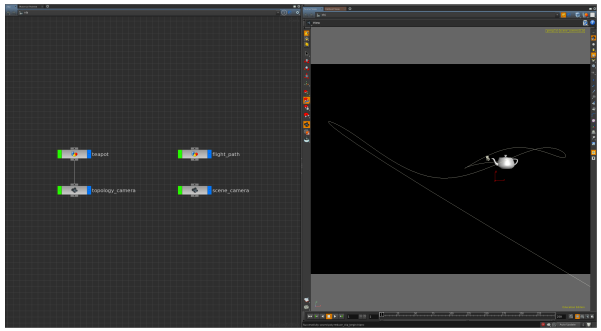
ADDITIONAL OPTIMISATION UTILITIES

Render times can be further reduced by utilising a combination of the Poly Reduce SOP and the Clip SOP for Polygon based geometry. Open the scene **polyreduce_clip_begin.hipnc**.

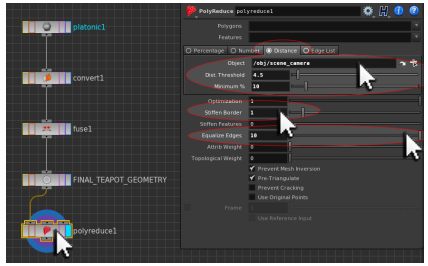


This scene contains a **Polygon** teapot **animated along a flight path**. **Parented** to the teapot object is a **topology_camera**. Currently there is no change to the topology of the teapot geometry relative to the scene_camera.

Inside the **teapot object**, append a **PolyReduce SOP** to the **FINAL_TEAPOT_GEOMETRY NULL SOP**. Activate its **Distance parameters** and specify:

Object	**/obj/scene_camera**
Dist. Threshold	**4.5**
Minimum %	**10**
Stifen Border	**1**
Equalize Edges	**10**

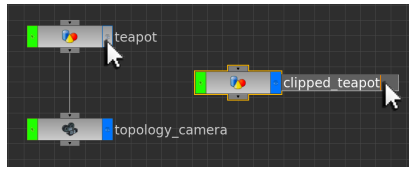
This will reduce the amount of polygons on the teapot, whilst retaining its overall shape and appearance.



When the teapot is far away from the scene_camera its topology is greatly reduced; however when the teapot gets closer to the scene camera, its original topology is gently restored.

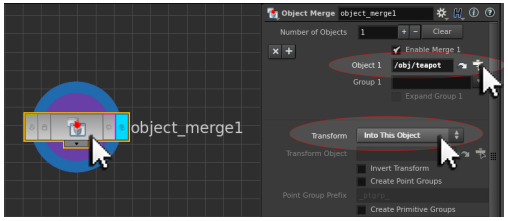
Examination of the teapot topology can be done by setting the **Viewer** to **look through** the **topology_camera** and pressing **PLAY**. Activating **Hidden Line Ghost Shading** will make the topology transition easier to see.

Polygon geometry can be **optimized further** by use of a **Clip SOP**. The **Clip SOP** can remove polygon geometry that starts going outside of the camera view. At **Object Level**, turn off the **Display Flag** for the **teapot object**, and alongside it **create** a **new piece of geometry**. Rename this new **geometry object** to **clipped_teapot**.

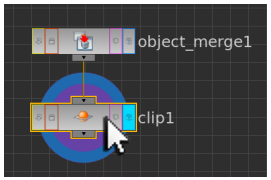


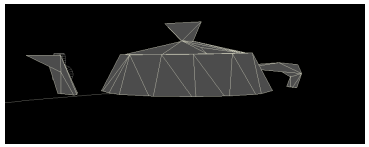
At the **Geometry Level** for the **clipped_teapot object**, **delete** the **default File SOP** and create an **Object Merge SOP**. In its **parameters** specify:

Object 1	**/obj/teapot**
Transform	**Into This Object**



This will read the animated teapot (including the PolyReduce SOP effect) into a new object where the geometry can be clipped independently from the animation.

