CS148 Homework 4

Homework Due: Jul 24th **(Thursday)** at 2PM - 5PM (In-Person) or 7PM - 8PM (Remote)

0.1 Assignment Format

This assignment has **4** **TODO** s. Like with the first two HWs, the TODOs are covered within the first few pages of the document. The rest of the handout consists of instructional tutorials.

As a break from last week's coding heavy assignment, this assignment will be all done in the Blender GUI, using Blender Cycles to ray trace your scenes. If you need a refresher on how to (1) enable Blender Cycles, (2) preview its visual results in the viewport, and (3) use it to render your scene as an image, then review the section on Blender Cycles in the HW2 document. Also like HW2, this assignment is quite open-ended – we encourage you to get creative with how you arrange your lights and apply your textures!

0.2 Collaboration Policy and Office Hours

All policies from here on are the same as they were for HW1. See the HW1 document for details.

Quiz 4

You will be randomly asked one of these questions:

- What does it mean for light to be attenuated? What concept do we borrow from Chemistry to model attenuation of light in ray tracing?
- Describe a visual phenomena that a distributed raytracer can easily render, but a Whitted raytracer cannot. How does the concept of sampling come into play for modeling the visual phenomena you described?
- How do we use the UV coordinate space for texture mapping? When might we want to extend UV coordinates past the range of 0 to 1?
- What is the purpose of having different textures for diffuse vs specular? How do these textures get factored into our lighting equation?
- Describe the idea of a normal map. What visual effect does it allow us to create, and how are the normal data stored in the map?

1 Assignment

1.1 **TODO 1**: Render 2 scenes with (1) soft shadows and (2) dramatic lighting.

Blender by default provides 4 different types of lights for you to place into your scene: point, area, spot, and directional lights. There are two ways to add any of these lights to your scene. One option is to add a new **Light** from the **Add** menu. The other is to modify an existing light in your scene in the Properties Editor under **Object Data Properties**. There, you can change your light between all four types.

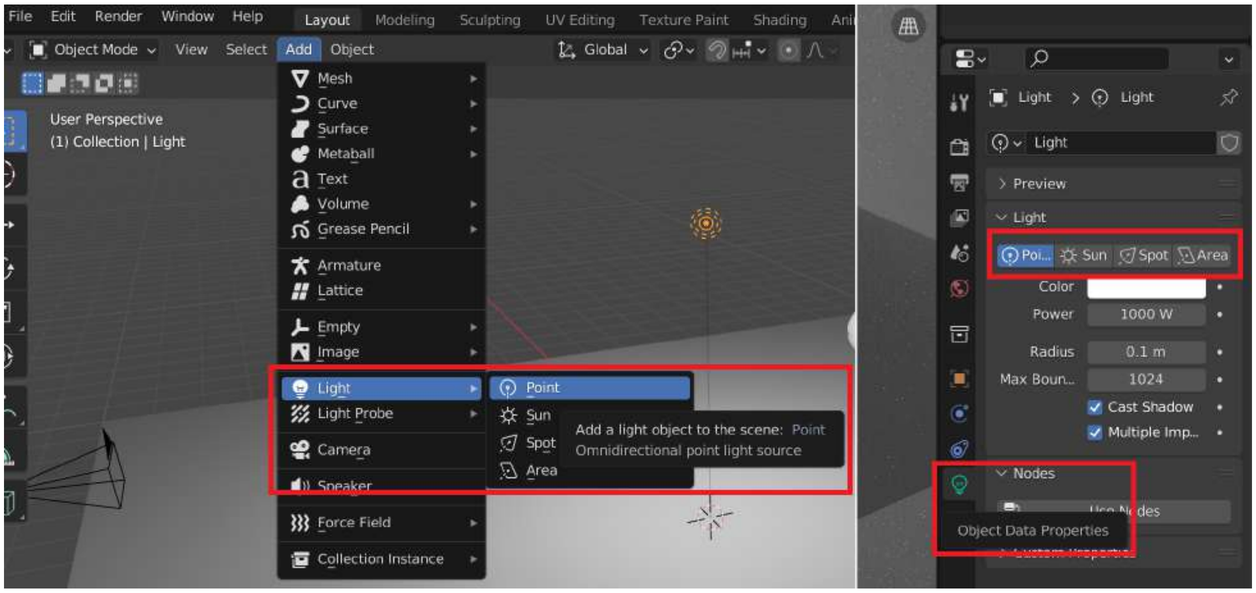


Figure 1: Ways to add a particular light type to your scene.

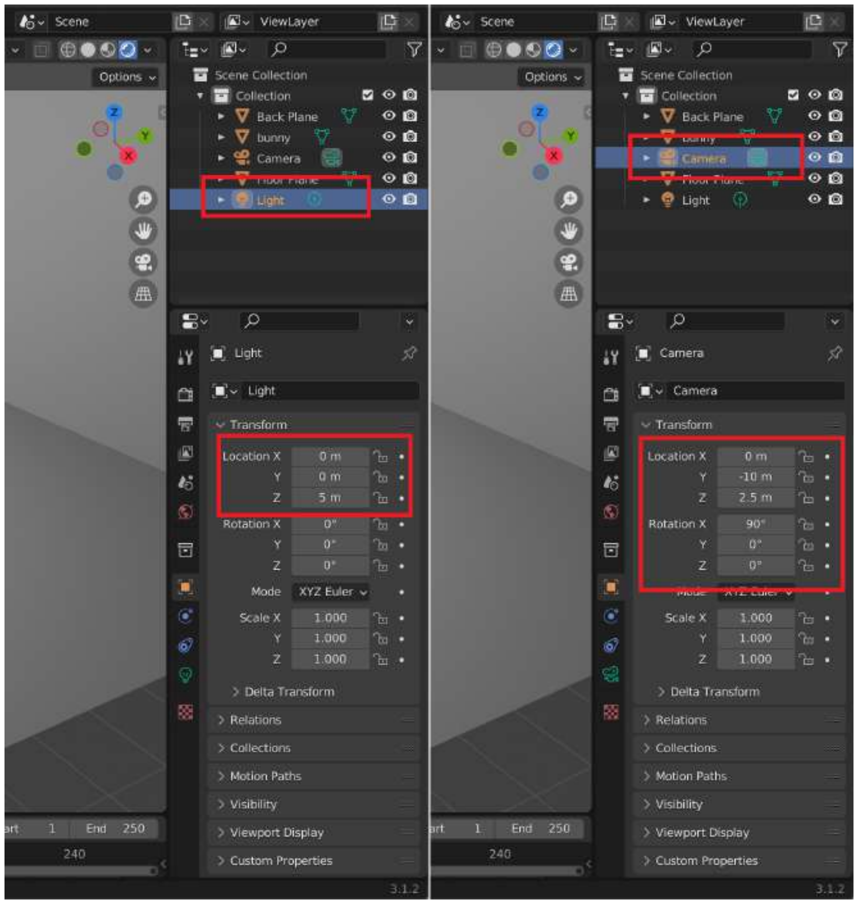


Figure 2: Transforming a light or camera using the Properties Editor. Note that in the case of a point light, only translations matter. Scaling has no effect on either.

Try using the different types of lights to light up your own scene. Use this as an opportunity to get a feel of how each light might come up for what you want to do in your final project. From Figure 3, we see that using a mix of lights can lead to a more natural look with softer shadows and more even illumination.

We want you to build two scenes, each with its own different lighting arrangements. For the first scene, we want you to find an arrangement of lights that give your objects soft shadows (try to get even softer shadow gradients than what's shown in Figure 3). For the second scene, we want you to try to intentionally create darker shadows for a dramatic lighting effect.

When you're satisfied with your two scenes, render and save them as two separate images using Blender Cycles (in an appropriate enough resolution to see the details) for the grading session. You can change the resolution of your output in the **Output Properties** tab of the Properties Editor, though the standard 1920x1080 at 50% should be good enough.

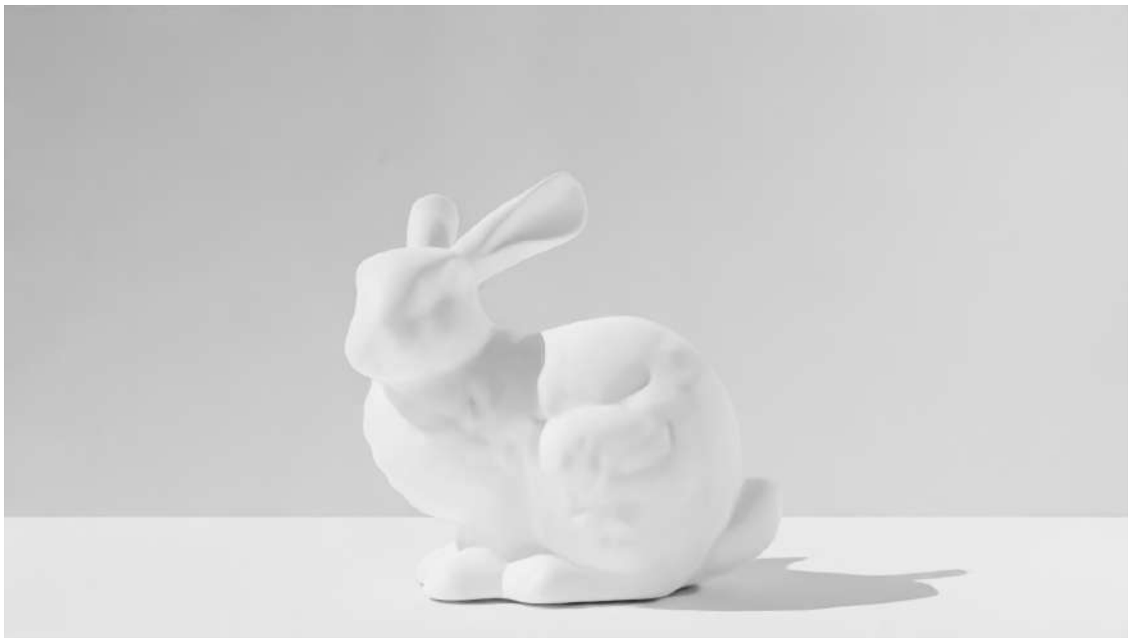


Figure 3: An example of a scene illuminated with all 4 types of light: point, area, spot, and directional lights. We end up with a more natural look with softer shadows and more even illumination than from just using one light type.

Show us: (2 pt) Two ray traced images using Blender Cycles of (1) your first scene with soft shadows and (2) your second scene with dramatic lighting. This part is extremely open-ended – as long as you make a good effort and can back up your lighting choices, you'll get credit!

1.2 TODO 2: Render a scene with environmental lighting.

In lecture, we talked about measuring incoming light and then using it to render synthetic objects in the scene. We can do this in Blender using (1) environment textures, also known as High Dynamic Range Images (HDRIs), or (2) the Nishita sky model. HDRIs are captures of real-world scenes, while Nishita Sky is a sky model that approximates the lighting of the sun and the sky.

Show us: (0.5 pt) A ray traced image using Blender Cycles of your own scene that has a HDRI or Nishita Sky applied and clearly visible.

