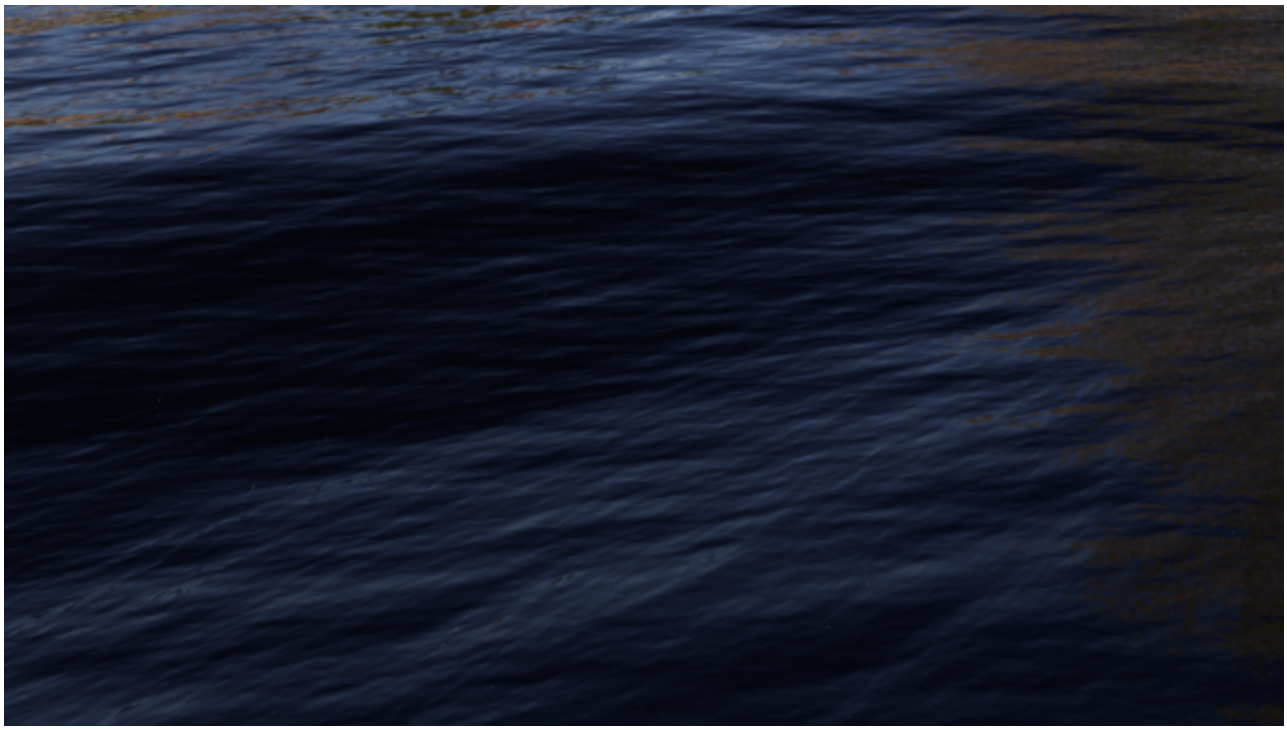
aaOceanPrmanShader

- Input Parameters
- Output Parameters
- Example Usage



On the Beautiful, Briny...

A Tessendorf Ocean generator. Code contributed by Amaan Akram (www.amaanakram.com).

This produces a bumped normal that can be fed into a displacement shader to create ocean-like waves.

Input Parameters

Resolution

Specifies generated texture resolution. Valid settings are 3 through 7. Each value generates the following resolution:

- A value of 3 generates a resolution of 128x128
- 4: 256x256
- 5: 512x512
- 6: 1024x1024
- 7: 2048x2048.

Ocean Scale

Size of the ocean patch in meters. Larger values "fit" a larger expanse of ocean into the rendered geometry.

Seed

Seed for the random number generator. Different values produce different ocean patches.

Current Time

Current scene time in seconds. Use an expression to drive this parameter.

Repeat Time

Waves loop (repeat their shape) after the specified amount of time in seconds.

Fade

Fades (scales) the ocean vector displacement strength.

Chop Amount

The sharpness of the peaks of the waves are controlled by this parameter. Also affects foam values.

Velocity

Controls the size of the waves. Higher velocities make fewer, but bigger waves. Lower velocities make a calm ocean.

Wave Speed

Speed multiplier for waves.

Cutoff

Defines a smoothing factor for the overall ocean surface. Cuts off (removes) waves with wavelengths smaller than the specified value.

Wave Height

Height mulitplier for waves.

Wind Direction

Direction the waves travel in.

Damp

Bias waves travelling in the direction of the wind. Value of 0 makes waves travel opposite to the wind direction. 1.0 makes the waves travel in the wind direction only.

