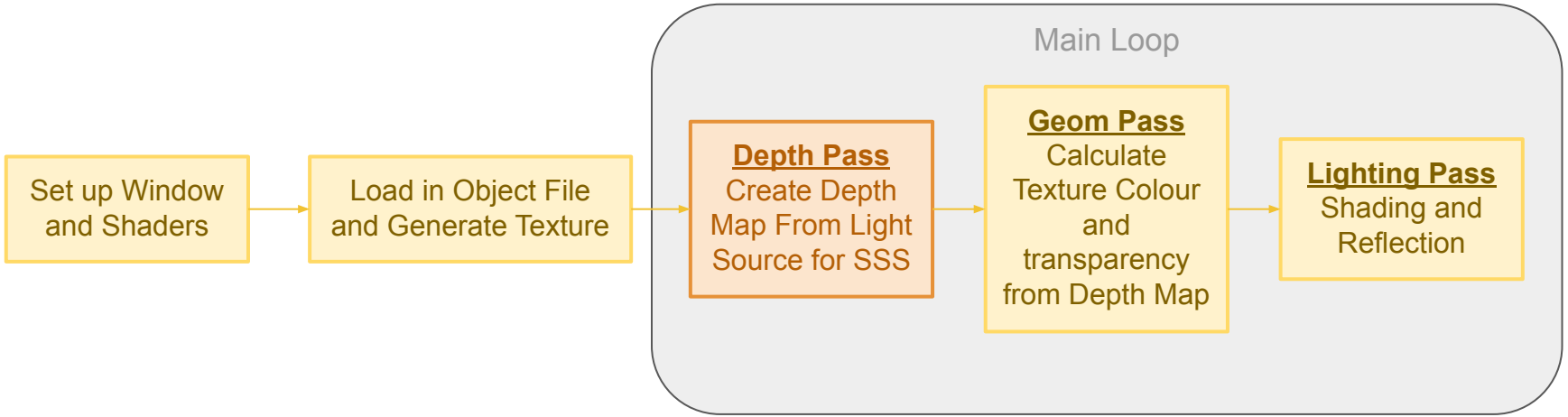
Smoothing + Turbulence

Texture Generation: Sine Wave

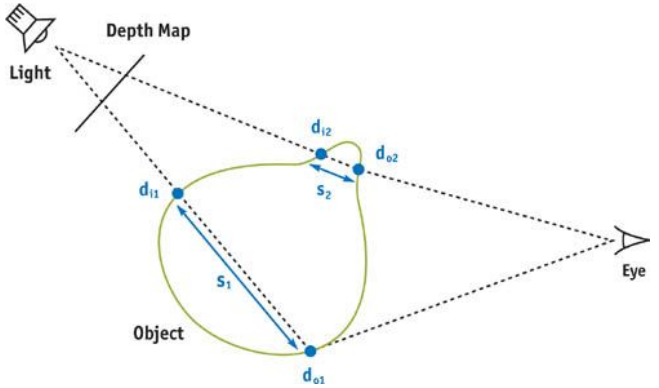


Generated Marble Textures

Pipeline

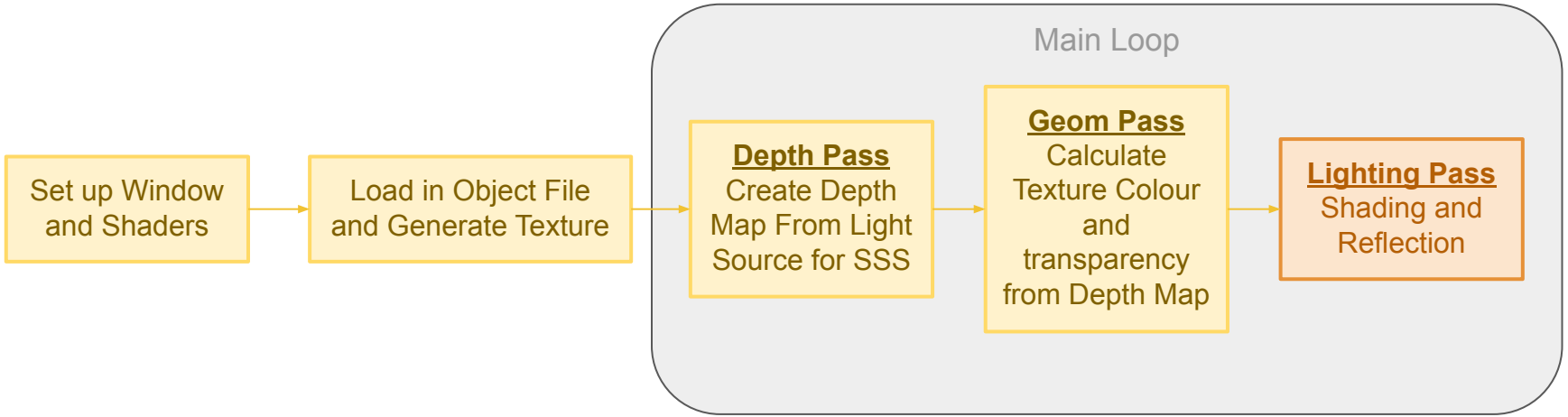

Main Loop
Set up Window and Shaders
Load in Object File and Generate Texture
Depth Pass
Create Depth Map From Light Source for SSS
Geom Pass
Calculate Texture Colour and transparency from Depth Map
Lighting Pass
Shading and Reflection

Subsurface Scattering



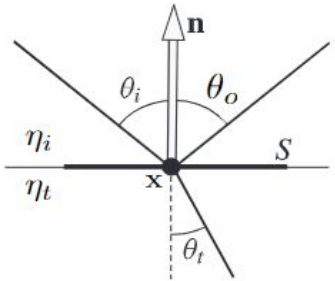
- 1st Pass: Calculate the Distance Light Has Traveled Through an Object Using a Depth Map
- Manipulate Color by taking into account the Depth Map

Pipeline


Main Loop
Set up Window and Shaders
Load in Object File and Generate Texture
Depth Pass
Create Depth Map From Light Source for SSS
Geom Pass
Calculate Texture Colour and transparency from Depth Map
Lighting Pass
Shading and Reflection

Reflection

- Fresnel Reflectance in real time
- Schlick Fresnel Approximation (specular reflection coefficient R)



$$R(\theta) = R_0 + (1 - R_0)(1 - \cos\theta)^5$$

where

$$R_0 = \left(\frac{n_1 - n_2}{n_1 + n_2}\right)^2$$

References

Real-time Rendering:
- https://learnopengl.com/

Procedural Texture:
- https://lodev.org/cgtutor/randomnoise.html
- http://physbam.stanford.edu/cs448x/old/Procedural_Noise(2f)Perlin_Noise.html

Reflections:
- https://belcour.github.io/blog/slides/2020-brdf-fresnel-decompo/index.html#/5/0/0

Subsurface Scattering:
- https://developer.nvidia.com/gpugems/gpugems/part-iii-materials/chapter-16-real-time-approximations-subsurface-scattering

Thank You!