1.3 TODO 3 : UV unwrap your own custom object.

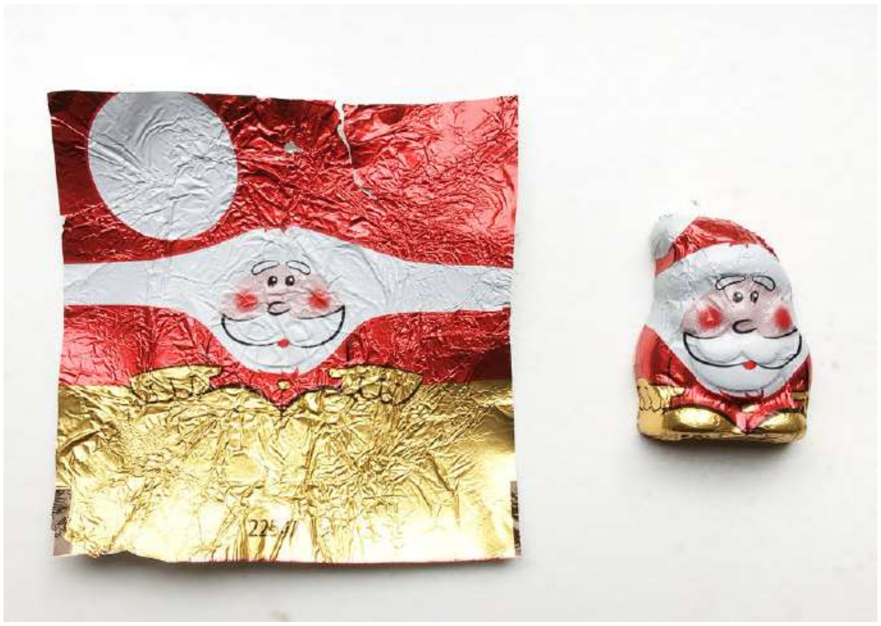


Figure 4: "Unwrapping" a Christmas chocolate. Source: Twitter.

In lecture, we discussed UV mapping (for a mesh) as the process of assigning UV coordinates to each vertex of the mesh. The UV coordinates are then used to sample the texture for the mesh. UV coordinates do not magically appear when we need them though — we have to actually start somewhere and create the UV coordinates from scratch. To get started with learning the process, we suggest watching this official Blender tutorial on UV unwrapping first, then read through the instructional material on UV unwrapping later in this document.

Show us: (0.5 pt) A UV map made by unwrapping an object that you made from scratch. This object should be more complex than a simple primitive or any of the Blender built-in objects. Use this as an opportunity to UV unwrap a custom object you might use for your project!

1.4 TODO 4 : Recreate a reference texture.

As discussed in lecture, texturing is a widely adopted method to add fine-grained detail to an object. There are different kinds of textures for different purposes: base color, normal, roughness, ambient occlusion, displacement, etc. **For this HW, we want you to first find a reference photo or image of an object that has a texture that you find interesting (e.g. maybe the marble patterns of a fancy sink, or the bumpy rock surfaces of a mountain range).**

Then, try to recreate the look in Blender using a combination of texture mapping (e.g. base color, normal, etc). Ideally, you use this as an opportunity to texture an object you'll want to include for your project – we can actually give you feedback on it during grading!

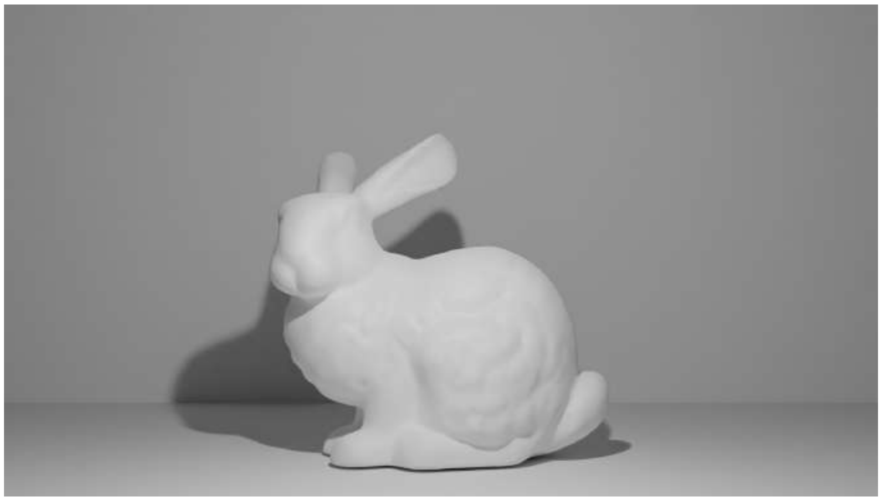
Show us: (2 pt) Your own recreation of a textured object based on a reference photo or image. Have your reference ready, and be ready to explain what types of texture mapping you used. This part is extremely open-ended – as long as you make a good effort and can back up your texturing choices, you'll get credit!

That covers all the TODOs! Everything else in this document consists of instructional (Blender) tutorials for you to reference.

2 Lighting in Blender

2.1 Point Lights

Let's try playing with a point light in Blender. First, we'll need to set up an actual scene. Here are the steps we took to set up the following scene:



1. Import this Stanford bunny model (with the default cube deleted), and translate it along the y-axis by 5. For speed and simplicity, you may just use the Properties Editor for transformations here, e.g. as shown in Figure 2 (for objects too in addition to the light(s) and camera).

2. Add a plane object via **Add** $\rightarrow$ **Mesh** $\rightarrow$ **Plane** , and scale it by 10 in both the x and y directions. This will be the floor plane.

3. Add another plane object, also scale it by 10 in both the x and y directions, then rotate it about the x-axis by 90 degrees, and translate it along the y-axis by 10. This will be the back wall plane.

4. Change the camera transformations from the default numbers to simply $(0, -10, 2.5)$ for the location and $(90, 0, 0)$ for the rotation.

5. Change the default light location to $(0, 0, 5)$. The rotation doesn't matter, since this is a point light (self check: can you answer why?).

6. Toggle to the camera view in the Viewport. Change your Render engine to Cycles, and toggle to the render preview. You should get a similar result to what's shown above.

Alternatively, if you've already begun to make objects for your final project, or just have some spare objects from previous work and experience, then you are very welcome to set up your own scene instead of the above! You may want to toggle the camera view back and forth when arranging your object(s), light(s), and camera. Additionally, you may want to toggle off the render preview and go back to the default solid mode view to lighten the computational load on your computer as you make modifications.

Once you're done setting up your scene, try experimenting with the placement of the point light(s). Remember from lecture that the irradiance (i.e. the light power or amount of light that is hitting an object surface) varies based on the tilt angle of the surface with the light as well as the distance to the light. See Figure 5.

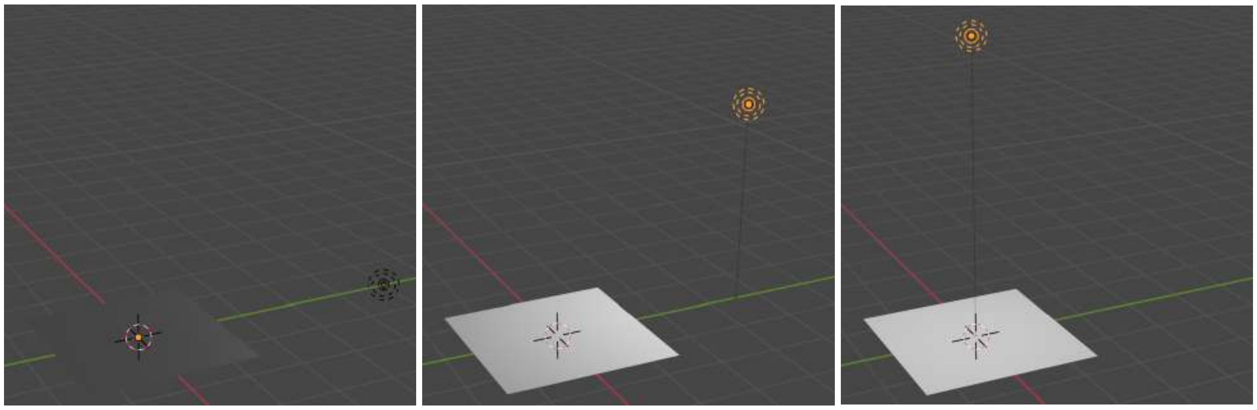
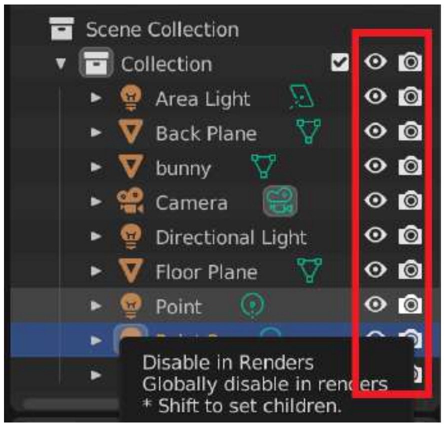


Figure 5: Irradiance varies based on the tilt angle and distance of the surface to the (point) light.

2.2 Area Lights

Now let's try playing with area lights. First, disable your point light(s) from Section 2.1 (using the eyeball and camera icons in the `Scene Collection` tree above the Properties Editor), and add an area light instead. Transform the area light however you see fit for your scene. Recall that the radiance of an area light varies based on the tilt angle between it and the object surface. See Figure 6.



Notice how the area light results in a much less uniform spread of light, as all the light rays are restricted to coming from a particular area. This is in contrast to the point light, where light rays are sent out indiscriminately in all directions around the point. **The larger area of an area light results in softer looking shadows (e.g. less dark contrast) than what might result from a point light** (see Figure 7 vs. Figure 8).

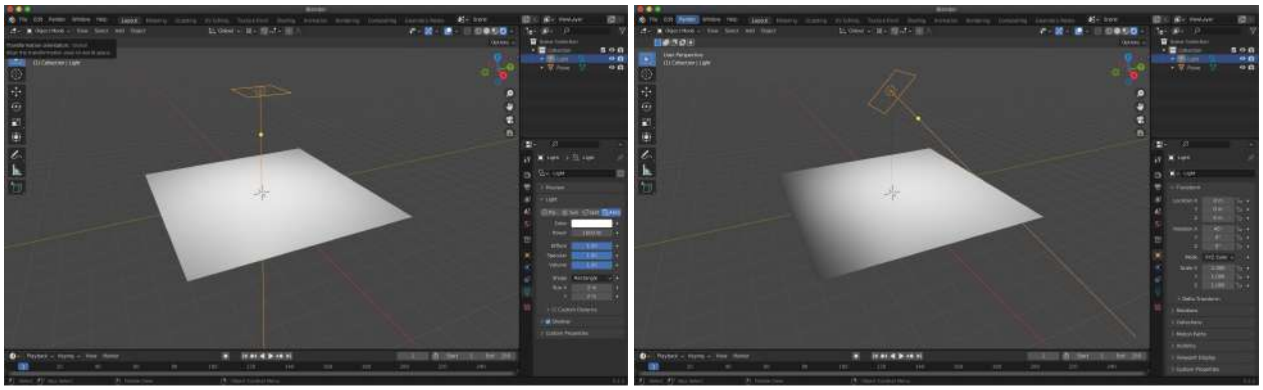


Figure 6: Radiance varies based on the tilt angle and distance of the surface to the (area) light.

