<< RETURN

UV Mapping tips and tricks Discussion.

Author Name: Renier Johannes Banninga
Contact Information: renierb@streamline-studios.com

The movie and game industry has come along in giant leaps and bounds over the last 10 years. We can create some pretty realistic cg creatures these days, featuring every bell and whistle you can possible imagine. From muscle dynamics to cloth simulations and even accurate liquid dynamics. Visual effects work has become a science and an art form all rolled into one.

But one subject that always seems to get the least attention is that of texture mapping. It's always the last part that the 3d modeler wants to deal with and the first thing that can make your texture artists go insane. You've just finished your latest creation, a fantastic multi-limbed creature with layers of internal mesh details and now its time to create the UV's. But in all its glory you've created a nightmare because none of the basic modifiers will work. It's an all out custom job. You have to make each surface have its own unique mapping with little to no distortion. That means no stretching or shearing and keeping a consistent pixel density over the entire character. And while you're doing this, come up with a texture map that's understandable by the texture artist. And with all this going on you still need to finish it to keep up with those lovely deadlines.

Welcome to the world of texture mapping, the least talked about subject in 3d. My goal with this discussion today is to arm you with every trick in the book when it comes to UV mapping. And giving tips as to what methods to use in every situation you can encounter.

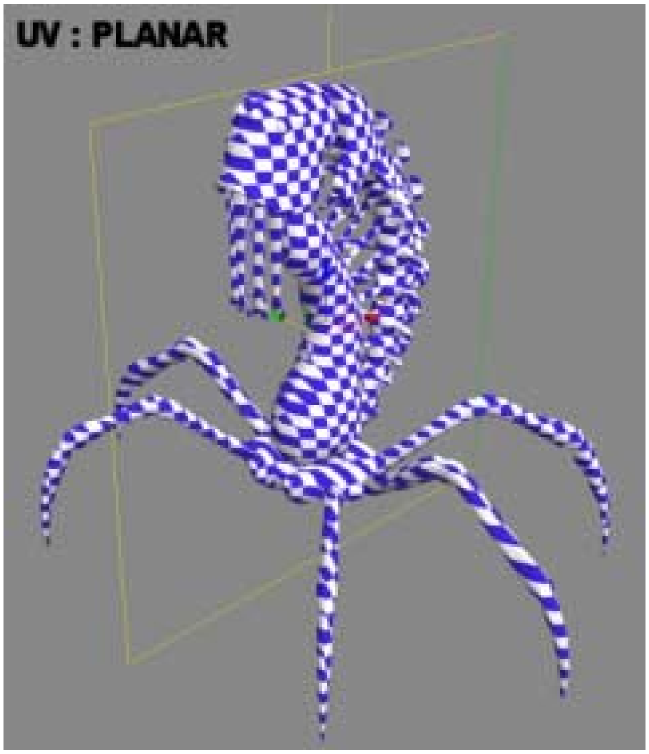
The second goal for this discussion is to share some tricks in achieving more texture detail in real-time games by the use of blending techniques using small modular detail textures.

The Basics

Mapping types and there uses

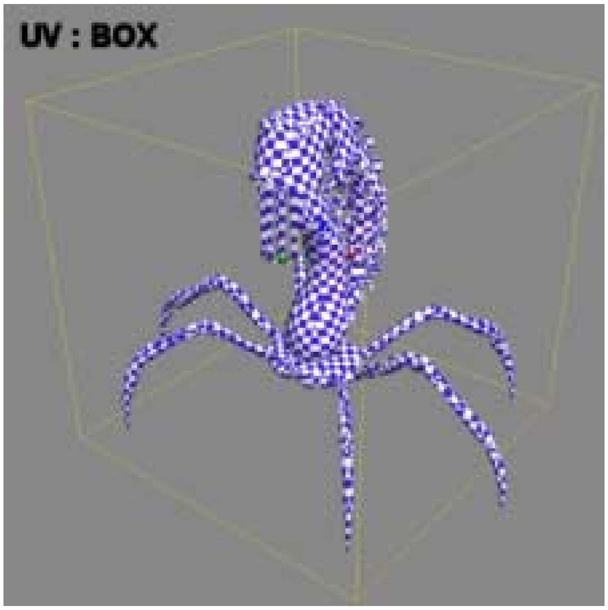
Planar

Planar mapping is the most basic of the mapping modifiers to apply to objects. It works by projecting the texture onto a model from one direction. So it's useful for things like walls and basic terrain. But it can't really be used on complex objects with lots of overlaps since this will cause your mapping to start streaking on the polygons that don't face the projected map directly. There are a few ways to fix these errors though but will be covering that later in the discussion.



