aaOcean for MODO 801SP4/901

Version 1.0.3

Philip Stopford - May 28, 2015

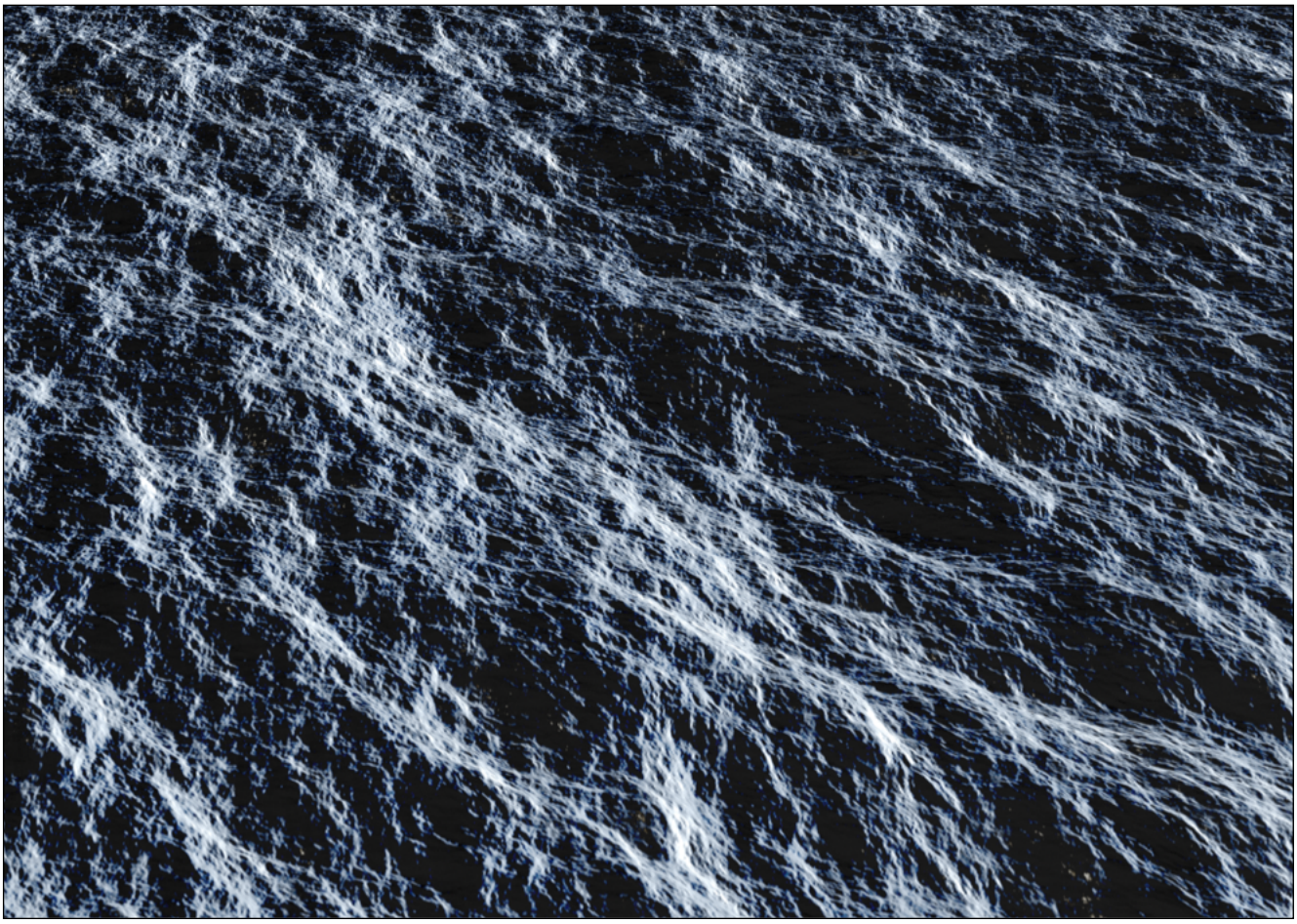


Table of Contents

Introduction

The aaOcean library was developed by Amaan Akram and is an implementation of Jerry Tessendorf's 2004 paper on simulating ocean waves. Version 2.6 of the library is the foundation of the port for MODO.

aaOcean for MODO consists of two separate plugins, each with their own advantages and disadvantages. Note that, due to the way the plugins work in MODO, identical settings may not give the exact same results between plugins (e.g deformer has different operating conditions than the texture system).

Note that aaOcean is not an ocean shader.

This is an open source project; you can find the entire source code, including dependencies and project files at https://github.com/philstopford/aaOcean_MODO/. It is made available under the terms of the GPL v3 license; aaOcean is GPLv3.

Release 1.0.2 updates aaOcean to align with patches applied by Amaan based on the issues identified by Andrew and related adjustments. It also enables the channel modifier.

Release 1.0.3 greatly improves the texture implementation, reducing memory to ~50% of the previous level and increasing render speed by more than 30%. A MODO SDK limitation currently prevents the vector displacement working in an ideal manner. This cannot be fixed without MODO SDK support, which will hopefully come in a Service Pack for MODO 901. This is one of the reasons that the wave height input for textures has been removed from the main interface (it still exists in the channel list, but this change reduces the scope for users to run into disappointing results). MODO 701 will be supported based on requests, but is not a high priority.

Requirements

aaOcean for MODO is built in its default configuration for MODO 801SP4 and 901 (a MODO 701 kit will depend on requests). It's available for Mac OS X, Linux and Windows.

Installation

aaOcean for MODO is delivered as a kit. There are two kit versions - one for MODO 801 SP4 and one for MODO 901. You can install aaOcean to your user Scripts folder or to the MODO kit folder by following the instructions below.

Install Location

1. Launch MODO.
2. From MODO's 'System' menu, select 'Open User Scripts Folder'
3. A file browser opens in that Folder.
4. Drag and drop the 'aaOcean_xxx' folder from the archive to this location, where xxx is 801 or 901 depending on the version(s) of MODO you want to use aaOcean with.
5. Restart MODO.

Mac OS X - Additional Steps

The Mac version is built against Yosemite (10.10), but should work back to Mountain Lion (10.8). It's only been tested on Yosemite, however.

Windows - Additional Steps

The Windows version is built using Microsoft's Visual C++ 2013 system. That requires updated runtime libraries on this platform (x64). They can be downloaded and installed from Microsoft's site here : http://www.microsoft.com/en-us/download/details.aspx?id=40784

Linux - Additional Steps

Under Linux, there are some additional dependencies that are required. These should be available from your distribution's package manager.

libfftw3f (and libfftw3f_threads if it exists, but, distribution-depending, this may be included in the libfftw3f package)

For example, Ubuntu 12.04 has this as libfftw3-3.

General aaOcean Information

aaOcean works, behind the scenes, in a pretty general manner. It generates a tile, the dimensions of which are set by the ocean size value and the detail of which is set by the resolution value. This tile is repeated over the geometry/surface as needed.

Common parameters are discussed here. Outputs are discussed under each respective implementation because the implementation defines the available outputs and how they can be used.

Inputs

Output Type : Choose the output type (not available for all aaOcean implementations)

Resolution : Choose the map size to set the amount of detail. Larger values will provide more detail, but take significantly longer to evaluate and require more memory. They may also give the impression that MODO has frozen. Give it time. See additional notes for the deformer with regard to this setting.

Ocean size : The physical size of the ocean tile that is being generated.

Wave height : This defines the wave height, essentially a contrast control for the wave patterns. (note that 1.0.3 removes this from the texture plugin interface, but it exists in the channel list)

Surface Tension : Defines the viscosity of the ocean.

Wind Align : Stretches the waves perpendicular to the direction of the wind.

Smoothing : Smooths out the surface, removing smaller waves from big ones.

Wave Direction : Angular setting to control the prevailing direction of the waves.

Wave Reflection : By default, the ocean will have no prevailing direction of travel for the waves. You can use this control to change that. A value of 1 will completely eliminate waves traveling in the opposite direction to the wind.

Wave Speed : Controls the speed of the waves traveling over the surface. Increasing the ocean size will seem to slow down the waves, as they will have a longer distance to travel.

Chop : This controls the sharpness of the peaks of the waves. Higher values sharpen the tops. If you set the value too high, the tops of the waves will self-intersect.

Wave Size : Sets the wave size, as you might expect.

Ocean Depth : This can be changed, but has a relatively minor effect on the ocean.

Seed : This is used to initialize the random number generator. You can use this to change the Ocean; if you find a value you like, you can use this in the same context in a different scene to get the exact same result.