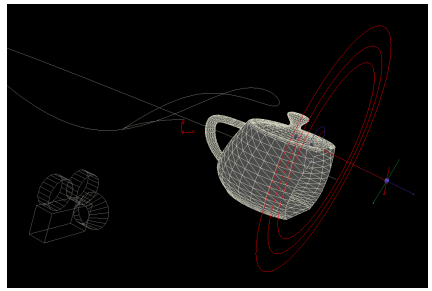




Append to the **Object Merge SOP** a **Clip SOP**. By default this node will create an infinite clipping plane where geometry above or below the specified plane is deleted.
In the **parameters** for the **Clip SOP** specify:

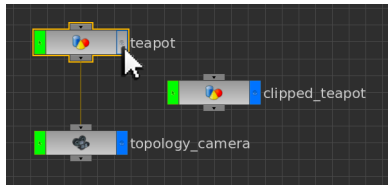
Keep	**Primitives Above the Plane**		
Origin	**2.1**	**0**	**9**
Distance	**0.6**		
Direction	**-0.8**	**0.4**	**-0.45**

This will create an **infinite clipping plane** just outside of the **scene_camera view** that will clip geometry no longer seen by the scene_camera view.



NOTE: The Clip SOP can also be positioned interactively in a Context View when in Clip SOP Tool Mode.

Even if parts of an **object** are **outside** of the **camera's frustum**, the **entire object's geometry** will be **loaded** and **processed** at **render time**. This **clipping method** can **explicitly cull** of **sections of the geometry outside** of the **camera frustum**, so **even if** the **entire geometry** is being **loaded at render time**, **only part** of it **will** actually **be processed**.



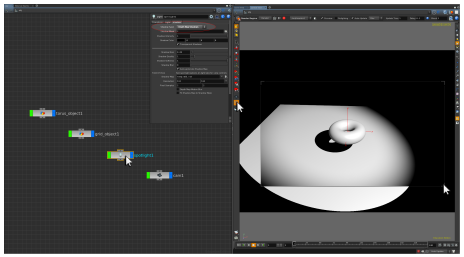
As a **final step**, the **Display Flag** for the **original teapot object** can be **deactivated**, so that only the **clipped_teapot** is **active** for **rendering**.

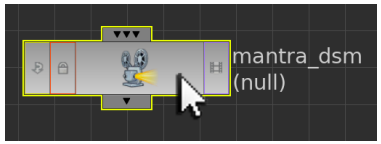
See file polyreduce_clip_end.hipnc

PRE-RENDERING DEPTH MAP SHADOWS

By **default**, a **depth map shadow image** will be **rendered out automatically** at **render time** when Depth Map Shadows are activated on a light. In order to **generate pre-rendered Depth Map Shadow files**, a **custom Mantra ROP** must be **created**, and the **resulting shadow map reloaded** into the depth map shadow scene light.

Open the scene **DepthShadowMap_begin.hipnc.** This scene contains a torus above a grid, illuminated by a spotlight with Depth Map Shadows activated.





At **Outputs Level**, create a new **Mantra ROP**, and rename it to **mantra_dsm**. In its **parameters** specify:

Camera /obj/spotlight1

☑ **Override Camera Resolution**

Resolution Scale **User Defined Resolution**

Resolution **1024** **1024**

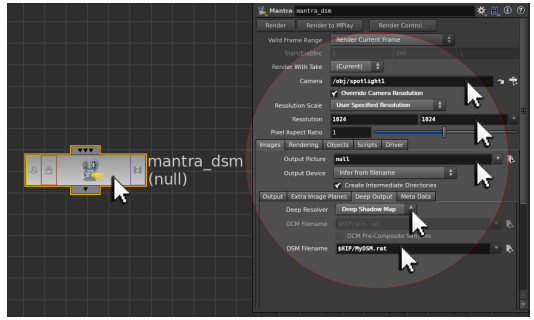
NOTE: The resolution of the shadow map must be square.

