# 3D TEXTURE MAPPING

*Nicola Frachesen, Richard Li, Natalie Rumak, Xiaoxi Zheng*

---

Basic Structure of Presentation:

- Introduction to 3D Texture Mapping
- 3D Texture Mapping Techniques
    - Bump Mapping
    - Normal Mapping
    - Displacement Mapping
    - Relief Mapping
- Maya Demo
- Other Techniques
    - Parallax Mapping
    - Horizon Mapping

Shortcuts:

- Nicola ➪ Ni
- Richard ➪ R
- Natalie ➪ Na
- Xiaoxi ➪ X

# START OF PRESENTATION

Slide 1 (Title): *Various*
Ni: Hello, everyone. My name is Nicola…
Na: I'm Natalie.
X: I'm Xiaoxi.
R: And I'm Richard.
Ni: We are the Texture Meowpers! Today, we'll be taking a look at 3D texture mapping.
R: We're going to talk a bit about the what, why, and how of 3D texture mapping - the various techniques that are used, et cetera.
Na: Also, we'll be presenting a number of visuals and live demos to show everyone how it works.
X: And if we have time at the end, we'll talk about other 3D mapping techniques and current research.
Ni: So let's get started! [transition into next slide]

---

# INTRODUCTION TO 3D TEXTURE MAPPING

Slide 2 (Introduction to 3D Texture Mapping): *Nicola*

So what is 3D texture mapping anyway? 3D texture mapping refers to a number of different techniques used in 3D computer modeling to give flat 2D surfaces the illusion of depth. But why is it that we need to create this illusion of depth? 3D models are already, as their name suggests, three dimensional, so why do we need to bother trying to make models look like they have depth, when we can just give them depth? Also, how exactly can a simple texture give the illusion of depth, and how good can it really make something look? We will be answering these questions and more in our introduction to the principles of 3D texture mapping.

Slide 3 (Low vs. High Poly): *Natalie*

3D texture mapping was created to make up for the processing power of computers when rendering a 3D object or scene. Rendering is the process of converting the 3D model with all its textures and effects applied into a 2D image, essentially like taking a picture of the scene. People wanted to be able to add more details to their models, but the computers could not handle how intricate they became when the models were changed directly. However, 3D texture mapping allowed for this detail to be added without causing a huge increase in processing time, which changing the model directly would do.

3D models are made up of polygons where the edges and vertices are connected to form the object. Vertices are found at each intersection between edges and they hold the information for how the model is shaped. Since adding detail directly to the model would require you to move vertices, you would need to have a higher polygon count to be able to make a model more detailed. Rendering an image from one 3D object requires the computer to access the information in each vertex on the model. The more polygons and vertices there are, the longer it will take the computer to render an image. This is why we choose to use 3D texture mapping techniques. It means we can add detail to the models without making more polygons and we can render images much faster as a result. This might not make a difference when it comes to small models that are not complicated in the first place, but when you start to have scenes with a lot of models that are all characters with lots of detail the 3d texture mapping techniques will make a huge difference in render time.

Slide 4 (Surface Normals): *Xiaoxi*

Even though they simply alter two dimensional images, 3D texture mappings can do an amazing job of looking like they have depth. This is all thanks to the visual properties of lights and shadows. The human eye can often tell that an object has depth by how light falls on it. This is because of what we call “surface normals,” which are vector representations of which direction a surface is facing. As shown in the image, if part of a surface is facing a light source, it will be brighter, but if it is facing away from a light source, it will be darker. This basic property allows 3D texture mappings to make things look like they have depth just by changing how bright or dark parts of a surface are, thus changing how light appears to hit them.

---

# 3D TEXTURE MAPPING TECHNIQUES

## Bump Mapping

Slide 5 (Bump Mapping): *Nicola*

And now for the first, oldest, and most common type of 3D texture mapping, bump mapping. This is the one that started it all, and allows one to quickly add details and depth to surfaces. It was created by our good friend Blinn in 1978. You might remember him from other groups' presentations as having worked a lot on lighting models. Typically, bump mapping is applied to a 3D model by using a grayscale image known as a bump map, as you can see on this slide. As we mentioned before with surface normals, parts of an object that recede tend to be darker, while parts that stick out are brighter. These bump maps can thus represent how deep or raised parts of the object are using this method, with the light parts of the bump map representing an area which should look like it's sticking out from the surface, and darker parts representing an area that should look like it's going inwards. When the bump map is properly applied to the object, along with a texture map for color and detail, the illusion of depth is created without physically changing the object at all.

