中国游戏开发者大会

CHINA GAME DEVELOPERS CONFERENCE

Ocean simulation and rendering in War Thunder

Tim Tcheblokov, NVIDIA

War Thunder

Ground and Air forces combat simulator

Tank mode

- Shooter-like level of details
- Features/detail size of 2-3cm

War Thunder

Ground and Air forces combat simulator

Destroyed by tank,
revenge with bomber!

- Air combat simulator level of details
- Realistic LOS, up to 160 km
- Features/detail size of several kilometers

War Thunder

Free To Play MMO

- Unlike premium games, people pay only if they like the game
- You should be running fast and look great on any client

War Thunder

Scalability is the key

- DX11, DX9, GL, GLES, PS4
- Windows, Linux, Mobile, Mac, Consoles
- Toasters to Titans

War Thunder

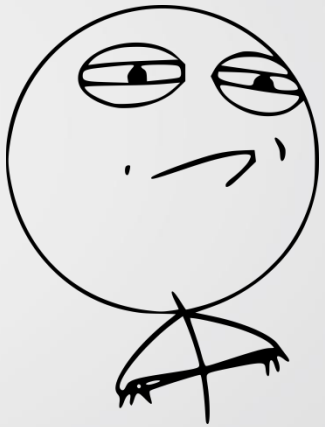
Permanent engine improvements

- New effects
- Performance tweaks
- Water tech lacked progress
- We need new water tech!

War Thunder

New water tech challenges

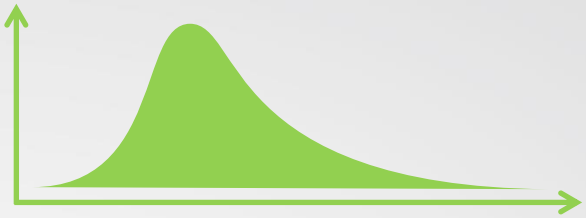
- Realistic look
- Explosions, wakes, shores, rivers and lakes
- High simulation and rendering performance
- Full-fledged citizen of in-game physics
- Scalable for each client's hardware capabilities



CHALLENGE ACCEPTED

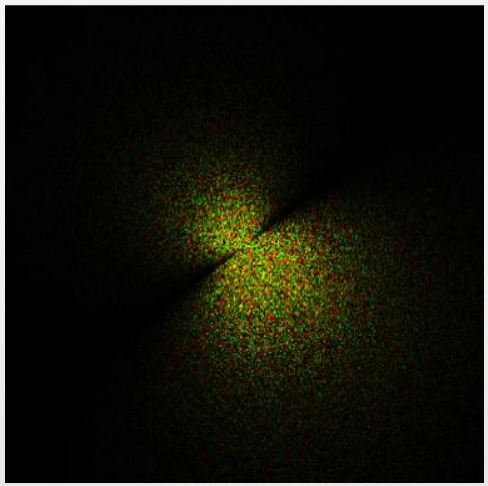
Ocean simulation

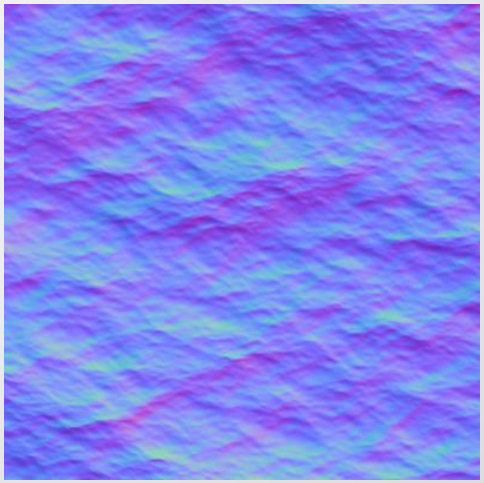
Classic approach



iFFT

- Jerry Tessendorf's paper: Simulating Ocean Water
- Based on empirical data: Phillips spectrum
- Simulation in frequency domain, inverse FFT to spatial domain
- Texture as result
  R, G, B = Dx, Dy, Dz

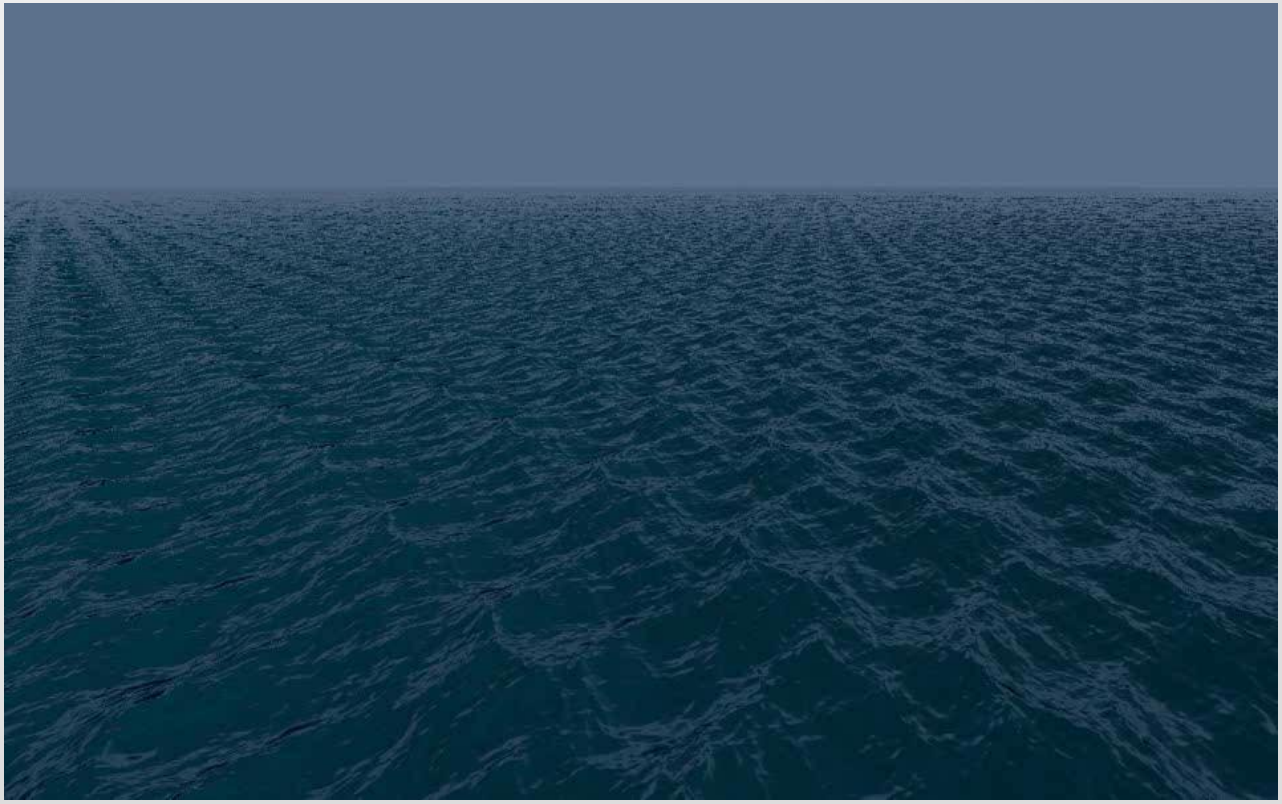




Ocean simulation

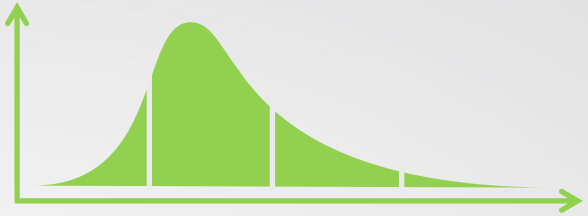
Classic approach

- Looks natural in close-up
- Suffers repeats at distance
- Can increase FFT size, but
- Large FFT sizes are expensive



Ocean simulation

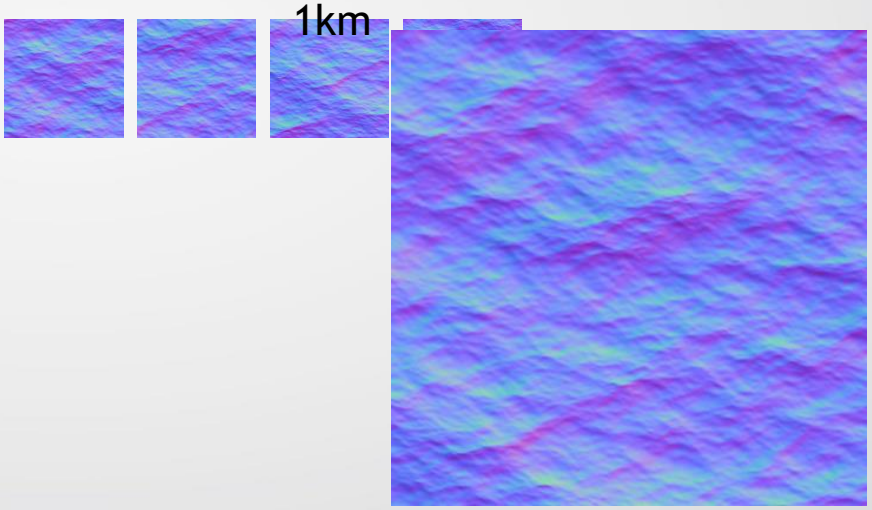
Cascades based approach



iFFT

1km

5m

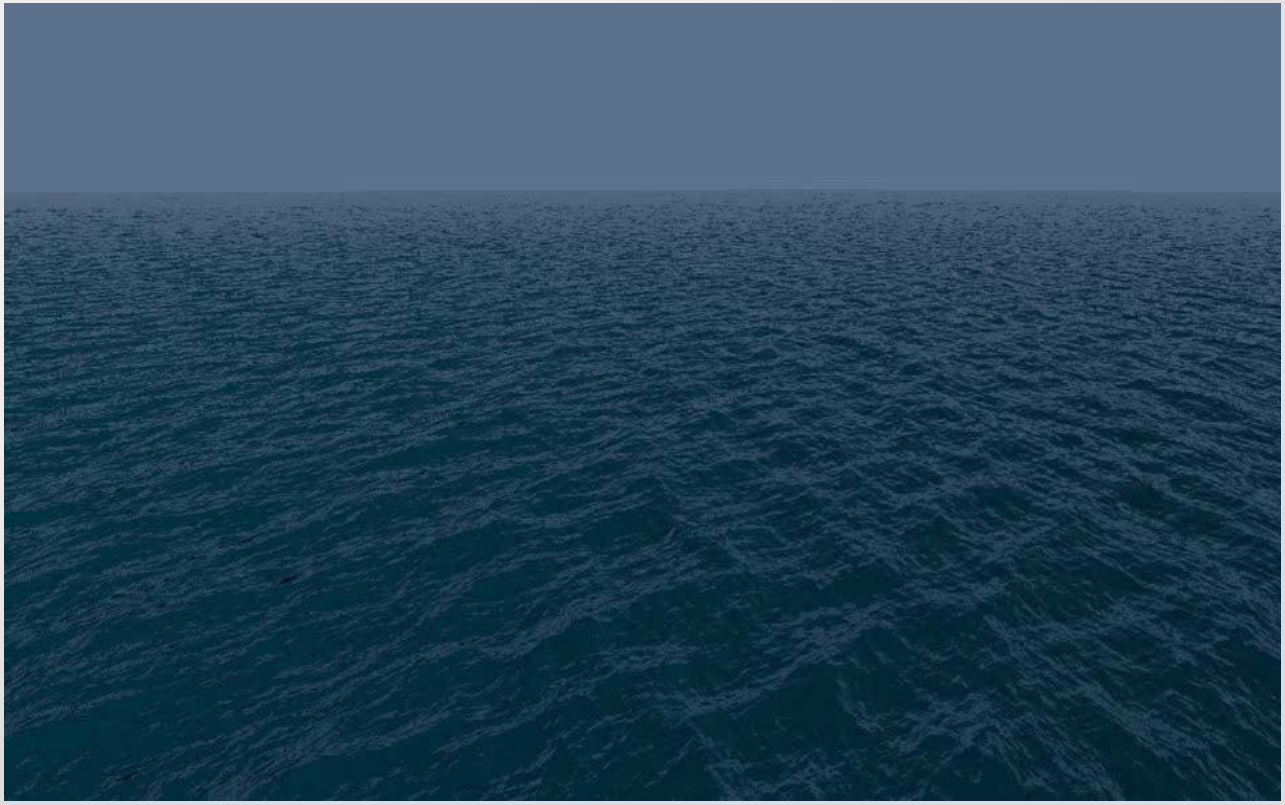


- Split spectrum to 4 cascades
- 4x Simulation in frequency domain, inverse FFT to spatial domain
- Set of textures as results
- Combine cascades to get results

Ocean simulation

Cascades based approach

- High dynamic range of wavelengths
- More details in close-up
- Less repeats at distance



