Real-Time Rendering of Ocean Water With Dynamic Skybox

Aaron Hornby

GOALS

- Implement a real-time simulation of ocean water/waves in a photorealistic manner.
- Implement a dynamic skybox and environment that influences water shading.
- Provide a well-documented open source repository for others seeking to learn the techniques.
- Provide a sandbox environment with many tweakable parameters to influence both water mesh representation and visuals.
- Multi-platform.

Motivation

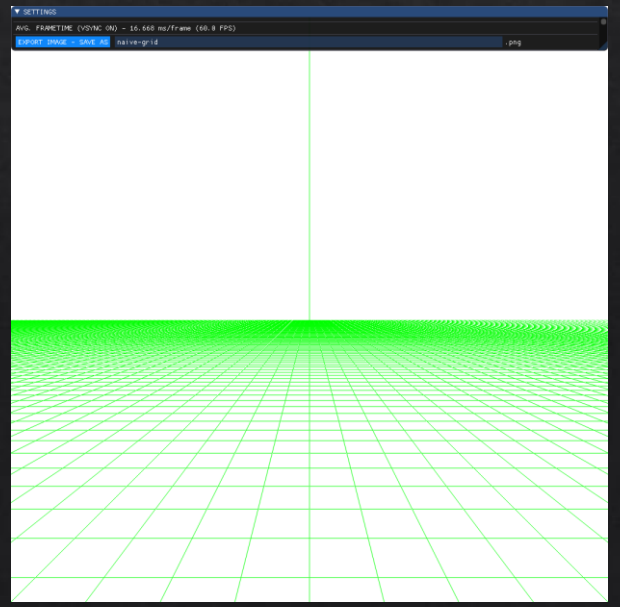
- Prevalent topic, important appeal.
- Every water renderer is different.
- Many subtleties that can be modified to get different looks.

The Challenge (water)

- Infinite surface – broad representation (requires LOD scheme), must move with the camera.
- Multiple surface properties (amplitude, frequency, speed, steepness, direction).
- Smaller scale noise for a more refined look.
- Shading is a composition of many layers (tint, clarity, edge smoothing, local reflections, local refractions, global reflections, Fresnel reflectance, sun specular highlight, fog, distortion, ambient light, etc.).
- Reduce amount of visual tiling / repetition.
- Knowing when to stop?

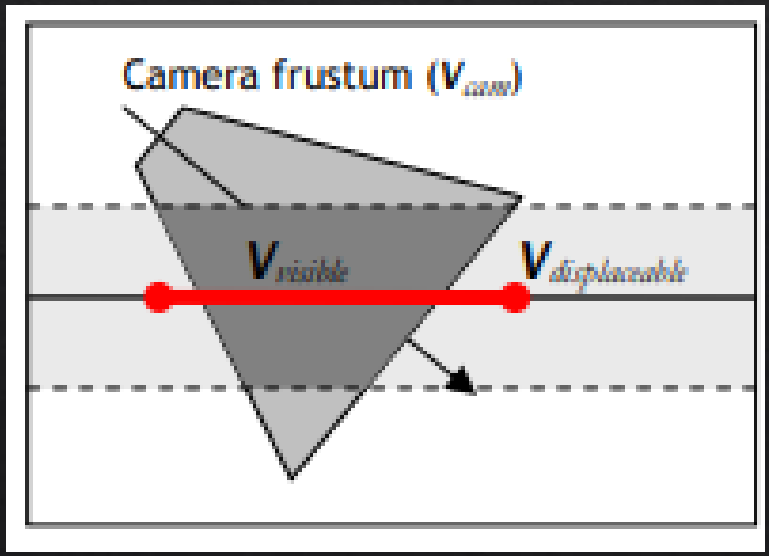
Surface Representation (naïve approach)

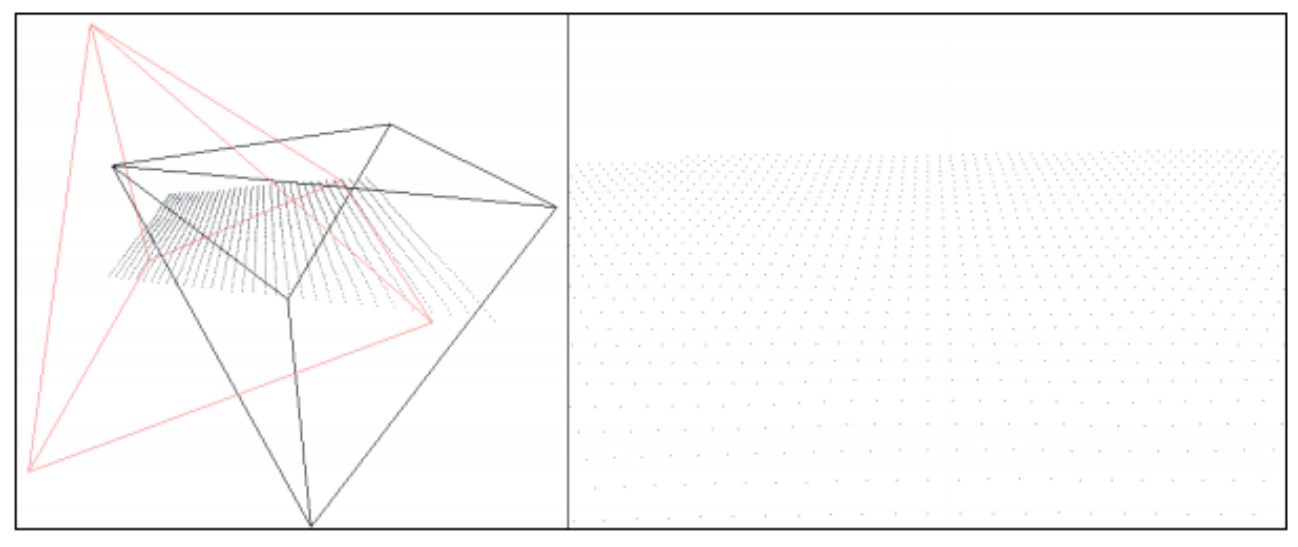
- Construct a planar grid along the XZ-plane in our scene, with equally distributed points in world-space.
- Problems?
- How do you represent an infinite surface that spans towards horizon?
- High close-detail means subdividing the entire grid, must of which is far away or offscreen (low vertex efficiency).
- Visual artifacts farther away.