Repeat Time : The time at which the ocean animation should repeat.

Do Normals : This channel is internally disabled within aaOcean due to compiler issues with a required internal method. I've left this available in the internals, but removed from most areas of the UI to avoid confusion. You might still see it in the channel list.

Do Foam : Where applicable, this enables foam generation for the ocean. This is required for the output type '1' to be operable.

Deformer

The deformer works against any mesh item that contains vertices. It has the benefit of providing immediate feedback in the viewport, working with the MODO deformer stack and also enabling vertex-based constraints of items (e.g. buoys, boats). It does require a dense mesh to provide convincing close-ups. You can use this with subdivision surfaces - the deformer will affect the cage.

Implementation Notes for Inputs

Resolution : in this mode, the final resolution of your ocean will of course depend on the number of vertices in your mesh. The resolution value only defines the granularity of the ocean map used to calculate the deformation. Increasing the ocean resolution on a low density mesh will not improve the output.

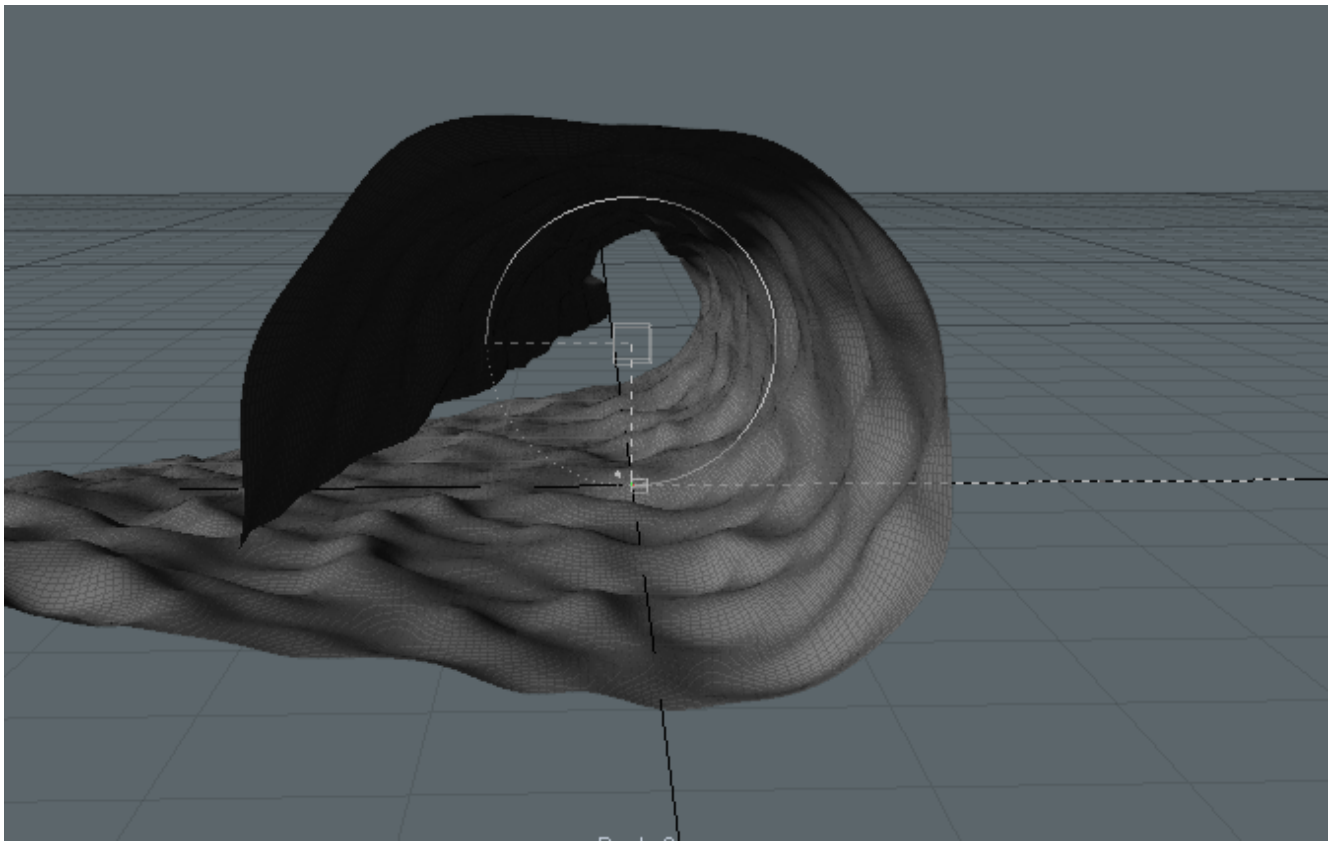
Ocean size : this defines the real world ocean size from which the vertices on your mesh are processed.

Wave height : defines the real world wave height of the deformer.

Outputs

The deformer only uses the displacement outputs of the aaOcean library. There is no output support for extended features (e.g. foam, eigenvectors) due to architectural limitations in MODO.

aaOcean for MODO allows you to take advantage of MODO's order of operations system to stack deformers. In this way, you can use a bend effector to create looming waves, a vortex effector to create whirlpools, etc.



Texture

The texture system is the fastest way of generating displacement for the ocean. You can take a simple quad poly and generate enormous oceans with compelling detail. You can use displacement or vector displacement, but note that the vector displacement can sometimes result in artifacts due to the high velocities resulting from wave folding.

The texture implementation of aaOcean for MODO also works pretty similarly to most texture items in MODO. It can use a texture locator to set the size, if you wish to work that way.

Using the output type setting, you have the option of choosing the information output in the RGB channels of the color texture. For the full set of outputs, you will need to instance (or duplicate) the texture multiple times and this will incur additional memory and computation expense.

Note also that you won't see any viewport updates from this texture in displacement mode - you'll need to have Preview running.

Implementation Notes for Inputs

Output type : Used to select what is output by the texture:

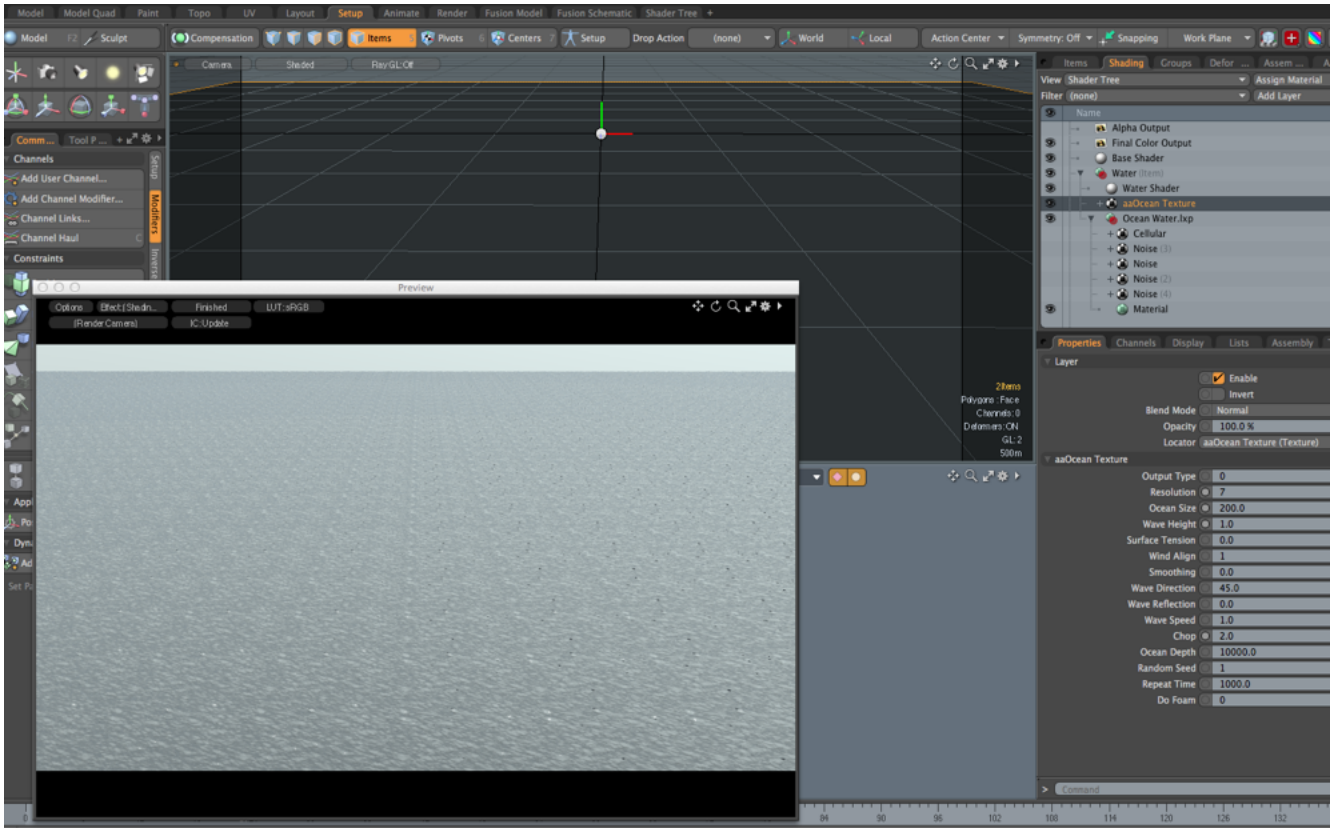
- Displacement data (RGB or scalar)
- Foam map (scalar)
- Eigenminus vector (RGB)
- Eigenplus (RGB)

Ocean size : You have the choice of using the ocean size control to set the ocean size, or you can set the texture locator size. The results are the same either way, so it depends on how you prefer to work - you can make all changes in the aaOcean panel or jump back and forth if the typical MODO workflow appeals. The strong recommendation is to use this setting rather than the locator, but the flexibility is there to use either or both as you see fit.