Ocean simulation & rendering

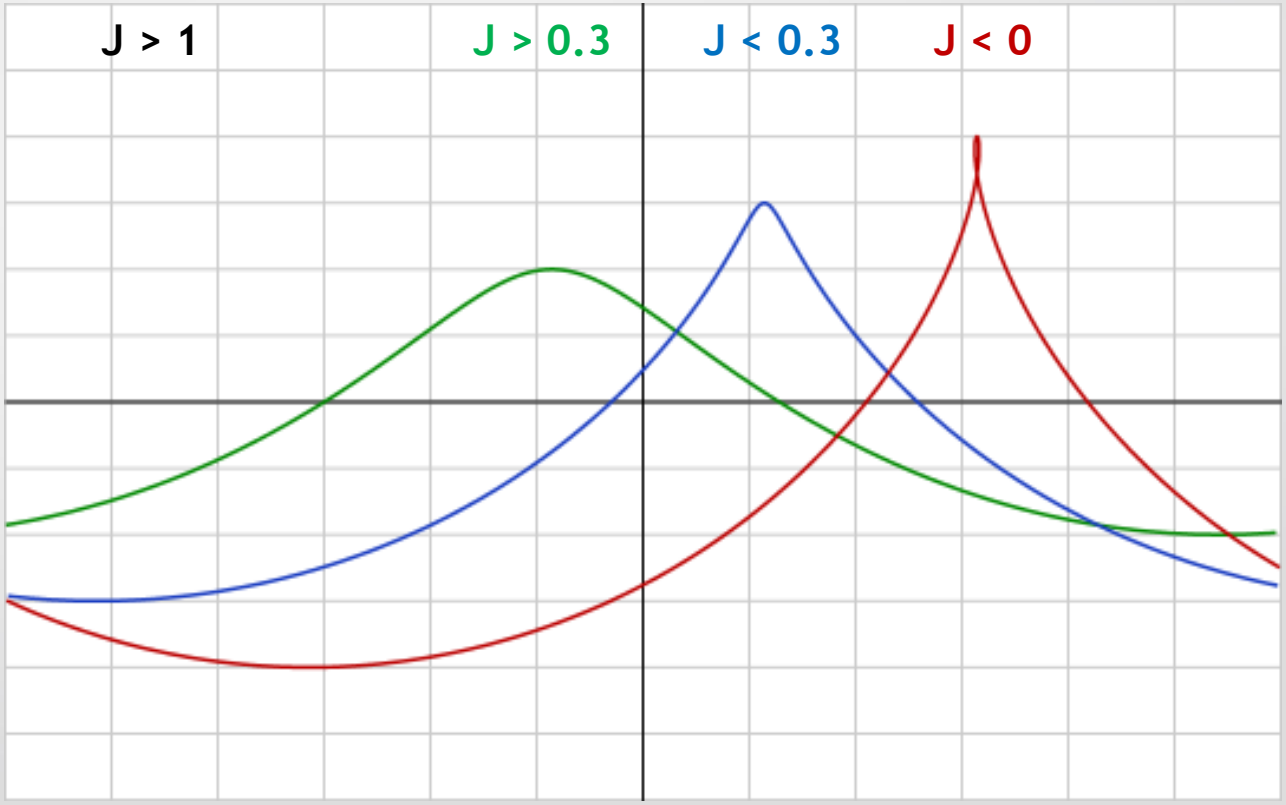
Cascades based approach

- Fade out & exclude cascades at distance
- Less noise
- Less work on GPU

Ocean simulation & rendering

Basic foam simulation

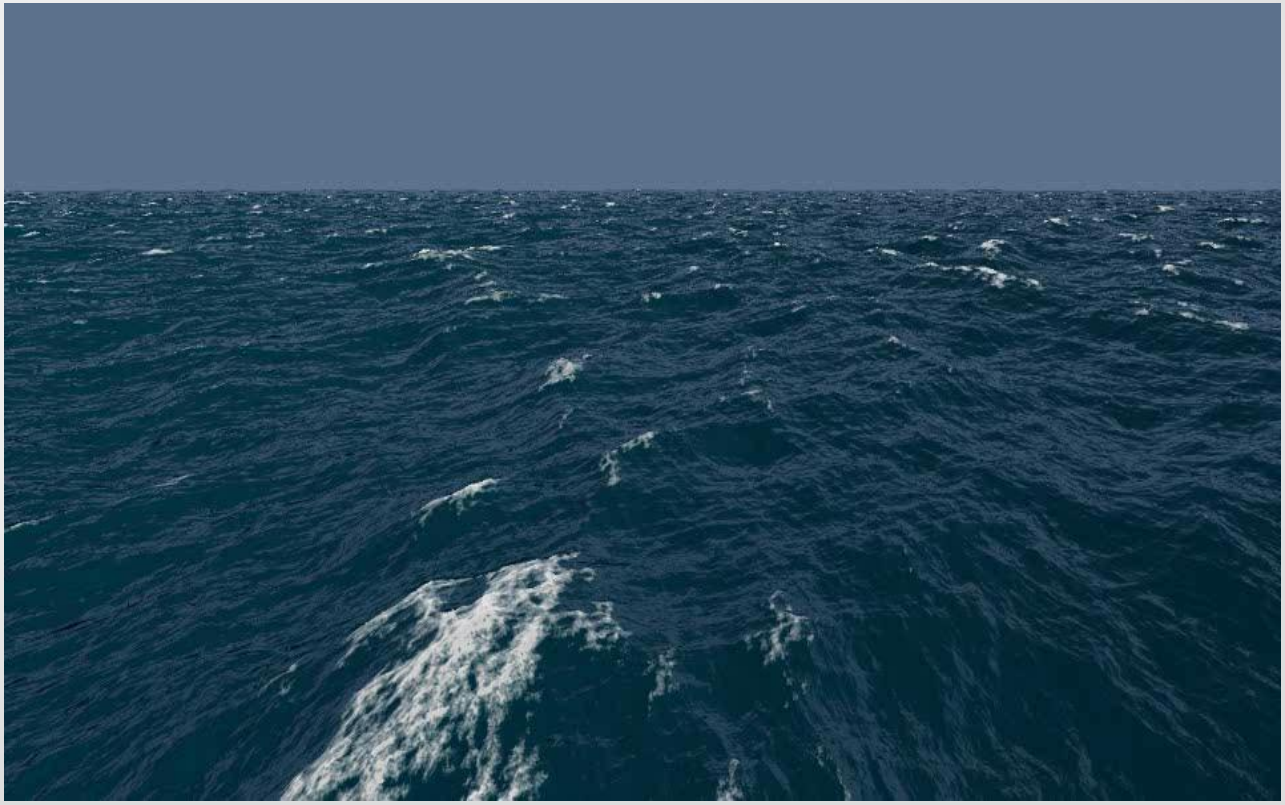
- Jerry Tessendorf proposal:
  use Jacobian of flat->displaced transform
- J > 1: stretched
  J < 1: squeezed
  J < 0: overlap
  Wave breaks!
- We use J < M
  M ~ 0.3 .. 0.5



Ocean simulation

Basic foam simulation

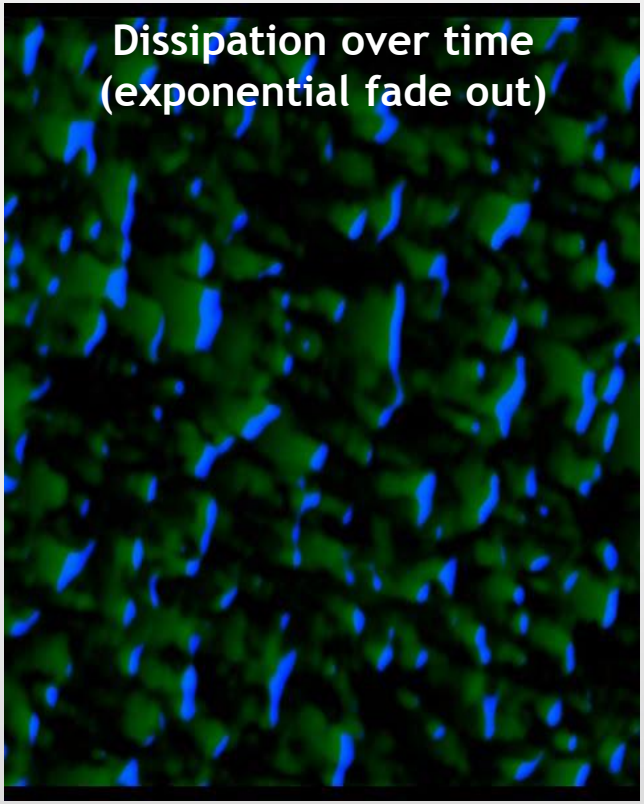
- Modulate foam textures by saturate(k*(-J+M))
- Breaking areas look better
- Foam disappears!

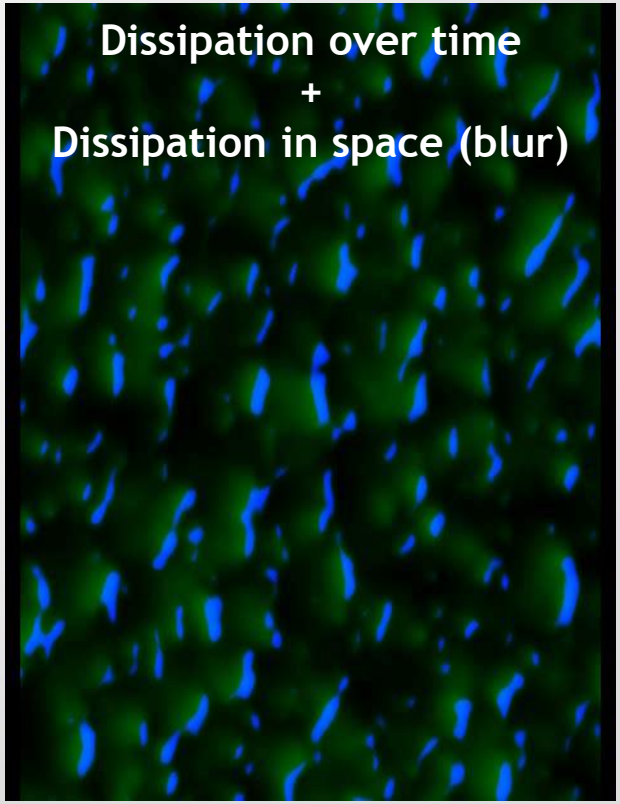


Ocean simulation & rendering

Advanced foam simulation

- Breaking waves inject turbulent energy to cascades
- Energy dissipates over time and in space
- Simulate for each cascade using PS
- Combine results





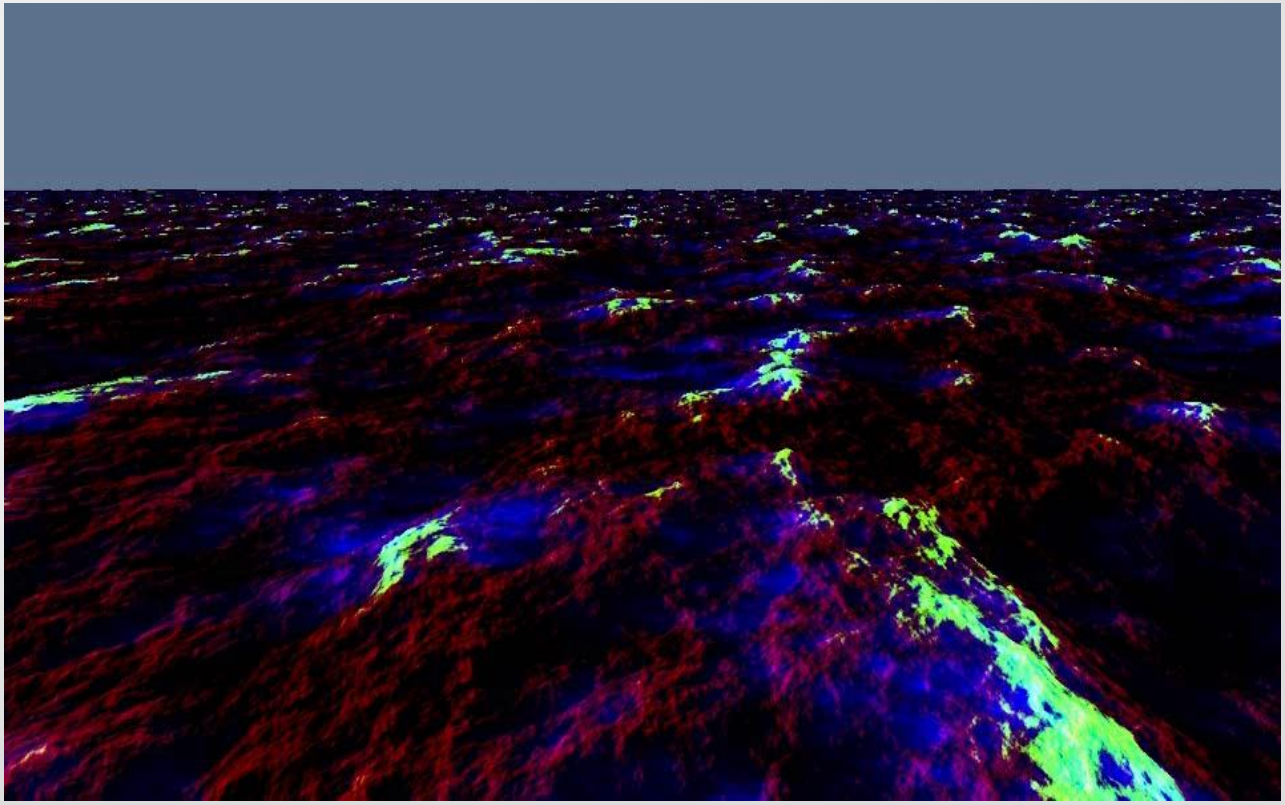
Ocean simulation & rendering

Advanced foam simulation

- Foam simulation results, color coded:

  Breaking areas
  Turbulent energy
  Surface stretching

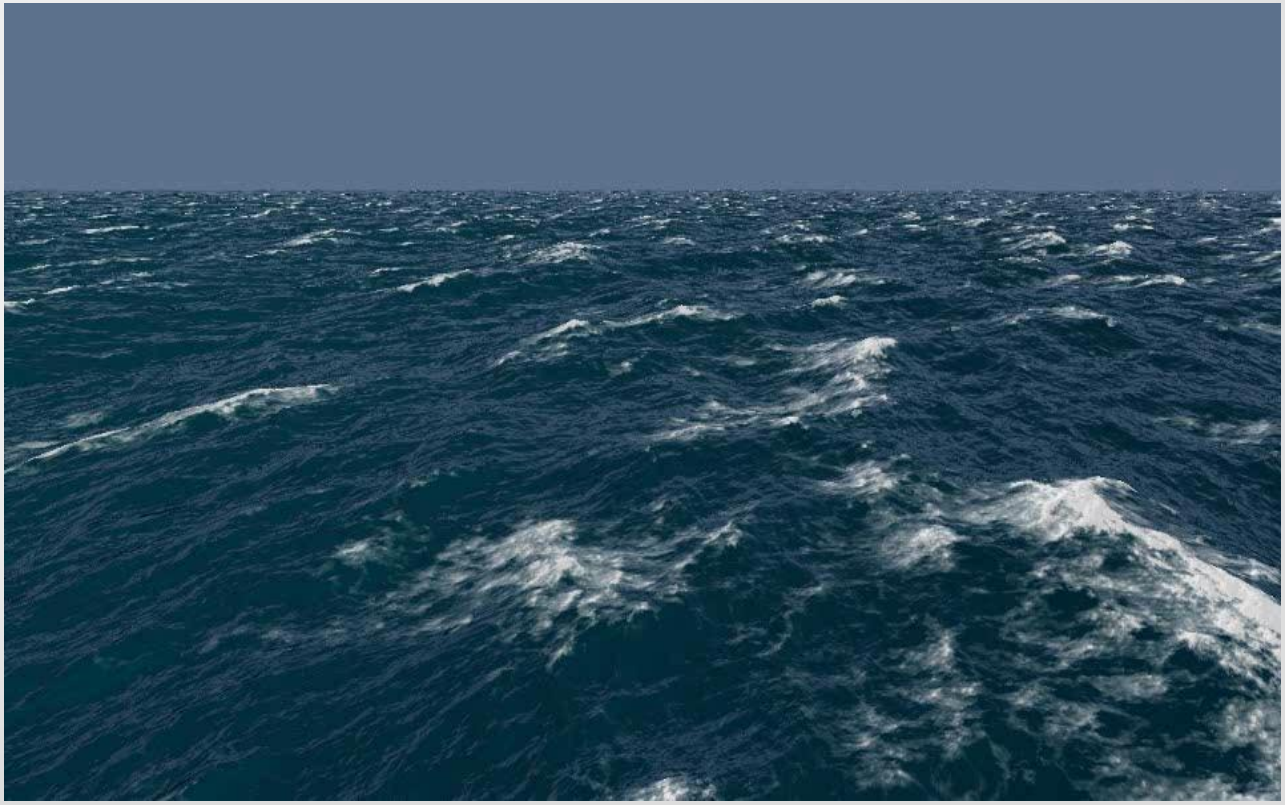
- Stretching is important for foam layer:
  stretched - thinner
  squeezed - thicker



Ocean simulation & rendering

Advanced foam simulation & rendering

- Turbulent energy = foam intensity
- Modulate foam textures by energy and stretching
- Add dense foam on breaking areas



Ocean simulation & rendering

Advanced foam simulation & rendering

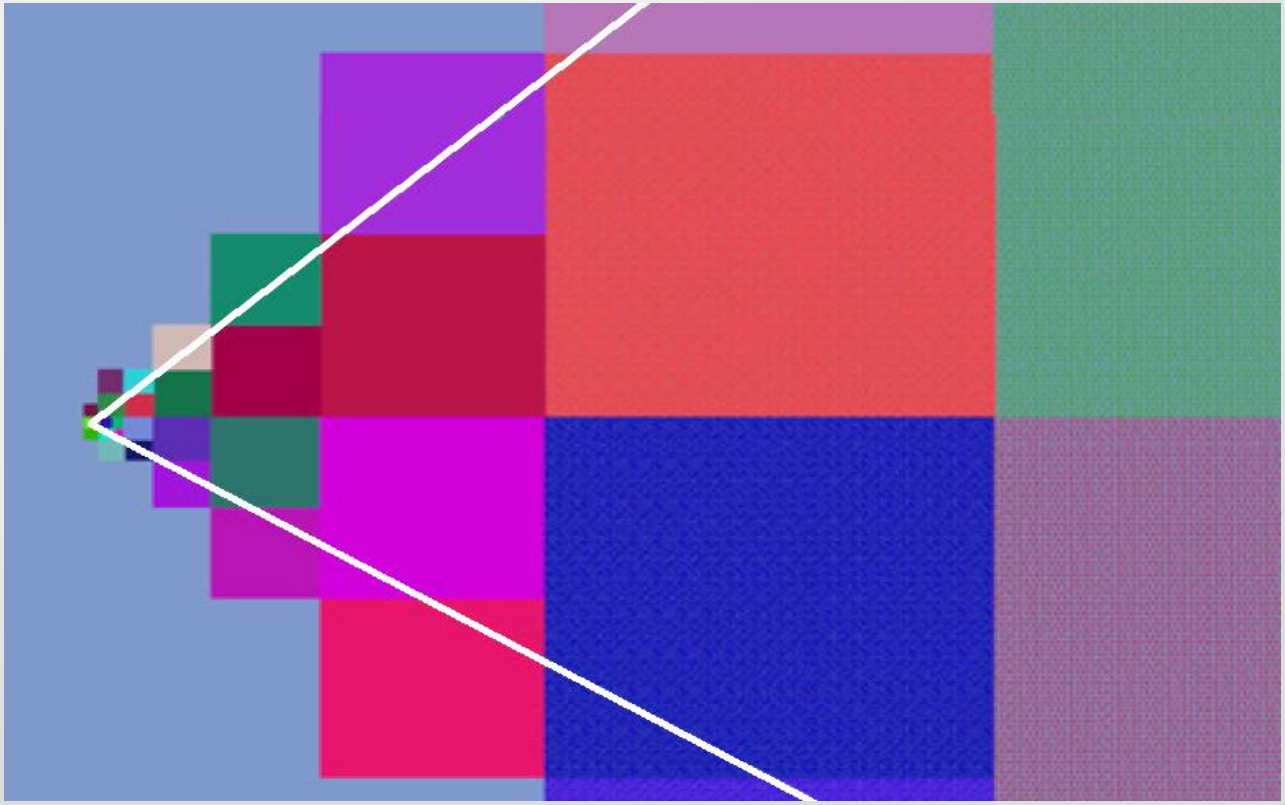
- Turbulent energy = foam intensity
- ..and mixed-in bubbles!
- Add “milkiness” modulated by energy to refraction color

Ocean simulation & rendering

Ocean surface geometry

Quadtree defined by view frustum

- Fast to build in runtime
- Each node is regular vertex grid + edge stripes
- Precomputed IB/VB, indexed triangle lists



Ocean simulation & rendering

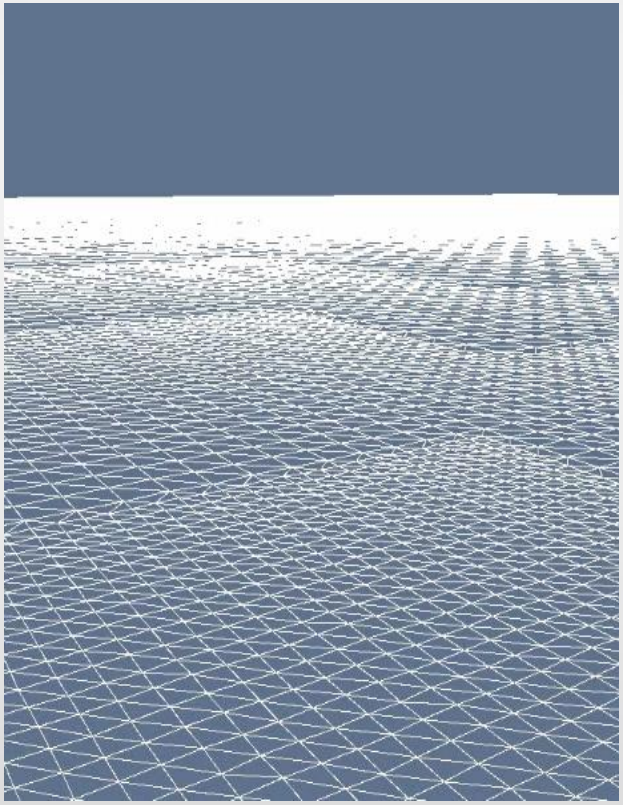
Ocean surface geometry

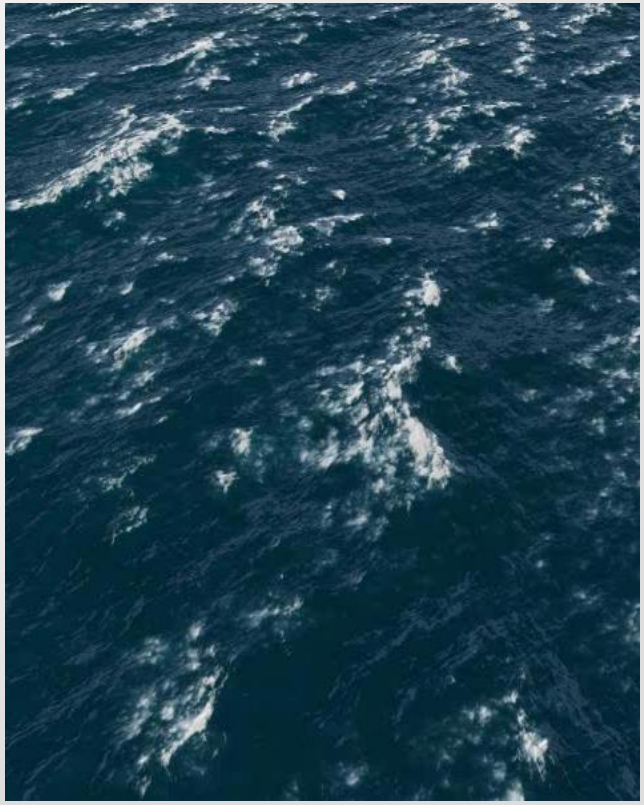
Quadtree defined by view frustum

- Good LOD balance:

  dense in closeup
  coarse at distance

- Suffers popping!



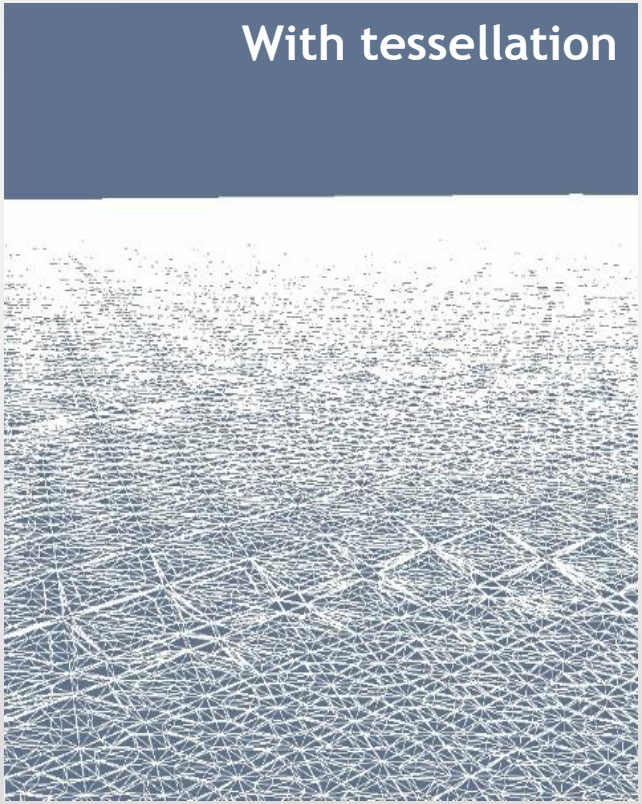


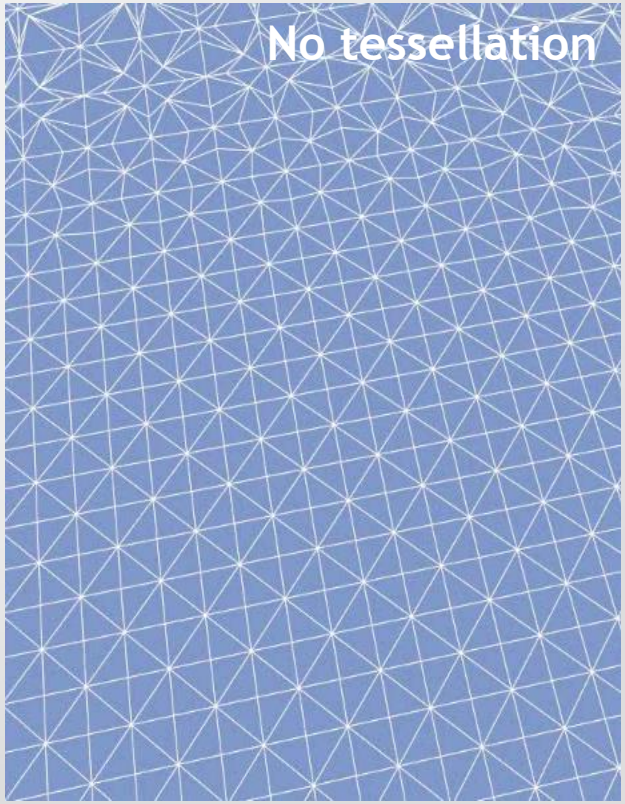
Ocean simulation & rendering

Ocean surface geometry

Geomorphing

- Smooth transitions between geometry LODs
- Done in VS
- Friendly to tessellation too!