The Projected Grid Concept (the solution)

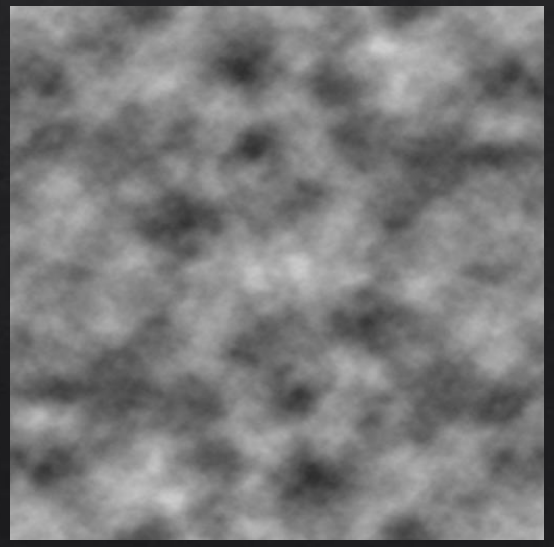
- Claes Johanson, “Real-time water rendering – Introducing the projected grid concept” [2004].
- XZ-planar grid fit into camera frustum, pseudo-LOD, roughly equal distribution of grid points in post-perspective space.
- Relatively simple to implement versus other LOD schemes.
- Problems? Mainly the “swimming artifacts”. Less relevant for water that naturally “swims”.





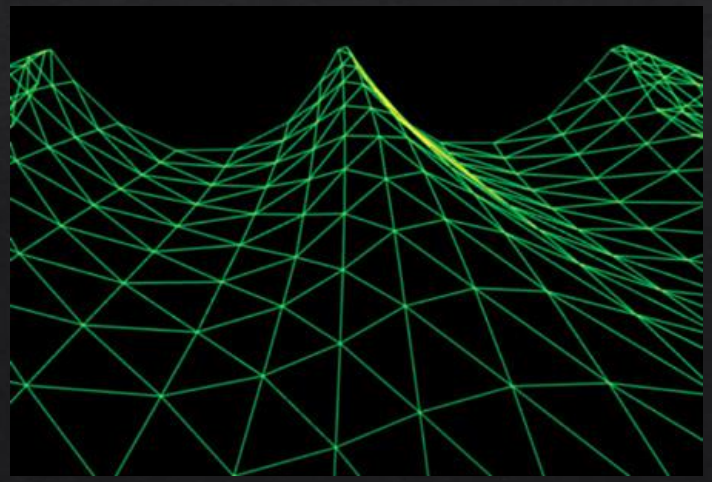
First Attempt (Heightmap)

- Sample a grayscale noise texture (the heightmap) and displace only the world-space grid points in the +y axis.
- Simple, but tiling is an eyesore.
- Not suitable for waves that require crests to be steeper.
- No directionality unless baked into heightmap and animated.



Second attempt (Gerstner Waves)

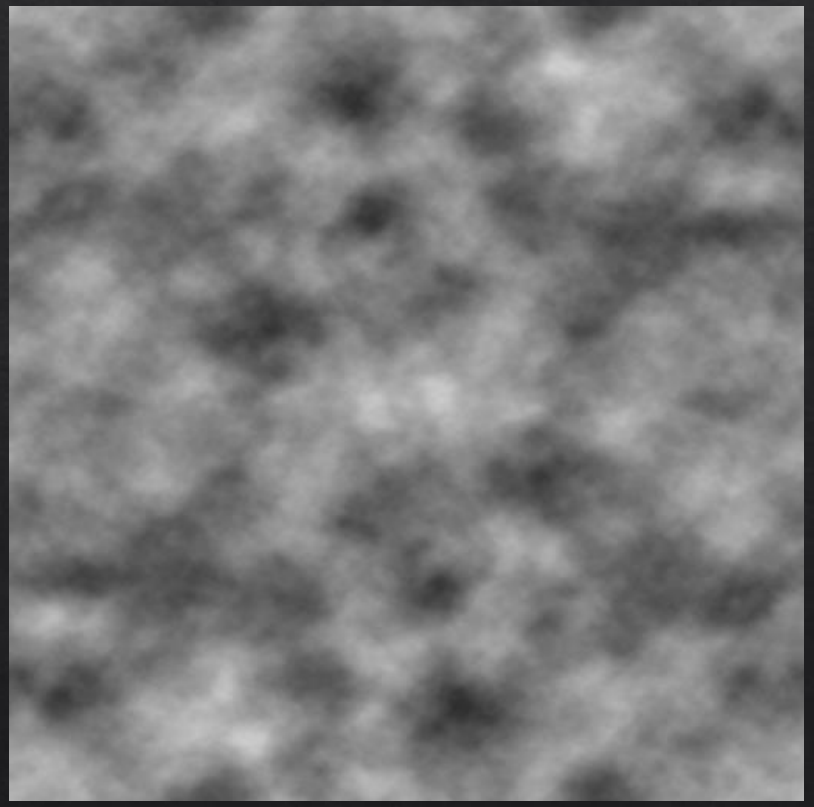
- Mark Finch, Cyan Worlds, “Chapter 1. Effective Water Simulation from Physical Models”. Nvidia GPU Gems. 5th Printing, September, 2007.
- Displacement of grid points in XYZ/
- Amplitude, Frequency, Phase Constant (~Speed), Steepness, Direction.