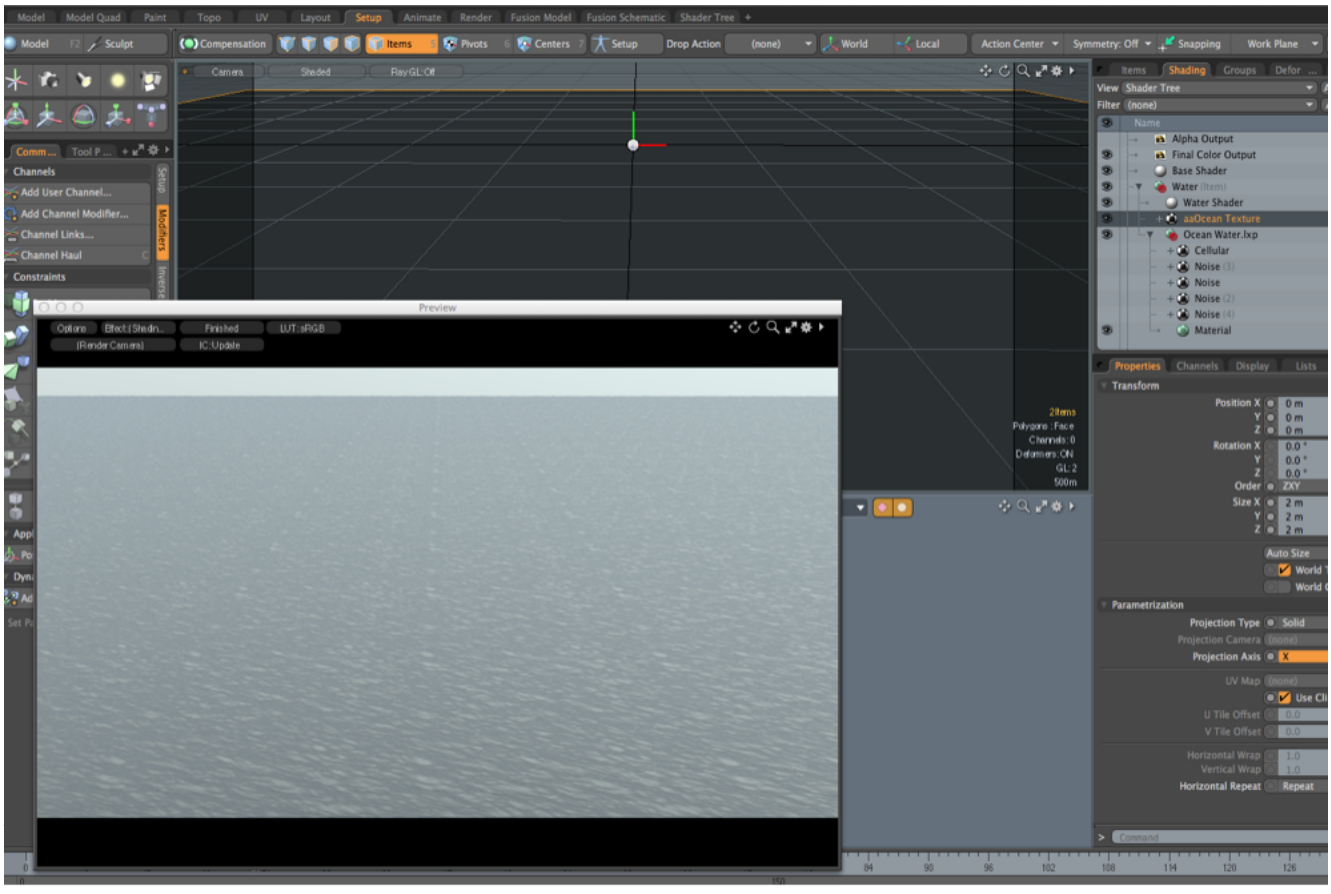
Like the deformer system, if you set the ocean to the exact same size as your surface, you will see the ocean tile perfectly match. To that end, and depending on your camera view, you might want to set the ocean to be slightly larger than your object's surface.

By way of illustration, the following images are for a 200 m ocean size (and the viewport grid is at 500 m) :

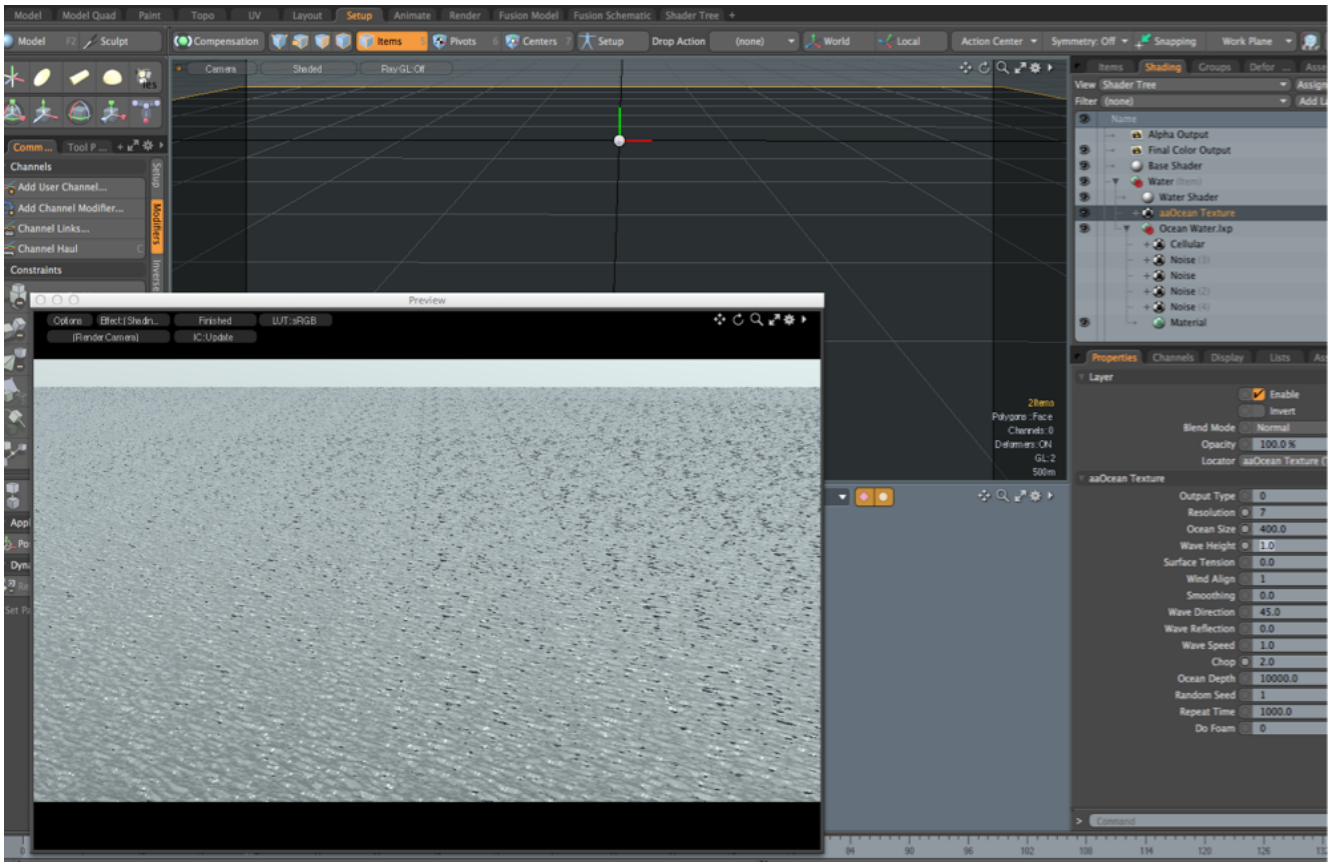
Texture locator size 1 m (notice the tiling):



Texture locator size 2 m (reduced tiling) :



Texture locator size 1 m, ocean size 400 m :



Wave height : **This control was removed from the user interface with 1.0.3 due to potential for confusion and as a result of an SDK issue that prevents proper usage. It remains available in the channel list for the determined user.** At 1.0, in a displacement texture, this works with material displacement height value to define the maximum wave height. For a displacement texture, setting this above 1.0 will result in clamping of the displacement. In other texture modes, this clamping will not be seen in the output.