Ocean simulation & rendering

Ocean surface geometry

Geomorphing result:

- Adaptive LOD with smooth transitions
- Mesh density adjustable for any client

Ocean simulation & rendering

Shore interaction

Nature:

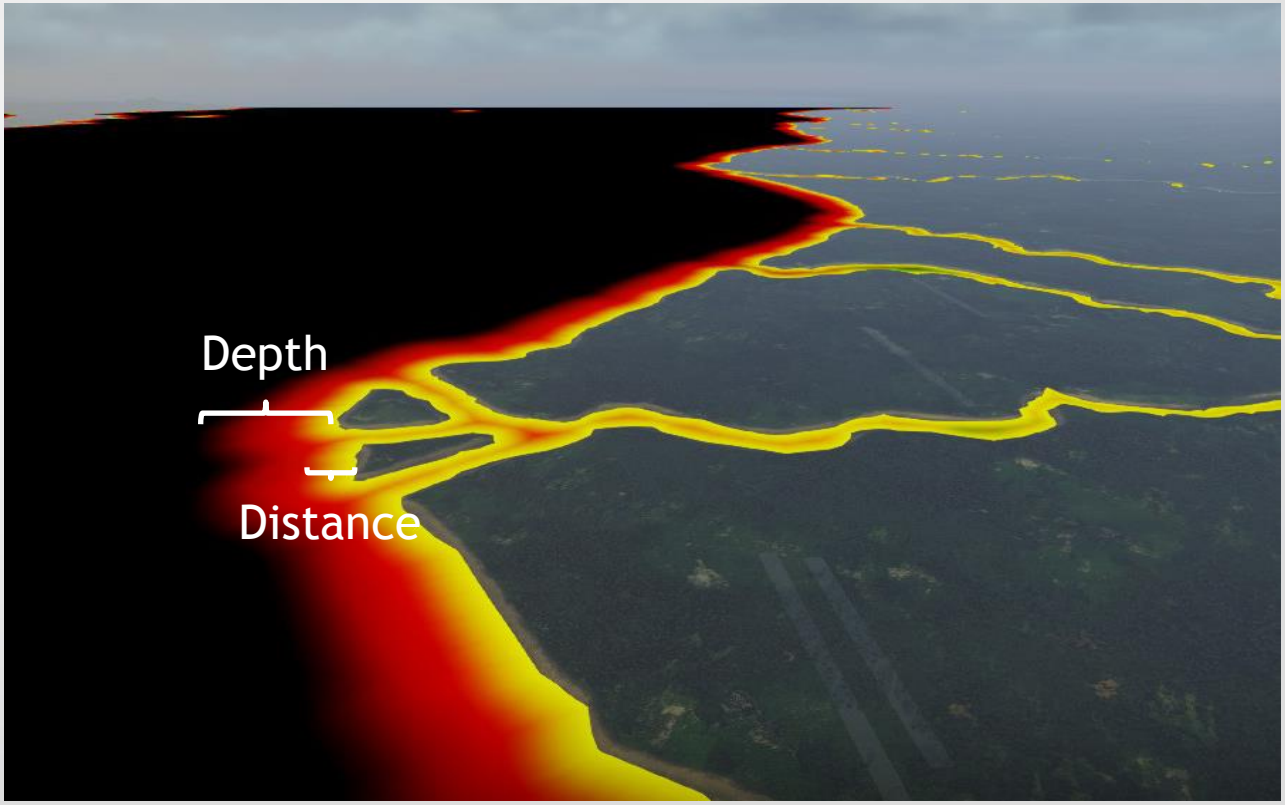
- Shore waves appear in shallow areas
- Mostly parallel to the shore
- Can use Distance field!

Ocean simulation & rendering

Shore interaction

Distance Field

- Distance to shore as phase for waves
- DF done on GPU on loading the location 4k*4k for 65km*65km world
- RGBA8 texture:
  R: Depth
  G: Distance to shore
  B,A: DF gradient



Ocean simulation & rendering

Shore interaction

Rivers and lakes:

- Terrain is obstacle for wind & ocean waves
- No shore waves
- No ocean waves

Ocean simulation & rendering

Shore interaction

Islands:

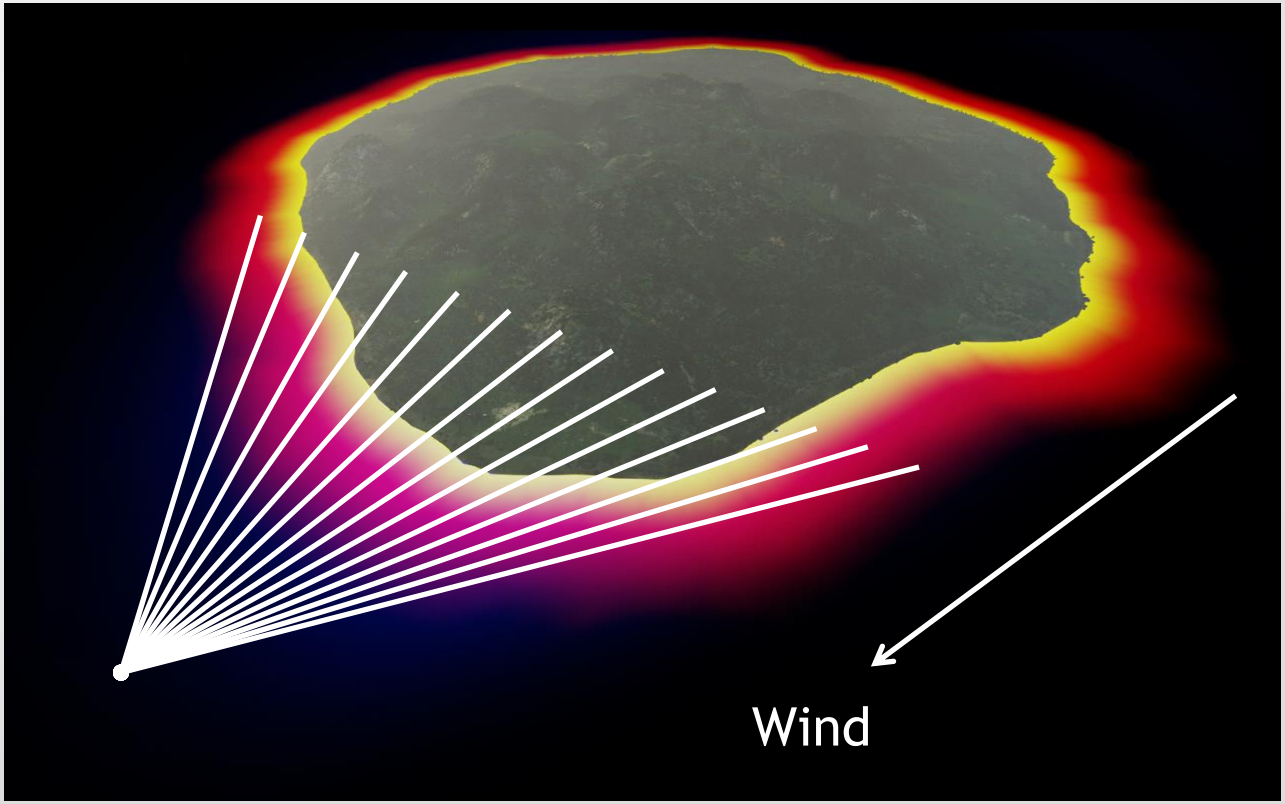
- Terrain is obstacle for wind & ocean waves
- Smaller shore waves
- Smaller ocean waves

Ocean simulation & rendering

Shore interaction

How much the terrain is obstacle for waves?

- Sample range of directions
- 1 (terrain)
  0 (water)
- Sum -> [0..1]
- Large texels -> blurred

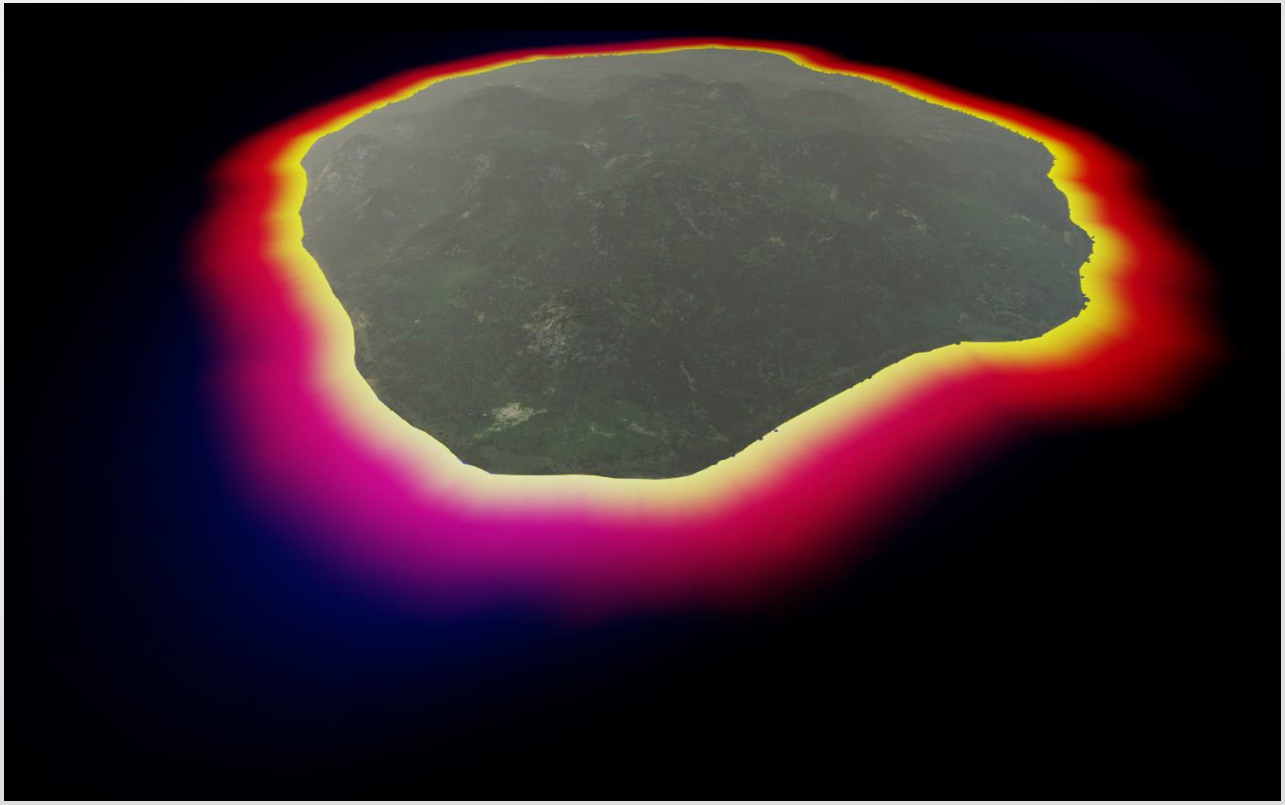


Ocean simulation & rendering

Shore interaction

Downwind texture

- A measure of “openness”
- Smaller ocean/shore waves
- No ocean/shore waves in rivers and lakes!
- Rivers and lakes get proper “openness” automatically