Wind Align

Align waves perpendicular to the direction of the wind, as observed in shallow waters.

Invert Foam

Makes wave peaks appear white, instead of make wave troughs white.

Gamma

Gamma factor to apply to the eigenvalue output.

Brightness

Multiplier for the eigenvalues.

Normalize

Whether to normalize the eigenvalues. See *fMin* and *fMax*.

fMin

fMin and fMax are used to normalize the eigenvalues, which can go into negative and positive values. See render log for typical values on a frame.

fMax

fMin and fMax are used to normalize the eigenvalues, which can go into negative and positive values. See render log for typical values on a frame.

Write File

Writes out a full-float OpenEXR vector displacement map sequence in object space.

Output Folder

Path to the folder for output images.

Postfix

Provides any postfix to apply to the generated file to identify it uniquely.

Current Frame

Set an expression for the current frame for frame padded sequence names.

Ocean Depth

Slows down waves as depth decreases.

Surface Tension

Capillary waves that run on top of the ocean surface. For small-scale effects only. See the original Tessendorf paper for details.

Invert T

Inverts the t parameter for the manifold.

Manifold

Provides the domain over which to apply the displacement. Defaults to s,t.

Output Parameters

outputDisplacementRGB

The bumped normal.

outEigenvalueFloat

The eigenvalues.

Example Usage

To quickly get a convincing-looking ocean render, we can use PxrSurface (using the Glass parameters) as the Bxdf. The only thing we want to tweak is the *Refractive Index* (eta), which we'll change to 1.33 to match the value of water.

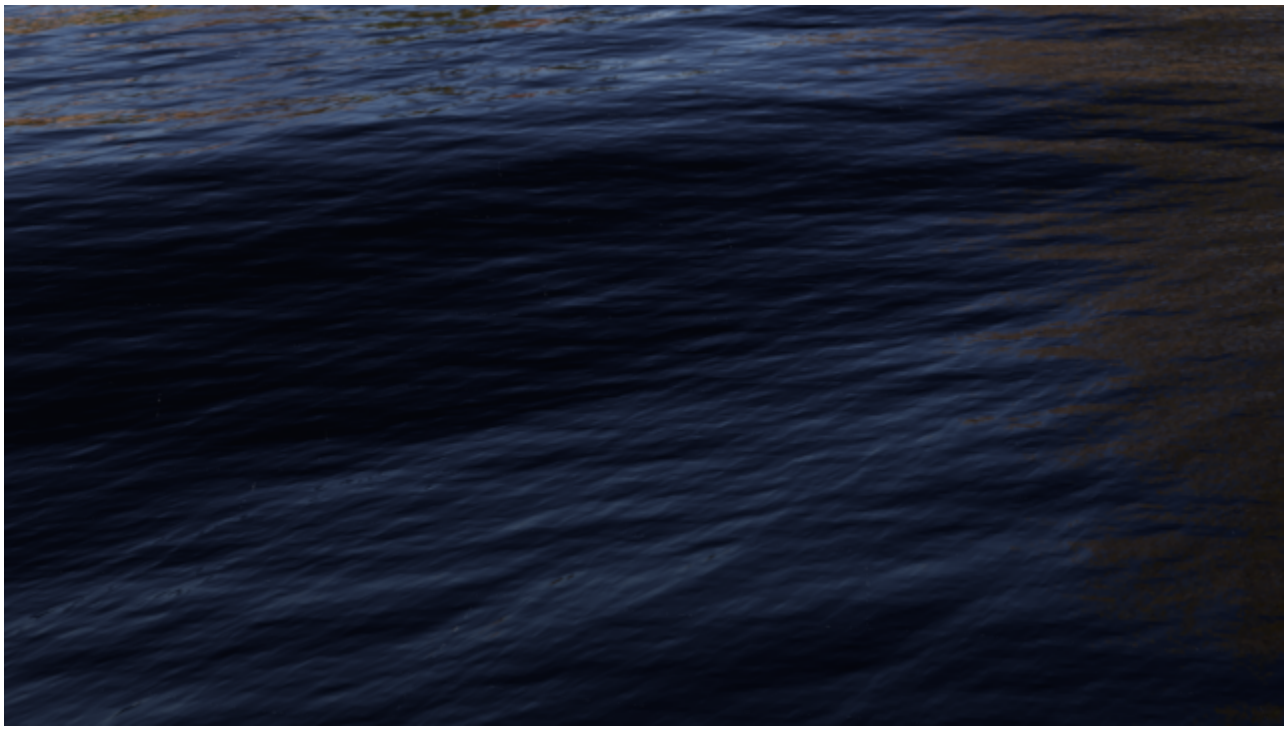
For the displacement, we can use the PxrDisplace shader. Connect aaOceanPrmanShader's *outputDisplacementRGB* output.