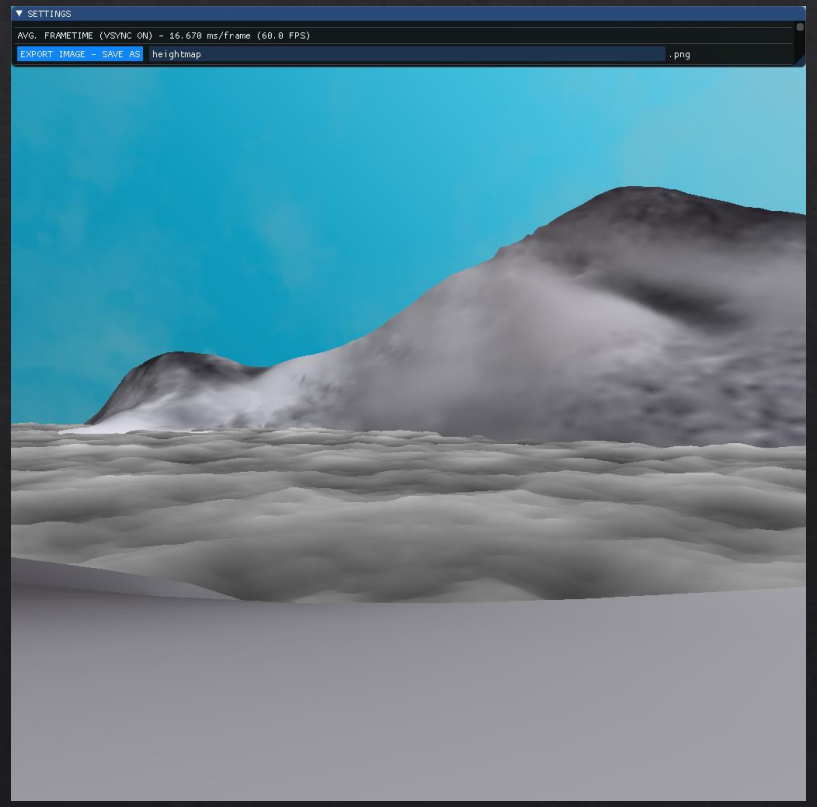


$$\mathbf{P}(x,y,t) = \begin{pmatrix} x + \sum\left(Q_i A_i \times \mathbf{D}_i.x \times \cos\left(w_i \mathbf{D}_i \cdot (x,y) + \varphi_i t\right)\right), \\ y + \sum\left(Q_i A_i \times \mathbf{D}_i.y \times \cos\left(w_i \mathbf{D}_i \cdot (x,y) + \varphi_i t\right)\right), \\ \sum\left(A_i \sin\left(w_i \mathbf{D}_i \cdot (x,y) + \varphi_i t\right)\right) \end{pmatrix}.$$

Heightmap (randomness)

- Two adjustable parameters: “roughness and steepness”.





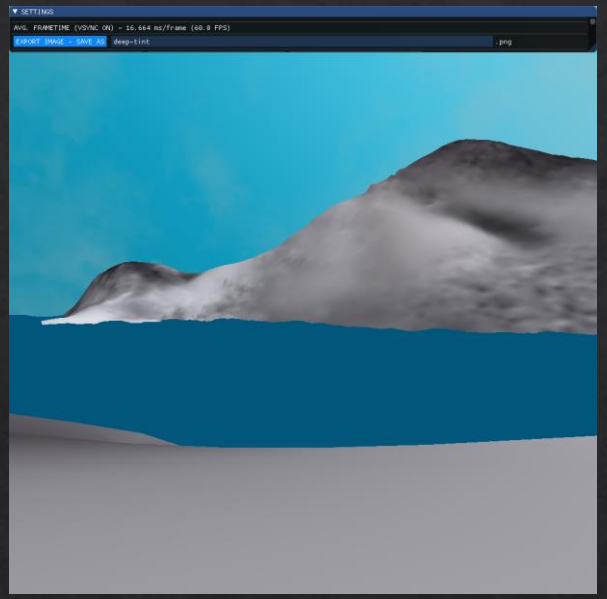
Overview of the shading

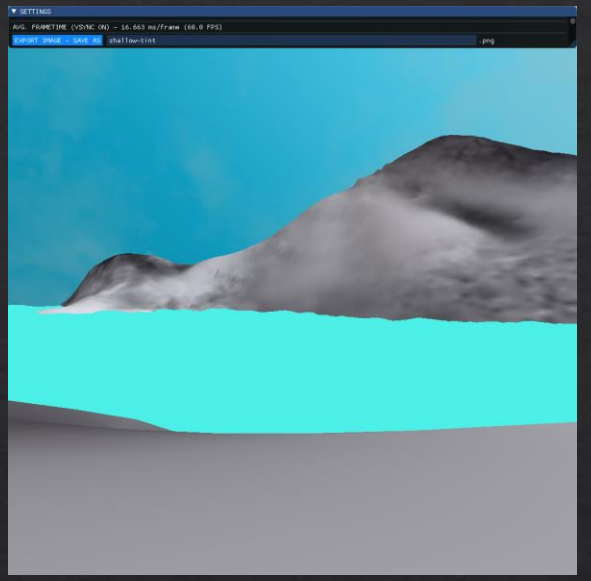
- A composition of many layers with distortion:
- Transmission Colour = {tint, refraction, clarity, depth}.
- Reflection Colour = {global/skybox, sun highlight, local}.
- colour = vec4(mix(waterFresneTransmissionColour, waterFresnelReflectionColour, Fresnel_f_theta), 1.0f);
- colour.a = edgeHardness;
- colour.rgb = mix(colour.rgb, fogColour.rgb, fogColour.a);

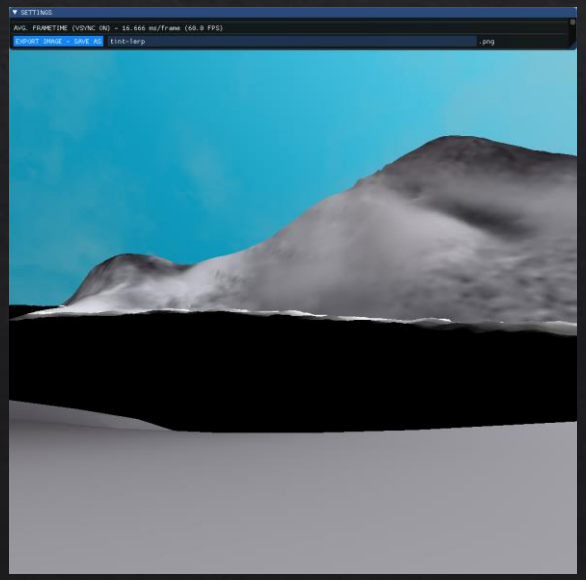
Tint

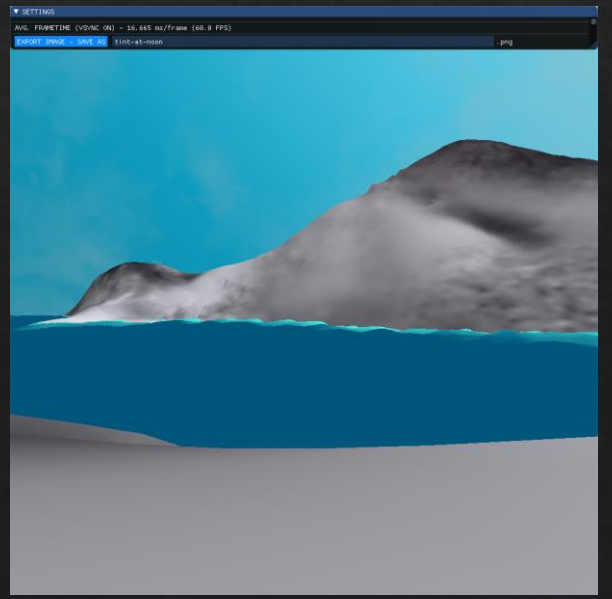
- Deep tint colour (Blue) ---> Shallow tint colour (Turquoise).
- Specify both colours at noon (brightest) which then automatically fade to black as sun lowers in sky.
- User parameter in [0.0, 1.0].

Tint (cont.)