Outputs

The texture output depends first of all on the way that the texture layer is being used in the ShaderTree.

In all modes of the texture layer, the value and alpha outputs are processed. The alpha is always set to 1.0 and the value output is always the Y displacement of the Ocean.

In color/vector modes, the texture layer output varies based on the Output Type set for the ocean texture :

In displacement mode, the red/green/blue channels are set to the respective displacement amounts in XYZ. In eigenvector modes, the output is the XYZ vector (in RGB form).

- A note about displacement is needed here. MODO expects all displacement output from a texture to lie between 0 and 1. 0.5 represents no displacement, 0 is a full negative and 1 is full positive. This value is used with the displacement height of the material to define the actual displacement at each spot on the surface during rendering.

Channel Modifier

The channel modifier is intended to drive features like particles and arbitrary item positioning based on the ocean configuration. The output is perfectly matched to the texture plugin for the same inputs, but the evaluation will generally be slightly slower. The controls and outputs are identical to the texture plugin. Some notes are below in the case you want to drive surfacing with the channel modifier :

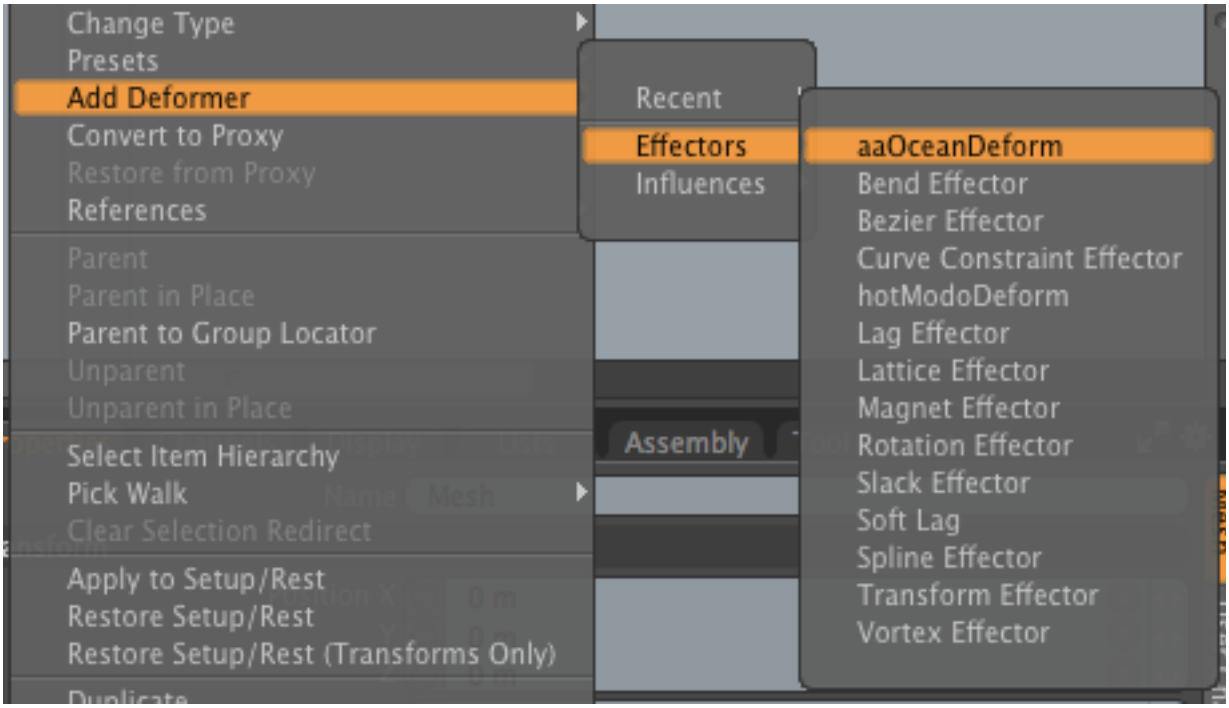
- Use the Sample Position World Position X and Z inputs to input X and input Z on the channel modifier.
- To drive displacement on the surface, add a constant in the ShaderTree and drag it into the schematic. Change the constant to the vector displacement mode and then wire the channel modifier displacement into the Color input of the constant node.

Let's Get Our Feet Wet

Deformer

We're going to start with the deformer system because it has the advantage of displaying its results in the viewport and the ocean works based on the mesh size.

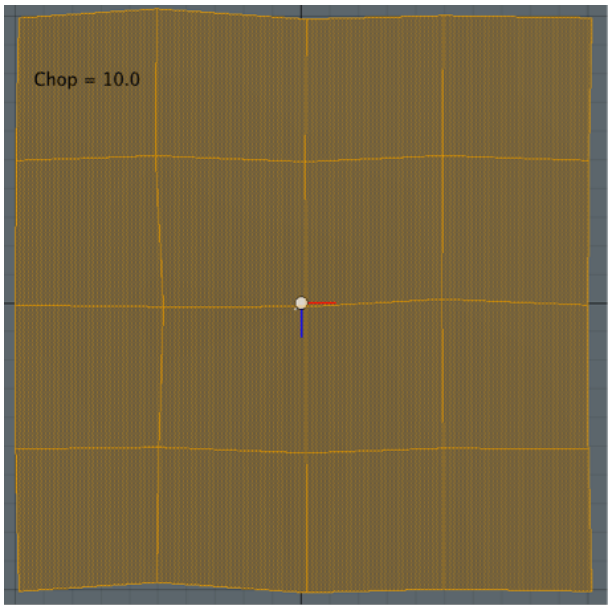
1. Add a unit plane to a mesh item. Scale it up to be 200 m on each side, centered on the world origin. If you sized the mesh item rather than the geometry itself, freeze the transforms on the mesh item.
2. Right click the mesh item in the item list and the Use the 'Add Item' drop down menu in the item list to add aaOceanDeform from the Deformers submenu.

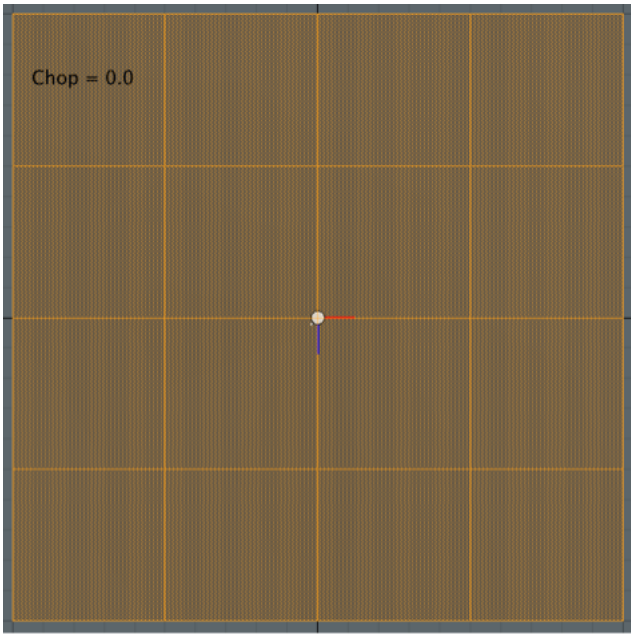


3. You should see a small shift in the viewport. It's not very convincing as an ocean, so let's subdivide the plane a few times.
4. In polygon mode, select your quad, hit Shift-D and hit return. This will turn your single quad into 4 polygons. You'll begin to see the ocean deform. Keep subdividing and increase amounts of detail will show up, but of course your mesh will become very dense.
5. Scrub the timeline to show how the ocean characteristics change over time.