Below is an example render with the default settings for aaOceanPrmanShader. The ocean is being lit by an environment light using the `Luxo-Jr_4000x2000.tex` texture that is included with RenderMan.



A calm ocean.

Depending on the size of the object, it may be necessary to increase the *Resolution* parameter. For the next render, because of the size of the plane, *Resolution* has been increased to 7. Also, *Chop Amount* is increased to 10.0.



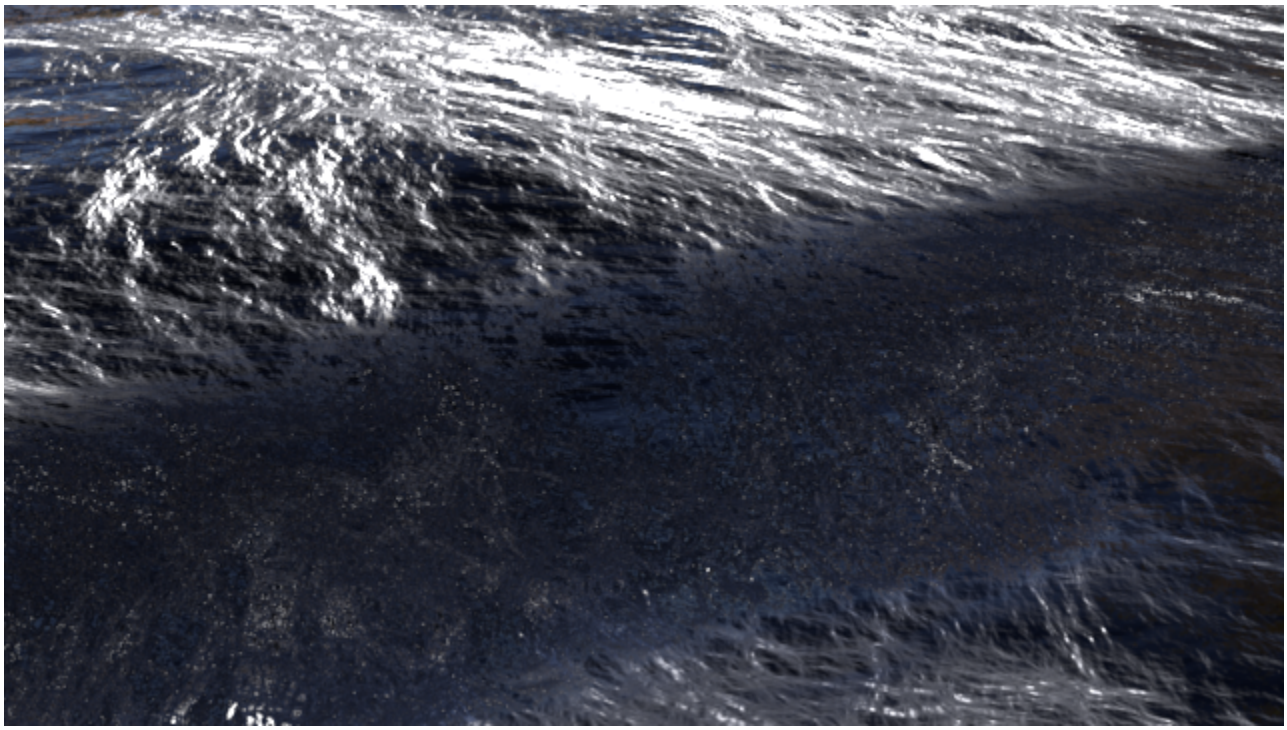
Things are getting bumpy.

To add some simplistic white foam, we can add a diffuse layer to our ocean, by adding diffuse to our PxrSurface.

Create a PxrLayerSurface shader, and connect the *outputDisplacementRGB* output to the (diffuse) *Bump* parameter and set the (diffuse) *Color* to something closer to white. Make sure that the specular and clear coat lobes have been disabled.

For the layer mask, we can use the *outEigenvalueFloat* output from the aaOceanPrmanShader pattern. It's necessary that the values fall into the 0-1 range. We can easily enforce that by using a PxrRemap node. Connect the PxrRemap to the layer mask parameter. Because PxrRemap only works with colors, we can use a PxrToFloat3 to convert the *outEigenvalueFloat* to a triplet. Set the *Input Min* and *Input Max* to -5 and 5, respectively, on the PxrRemap node.

Lastly, turn on *invertFoam* in the aaOceanPrmanShader node so that the white appears at the wave peaks. You should then get a render similar to the image below.



Not so calm anymore.