Images >

Output Picture **null**

Images > Deep Output

Deep Resolver **Deep Shadow Map**



In the **now active DSM Filename parameter** specify:

DSM Filename **$HIP/MyDSM.rat**

NOTE: for animated shadow map sequences, **$F4** should be used, and a **frame range specified** on the **mantra_dsm ROP**.

Finally press the **mantra_dsm Render Button** to **render** the **shadow map to disk**.

Return to **Object Level**, and in the **Shadow parameters** for the **spotlight1** specify:

☐ Auto-generate Shadow Map

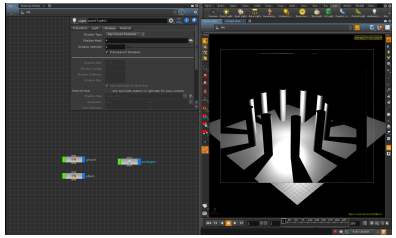
Shadow Map **$HIP/MyDSM.rat**

Now when the scene is rendered, the shadow created by spotlight1 will be read in from disk before render time rather than being auto-generated.

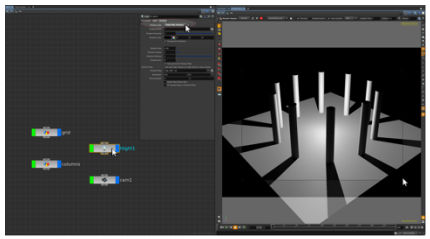
See file **DepthShadowMap_end.hipnc.**

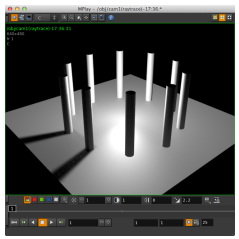
OMNI SHADOWS FROM FAUX POINT LIGHTS

By default a **Point Light** will **render** with **Ray Trace shadows**, creating an **omni-directional shadow**. By default **a shadow map** generated by a point light will only render through the +Z Axis. This means the shadow won't be omni-directional, but directional relative to the orientation of the light.



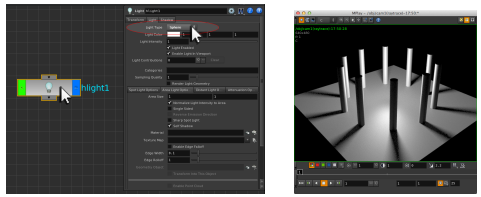
Open the scene omni_shadows_begin.hipnc. When the **scene** is **rendered** with **Ray Traced Shadows** on the **point light**, an **omni directional shadow** is generated, but these shadows will always be sharp. When **Depth Map Shadows are activated**, the **Render Region Preview** still **displays omni-shadows** due to **Houdini** using **ray tracing** to generate shadows in the context of a Render Region Preview.





If however the scene is formally rendered to MPlay, the shadows are only created in the +Z direction of the point light, but can be blurred and softened. Blurring depth map shadows can sometimes however result in render artefacts.

While it is **possible** to **configure a series** of **depth map point lights** for each shadow direction being rendered, using Channel Referencing so that their combined effect is the same as a single point light; a **simpler method** is to **generate omni-shadows** from an **Area Light instead**.

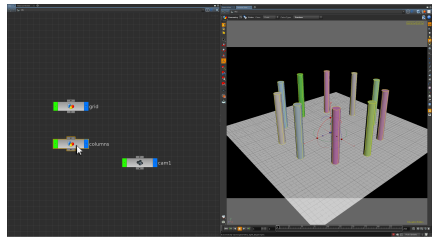


In the **parameters** for the point light, set the **Light Type** parameter to **Sphere**, and reset the **Shadow Type** to **Ray-Traced Shadows**. When the scene is rendered again, soft omni-shadows are generated.