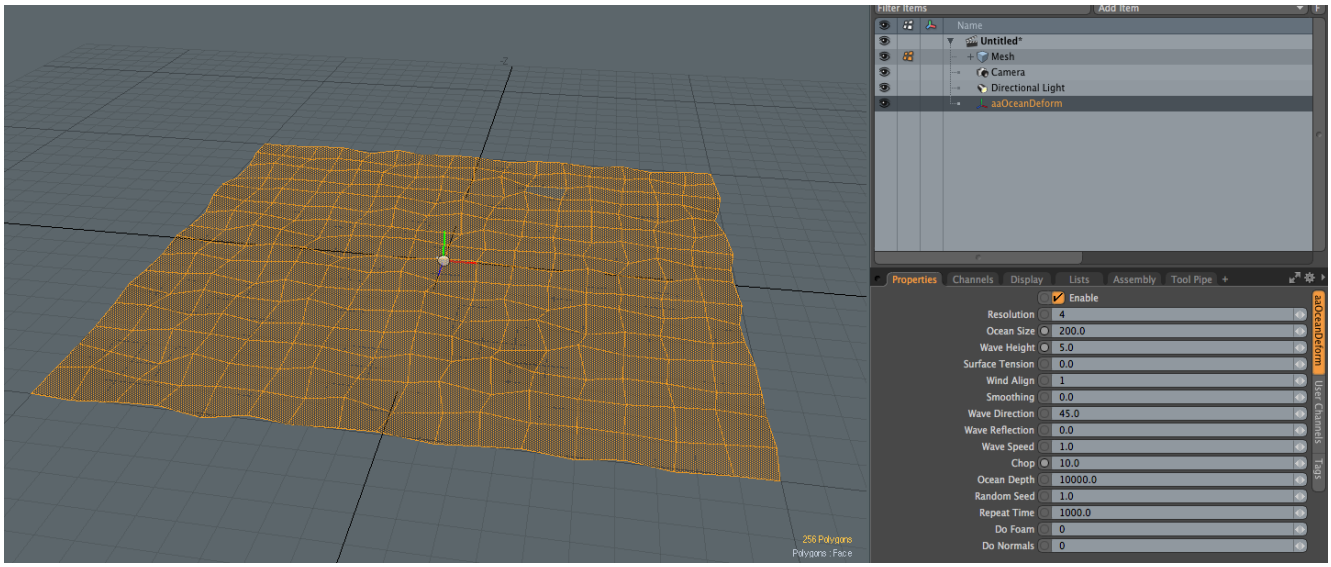
6. Change the wave height to, say, 5 m and you should begin to more clearly see the ocean effect.

7. Let's quickly review the chop effect. By default, it's set to 1.78. Switch to a top view of your mesh. You'll see that the edges are all irregular. If you set chop to 0, the edges will become orthogonal. Chop, basically, adds lateral movement to the displacement of the vertices. No chop means only vertical displacement is applied to each vertex.





8. At 256 polygons, the ocean starts taking some basic shape. Let's keep subdividing to 4096 polygons. A reference is below.



9. If you scrub the timeline now, you will be able to see the wave interactions (forward and reverse).

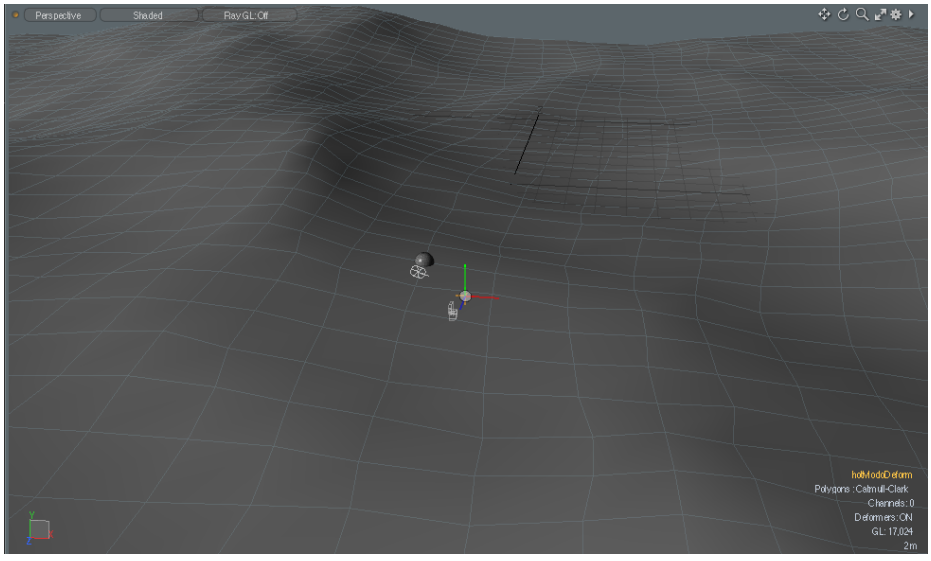
10. Let's add a point of interest. Add a unit sphere to the scene in a new mesh item.

11. Select the aaOceanDeform entry in the item list and deselect 'Enable' to temporarily turn off the deformer.

12. Select the Sphere. Shift-select the ocean mesh.

13. From the 'Setup' workspace, under 'Modifiers' choose 'Vertex Position'. This will set up a constraint from the ball to the nearest vertex on the ocean.

14. Re-enable the aaOceanDeform deformer. You should see something like this :



15. As you scrub the timeline, the ball will follow the deformed ocean. This is one of the key benefits to the deformer mode for aaOcean. It's very easy to set up and it's trivial to create constraints for objects.

16. Now that we have a lightly detailed ocean mesh thanks to the subdivision, let's explore how the resolution parameter affects the look and feel. Go ahead and change it from the default. Lower values evaluate extremely quickly, but lack any subtle motions that impart realism. At the higher end, impressive detail is added, but the evaluation slows down markedly.

17. Try also adjusting the ocean size. This is where you have to pay some attention to your settings. The ocean size relates directly to your ocean mesh size. For our example here, the default value of 200 matches with our geometry because of the sizing we performed in step 1. If you change the ocean size to 100, you will see tiling in the pattern because the ocean map needs to affect the entire mesh and so it has to repeat twice in both directions. If you make the ocean size 400, the ocean will appear blurred because a significant portion of the map is never applied to the mesh because it falls outside the mesh itself. In general, matching this value to your mesh size should be the first step in setting up your ocean. Detail is added using the other settings.

- Note that scaling the mesh item without freezing it will not change the effect of the deformer on the ocean mesh item.

18. The surface tension control is, as the name might suggest, a way of adjusting the apparent viscosity of the ocean. Higher values will give the effect of a lower viscosity fluid.

Let's take a look at the wind settings.

19. Set your ocean size back to 200.0. Set resolution at the highest value you can live with, e.g. 1024. Set wave height to 7.0 and chop to 0.0 for the moment. Set surface tension back to 0.0, wave speed to 2.0 and wind align to 0. If you scrub the timeline now, you'll see that the waves behave as though there is no wind in the scene. Change Wind Align to 1 and this will change. You can use the Wind Direction control to steer the waves.

20. The smoothing value, at higher values, will work to suppress the waves on the ocean.

21. Wave Reflection affects the amplitude of the waves traveling against the prevailing flow. At a setting of 0.0, the waves have the same amplitude in both directions; for a value of 1.0, there are no waves traveling against the prevailing flow. You can adjust this during playback to get an idea of the response to this control.

22. The seed value can be used to define the input to the ocean's internal random number generator to change the ocean look for the given set of parameters.

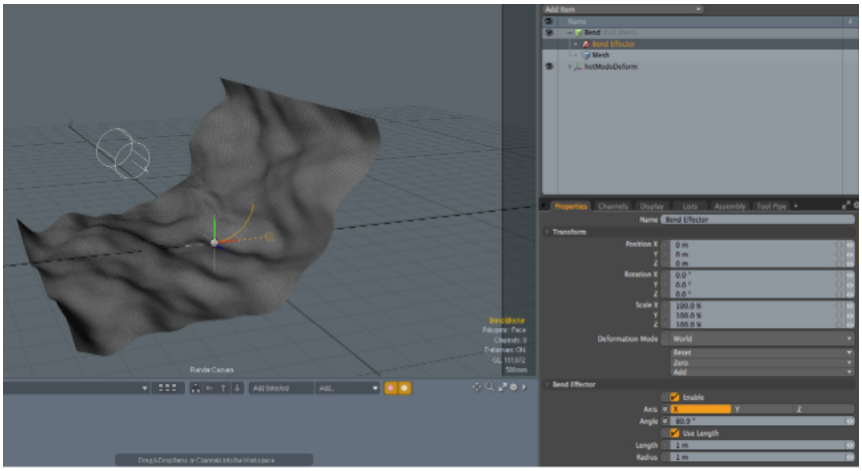
23. Repeat Time defines the time at which the animation of the waves should loop.