▼ SETTINGS
AVG. FRAMETIME (VSYNC ON) - 16.664 ms/frame (60.0 FPS)
EXPORT IMAGE - SAVE AS
deep-tint
.png


▼ SETTINGS
AVG. FRAMETIME (VSYNC ON) - 16.663 ms/frame (60.0 FPS)
EXPORT IMAGE - SAVE AS
shallow-tint
.png

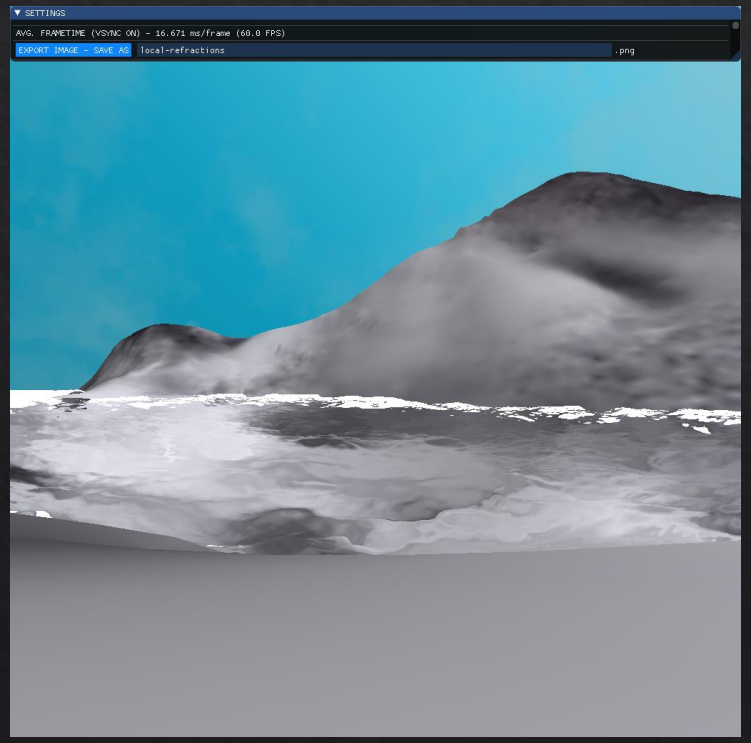

▼ SETTINGS
AVG. FRAMETIME (VSYNC ON) - 16.666 ms/frame (60.0 FPS)
EXPORT IMAGE - SAVE AS
tint-lerp
.png

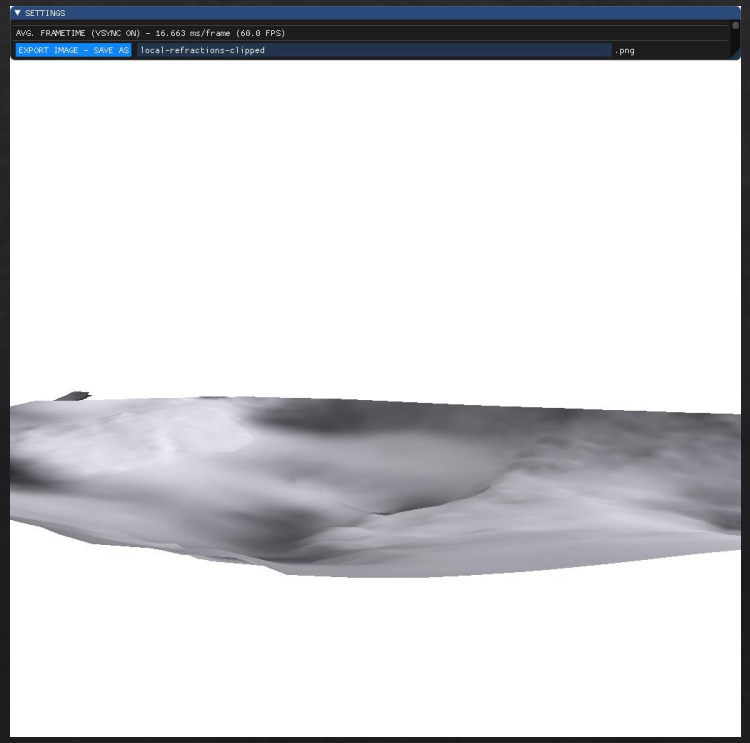

▼ SETTINGS
AVG. FRAMETIME (VSYNC ON) - 16.665 ms/frame (60.0 FPS)
EXPORT IMAGE - SAVE AS
tint-at-noon
.png

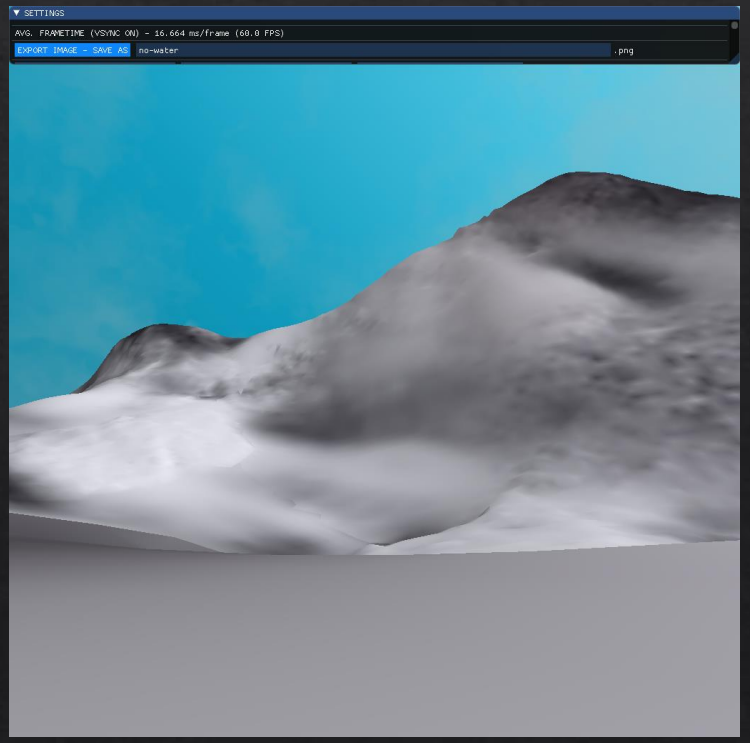
Local Refractions

- Render scene to texture
- Render geometry with world-space position.y at 75% (n_air = 1.0003 / n_water = 1.33)
- Clip plane at XZ-plane (leaves geometry below plane)
- Enable back-face culling with standard winding (counter-clockwise) – avoid higher geometry now occluding your view.
- Problems? Approximating the surface with a plane means that larger error when amplitude increases. Noticeable at intersections, but will be hidden by other effects (foam, soft edges)

Local Refractions (cont.)