See file omni_shadows_end.hipnc

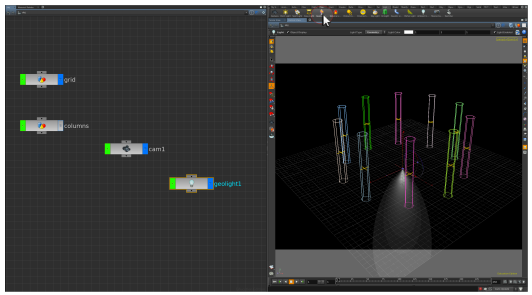
THE GEOMETRY LIGHT

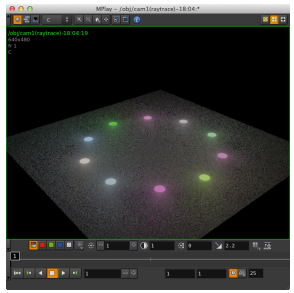
A variation on the **Area Light** is the **Geometry Light**. This can be used to **make an object emit light**, **colour** and **texture information**. **Open the scene geometry_light_begin.hipnc**



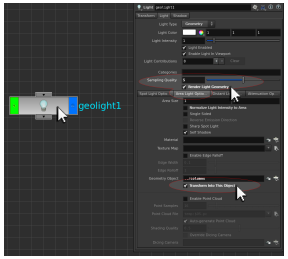
This scene contains some primitive columns on a grid. The columns have been coloured using a **Color SOP** at Geometry Level.

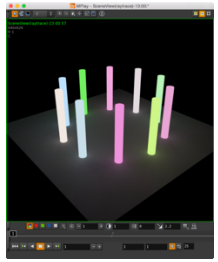
Select the **columns** object in the **Network Editor**, and from the **Lights and Camera Shelf activate** the **Geometry Light button**. By **default**, this will **turn of the display** of the **selected object** so that it does not render. When the **scene is rendered**, the **effect** of the **geometry light columns** can be seen on the grid.





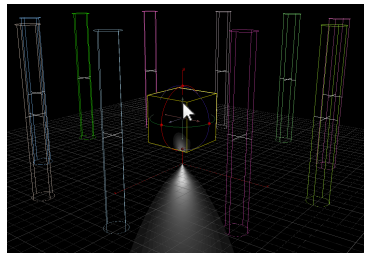
The **parameters** of the **Geometry Light** can also be modified to improve the rendering quality. As with **Area Lights** and **Environment Lights**, this is done using the **Sampling Quality parameter**.

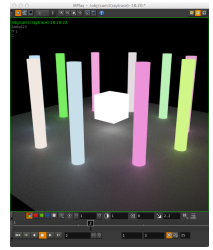




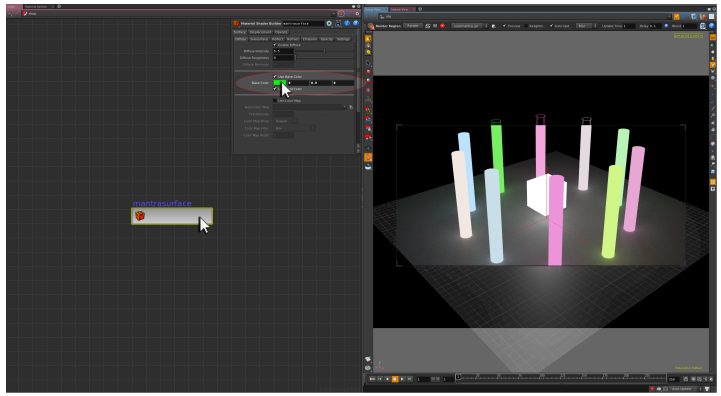
The **columns** can be **seen in the render** by activating the **Render Light Geometry option**. If an **animated object** is used as the **Geometry Light source**, the **Transform Into This Object tick box** can be activated, so the Geometry Light always follows the animation on the source geometry.