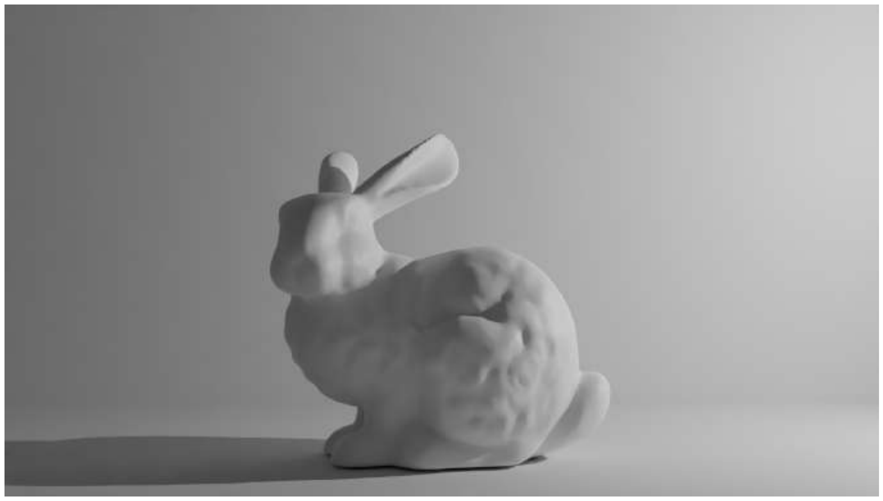


Figure 7: An example of a scene illuminated with only an area light. The light rays are restricted to coming only from a particular area, though having multiple rays going in the same direction can lead to softer shadows than what would be produced by a point light.

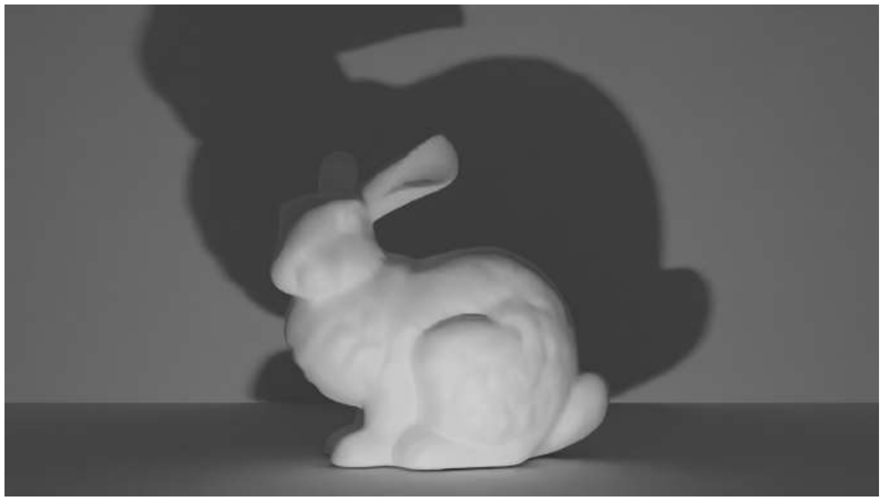


Figure 8: An example of a scene illuminated with only a point light. Point lights tend to produce much harsher shadows with darker contrasts to that of the surroundings, as each direction only gets one ray of light.

2.3 Spotlights

Spotlights model lights that are engineered to emit light rays in a cone-like shape. Imagine lamps or streetlights in real life where the light is often surrounded by some sort of lampshade or outer object designed to concentrate the light. This leads to a light type that aims its light rays at a specific area, like area lights, but still spreads its rays in multiple directions, similar to point lights.

Spotlights are popular in modeling due to how common they tend to be in our everyday lives. They are also able to produce more dramatic lighting compared to a point light or area light. Consider Figure 9 compared to Figure 8. It's much easier to concentrate the light from a spotlight on a particular part of a scene than from a point light, whose light rays go in every direction.

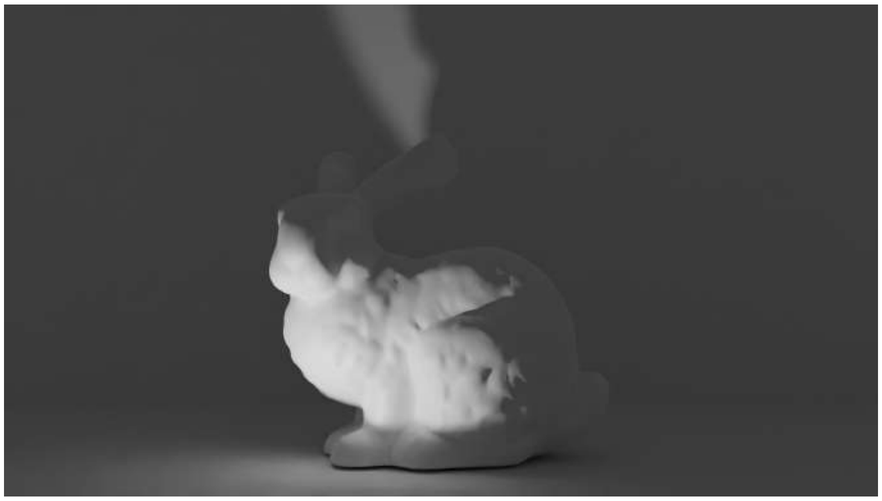


Figure 9: An example of a scene illuminated with only a spotlight for more dramatic lighting. The spotlight excels at concentrating light all into one spot in the scene.

2.4 Directional Lights

The last type of light is the directional light or "sunlight" as Blender calls it. The idea of directional lights is that they don't have their own locations (i.e. no actual positions in world space), but rather, they shoot light rays uniformly across all of space in the same direction. Think of the sun in real life. The sun is so far away from us that all its light rays at any given instant might as well be shining in the same direction. Hence why Blender calls them sunlights.

Mathematically, when we consider directional lights for lighting computations (i.e. the lighting equation), we just ignore any concept of distance and use the same direction for the light vector regardless of where we are in the scene. So unlike what happens with point lights or area lights, an object that's twice as far away as another from a directional light still gets the same amount of light. Only the direction matters for a directional light.

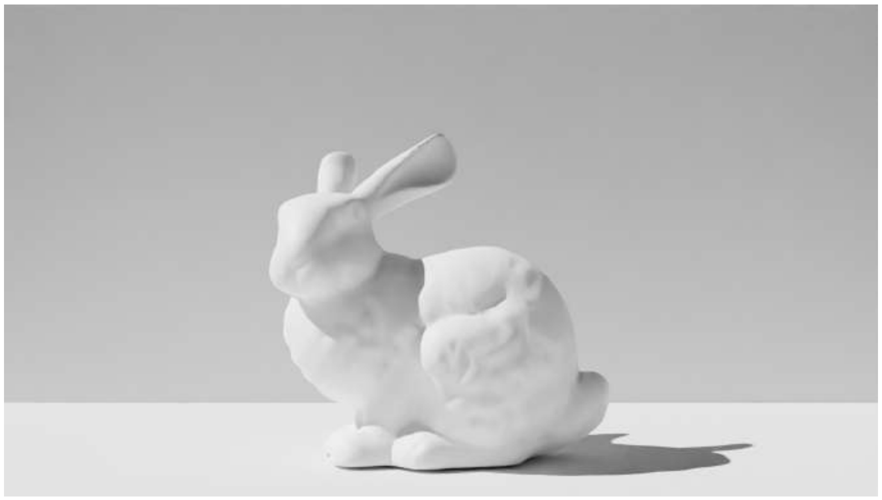


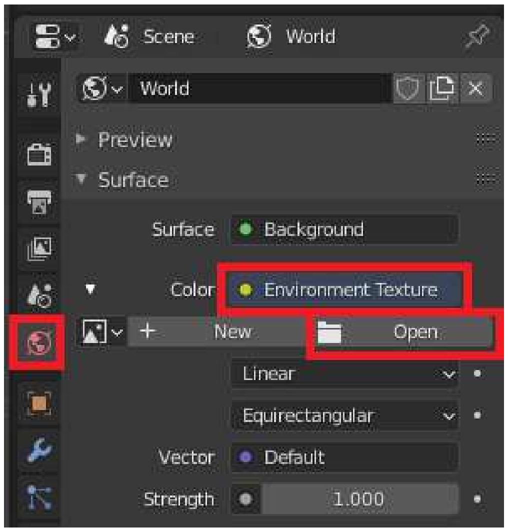
Figure 10: An example of a scene illuminated with only a directional aka "sun" light. Directional lights have no concept of position. Instead, they shine uniformly across all of space in one direction.

3 Environment Lighting

3.1 Environment Texture - HDRI

High Dynamic Range Image, or HDRI, is a type of image frequently used for realistic lighting in 3D environments, usually in EXR format. It is created by combining pictures taken in the same scene with different exposure values, allowing it to contain much more color and illumination information than normal images. It used to be that we needed professional equipment to capture HDRIs, but nowadays you can do it yourself using 360 cameras. There are free CC0 HDRI libraries online, such as the popular HDRI Haven, which has a wide variety of environments from sunny outdoors to indoor studios.

To use HDRI textures in Blender, download an HDRI online (1k resolution is enough for our use) or pick one that comes with Blender (in the installation folder: `Blender 3.5/3.5/datafiles/studiolights/world`). Then, in the Properties Editor, go to `World Properties` and click on the yellow dot in the `Color` field to change the background color to `Environment Texture`. From there, click `Open` to browse to your HDRI. Swap to the render preview (with Cycles enabled) to see your HDRI in effect.



Surface: Background
Color: Environment Texture
Linear
Equirectangular
Vector: Default
Strength: 1.000

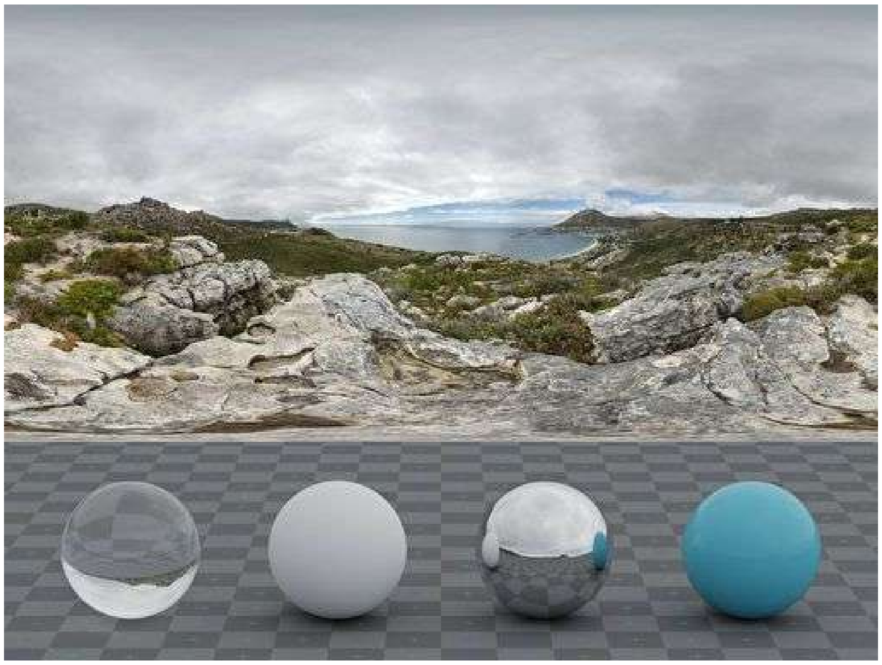
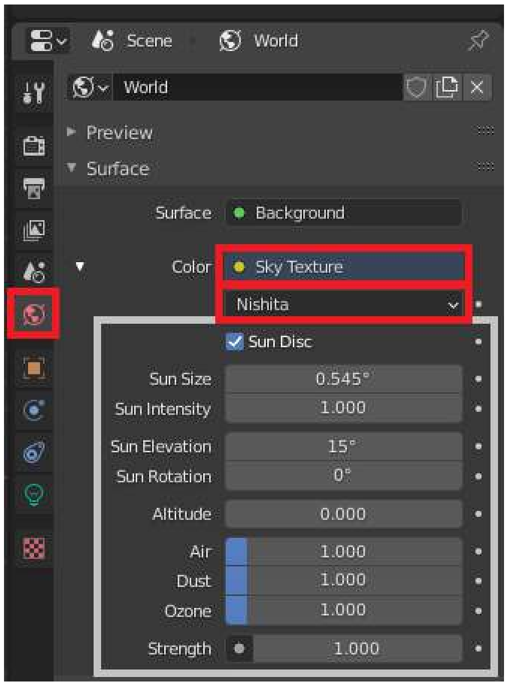


Figure 11: A scene rendered with Blender Cycles using a HDRI Haven texture.