24. In the deformer, the 'Do Foam' and 'Do Normals' options are exposed, but don't yield anything in the current MODO system.

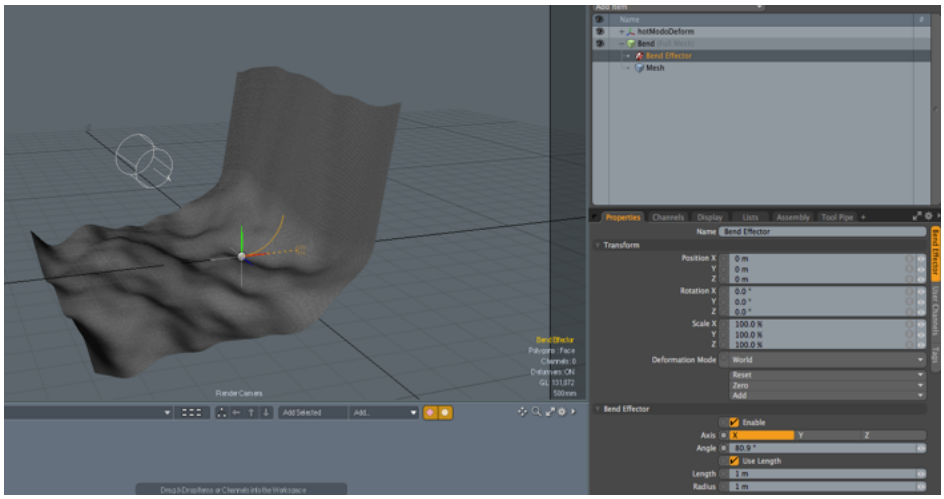
25. Now, let's explore the deformer stack options for aaOcean. Go back to the item list and right click on your ocean mesh. Add a bend effector from 'Add Deformer -> Effectors' from this menu. Hop over to the Deformers list. If all went well, the Bend effector will be at the top of the listing, with aaOceanDeform at the bottom.

- You might find some ugly shearing being displayed in the viewport - to resolve this, you need to increase the radius value of the bend.

26. Let's adjust the bend settings on the Bend Effector (you can expand the Bend group in the Deformer list, or hop over to the Item List and select the Bend Effector directly). The Properties panel will then show the controls we want to adjust. Choose the X axis and increase the angle.



27. Now had back over to the Deformers list and move aaOcean to the top of the list. You'll see something like below. This is because aaOcean uses the X and Z positions of vertices to define the deformation in the Y axis.



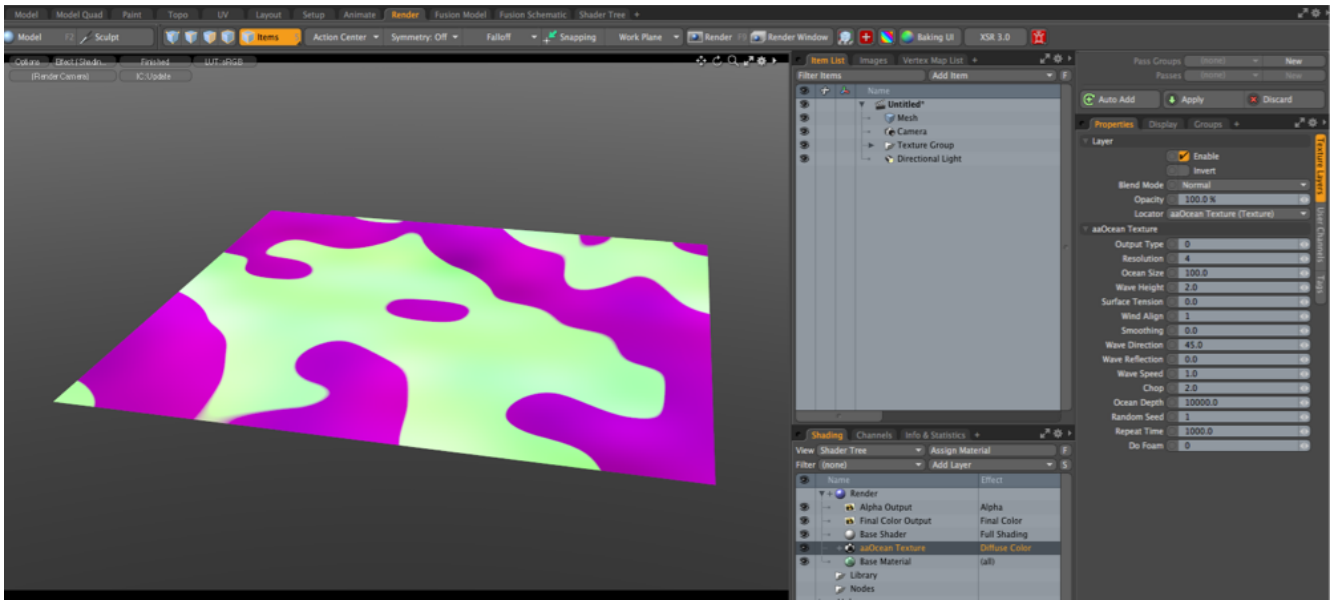
28. You can also use falloffs with the aaOcean deformer. Just right click the aaOceanDeform to add any falloffs that you would like to use.

Remember that you can also use Subdivision or Catmull-Clark Subdivision surfaces with the deformed mesh.

Texture

Please go through the Deformer introduction before this section. By doing so, the basic principles will be familiar to you and we can skip to the interesting aspects of this part of aaOcean for MODO.

1. Start with a fresh scene. As before, add a unit plane and scale it up to be a reasonable size (e.g. 20-50 m on each edge). If you resized it in item mode, freeze the transforms.
2. Let's now jump over to the render workspace in modo. All of our work will be done here, with our single quad ocean. Add an aaOcean Texture from the aaOcean Texture group in the 'Add Layer' drop-down of the ShaderTree. The default mode is 'Diffuse Color' and we'll leave it like this for a moment. Most of the time, you will probably be using aaOcean as a displacement or vector displacement texture (except for foam). The render view will show you a colorful pattern. Head over to the Texture Locator panel and set the locator size values to 1.0 in each axis