<u>Slide 6 (Bump Map vs. No Bump Map)</u>: *Nicola*

Here we can see what that brick wall from the previous slide looks like with and without bump mapping, allowing you to get a better sense of how the light and dark areas on bump maps can alter what an object looks like. These areas between the bricks appropriately look darker, and the bricks themselves pop out, and have visible cracks and other nuances. Now, an important thing to note about bump maps is that they are limited to 8-bits of color information per pixel. Thus, only 256 variations of black, gray or white can be calculated to determine how lowered a pixel should be. This limits detail to an extent, and makes it difficult for bump maps to make it look like things are sticking out in different directions, other than just straight up or down from the surface. Additionally, with few exceptions, the silhouette of the geometry that the bump map is applied to will be unaffected by it. As such, bump mapping is an easy, simple, and computationally efficient way of creating the illusion of depth on a flat surface.

## <u>Normal Mapping</u>

<u>Slide 7 (Normal Mapping)</u>: *Natalie*

Normal mapping is another technique which adds an illusion of depth to a 3D model. It is a specific type of bump mapping, but it works a bit differently which I will explain in a bit. When artists are creating normal maps, they will usually have two versions of a 3D model, one with a lot of polygons (high poly) and one with much less (low poly). The high poly model has the detail that we want to add to the low poly model by applying a normal map. Programs such as Zbrush, xNormal, and Maya can take the high and low poly models and compare them to create a normal map. They need to be created this way because unlike bump maps which are greyscale, normal maps are stored as RGB images and are much harder to just draw on your own.

Slide 8 (Normal Mapping Examples 1): *Natalie*

Here you can see how a normal map is created. On the top you see the normals of the model and how they follow the curves of the mesh. Those curves are the detail we want to add into our normal map. When creating the normal map, the program will access each pixel of the high polygon mesh and check its normal orientation. It will then check the UV map of the model and color the pixel on the normal map corresponding to the pixel from the 3D model. The color depends on the orientation of the normal on the high poly mesh. The reason that normal mapping is more accurate than bump mapping is because it can access the orientation of x, y, and z coordinate systems, whereas bump mapping does not have x and y.

Slide 9 (Normal Mapping Examples 2): *Natalie*

In this slide you can see an example of a normal map. Each color on the map represents the direction the surface normals were pointing on the high poly 3D model. This information is then applied to the low poly model to make it appear to have the same amount of detail as the high poly version. The color red stores the X coordinate information, while green stores Y coordinate, and blue stores Z coordinate. The value from 0 to 255 of the RGB channels is converted to a value between -1 and 1 for the X and Y, and from 0 to 1 for Z coordinate systems. Since the Z value determines if the normal is facing towards you or away from you, it should not have a negative value, or it would not appear visible. This is why normal maps have a blue tint to them. Since the z value has to be 0 to 1, if a normal is pointing straight at you with no x or y orientation changes, it will have rgb value of (128, 128, 255). Each pixel on the low poly 3D model that has a normal map applied will access a pixel on the normal map based on its UV mapping. It will convert the single color value into the red, green, and blue channels. Then convert each channel value into the X, Y, and Z coordinates that will determine the orientation of that pixel's normal. Something that is pointing right will then have an x value from 0 to 1 and will add red to the rgb value. Something that is pointing left will have less red and so will be this darker blue. If it is pointing up then it has more green and down has less green.

Slide 10 (Normal Mapping Examples 3): *Natalie*

Here you can see some examples of how an 3D model would look with an applied normal map. The tire's normal map is seen in the bottom right of the image, which is applied to the model in the top left. Once the normal map and other texture maps are applied to the tire, it appears to have the detail that the high poly 3D model had. Similarly, the low poly rocks in the center are given detail by applying the normal map to create the render on the right. This one (point to right) is not a high poly model, it simply has a normal map applied. Thus, normal mapping is like a more detailed and advanced version of bump mapping which also does not require much processing power.