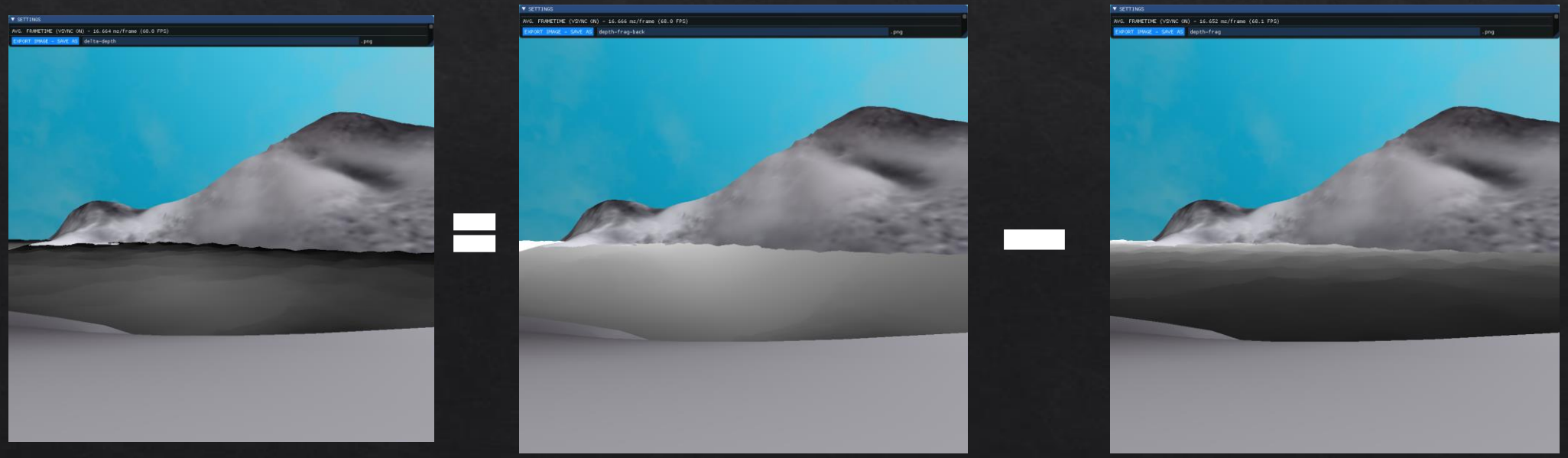






Depth & Clarity

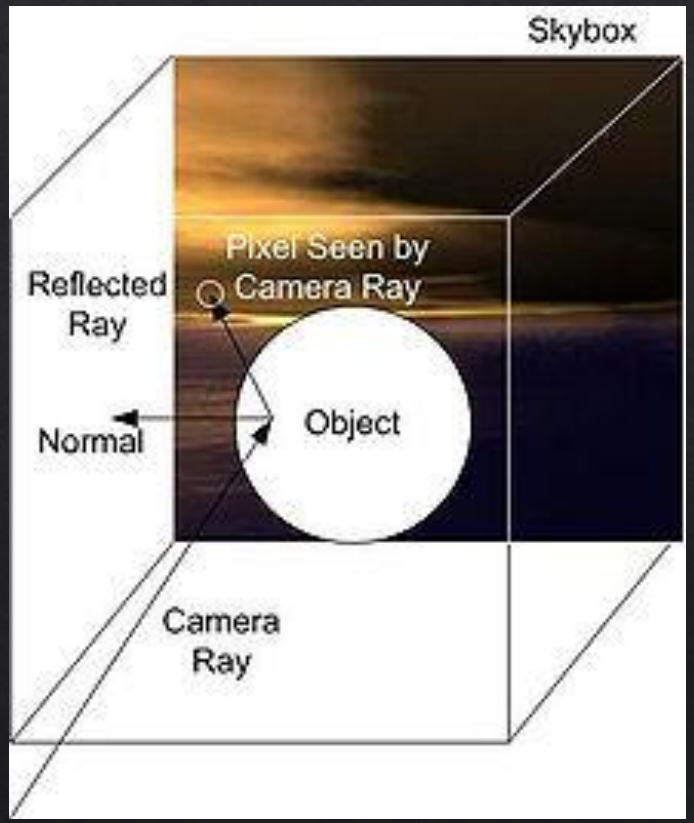
- Depth used for many effects.
- Important notice: make sure to take advantage of the full 24-bit depth buffer precision when performing comparison. Otherwise, you end up with larger colour bands and some jagged edges.
- User parameter (multiplier of the depth delta – how far into water can we see refractions).