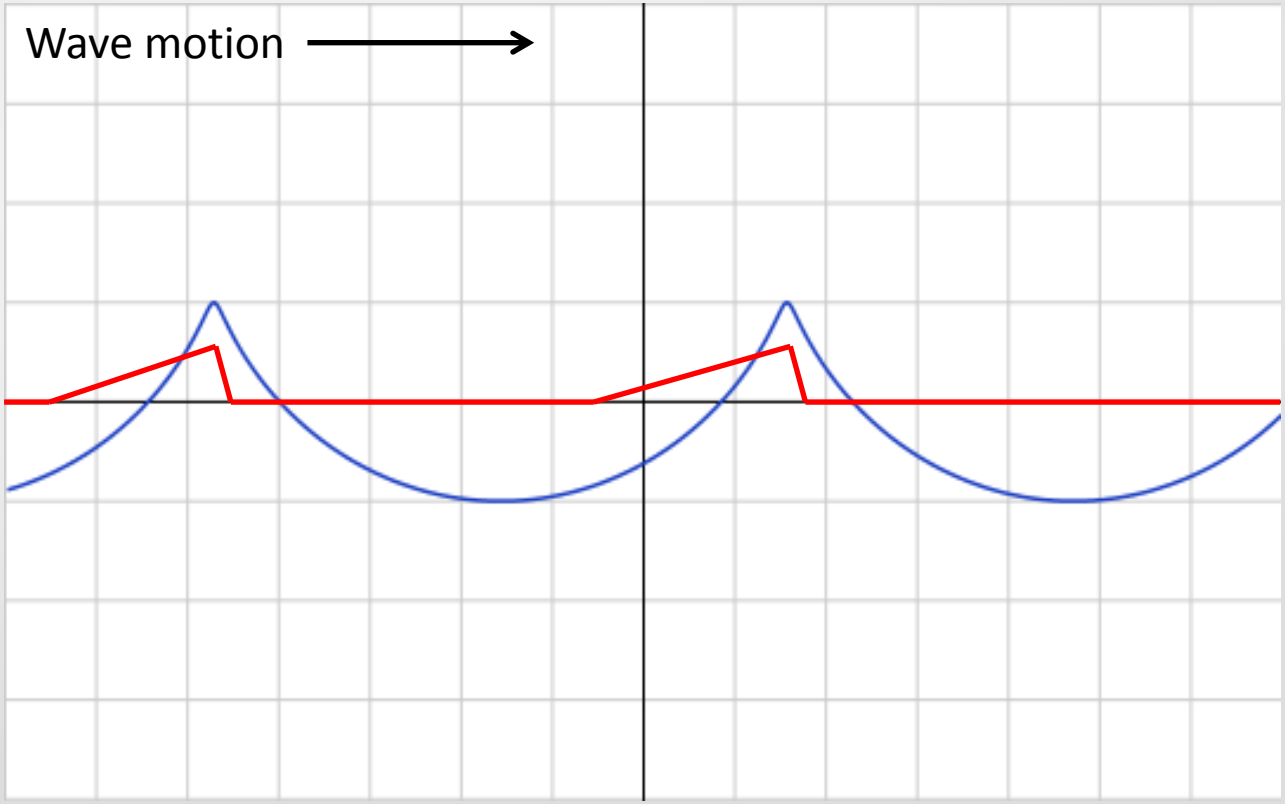


Ocean simulation & rendering

Shore interaction

Gerstner waves

- A = Significant Wave Height
- ω, speed derived from A
- Normals calculated analytically
- Sawtooth for foam

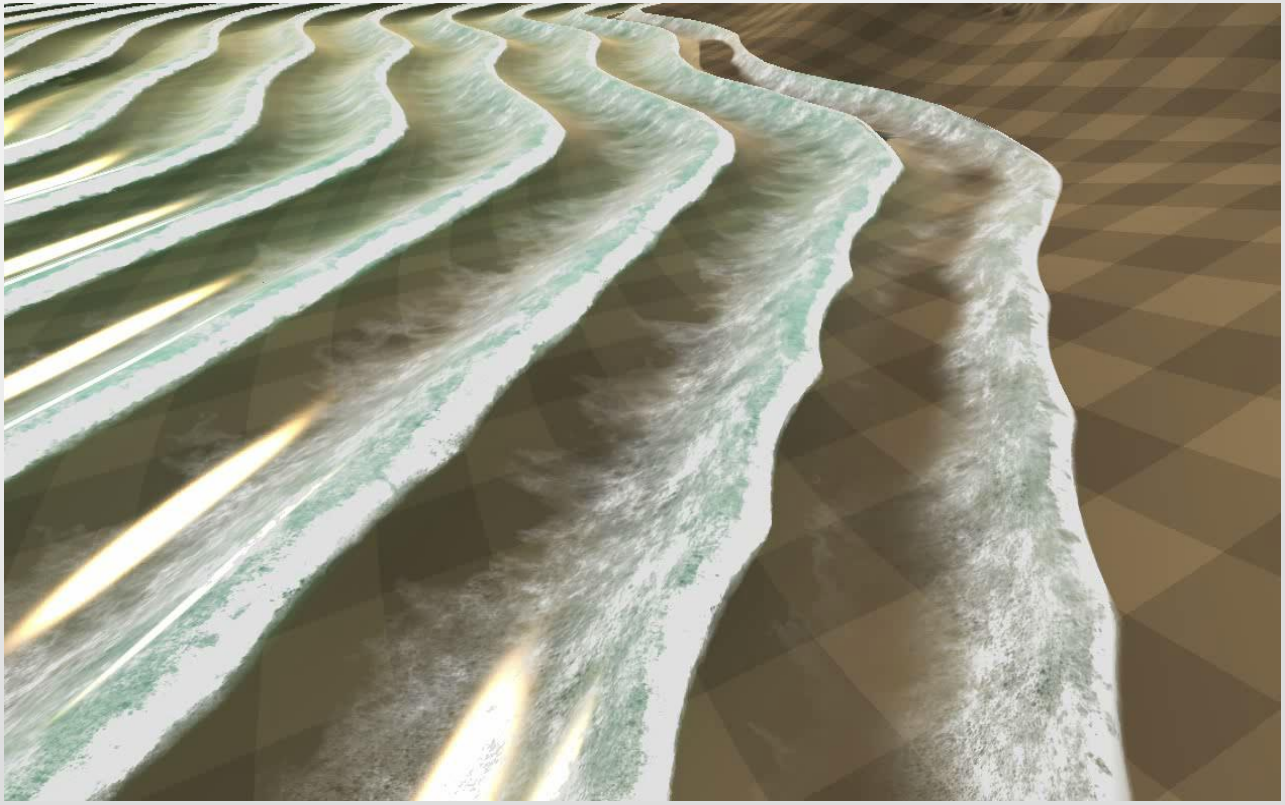


Ocean simulation & rendering

Shore interaction

Gerstner waves

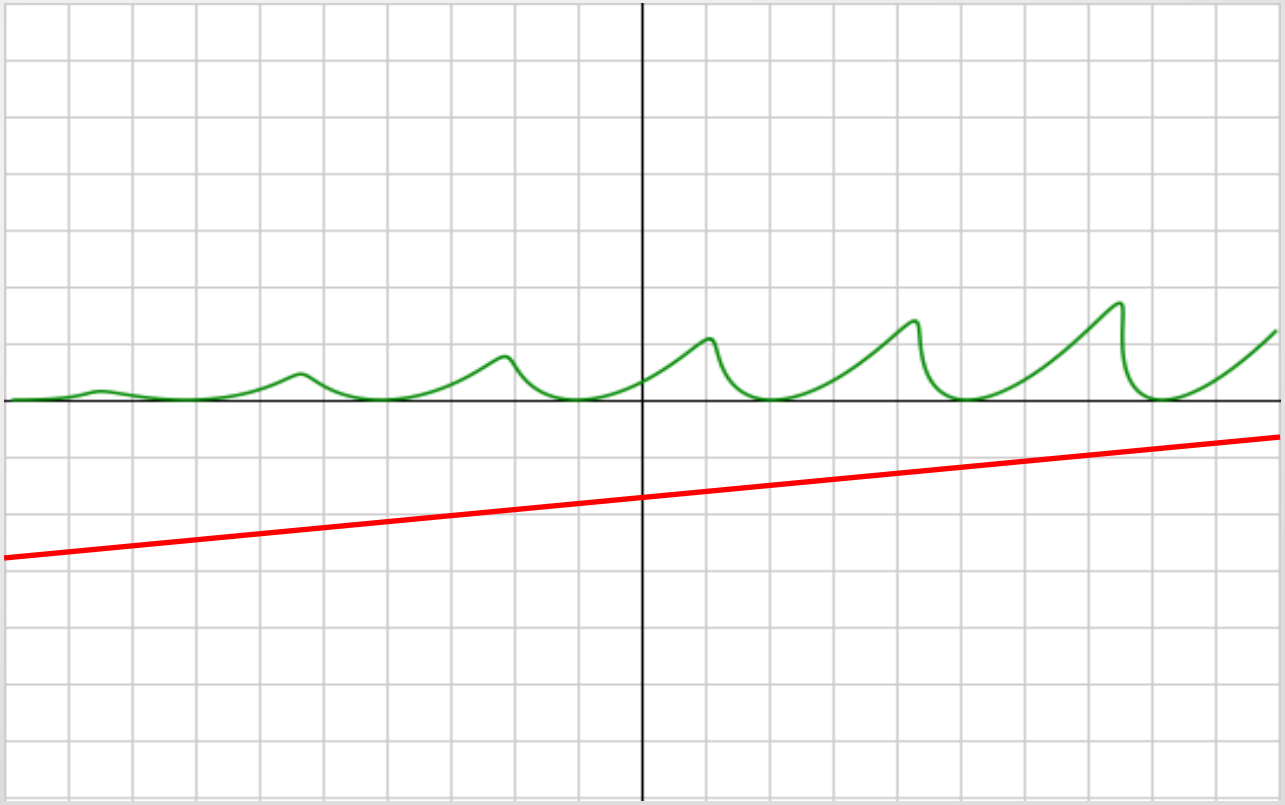
- A = Significant Wave Height
- ω, speed derived from A
- Normals calculated analytically
- Sawtooth for foam



Ocean simulation & rendering

Shore interaction

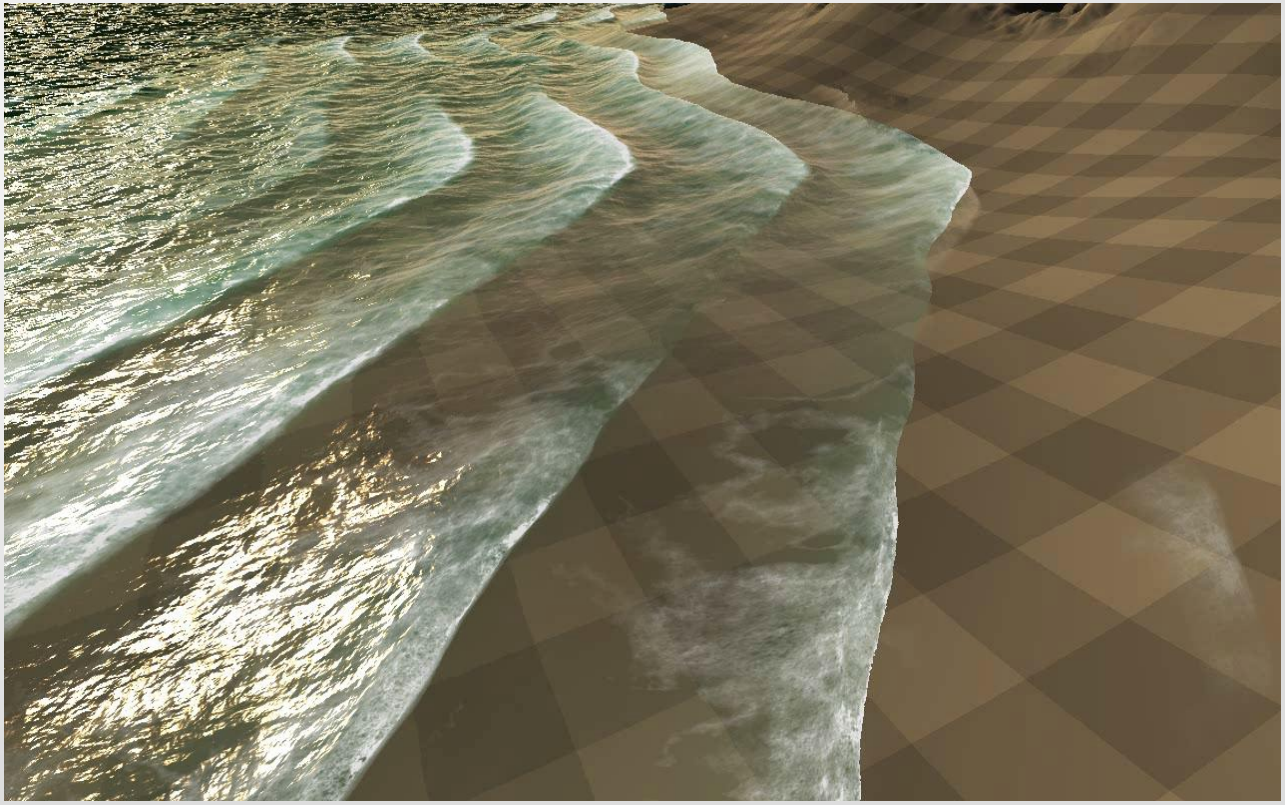
- Scale shore waves according to depth
- Move wave tops forward
- Seabed drag



Ocean simulation & rendering

Shore interaction

- Scale shore waves according to depth
- Move wave tops forward
- Seabed drag
- Ocean -> Shore lerp for displacements and normals