## Displacement Mapping

Slide 11 (Displacement Mapping): *Xiaoxi*

So now after gaining a good understanding of basic 3D texture mapping techniques, we will move on to what's called displacement mapping. Displacement mapping is an interesting alternative to some of the other mapping techniques which can be used on its own or in combination with the others. Unlike the other techniques, which only altered the surface normals, this one actually changes the surface of a 3D model when it is time to render. This makes displacement mapping useful for when the details of a 3D model have to visibly protrude from the surface. Usually, this is done by methods of tessellation, which you might remember from earlier presentations, is about creating many small triangles to represent the displacement map on the surface. On a side note, per-pixel displacement mapping is also available for those with the ambitions. As the name suggests, per-pixel displacement mapping can create extremely detailed protrusions and is easily refinable, and it can be implemented with some of the most advanced modern GPUs. As you can imagine though, that's a lot of vertices to be displaced at render time and not recommended for normal "everyday" usage. As seen in this slide, without the displacement map, the plane does not look like it has any depth. Simply adding this map will give the plane the indents and protrusions necessary to create a good amount of depth.

Slide 12-13(Displacement Mapping Examples): *Xiaoxi*

By combining the previous techniques (bump and normal mapping), you can add small details to the displaced 3D model, and even have the displacement change its silhouette. When it comes time to render, the model's surfaces will change according to the displacement map and then bump mapping and/or normal mappings can be applied to add even more detail and realism without adding polygons to the model. Unlike the previous mapping techniques, which do not change how the object looks from the side, displacement mapping will make to object look as it should from the side, rather than just being flat This makes displacement mapping look quite good, but it comes at the cost of greater render time than other 3D texture mapping techniques, since the altered surfaces take time to create and render. In later sections we will introduce other advanced techniques to target specific needs like self-shadowing on top of displacement map. Before we move on to other advanced techniques, are there questions at this time for the three most fundamental 3D texture mapping techniques?

The next technique we'll look at will provide similar solutions of looking at things from the side in a less computationally expensive way.

## Relief Mapping

Slide 14 (Relief Mapping): *Richard*

Now for a mapping technique that is a little more advanced and niche. Relief mapping is a newer technique that hasn't been around as long; it was first mentioned in academia in 2000 (officially by the name of "relief texture mapping") in a paper published by Oliveira et al. at the University of North Carolina at Chapel Hill. It fixes one of the shortcomings of regular texture mapping, which is that of parallax: when seen by a moving observer, the absence of parallax reveals the flatness of the surface. You can fix this with a height map, but that ends up requiring ray-tracing or a direct forward mapping (a form of image warping) because screen coordinates cannot be directly converted into texture coordinates with a height map. Forward mapping is actually what relief mapping will use, but we'll get to that.

Here, the left image is rendered using traditional mapping, the right image is rendered with relief mapping. The footnote mentions that there are exactly the same number of polygons in both images - keep that in mind. Notice, however, (it may be a little hard to see, but) check out the bricks on the wall, and the dormers on the roof. They look like they're sticking out even though you're looking at them from the side. This is the problem of parallax error, which is most obvious if you're actually moving the camera around this "world." If you moved around in the left image, you would immediately be able to tell that the surface is actually flat - it just doesn't look right. Even here it looks somewhat problematic - like someone tried to make it 3D but didn't quite succeed. In general, whereas bump mapping and normal mapping can result in complex objects appearing strange when you aren't looking at them head on, relief maps can handle it while still keeping it realistic.