In the **Network Editor**, create a **Box Object**, and translate it up so it sits above the grid. From the **Material Palette LMB Drag** and **Drop** and **Mantra Surface Material** onto the **box object**.



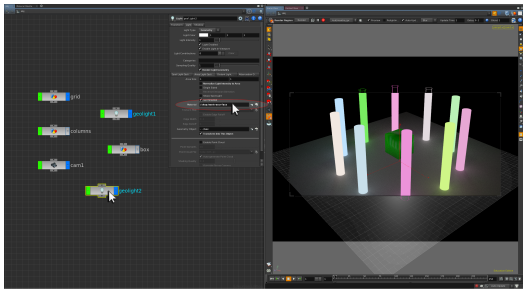


With the **box object selected**, activate it as a **Geometry Light** and set this **second Geometry Light** to **Render the Light Geometry**.

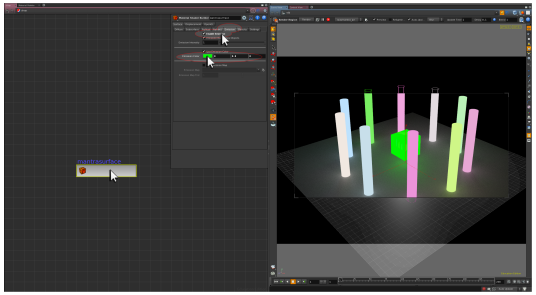


At **SHOP Level**, adjust the **Diffuse Color parameter** of the **Mantra Surface Material**, and **note** that it **does not affect** the **colour** of the **Geometry Light** in the resulting render.

In the **parameters** for this **second Geometry Light**, load the **Mantra Surface Material** into the **Material parameter**, and note that the **box colour** changes to the **Diffuse Color specified**; however the **effect** of the **Geometry Light** is **negated**.



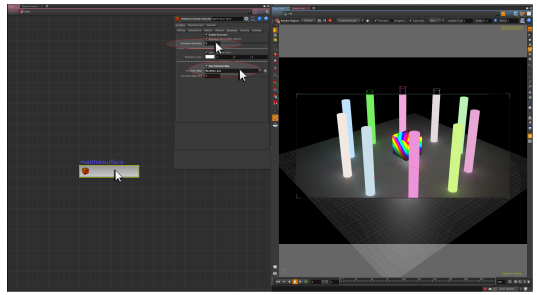
Back at **SHOP Level**, **deactivate** the **Diffuse parameters** for the **Mantra Surface Material**, and **instead enable** the **Emission parameters**. Adjusting the **Emission Color parameter** will cause the **Geometry Light** to **reactivate** relative to the specified **emission colour**.



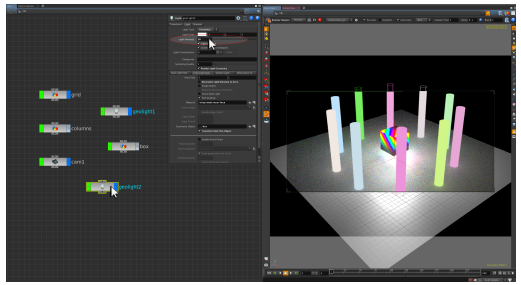
Reset the **Emission Color** parameter to **white**, and activate the **Use Emission Map tick box**. Set the **Emission Map parameter** to:

Emission Map **Rainbow.pic**

This is a Houdini native utility image, useful for quick texturing tests (similar to Mandril.rat).



Increasing the **Emission Intensity parameter** to **5** will **increase both** the **vibrancy** of the **Rainbow.pic image** as well as the **effect** of the **Geometry Light**.



As a final step, return to **Object Level**, and increase the **Light Intensity** parameter of the second **Geometry Light** to **20**. Note that this Intensity parameter **increases** the **intensity** of the **Geometry Light effect** without adjusting the **Emission Intensity** of the **rainbow box**.

See file **geometry_light_end.hipnc**