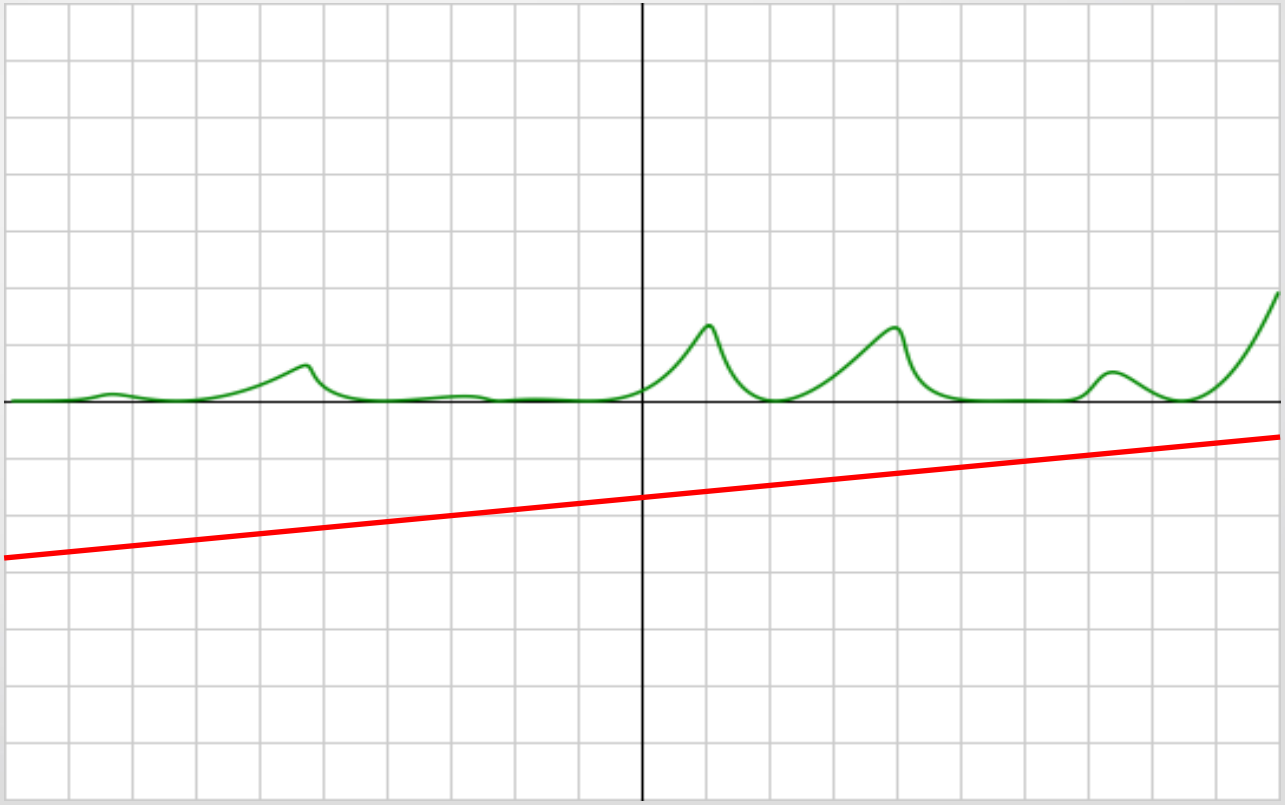


Ocean simulation & rendering

Shore interaction

Break regularity

- Add noise
- Apply group speed (phase speed / 2)

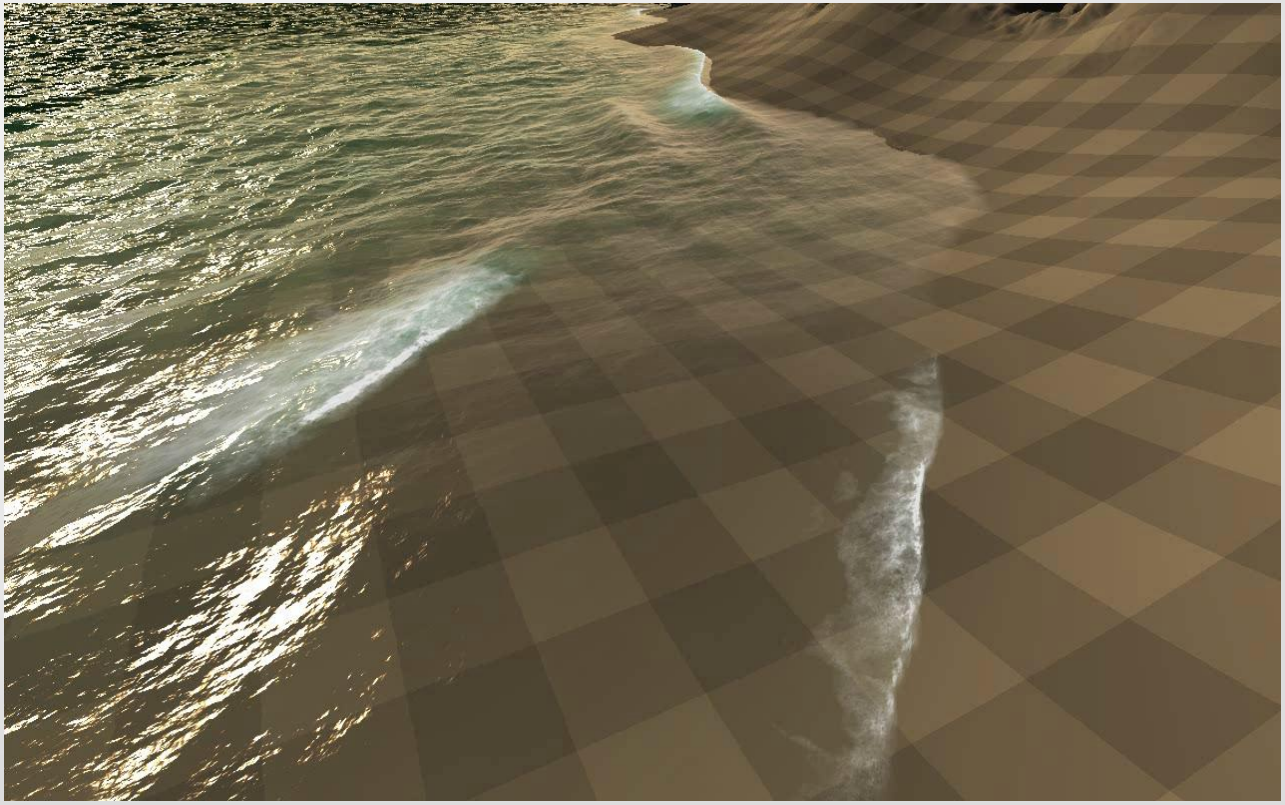


Ocean simulation & rendering

Shore interaction

Break regularity

- Add noise
- Apply group speed (phase speed / 2)

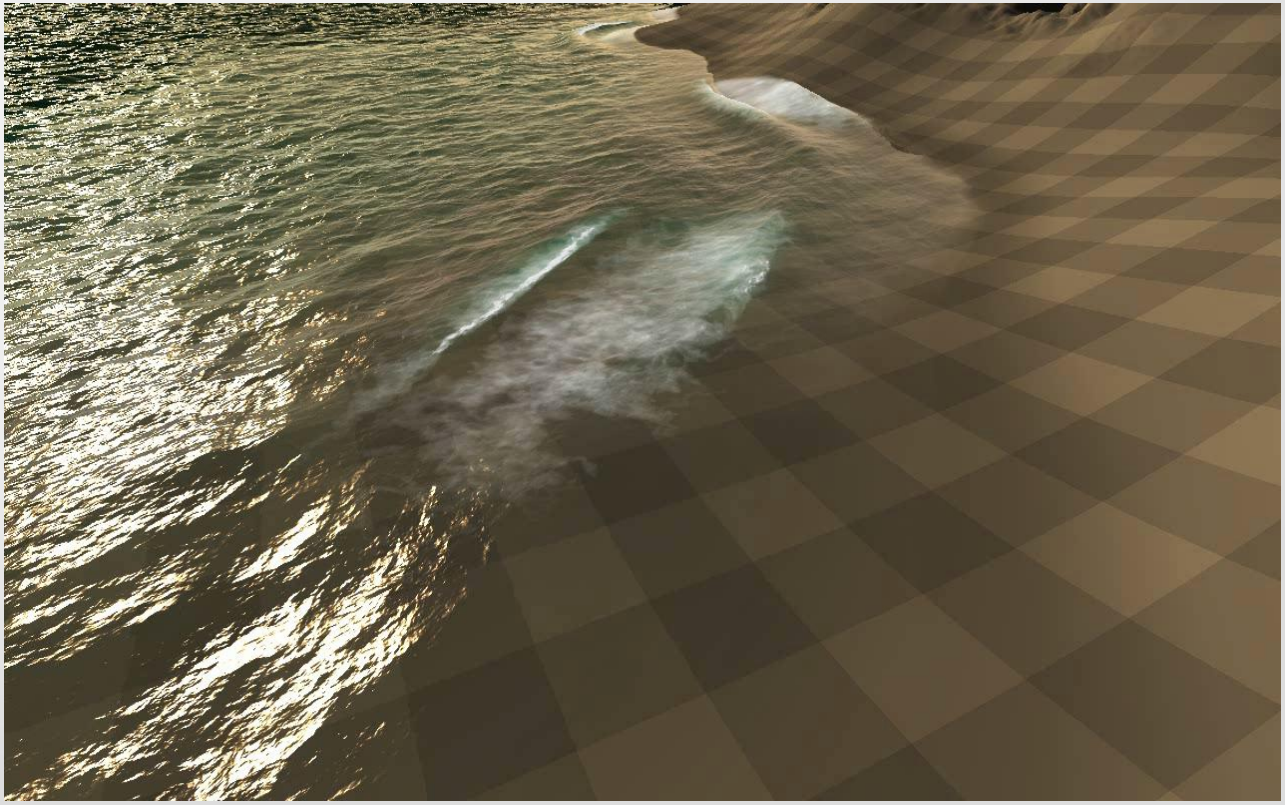


Ocean simulation & rendering

Shore interaction

Shade terrain

- Sand becomes wet and reflective
- Water and foam roll back



Ocean simulation & rendering

Shore interaction

Distant view

Ocean simulation & rendering

Rendering performance

Performance <->
IQ tweaks

	Close up	Mid range	At distance
Displacements and normals	All cascades	Some cascades	Largest cascade
Water geometry	Dense grid	Coarse grid	Flat quad
Refractions	Distorted	Simple	None
Reflections	Distorted	Blurred	Blurred
Shore interaction	Full	Full	Normals + Foam
Wakes & splashes	Full	Normals only	Normals only
Light scattering	Yes	Yes	No

Ocean simulation for in-game physics

Physical simulation challenges

- As fast as possible, High-End PCs down to toasters:
  - WaveWorks: CUDA, DC and CPU simulation
  - Gaijin: added simulation support for some other platforms
- Same physics for every player
  - Physics displacements vs Graphics displacements
- Interaction with the world
  - Displacement readbacks for vessels and hydroplanes

Ocean simulation for in-game physics

Simulation for in-game physics

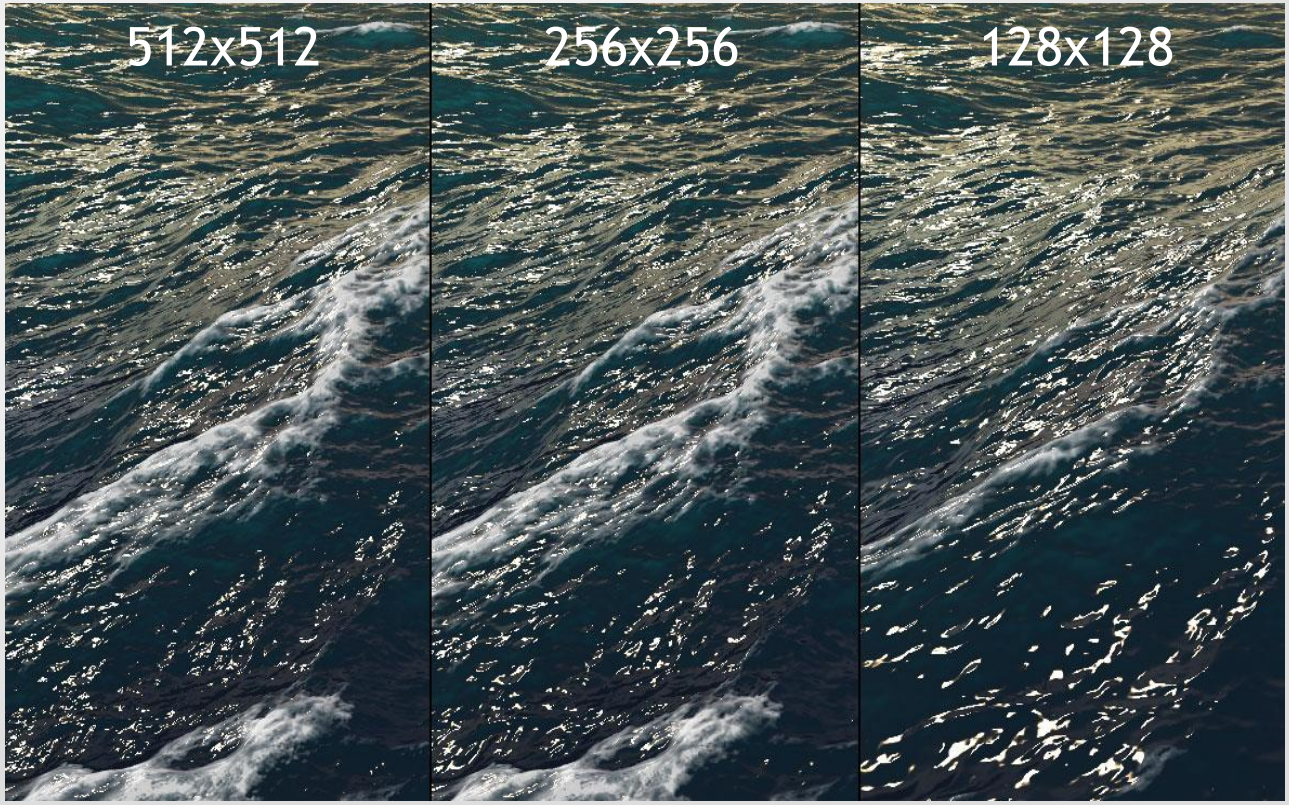
- Simulation for physics runs on CPU
  - Ensure same physics for everybody
  - Server + each client
  - Fixed timestep, 48 ticks/second
- Simulation for graphics runs on CPU or GPU
  - Prefer GPU if possible
  - Client only
  - Variable timestep, each frame

Ocean simulation

Simulation for in-game physics

Large FFT is expensive for CPU!