3.2 Nishita Sky

Nishita Sky is a recent addition to Blender starting from version 2.9. It is an improved version of the sky model proposed by Nishita et al. in 1993. This model lets you control the sky texture with only a few intuitive parameters. You can read more about the Nishita sky model here and see a video demo in Blender 2.9's release notes.

In Blender, we can use Nishita Sky by setting the `Color` field to `Sky Texture` in `World Properties`. Play with the settings to get the sky you like. Note though that if you want a more detailed sky background (e.g. with clouds or buildings), then you might prefer a HDRI texture.



Surface	Background
Color	Sky Texture
	Nishita
	✓ Sun Disc
Sun Size	0.545°
Sun Intensity	1.000
Sun Elevation	15°
Sun Rotation	0°
Altitude	0.000
Air	1.000
Dust	1.000
Ozone	1.000
Strength	1.000

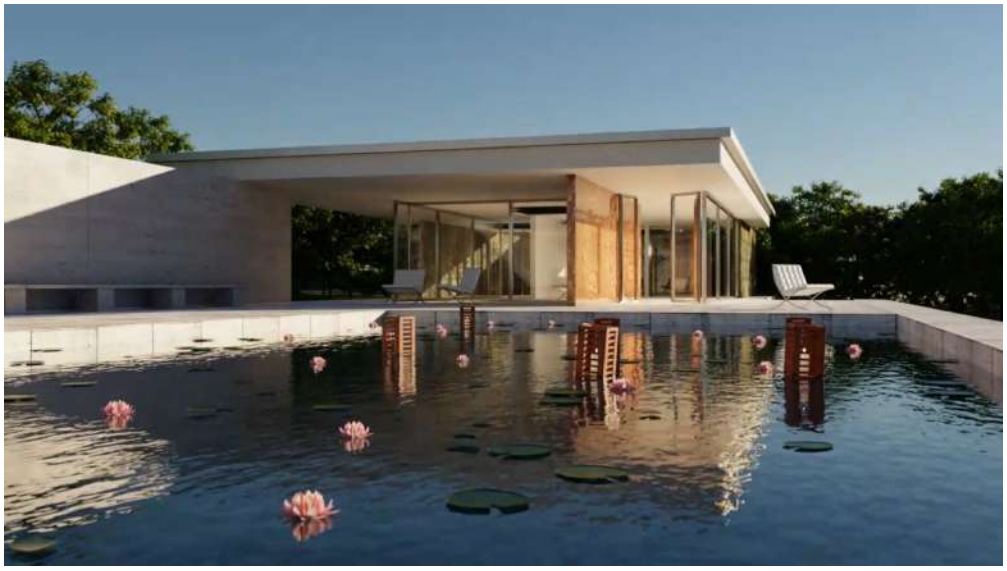
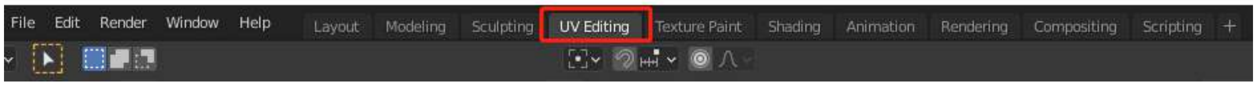


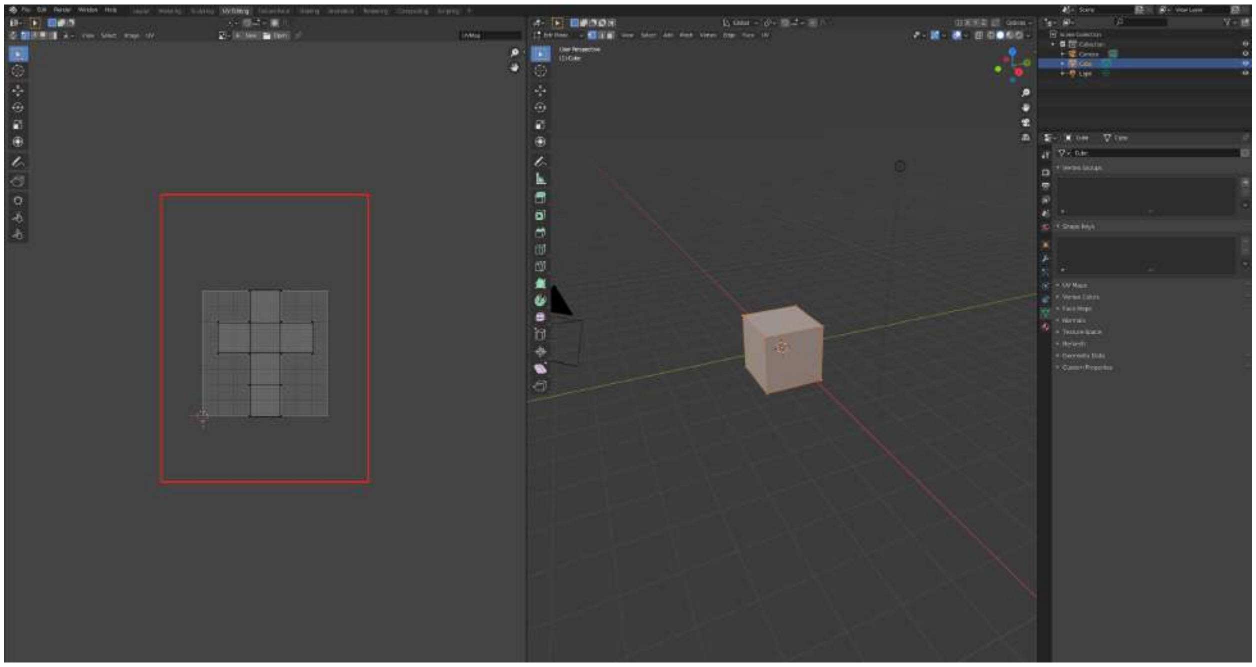
Figure 12: Nishita Sky Texture for Pavilion Barcelona. Source: Blender Wiki.

4 UV Unwrapping

4.1 Visualizing a UV Map

The first step to UV mapping is do a process called UV unwrapping (of an object mesh) to generate a UV map. Let's visualize what a UV map looks like in Blender with a simple example. Open a new Blender project and select the default cube. Then, toggle to the UV map editor view by clicking on **UV Editing**. You should be able to enter **Edit Mode** and see the UV map of the default cube object on the left side of the screen.



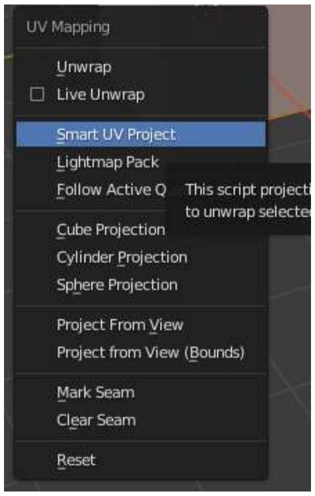


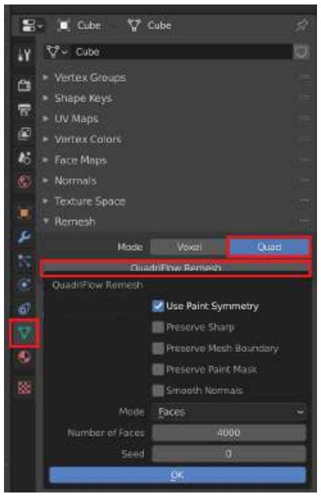
Notice how the cube "unwraps" its surface area into 2D.

4.2 Generating your own UV Map

Now let's generate a UV map for your own object. Open your Blender project from HW7 and select the object you created yourself. For demonstration purposes, we will use a simple stool object for the example in this HW. Toggle to the UV map editor view, select all the vertices of your object, and press **U**, then click on **Smart UV Project**. You will only UV unwrap what you've selected. If you accidentally deselected an object in **Edit Mode**, then you can press **A** with the 3D Viewport in focus to select all its vertices.

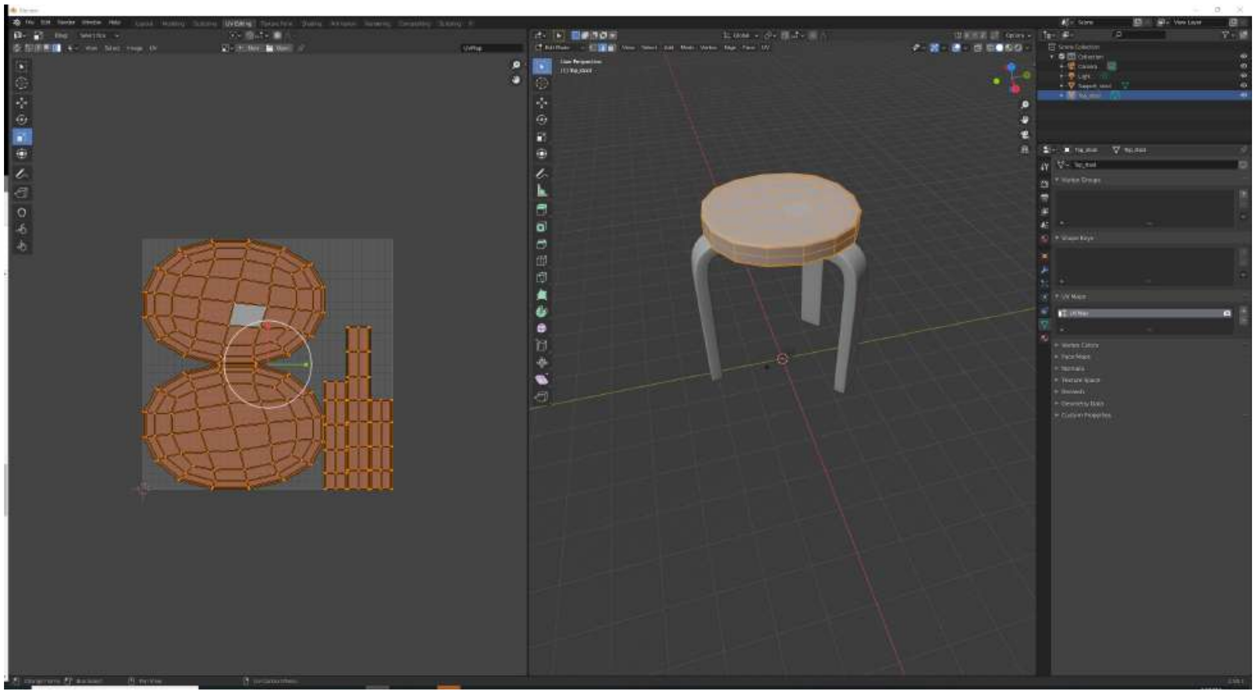
Note: If your object does not have a clean topology (especially if you sculpted it with dynamic topology), then you may need to tidy up the mesh before unwrapping using retopology tools such as quadriflow from **Object Data Properties → Remesh → Quad → Quadriflow Remesh** in the Properties Editor.