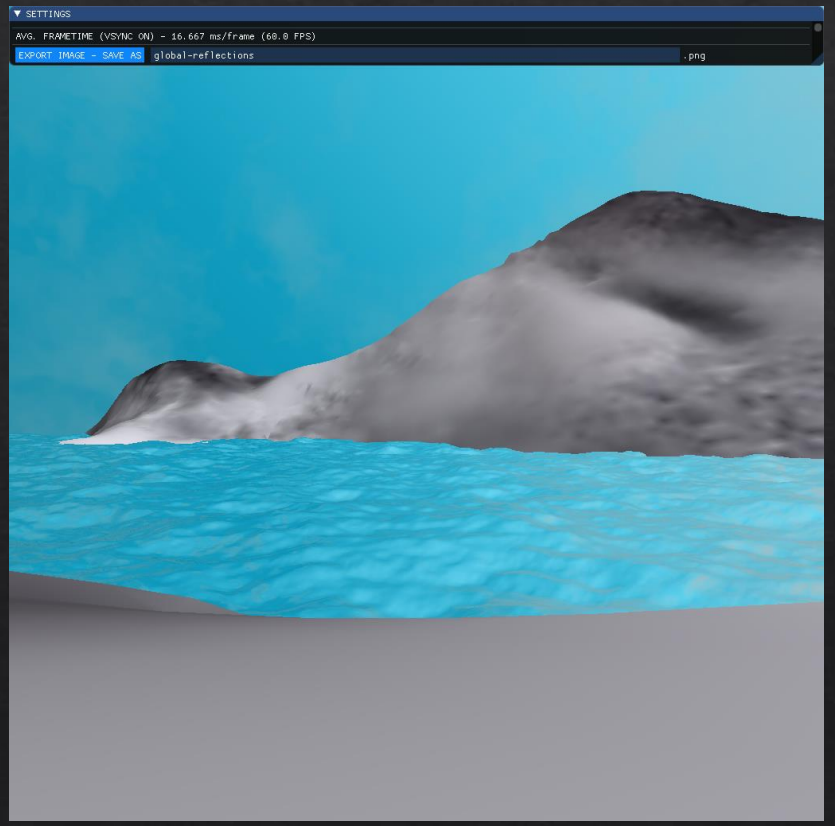


Pitfall – early discretization

Global Reflections

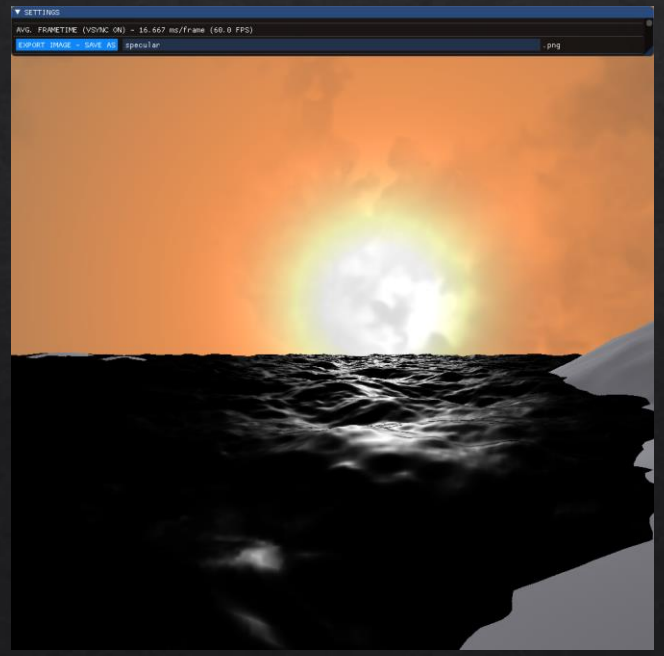
- Simple skybox mapping.

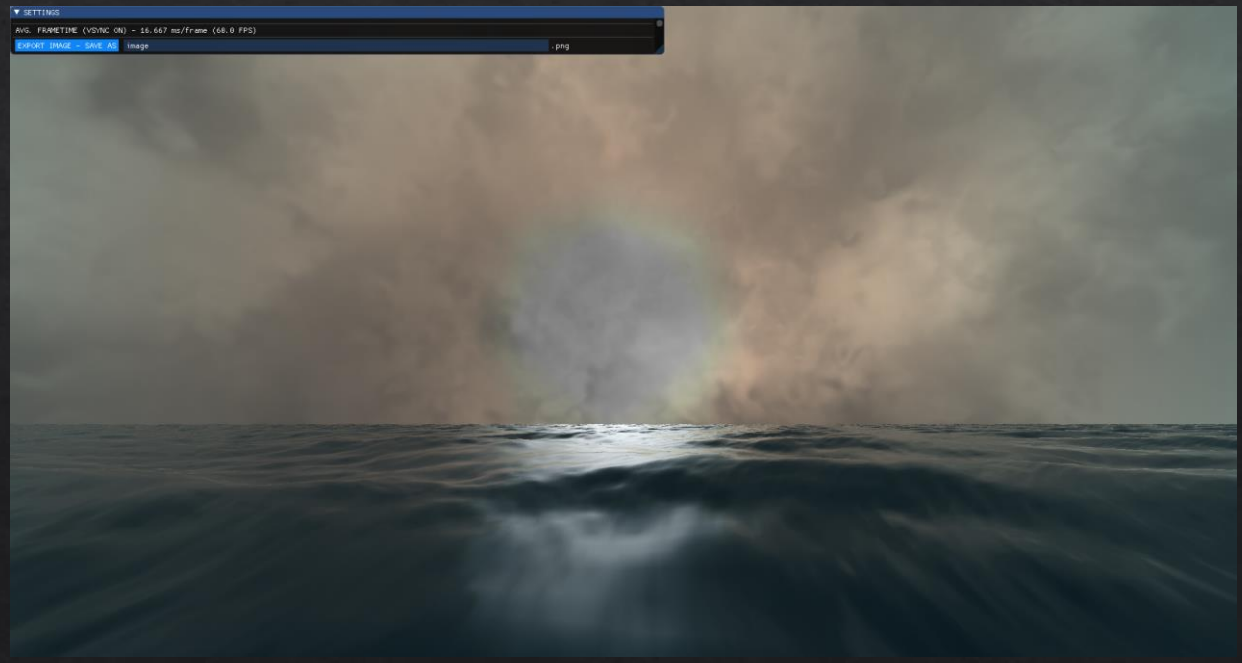




Sun (Specular Highlight)

- Employs the same specular highlight formula as skysphere.