- Spectrum downsampling
- Physics: 128x128
- Graphics: 128x128 .. 512*512
- Close enough!
  < 5 cm
  @ 3m amplitude



Ocean simulation for in-game physics

Simulation for in-game physics

- War Thunder is online game
- Client->Server lag is up to 500 msec
  - Up to 24 simulation steps to re-simulate on server in worst case
  - Normally 0 to 2 steps to re-simulate – up to 2x work
- Server->Client lag is up to 1500 msec
  - Up to 72 simulation steps to re-simulate on client in worst case
  - Normally 0 to 5 steps to re-simulate – up to 5x work
- Too Expensive to re-simulate ocean surface!

Ocean simulation for in-game physics

Simulation for in-game physics

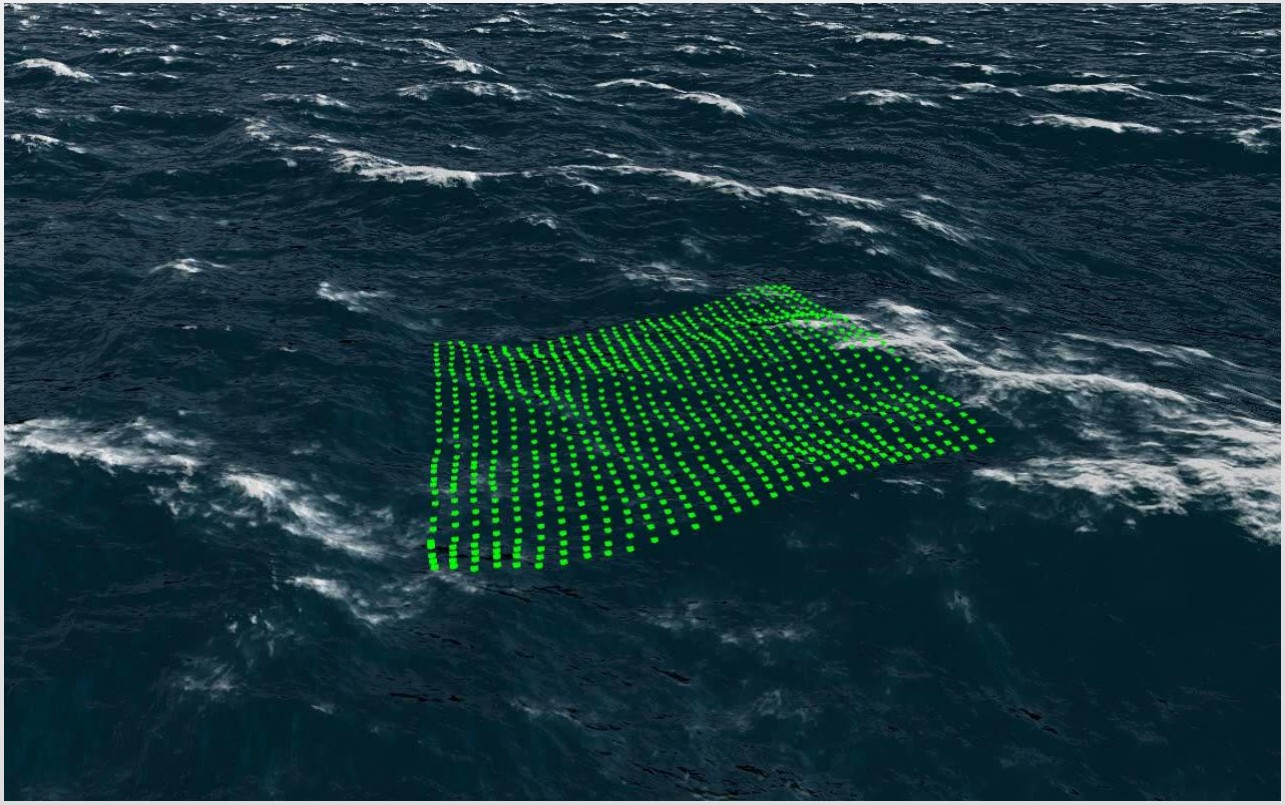
- Simulation cache
  - Simulate once, store, read when “past” results are needed
- Array of simulation results (2D arrays of displacement)
  - N(slices) * 4(cascades)*128x128(FFT size)*3(dx,dy,dz)
  - ~ 800 kb per slice
- Max lag = 1.5 sec, 48 ticks/sec -> 72 slices -> ~57MB
  - Can exclude cascades with small waves
  - Can simulate for physics at half rate

Ocean simulation for in-game physics

Readbacks for in-game physics

Readbacks for vessels and floating objects

- Sum displacements from all the cascades and return to the app
- Done on CPU
- Few readbacks + a bit of math = plausible vessel movement
- **Not for projectiles!**



Ocean simulation for in-game physics

Raycasts for in-game physics

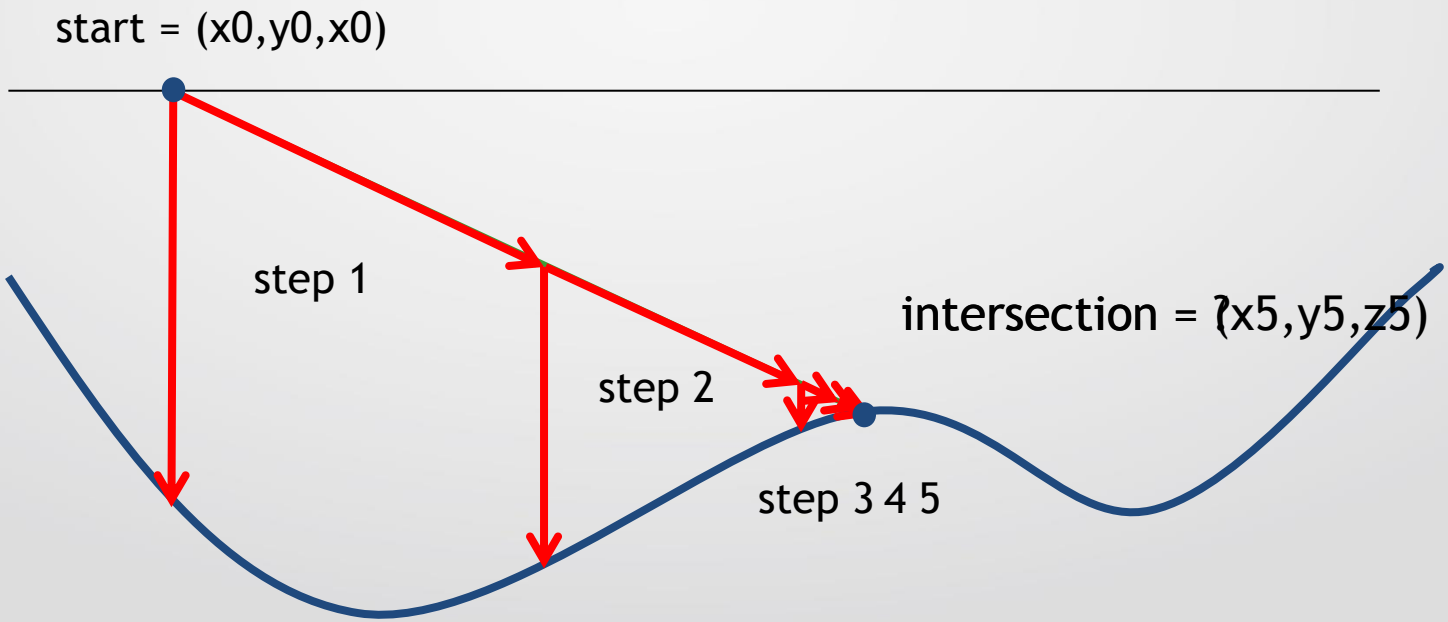
- Need ray/surface intersection tests for projectiles
- Raycasts for projectiles & water collisions
  - Require tracing through water volume
- Much more expensive than readbacks
  - Potentially infinite iterative process
    - Exit when refinement at current step < threshold (5 cm in sample)
    - Or # of steps is > predefined N
- Each tracing step is also not a simple readback!

Ocean simulation for in-game physics

Raycasts for in-game physics

Tracing water volume

Position += direction * (current_Y - water_Y)

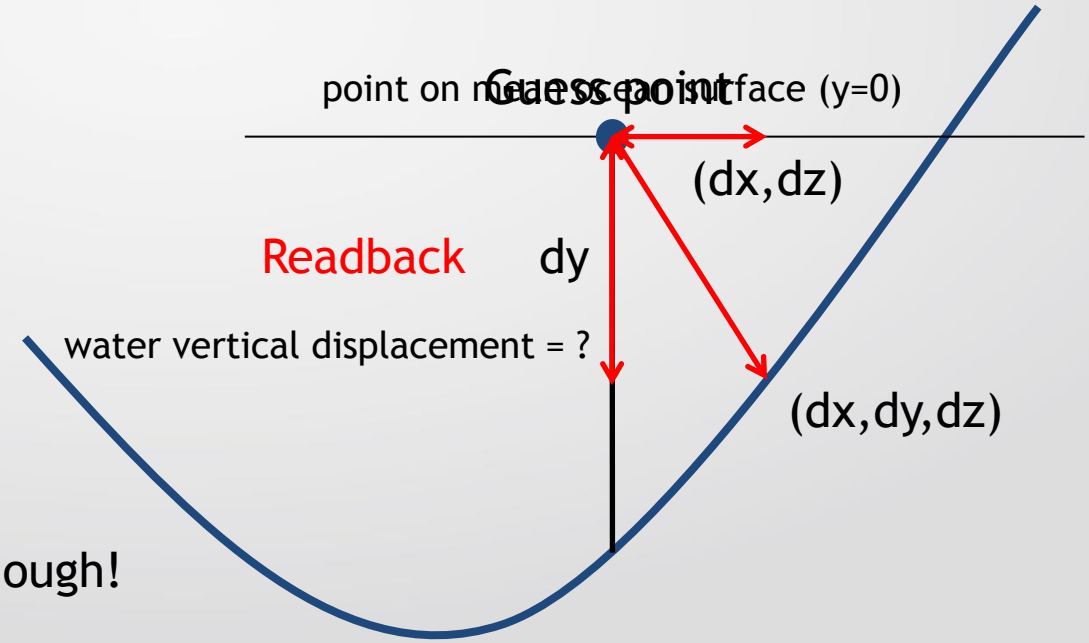


Ocean simulation for in-game physics

Raycasts for in-game physics

Assume that the displacements are locally linear

- Set “guess point” x,z to the sample point
- Read the displacements at “guess point” and move “guess point” back from sample point by the x,z displacements
- Repeat N times. 4 steps is enough!

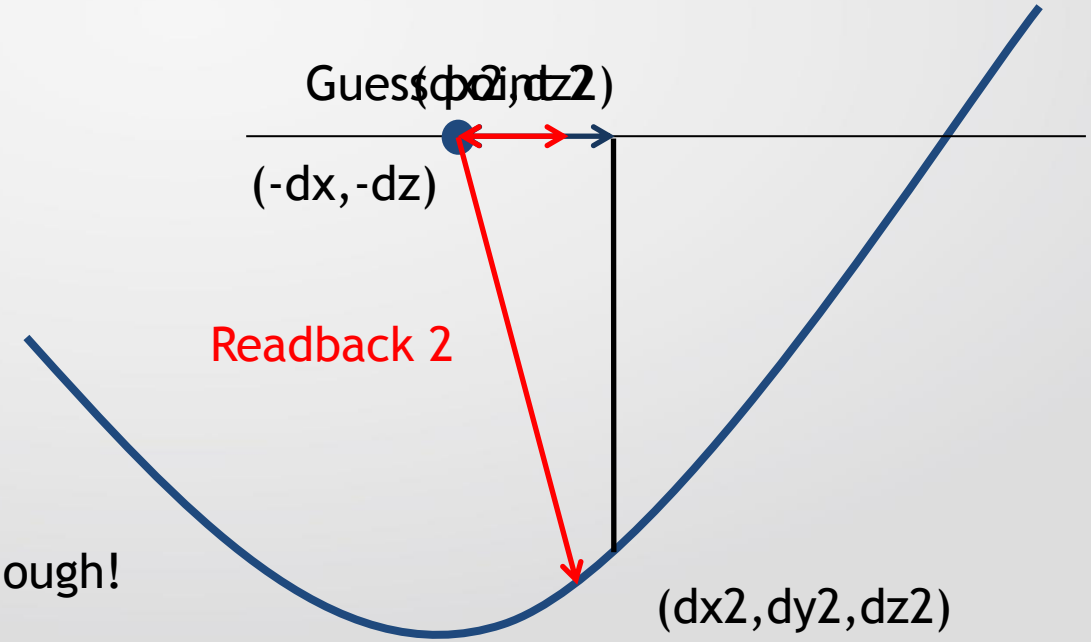


Ocean simulation for in-game physics

Raycasts for in-game physics

Assume that the displacements are locally linear

- Set “guess point” x,z to the sample point
- Read the displacements at “guess point” and move “guess point” back from sample point by the x,z displacements
- Repeat N times. 4 steps is enough!