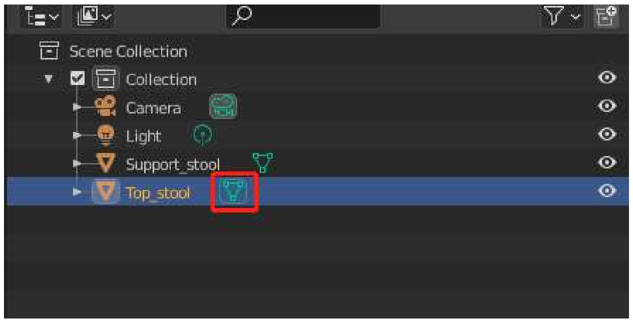


Mode Voxel [✓ Quad]
QuadriFlow Remesh
[✓] Use Paint Symmetry
Preserve Sharp
Preserve Mesh Boundary
Preserve Paint Mask
Smooth Normals
Mode: Faces
Number of Faces: 4000
Seed: 0

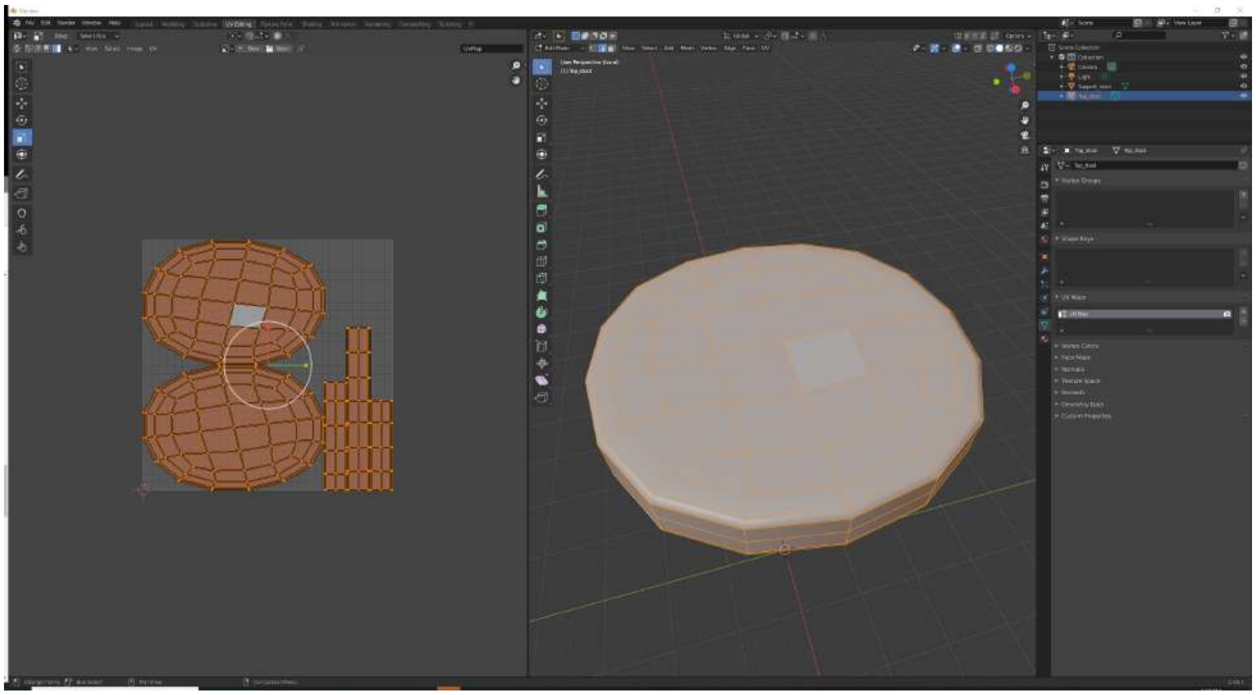
You should be able to see an auto-generated UV map for your object, like so:



Note: You will likely UV unwrap multiple objects in the same .blend file as you work on any project. If you want to quickly switch between them when editing, then you can click on the mesh icon next to your target object in the outliner:



On the flipside, you can also use “local view” to hide non-selected objects so that they don’t get in the way of your editing. Select an object and press / (the forward slash on the keyboard) or go to **View → Local View → Toggle Local View** to toggle to local view. In our case, we did this for the top part of the stool, thus hiding the leg parts from view.



4.3 Editing a UV Map

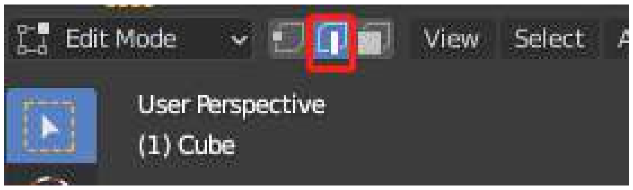
Sometimes the auto-generated UV map of a complex object will be far from perfect. Each connected part of the unwrapped mesh is called a “UV Island”. A good UV map should maximize its area covered by the islands with small distortion (i.e. little stretching) and a reasonable amount of empty space between the islands. There are ways to improve a UV map though if it is not what we want. We’ll explore two such techniques here. For a full list of UV editing tools in Blender, you can check the official manual here.

4.3.1 Adding UV Seams

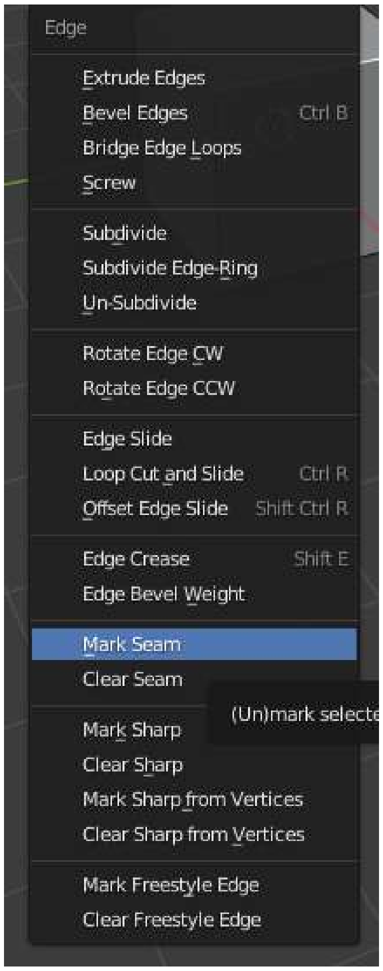
Just like in sewing, a seam is where the ends of the image (cloth for sewing) are sewn together. In UV unwrapping, the mesh is unwrapped at the seams. Think of this method as peeling an orange. You make a series of cuts in the skin, then peel it off. You could then flatten it out, applying some amount of stretching. The cuts are the same as seams.

Smart UV Project creates seams automatically, but we can specify the seams to have finer control over the unwrapping process. Generally, artists place seams at sharp edges, or places where non-overlapping textures are less noticeable (e.g. the back or the bottom of an object). Faces with edges not marked as seams will strictly adhere to each other. Sometimes we might want to add seams to separate a flat surface so the mesh could be better unwrapped. You can read more about seams in Blender in the official manual here.

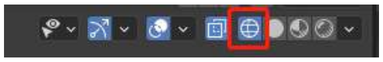
Let’s try adding a seam to your object. In the 3D Viewport, select edge selection mode via the following icon:



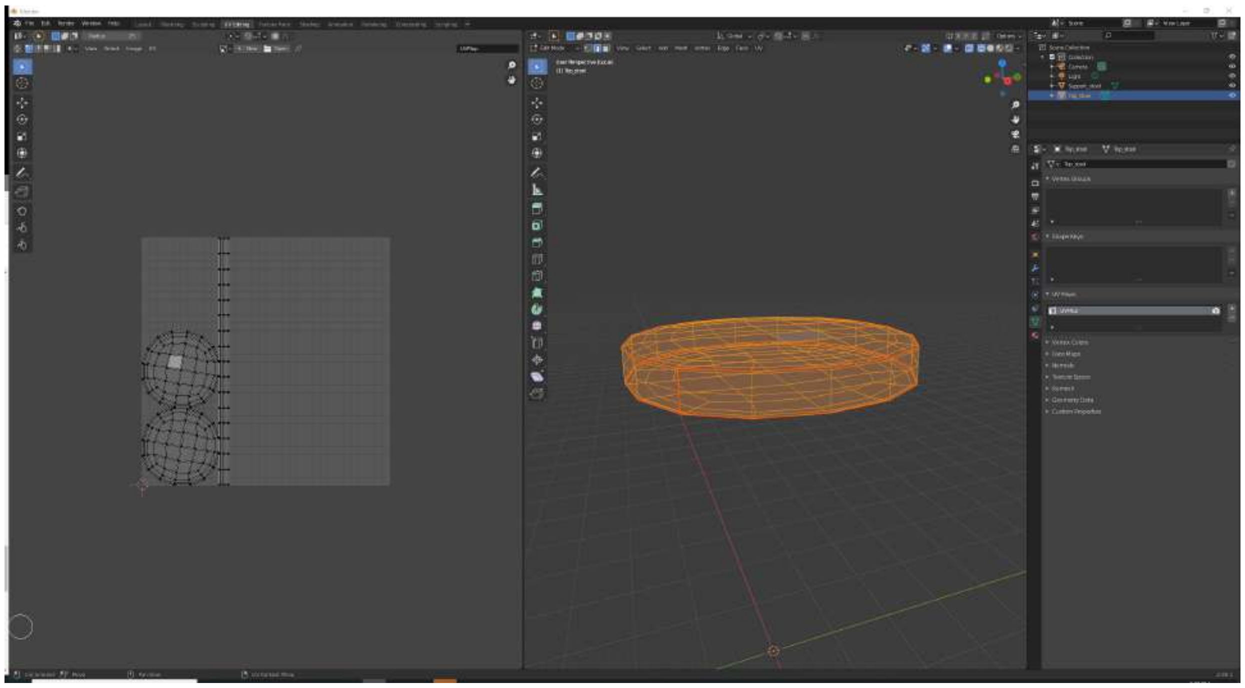
Hold shift, then start clicking to select all the edges that you want to mark as seams. When you're done selecting, press **Ctrl/Cmd + E** for the **Edge** menu and click **Mark Seam**. Seams will be marked in red in the 3D Viewport.