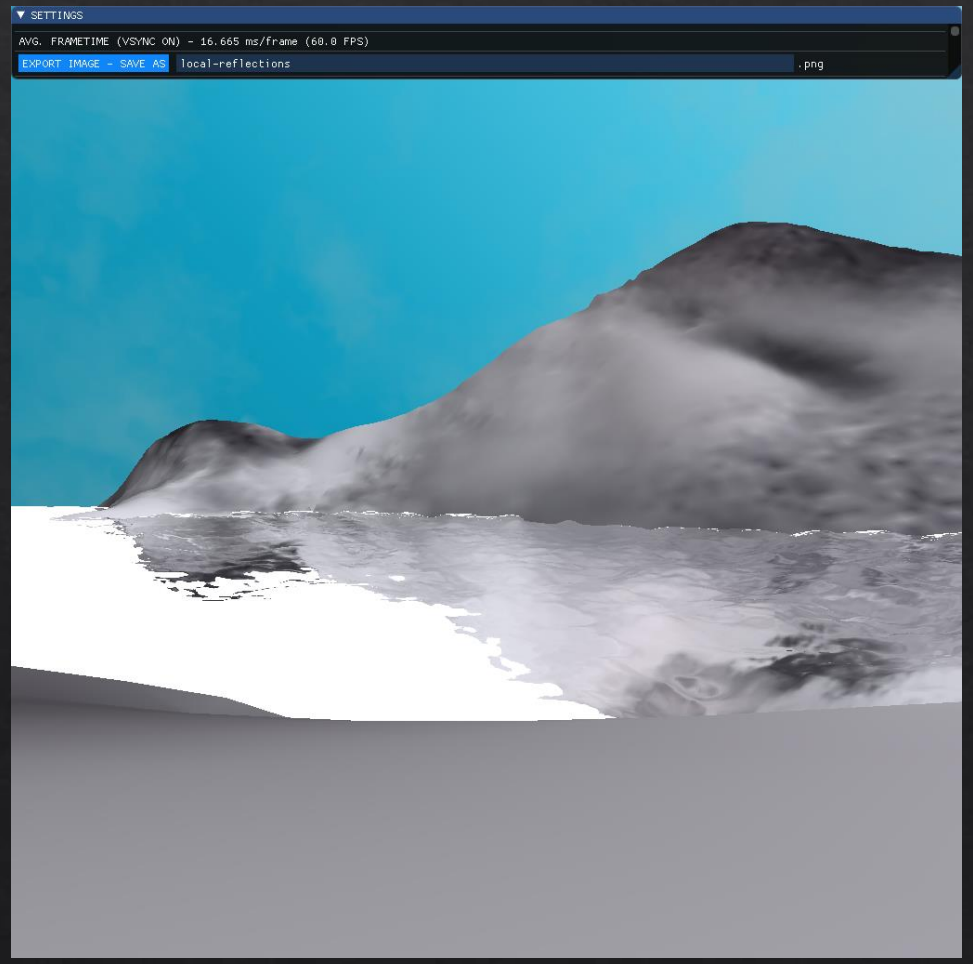


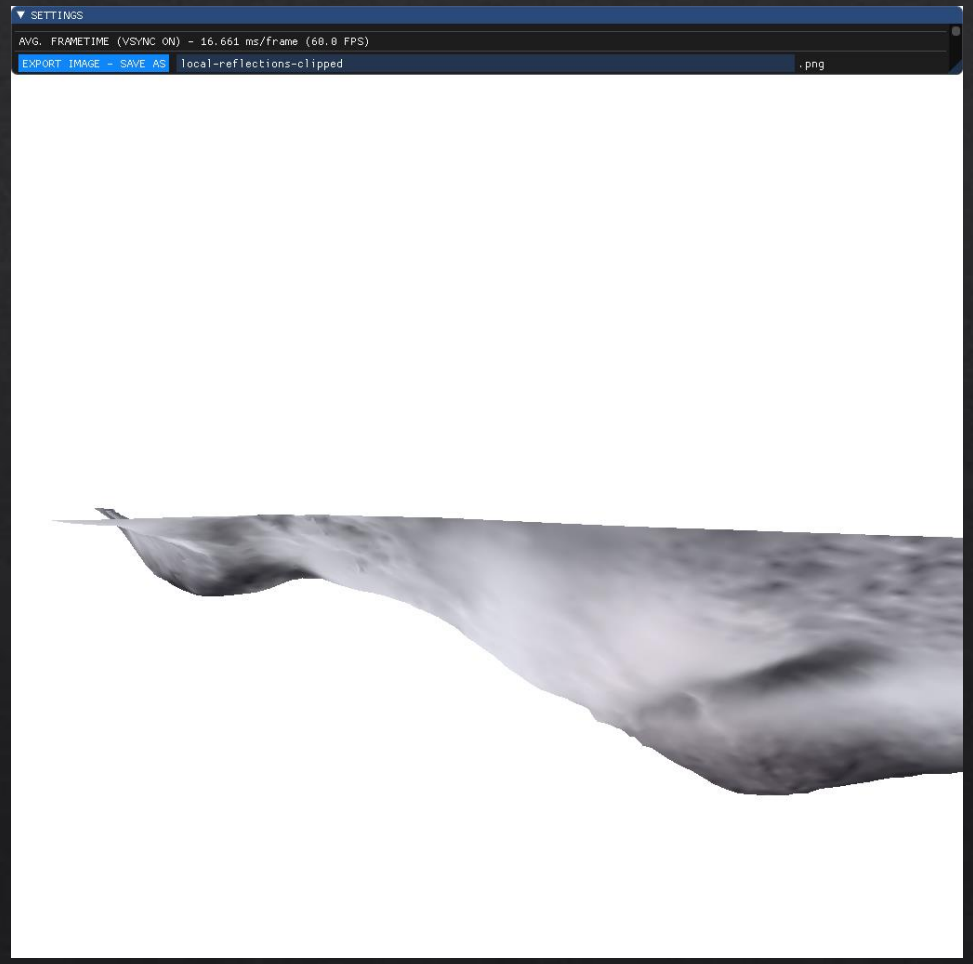


Local reflections

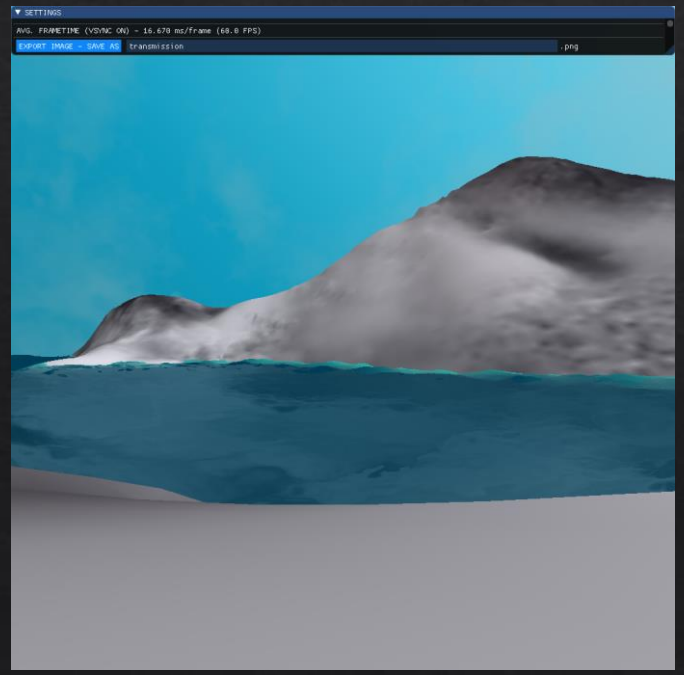
- Render scene to texture.
- Apply a mirror matrix that flips the y-component of the world-space position, since our reflection plane is the XZ-plane.
- Clip plane at XZ-plane (leaves geometry below plane).
- Enable back-face culling with reversed winding (clockwise) – avoid lower geometry now occluding your view.
- Problems? Approximating the surface with a plane means that larger error when amplitude increases. Noticeable at intersections, but will be hidden by other effects (e.g. soft edges).

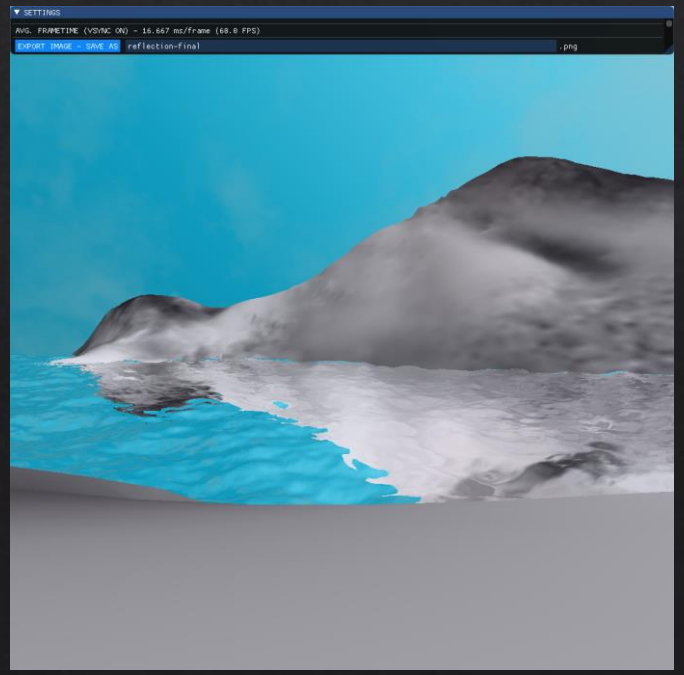
Local Reflections (cont.)