Slide 15 (Relief Mapping Example): *Richard*

So how does relief mapping actually work? To relief map something, you have to first create a relief texture, which is essentially an average 2D texture that also contains values for orthogonal displacement, pretty much like in normal mapping. (For reference, the word "relief" means differences in elevations on a surface - so you get why it's called that.) Relief textures get put onto a 3D model in a two-step process. First, it's converted into a normal texture using a forward transform (called "warping"), which involves some math and a set of equations and whatnot. Second, now that you just have a regular texture map, you apply that texture as usual to create the image. The big difference is that the program will recalculate the new texture whenever the perspective changes - unlike the other 3D texture mapping techniques which went in and explicitly updated the textures to simulate 3D, relief mapping instead draws from the same model each time, and generates an image that appropriately "pops out" in response to the viewing angle. For that reason, unlike bump mapping and normal mapping, relief maps are able to handle things like view motion parallax, self-occlusion, and self-shadowing. And despite having to warp the image with each frame, it does these things with much less processing time, the reason for that being that implementing occlusion and self-shadowing in bump and normal mapping you need to add more polygons, which increases rendering complexity and time.

Alright, now to wake y'all up we have a demo!

---

# MAYA DEMO

---

# OTHER TECHNIQUES

## Parallax Mapping

Slide 16 (Parallax Mapping): *Richard*

Now we will be discussing some of the more advanced research being done today on the subject of 3D texture mapping.

Let's start with parallax mapping. First published about in 2001, it was specifically promoted in order to represent motion parallax in real-time rendering. Sounds a lot like relief mapping, but not quite: parallax mapping is actually an extension of bump mapping, and it uses a heightmap and the current viewing angle as part of a function to determine what the final image looks like. In general, it creates more of a sense of depth by distorting the pixel heights in the texture in real time in such a way that when viewed from an angle, high points obscure lower points behind them, thereby simulating parallax. It's actually able to do this very efficiently - almost as fast as normal mapping.

Anyway, for this example, you can see that normal mapped objects can still appear flat when viewed from the side. The middle image shows how parallax mapping makes that a little better - note the distortion on the pillars and on the lion's mouth to try and simulate the viewing angle. Unfortunately, one of its shortcomings is that it's a single-step process, so it doesn't handle occlusion. The image on the right, however, does the 3D thing a lot better. Pay attention to how the lion looks a lot more 3D and how even the pillars have been distorted to the point that they pop out from the wall. Steep parallax mapping is able to handle self-occlusion and self-shadowing (it's still just using a bump map! To reiterate, this is still a flat image that has just been distorted to look 3D) and it does this by using a ray-tracing algorithm to find the first visible point on the height field. It's one of several techniques that enhance the capabilities of parallax mapping using various methods; others include parallax occlusion mapping and interval mapping. That's all cool stuff, but it's mostly just different approaches to the same issue, different ways of approaching the problem to resolve some specific purpose. If you're interested in understanding the specifics, you can look up published academia on all those methods. For now I'll let my group talk about something else.

## Horizon Mapping

Slide 17 (Horizon Mapping): *Nicola*

Horizon mapping is another advanced mapping technique that focuses on creating shadows for added depth, realism, and effects from lighting. Generally, 3D texture mapping techniques such as bump and normal mapping do not actually create shadows when applied to an object, but horizon mapping can fix that. It produces realistic-looking shadows by calculating the shadow that would be created by the more raised parts of a 3D texture map. Normally, shadows can only be created from other 3D objects, but here 3D texture mappings can cast shadows on their own objects. On the screen you can observe this technique in action. On the left is a brick wall that has had normal mapping and parallax mapping applied to it. The wall appears to have depth, but no areas look like they are really being affected by the light in the scene. However, on the left, you can observe the sharper contrast created with light and shadow on and around the stones. Thus, horizon mapping adds more realism and interaction with the rest of the scene.

---

# CONCLUSION

Slide 18 (Bringing It All Together): *Xiaoxi*

Today, we covered a variety of 3D texture mapping techniques. So, to recap the topics we have spoken about today, 3D texture mapping encompasses bump mapping, normal mapping, displacement mapping, relief mapping and other advanced techniques. All these techniques are being used in combination to target specific needs in the gaming, art, and film industries. It is common to witness artists using different techniques due to time and complexity constraints. Please keep in mind that there is a lot of research still being done today, and we can only cover so much for you within this time constraint. As a group, we hope you enjoyed this presentation and we hope we presented enough to spark your interest in this subject.

At this time please feel free to let us know if you have any questions.