You may also want to toggle to the wireframe view to better inspect the object's edges:



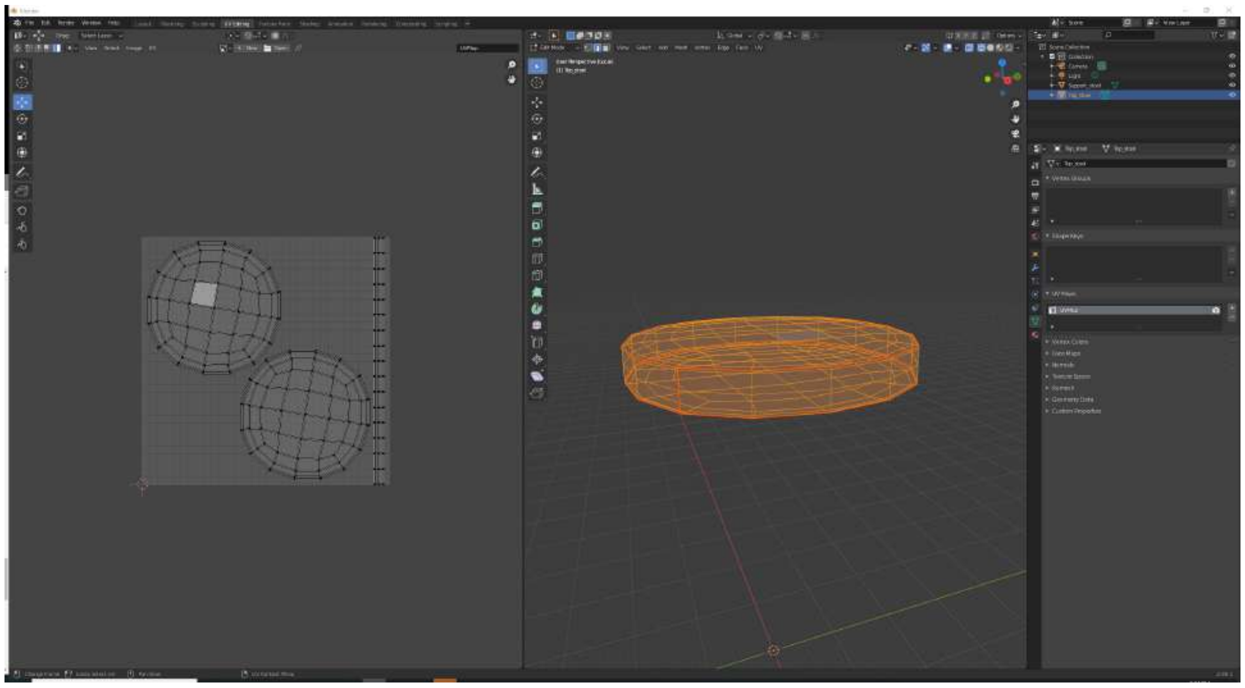
After adding seams to an object, press **U** and select **Unwrap**. You should be able to see a new UV map with the seams you specified.



4.3.2 Directly Editing the UV Map

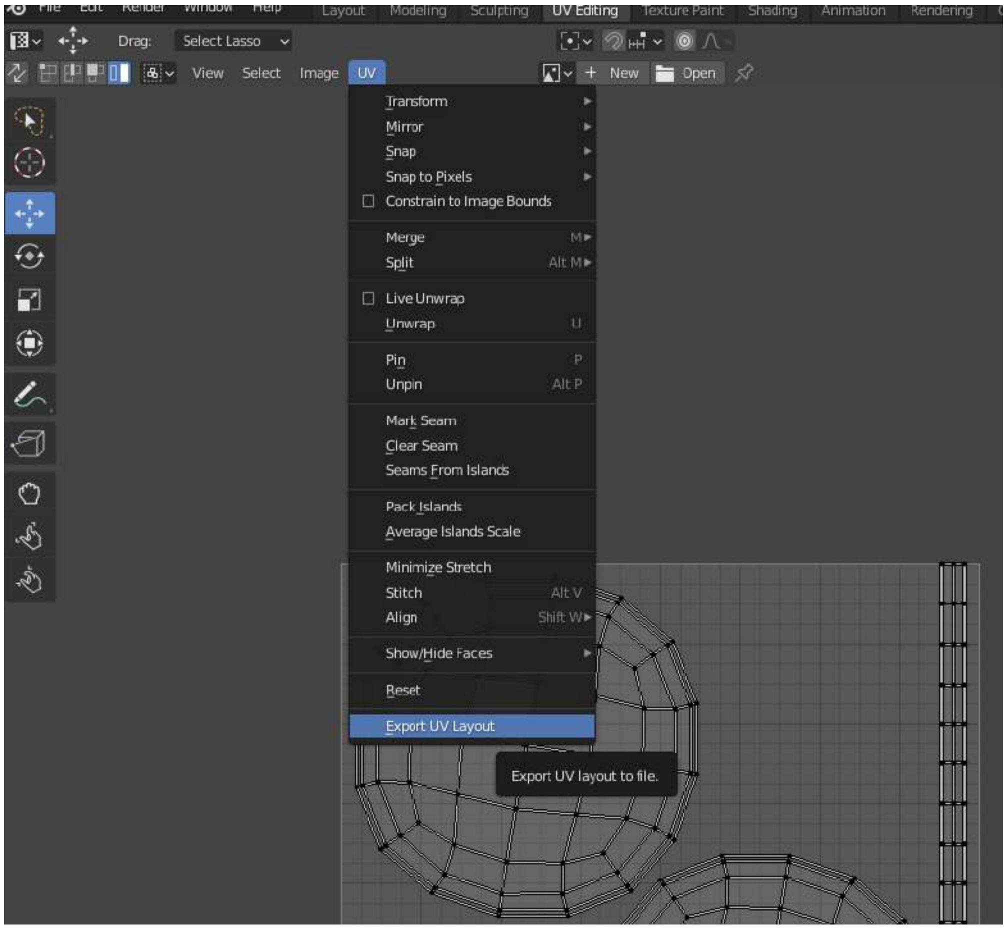
We can also directly edit the UV map. There are four different selection modes on the top left of the UV editor: `Vertex, Edge, Face, Island` modes. You can try whatever is suitable in your case and move, rotate, and scale the selected target. For instance, if you are in vertex mode, then you can move the selected vertices in the UV editor.

We show a better UV map of the stool top below after editing. A larger area of the square is now covered by the unwrapped meshes. This will allow more pixels on the texture to be mapped onto the object, giving us better detail.



4.3.3 Saving the UV Map

You can save your edited UV map through the **UV → Export UV Layout** menu, so that you can create textures with it later.



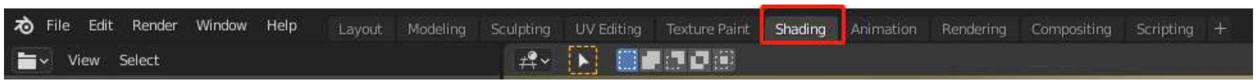
5 Texturing

For demonstration purposes, we will apply a rock ground texture to a sphere. You can download these example textures (e.g. 2k resolution diffuse maps, normal maps, and displacement maps) from this popular website. If you feel comfortable making your own textures, then you can also do that and apply them to your models.

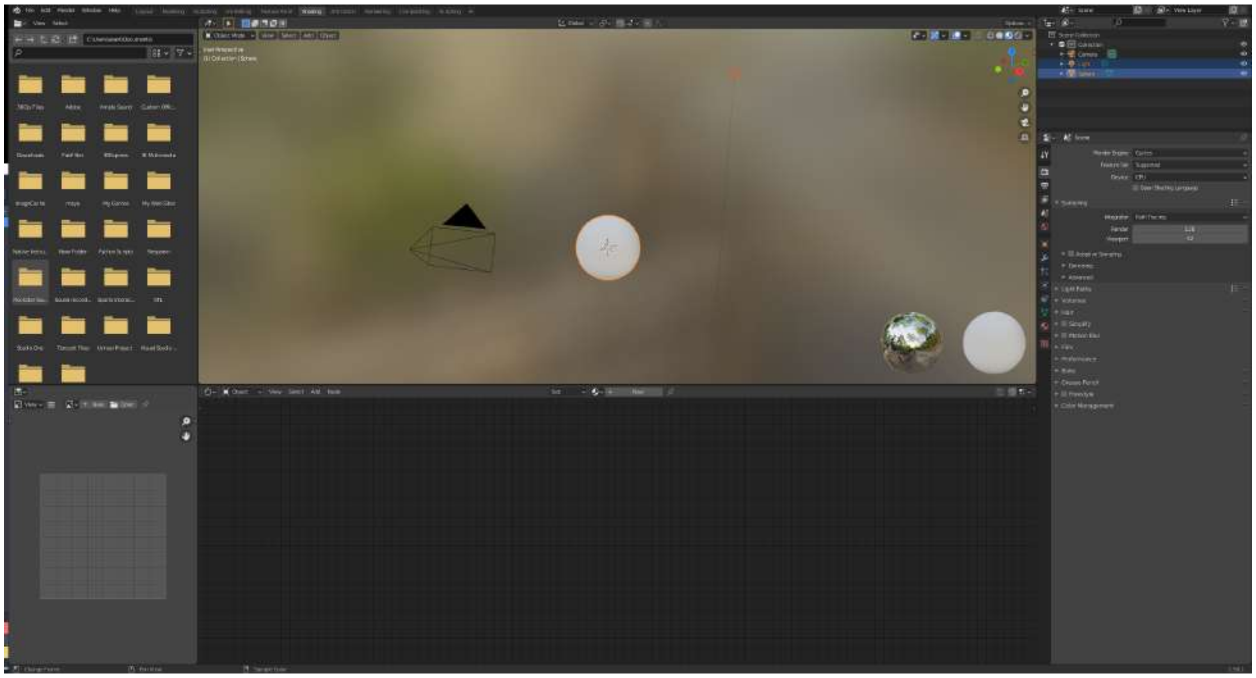
Note: Texturing an object requires that the object has an associated UV map. So make sure to complete the steps in the last section first on UV Unwrapping.

5.1 Base Color

Open a new Blender project, replace the default cube with a UV sphere, and enable smooth shading as well as Blender Cycles for ray tracing. Now open the Shading Editor via the **Shading** tab:

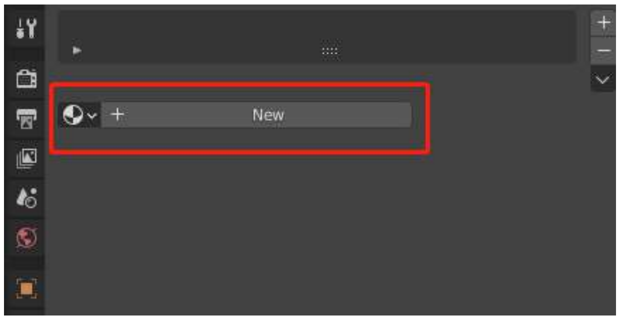


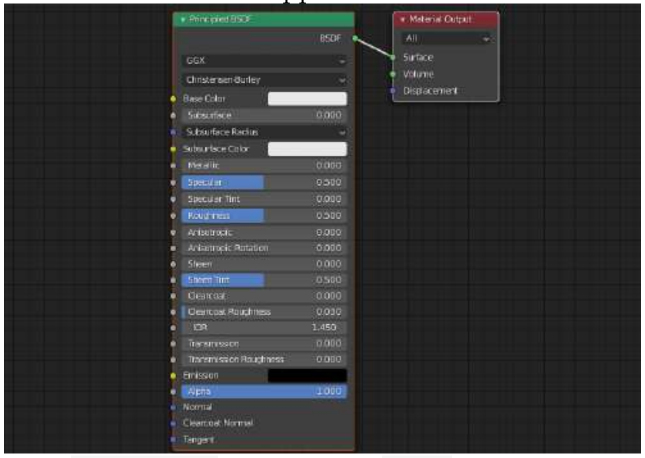
It may take a few seconds to load, but you should see an interface like the following:



The Shader Editor is used to edit materials. Materials used by Cycles are defined using a node tree. In HW4, we played with materials in the Properties Editor as a list of parameters, but under the hood, the materials are actually handled via a tree structure in the **Shading** tab. In the Shader Editor, you can create and edit the node tree more easily. You can read more about it in the official Blender manual.