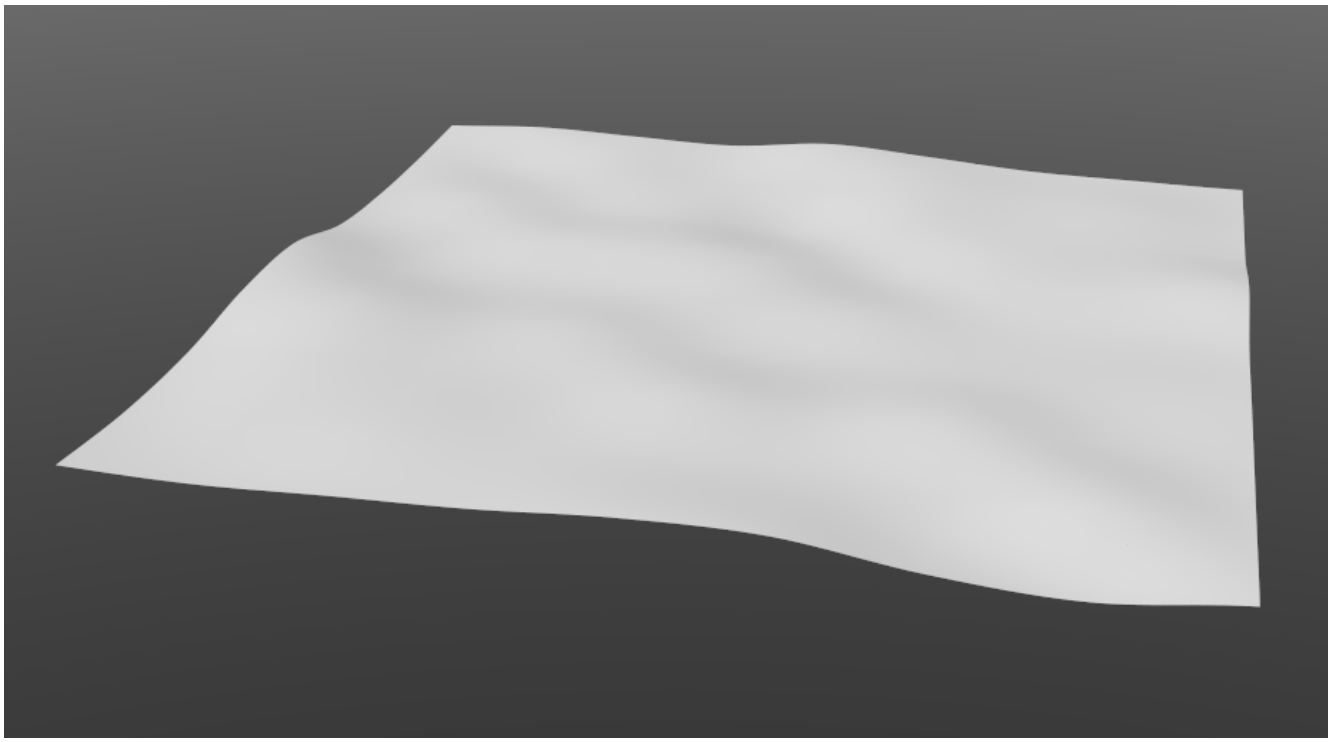


3. Optional : Set the ocean size to 1.0. In the Texture Locator properties, you can use the gang edit to change the size values to match the X and Z dimensions of your quad polygon. Set the Y value to the same value as the X and Z channels. When you're done, set the texture locator size back to the original values. It tends to be easier to use the texture controls to configure the ocean, so we will focus on that. However, you have the ability to exploit any aspect of modo to accomplish your task

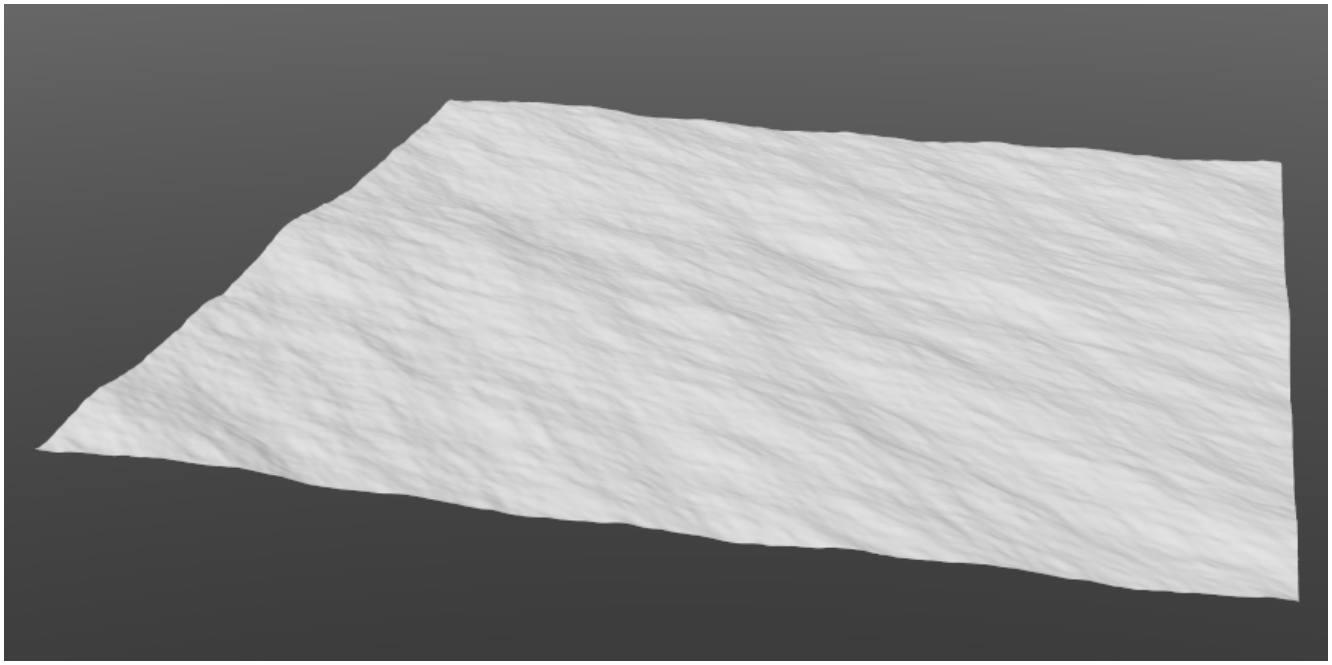
4. aaOcean can work with the various texture projections available to it (solid, UV, etc.) and the various related settings (UV rotation, etc). You can see the effect by changing the the various values. For most purposes, Solid works nicely, so we'll go with that one. Set the values back to the defaults if you changed anything other than the Texture Locator size (right click on the small circles for those settings you changed and 'Remove Edits')

5. Let's try the vector displacement option. Right click on the aaOcean Texture row in the item list, within the effect column and change from Diffuse Color to Vector Displacement (found under Surface Shading). You'll probably immediately be disappointed because nothing has happened. This is the first thing to remember about the displacement texture system in MODO. The textures only set the relative strength of the displacement over the surface. The amount of displacement is defined in the associated material.

6. On the associated material, which in this basic scene is our Base Material, find the 'Displacement Distance' parameter on the Material Ref sheet and increase it to 4 m. As you do so, the displacement will be apparent in the render preview.



7. Change the ocean size to 50.0, making the internal map better fit our object size and crank up the resolution to something like 128 or 256.



• Preview can sometimes get stuck and hold on to the same displacement in different frames, or only draw a very poorly evaluated mesh. If that happens, toggling the displacement texture on and off in the ShaderTree with the eye icon will force Preview to update.

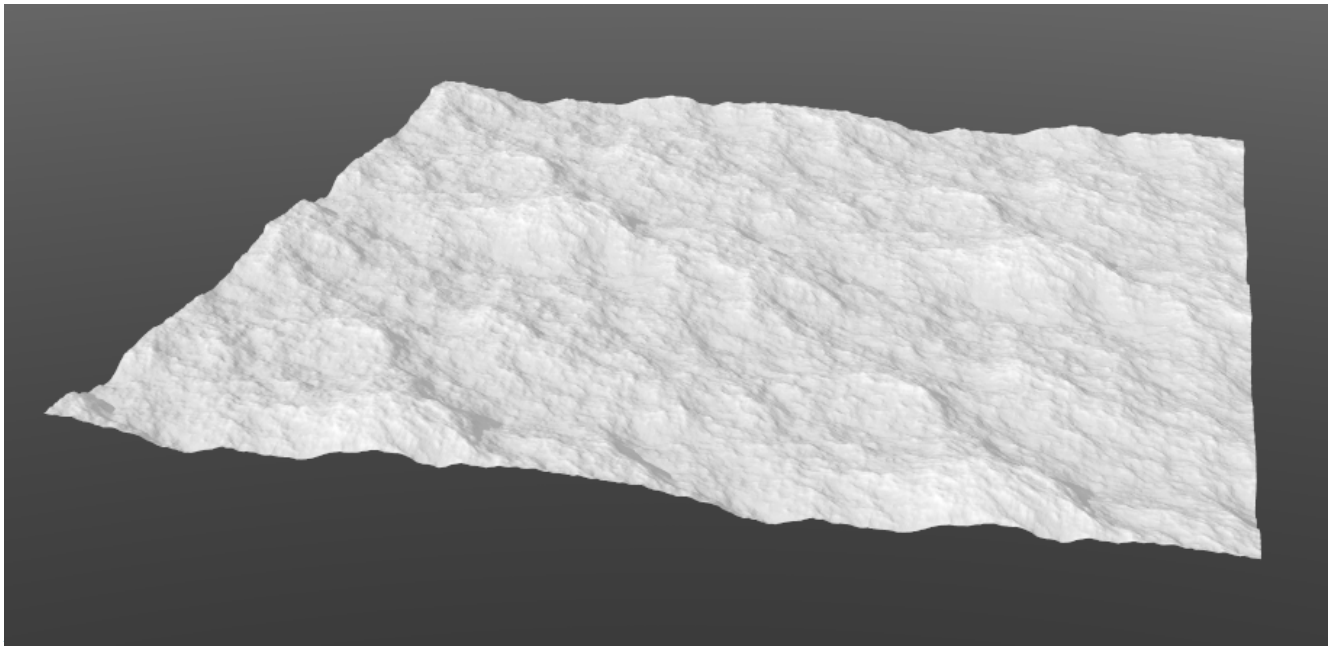
8. Unlike HOT, increasing the resolution in aaOcean doesn't radically change the ocean shape - it just adds more detail. Note that, at the maximum setting of 4096, you'll be getting over 16 million points on the ocean map that defines the displacement!