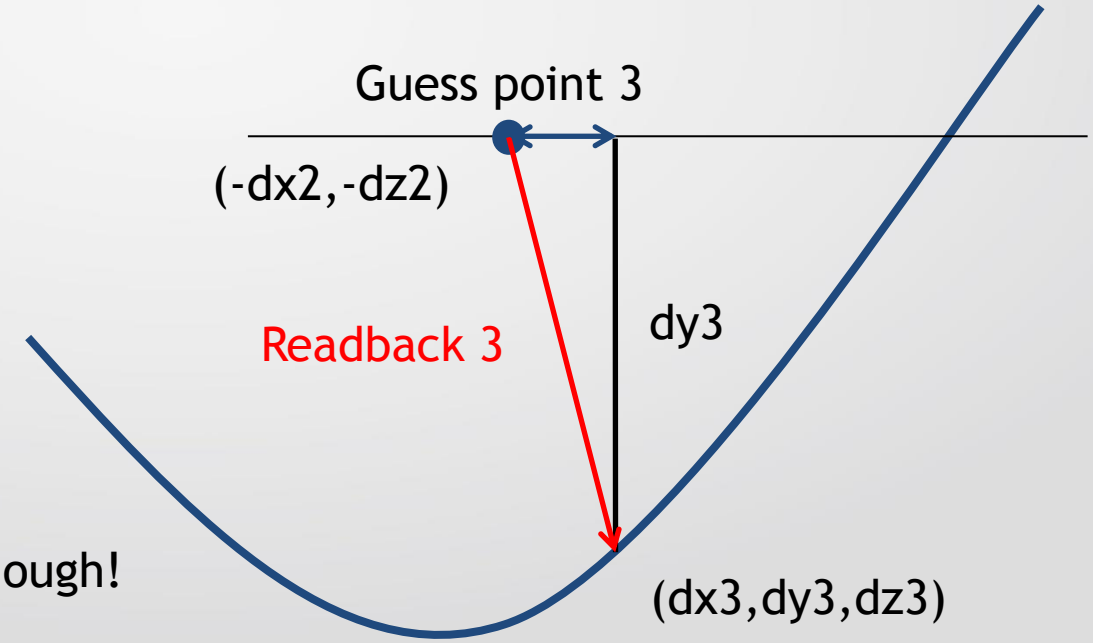


Ocean simulation for in-game physics

Raycasts for in-game physics

Assume that the displacements are locally linear

- Set “guess point” x,z to the sample point
- Read the displacements at “guess point” and move “guess point” back from sample point by the x,z displacements
- Repeat N times. 4 steps is enough!

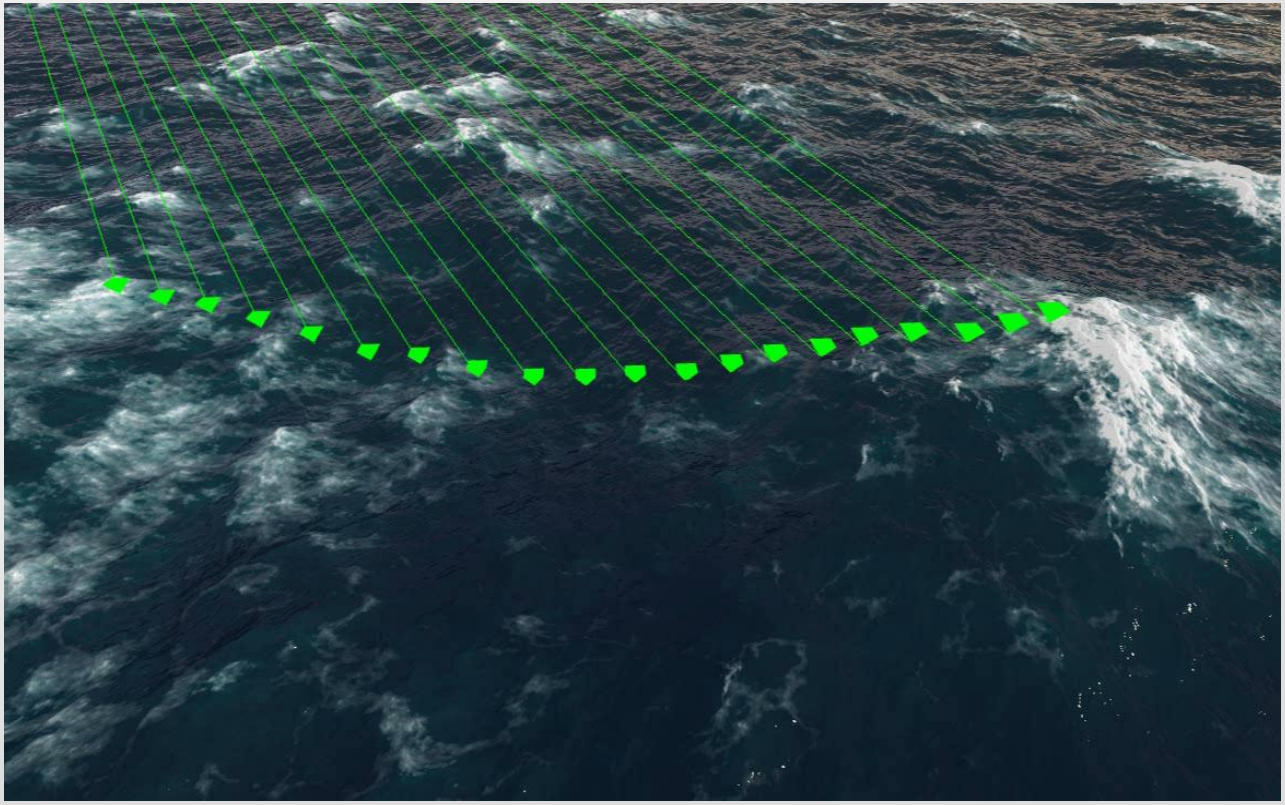


Ocean simulation for in-game physics

Raycasts for in-game physics

Result

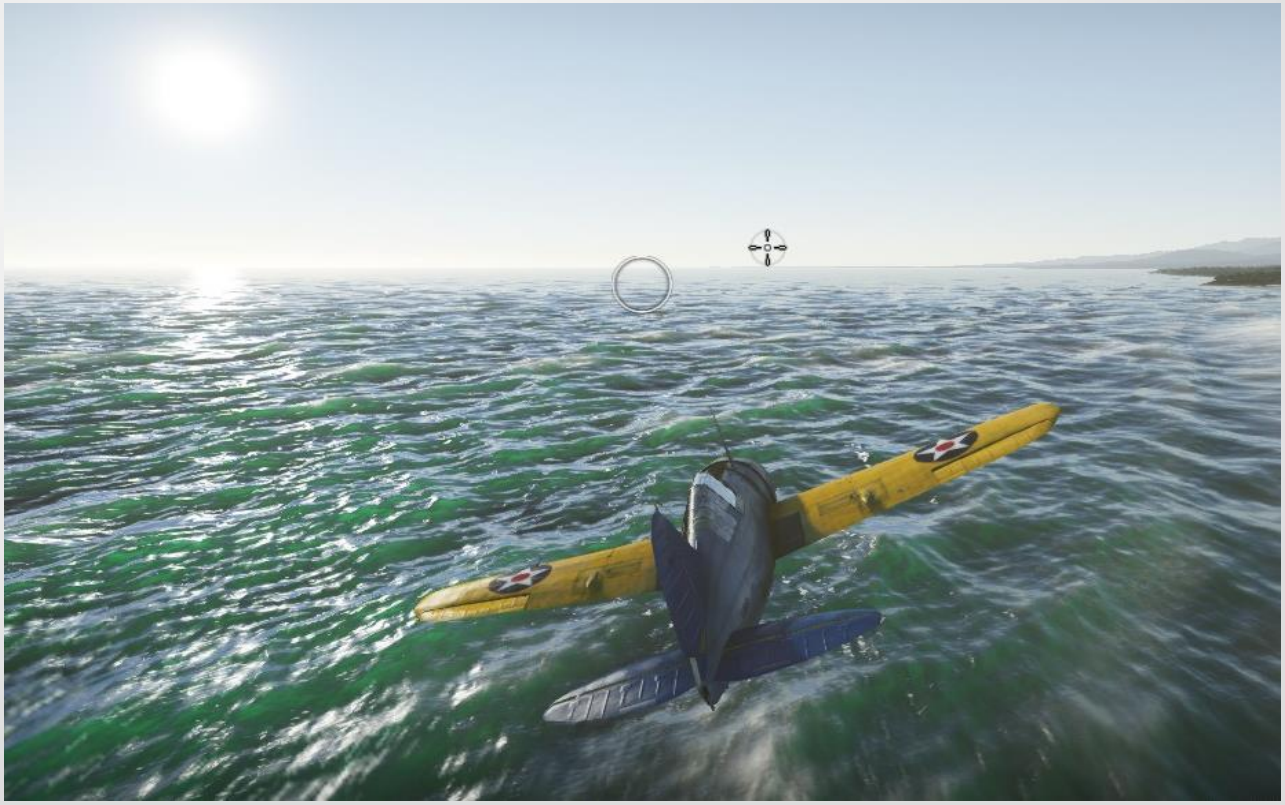
- Not cheap,
  but just a few of
  projectiles in flight
  need raycasts
- Up to ~2000
  projectiles
  in flight
  at 48 simulation
  ticks / second



Integrated in War Thunder

Not public yet

Demo & sample applications in WaveWorks distro

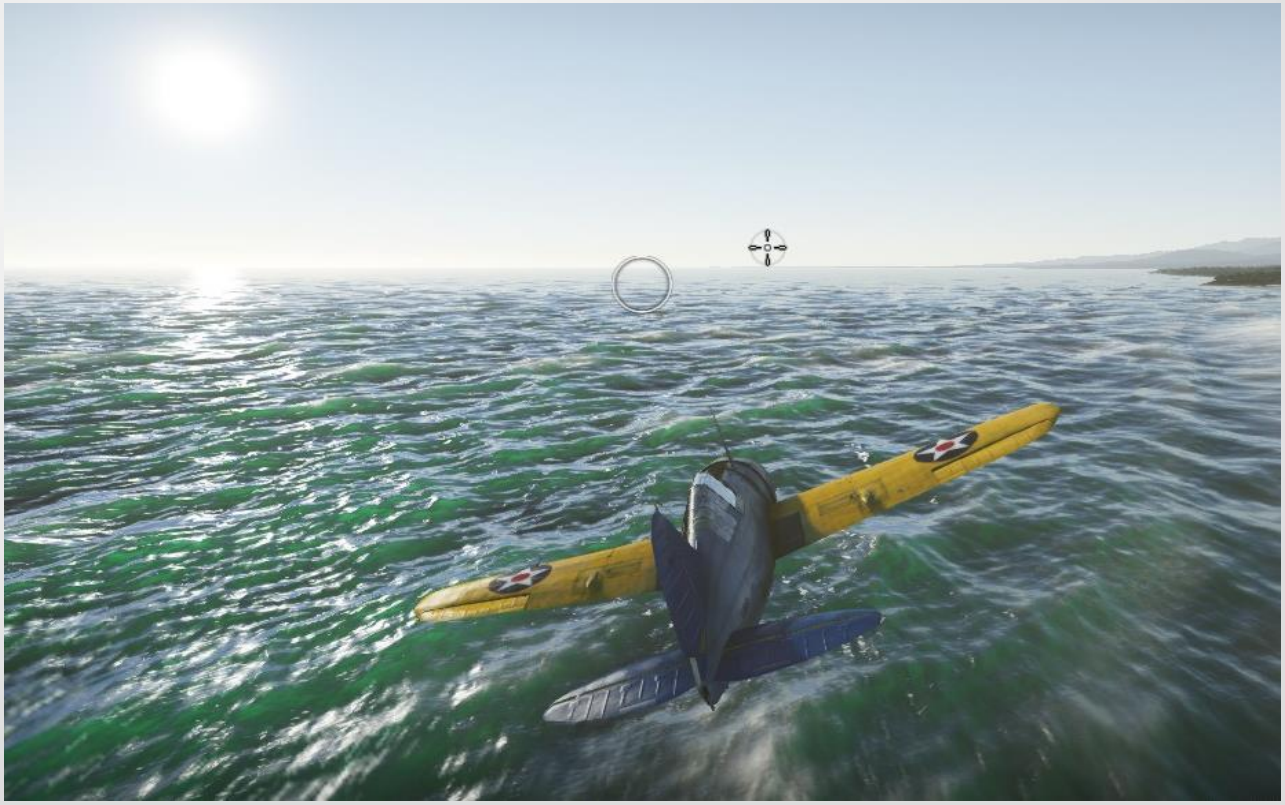


Integrated in War Thunder

Not public yet

Demo & sample applications in WaveWorks distro:

- Ocean simulation
- Rendering
- Geomorphing
- Shore interaction
- Raycasts
- Readbacks



Demo

D3D11 WaveWorks sample application

Android test application

Thank you!

Questions and answers

Tim Tcheblokov, NVIDIA


CGDC 2015 中国游戏开发者大会
CHINA GAME DEVELOPERS CONFERENCE


分享智慧
LET US SHARE

Simulation and rendering times

Simulation:

~0.5 msec on GTX770 (Middle range Kepler)

~5 msec on GK108 (Low-end mobile Fermi)

Rendering:

~0.5 msec on GTX770

~2 msec on GK108