Add a new material for the sphere, and a new material node will appear in the Node Editor:

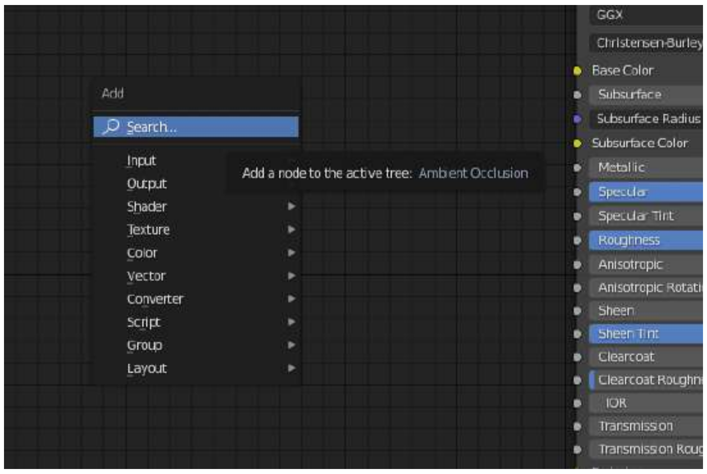


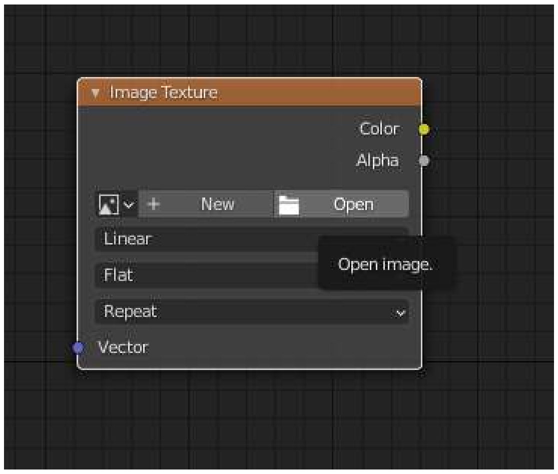


Principled BSDF
BSDF
GGX
Christensen-Burley
Base Color
Subsurface 0.000
Subsurface Radius
Subsurface Color
Metallic 0.000
Specular 0.500
Specular Tint 0.000
Roughness 0.500
Anisotropic 0.000
Anisotropic Rotation 0.000
Sheen 0.000
Sheen Tint 0.500
Clearcoat 0.000
Clearcoat Roughness 0.030
IOR 1.450
Transmission 0.000
Transmission Roughness 0.000
Emission
Alpha 1.000
Normal
Clearcoat Normal
Tangent

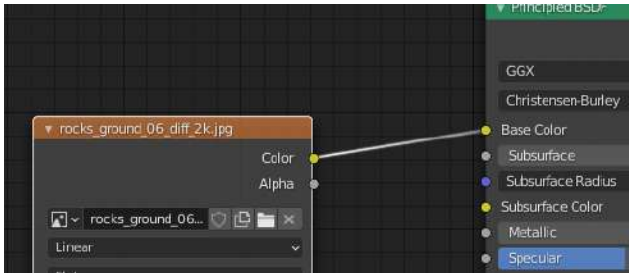
Material Output
All
Surface
Volume
Displacement

Go to the menu bar of the Shader Editor or press **Shift + A** to open the **Add** menu. Search for “image texture node” or go to **Texture → Image Texture** to get an image texture node. Click anywhere to add it to the graph.

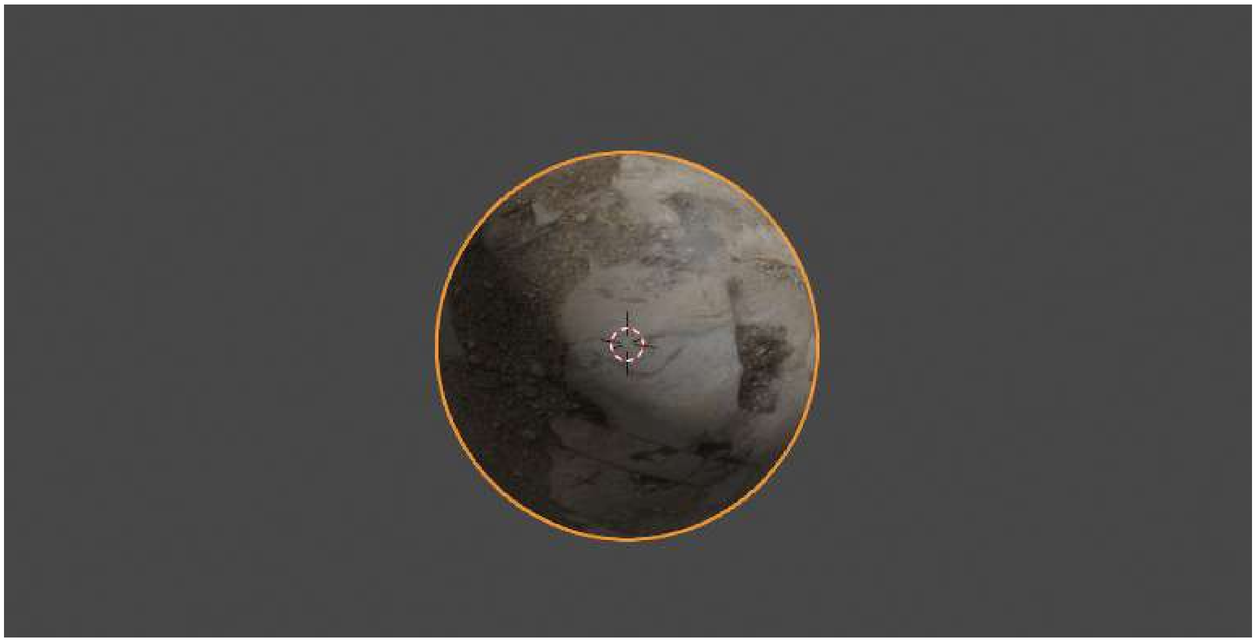




Click **Open** and select the downloaded base color texture **rocks_ground_06_diff_2k.jpg** . Then, connect the image texture's **Color** channel to the material's **Base Color** . You can also directly drag an image from your file explorer or finder window into the node editor to create an image texture node.



Now if you toggle the 3D viewport to the render preview, then you should see a textured sphere as follows. Move around in the viewport (e.g. with your mouse) to see what the sphere looks like. You may notice a seam on the sphere, which is from the UV map. If you're interested, then you can also modify the UV map and see how the texture changes accordingly.

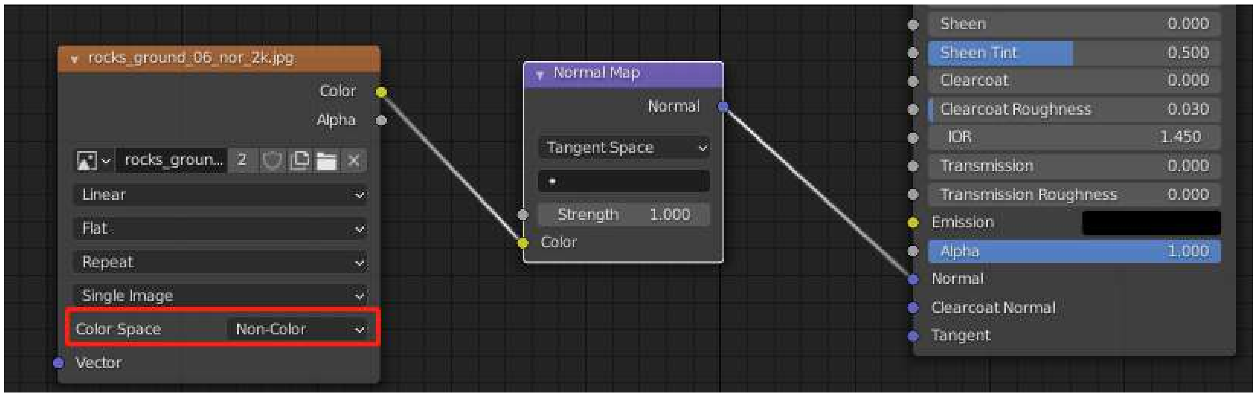


5.2 Normal Map

Normal mapping is a texture mapping technique used for faking the lighting of bumps and dents. Keep in mind that normal mapping will only change the shading of the object (i.e. how light interacts with the normals of the object), not the actual geometry of the object.

Working off the textured sphere from the last subsection, add a new image texture node, then a normal map node (`Add` → `Vector` → `Normal Map`) to the node tree. Select `rocks_ground_06_nor_2k.jpg` for the image texture node. Since the pixel value of a normal map does not represent actual colors, but rather it stores the XYZ values of normal vectors (in the RGB channels of the image), we should set the `Color Space` of the image texture node to `Non-Color`.