9. aaOcean also works with the regular displacement mode. Sometimes, this can even work better than the vector displacement, but you will probably need to greatly reduce the wave height value (e.g. from 1.0 to 0.05) {NOTE : in 1.0.3, this wave height property is available in the channel list - the input was removed from the main user interface]. Switch over to compare the results, in the same way that you changed from Diffuse Color to Vector Displacement back in step 2.

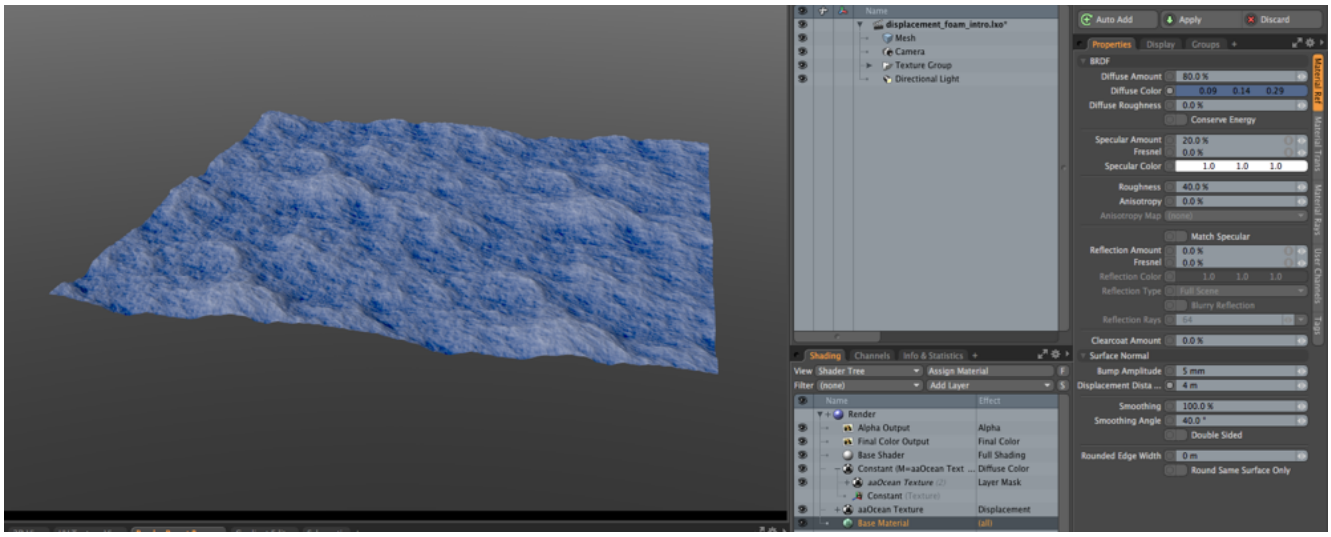
10. Now let's try something more advanced. Pause Preview to make things easier.

11. On the base material, in the Material Ref sheet, change the diffuse color to a dark blue. In the Material Trans sheet, change the Luminous Color to a light blue.

12. Instance the aaOcean texture layer. On the instanced layer, change the Output type to output foam. Accordingly, also set the 'Do Foam' value to 1. Invert the layer and change the layer effect from Vector Displacement to Luminous Amount. Restart Preview. You should see something like the below.



13. The more effective way to use this is as a layer mask. Change the material color to a blue, and add a constant to the shader tree that is set to diffuse color and white. Drag and drop the instanced aaOcean texture layer on top of the constant to use it as a layer mask. You should get something like below.



14. For larger oceans, you will find that you need to blend multiple oceans of different sizes and seeds in order to eliminate tiling whilst preserving detail in close-up shots.

FAQ

The plugin isn't showing up in 801 (release, SP1, SP2, SP3). Why?

This plugin is built against MODO 801 SP4's SDK. It won't show up in earlier versions.

I get tiling - what can I do?

The ocean system uses tiles that are repeated over the ocean surface. You'll need to blend multiple oceans with different seeds and sizes to break this up. For the texture and channel modifier plugins, this is easy - you can just stack them up. For the deformer, this presents a problem - you're best off using the deformer for crude deformations and then using displacement or bump mapping using aaOcean to hide the tiling from each part.

I don't see any displacement in my render (using displacement or vector displacement). What's up?

Check that you have a material affecting the surface and ensure that the displacement height on this material is appropriately set. Due to MODO's architecture, we can't set this from the texture plugins.

Linux : MODO won't start after installing this. Why?

This is likely the result of at least one missing dependency. On Mac, this doesn't appear to prevent MODO starting up, but Linux installs seem to be less tolerant. Check that you have the libraries installed (see Installation section, Linux : Additional Steps). Note that these libraries will need to be installed on Linux render nodes for the plugin to work there as well.

Linux : MODO renderslaves are having trouble. Why?

This is likely the result of at least one missing dependency.

Windows : I'm getting an error that the program cannot start due to MSVCR120.DLL being missing.

The Windows kit is compiled using Microsoft's Visual C++ 2013 software. That requires updated runtimes on many Windows systems. You can find a link to the installer in the Installation section, Windows : Additional Steps.

Any wake or similar functionality?

Not in aaOcean itself, no. However, the channel modifier can drive particles.

Known Issues in 1.0.3

Vector displacements are not entirely correct. They look OK, but there is a potential issue with clipping and there is no ability to generate decent chopped waves due to the inability to convert coordinates, as a result of a MODO SDK limitation.

Development Notes

This section is purely for those interested in the code and approach taken. It is not intended for end-users; it's more of a lessons learned and MODO SDK section for developers.

MODO SDK

It will build against 801 or 901. For this release, I settled on the 801 SP4 SDK and the 901 SDK.

Currently, vector displacement is not correct due to a MODO SDK issue that prevents coordinate conversion from the object space returned by the library to the tangent space needed by MODO. The dpdu/dpdv derivates in the current SDK, in textures, are always zero, making it impossible to manage the conversion.

OpenMP

aaOcean expects OpenMP, but this isn't available in Apple's developer systems. To that end, the OpenMP aspects are made optional in this code with a preprocessor directive, per the OpenMP documentation, in aaOceanClass.h. If you set _OPENMP, you will be able to explore OpenMP builds :

```
#ifdef _OPENMP
#include <omp.h>
#else
#define omp_get_num_procs() 1
#endif
```

An OpenMP enabled version of clang was used to build this plugin, via Homebrew (https://clang-omp.github.io)

Linux Build

MODO is built with quite an old toolchain, GCC/G++ 4.1, due to the target of RHEL 5.4. This toolchain is generally not available on newer systems, and is also incompatible with the compiler requirements of the aaOcean class. At the time of writing, building with the 4.8

toolchain appears to work just fine and MODO is happy with the resulting binary. Note that 1.0.3 uses newer C++ standards, so a corresponding parameter is set in the Makefile.

To compile this, you need to install the following dev packages from the package manager (building these dependencies from source can be rather painful)

- libfftw3f-dev
- libfftw3f_threads-dev

With these packages installed, you will need to adjust the LXSDK location in the Makefile in the main source folder. You can then just run 'make'. The plugin will be available at Linux/build/aamodo.lx

The resulting binary should work fine as-is on other systems as long as the library dependencies (libfftw3f, libfftw3f_threads) are met.

Threading

Unlike HOT, aaOcean is intended for threaded usage, so the approach taken is pretty standard. However, MODO makes it a little tricky to efficiently share data between threads. Andrew Helmer deserves much of the credit here for his extensive help with the channel modifier implementation. I used this as the basis for the texture plugin overhaul for 1.0.3.

Extended Information - Implementation Limitations

There are some implementation limitations for plugins in MODO that limit the ability to output the broad range of values from aaOcean in every mode. The deformer can only deform (and it cannot set vertex map values); the texture plugin can only output to the standard texture fields, hence the output mode setting in order to make the extended features available. This is the reason for the channel modifier which is, sadly, quite slow due to the internal design of this part of MODO.

A custom material is being considered as a possibility.

Acknowledgements

- Amaan Akram for creating aaOcean and being kind enough to respond to e-mails asking for implementation insights.
- Matt Painter for the generous support.
- Andrew Helmer for figuring out the root cause for the infuriating crash bug that drove me mad for weeks.
- Erwin Zwart for gently goading me into action.
- Greg Duquesne, Gwynne Reddick (of The Foundry), for help with HOT that enabled me to make faster headway with aaOcean, and for the encouragement throughout.
- Pete Segal, Matt Cox, Arnie Cachelin, Milan Bulat (of The Foundry) for extensive and patient guidance during both HOT and aaOcean efforts.

Copyrights and Licenses