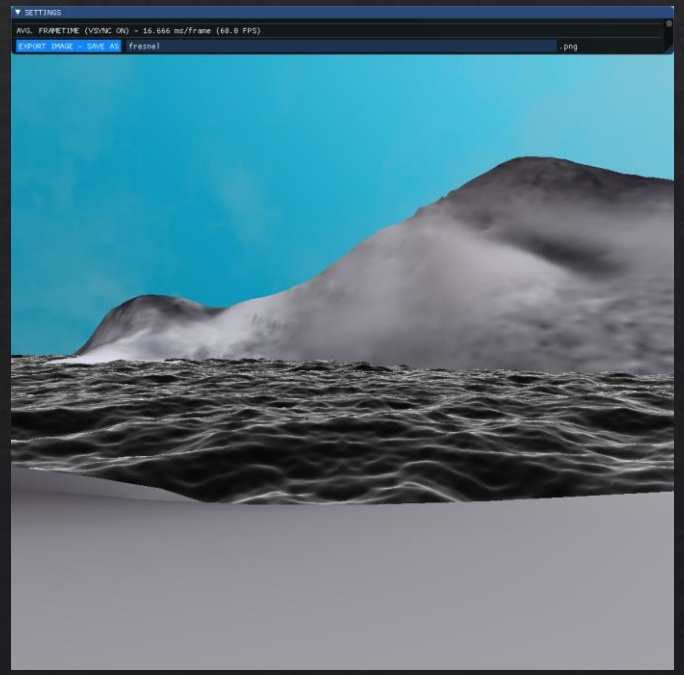




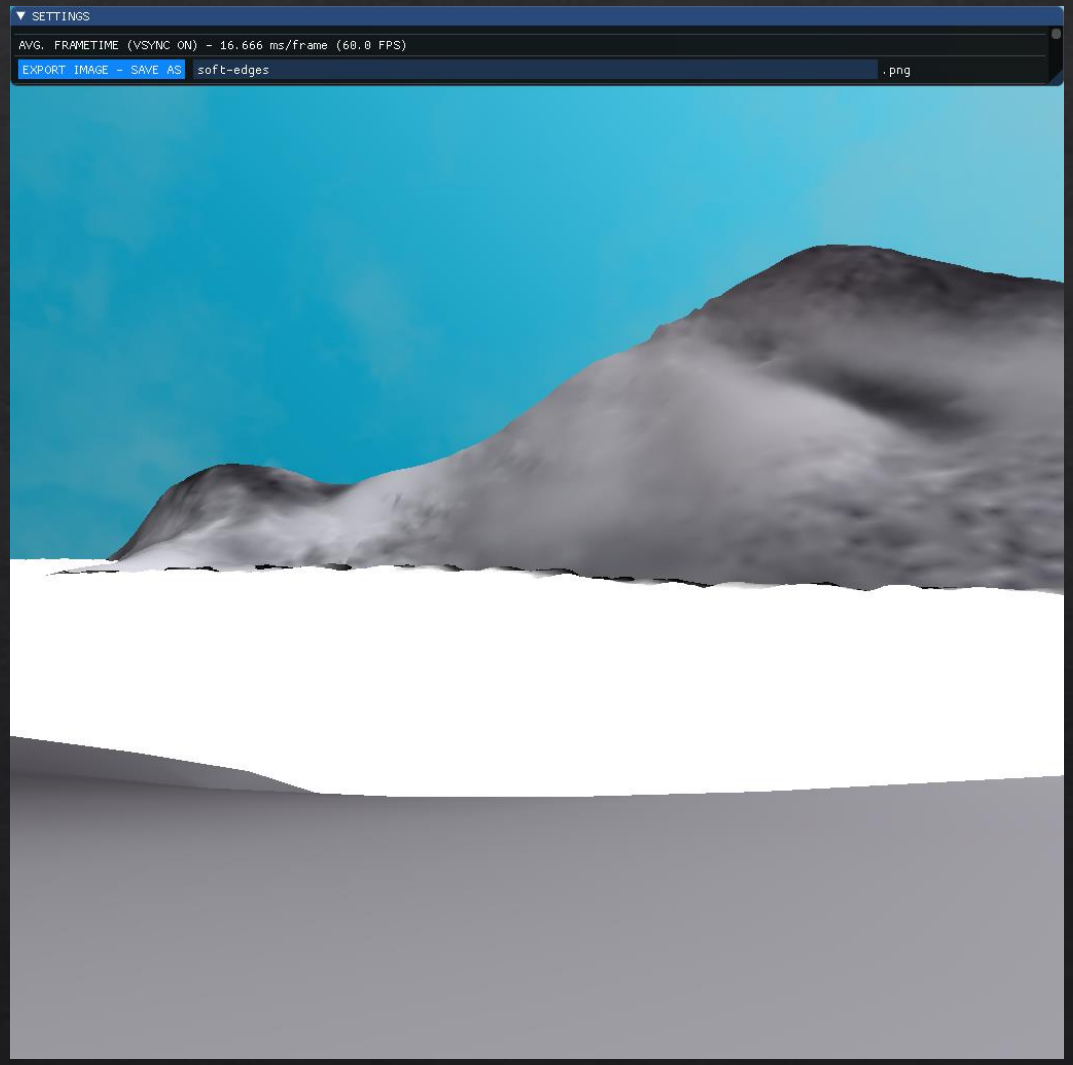
Fresnel Reflectance (final composition)







Soft-Edges


SETTINGS
AVG. FRAMETIME (VSYNC ON) - 16.666 ms/frame (60.0 FPS)
EXPORT IMAGE - SAVE AS
soft-edges
.png

Future Work

- Underwater effects (light attenuation, shallow caustics)
- Boat wakes
- Shoreline “wetting”
- Cheap “particle splashing” on contact with steep shores / rocks
- More realistic skybox (animated clouds, flatter look)
- Apply dithering technique to reduce visible colour banding everywhere in gradients
- Deferred-rending / G-buffers
- Better animation
- Buoyancy / rocking in water