Connect the `Color` channel from the image texture node to the normal map node, and the `Normal` channel from the normal map node to the Principled BSDF node like so:



rocks_ground_06_nor_2k.jpg
Color
Alpha
rocks_groun... 2
Linear
Flat
Repeat
Single Image
Color Space Non-Color
Vector

Normal Map
Normal
Tangent Space
Strength 1.000
Color

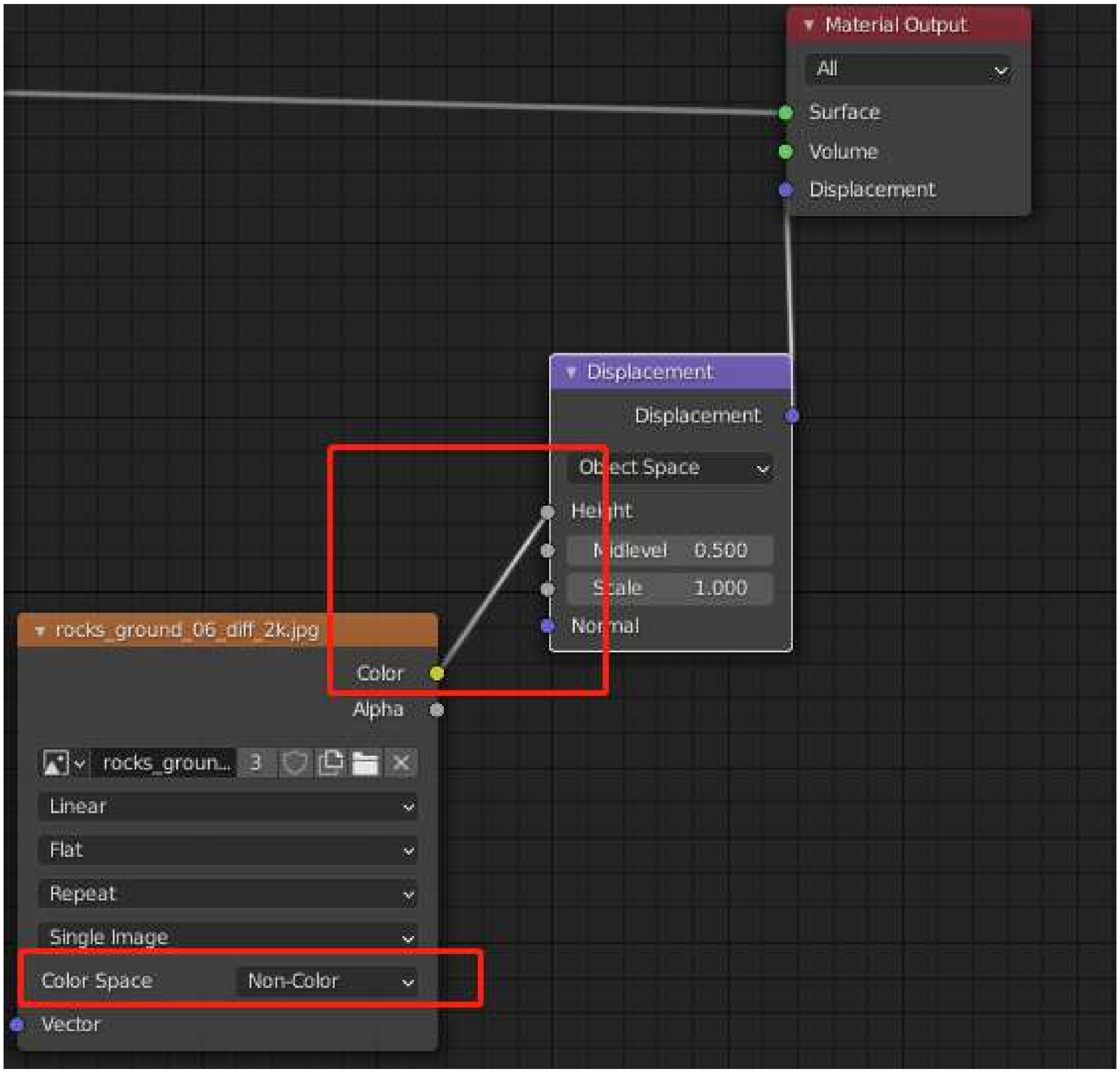
Sheen 0.000
Sheen Tint 0.500
Clearcoat 0.000
Clearcoat Roughness 0.030
IOR 1.450
Transmission 0.000
Transmission Roughness 0.000
Emission
Alpha 1.000
Normal
Clearcoat Normal
Tangent

Your rocky sphere should now have a sense of depth to it:

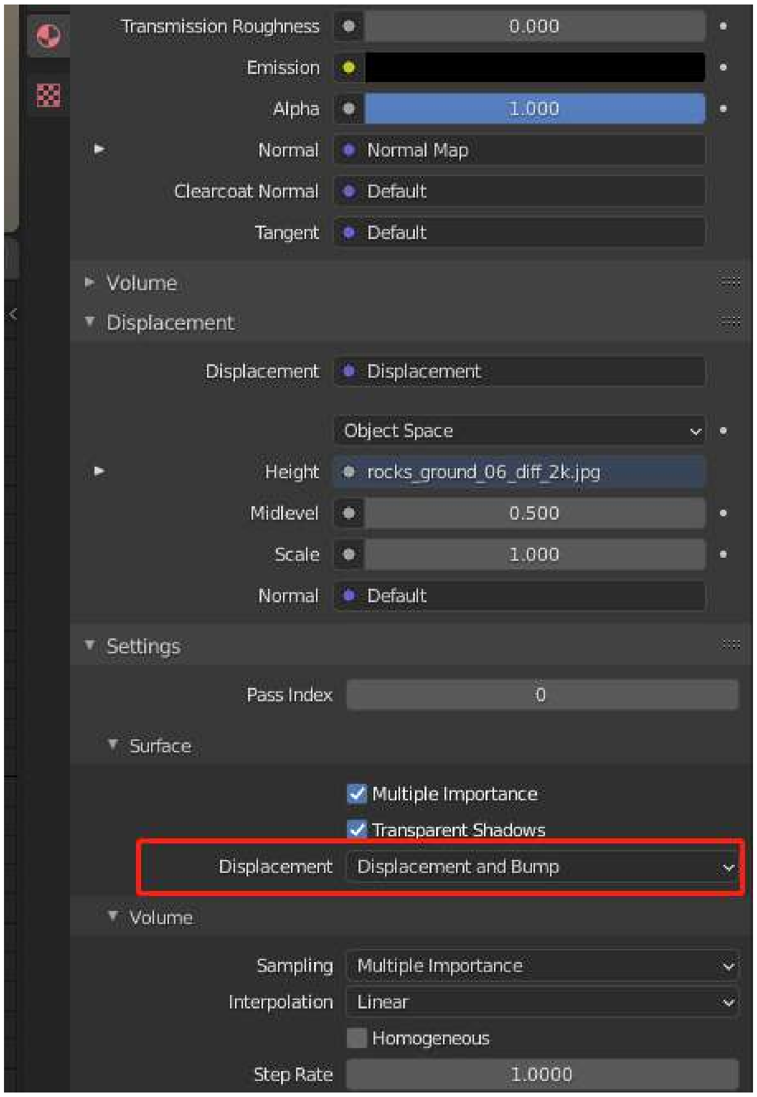
5.3 Displacement Map

Displacement mapping is an alternative texturing technique that uses a texture map to modify the actual geometry of the object in addition to its normals. Any vertex that gets covered by the displacement map texture actually has its position (and local surface normal) modified or displaced (hence "displacement" mapping) based on the values stored in the texture.

Working off the previous subsection, add another image texture node along with a displacement node (**Add** → **Vector** → **Displacement**). Select **rocks_ground_06_disp_2k.jpg** as the image for the image texture node. Set the **Color Space** to **Non-Color** as before. Connect the **Color** channel of the image texture node to the **Height** channel of the displacement node, then the output of displacement node to the **Displacement** channel of the **Material Output** node (not the Principled BSDF node!).

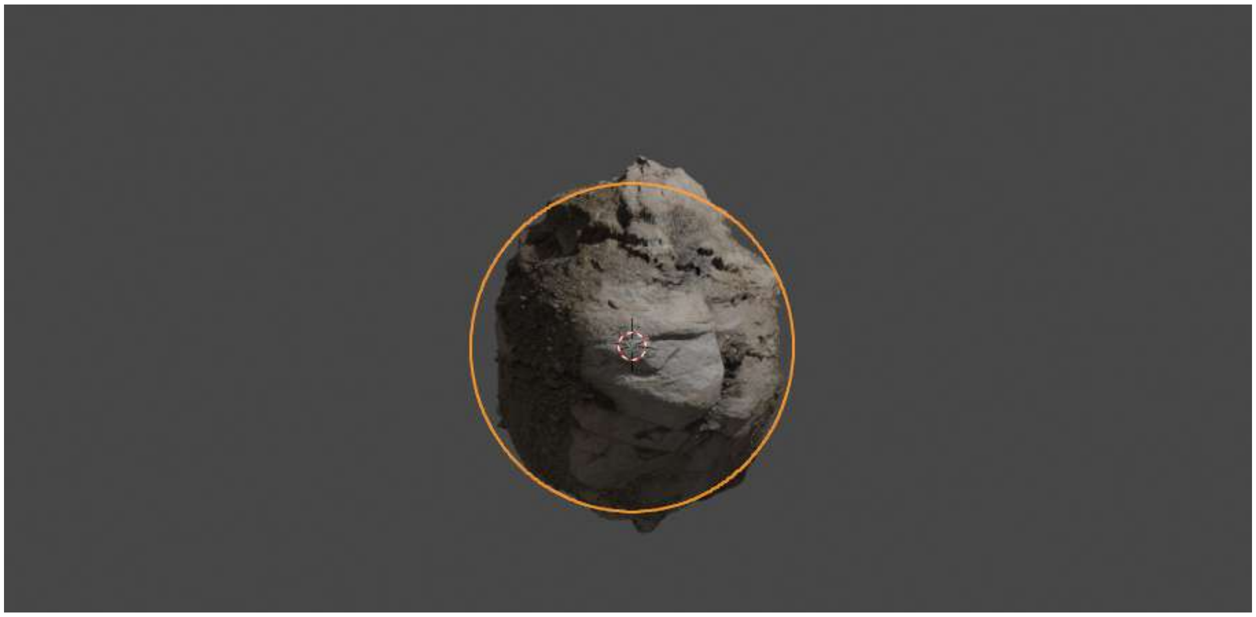


Then in Material Properties , set Displacement to Displacement and Bump .


Transmission Roughness 0.000
Emission
Alpha 1.000
Normal Normal Map
Clearcoat Normal Default
Tangent Default
Volume
Displacement
Displacement Displacement
Object Space
Height rocks_ground_06_diff_2k.jpg
Midlevel 0.500
Scale 1.000
Normal Default
Settings
Pass Index 0
Surface
✓ Multiple Importance
✓ Transparent Shadows
Displacement Displacement and Bump
Volume
Sampling Multiple Importance
Interpolation Linear
Homogeneous
Step Rate 1.0000

Since displacement mapping perturbs the actual mesh vertices, and the default UV sphere has a low resolution, we need to add a subdivision modifier. As a reminder from HW7, go to the Properties Editor, then Modifier Properties and add a Subdivision Surface modifier. Set the Levels Viewport to 3.

You should see that the sphere looks very bumpy now due to the displacement map being applied to the subdivided surface. You can change the Scale value of the displacement node to fine-tune the amount of displacement. With a bit of scaling, you can get an image like the following.



6 Texture Resources

As you might have noticed, to get realistic materials, you need to apply multiple types of texture maps. Base color aka diffuse maps can often just be any high resolution image that you find online, but getting the other maps (like normal and displacement maps) can be tricky. Here are some websites that provide free textures sets:

- TextureHaven
- Poliigon
- CC0 Textures
- 3D Textures
- texture.ninja

If you don't find anything that fits what you're looking for, then you can also create your own maps from a diffuse map. You can use image editing software such as Photoshop, Gimp, or Krita to create the other maps. There are also dedicated tools, such as CrazyBump, ShaderMap, and Materialize. There is also this nice web app as well as a way to create these maps in Blender itself.

Some other topics:

- If you want to start from scratch, then you might find useful this tutorial on how to make all the maps from a photograph.
- For tiling textures without repetition in Blender, there's a handy node group created by Blender Guru, explained in this video.
- If you want to explore advanced texturing, then you can try out the industry-standard texturing software Substance Painter, which has a free educational license.

Here's a comprehensive list of all the Blender-related resources: awesome-blender.