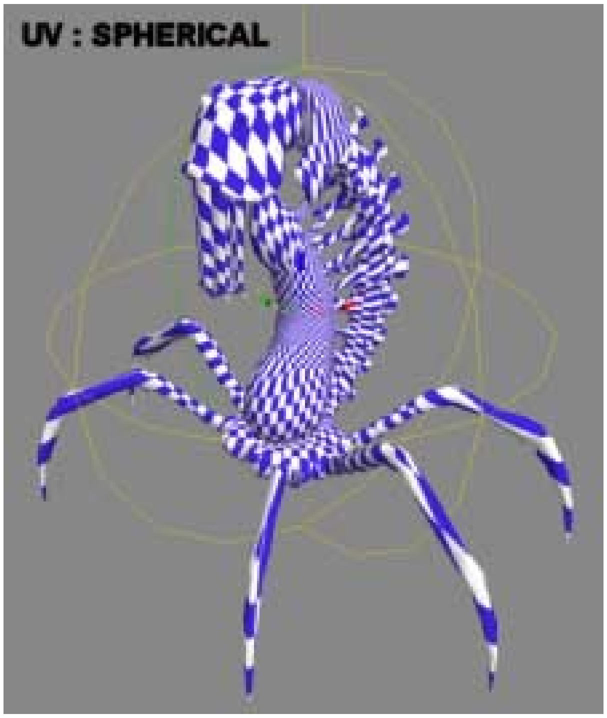
Box

Box mapping works by projecting the desired texture onto the model from 6 sides. This is very handy when working on technical or architectural objects, and when you just need quick mapping for certain less relevant parts on an object. But it's not very useful on organic objects where you need accurate mapping. A good example would be tiny screws on a cargo container, and some metal bars or grates in a factory. It becomes less useful when you start working with more complex objects that require specific mapping.



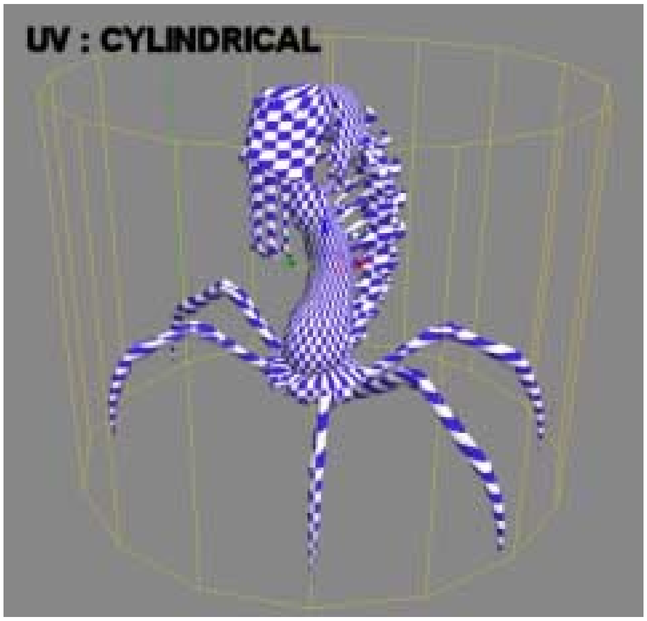
Spherical

Spherical mapping is just that, it projects the texture in a spherical pattern onto an object. Space junkies love using this one to map asteroids and planets, but the one side effect that it does cause is very high pixel density at the polls of the spheres mapping. This causes a pinching effect that's hard to counter when painting texture. It's helpful in blocking out mapping on human heads but with a lot of tweaking afterwards. It starts to lose it usefulness on models with lots of overlapping parts.



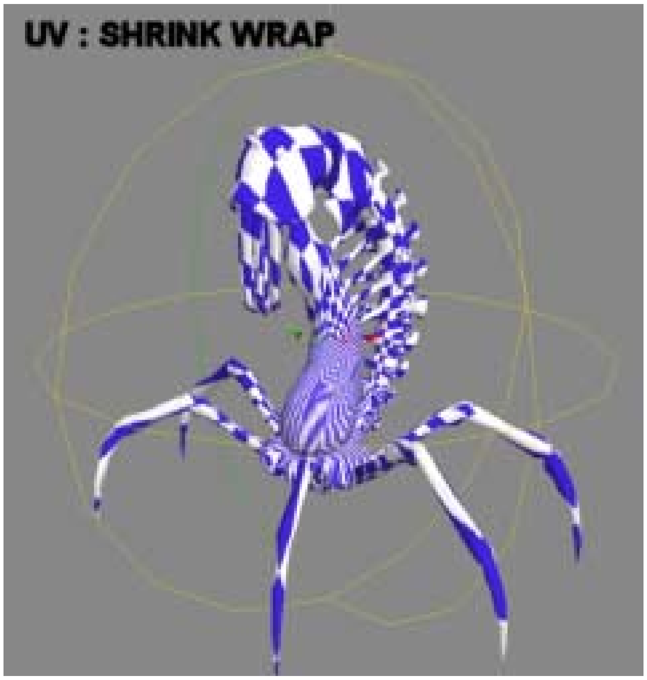
Cylindrical

Cylindrical mapping is actually used more widely then any of the other mapping types. Projecting the texture in a radial pattern inwards makes it very useful for mapping objects like tree trunks, arms, torso and legs. It's very handy to block out mapping on various types of meshes. But it still requires a lot of tweaking afterwards in the UV editor. Like the other mapping options it never works for every situation.



Shrink Wrap

Ah yes shrink wrap… I have yet to see anyone use this modifier in mapping objects. I have yet to figure out its purpose or how it works exactly, because it seems to create some of the most useless mapping solutions of them all. I can't recommend using this one on any object. If somebody finds a use for it do let me know, for I am rather curious to see it put to any sort of practical use.

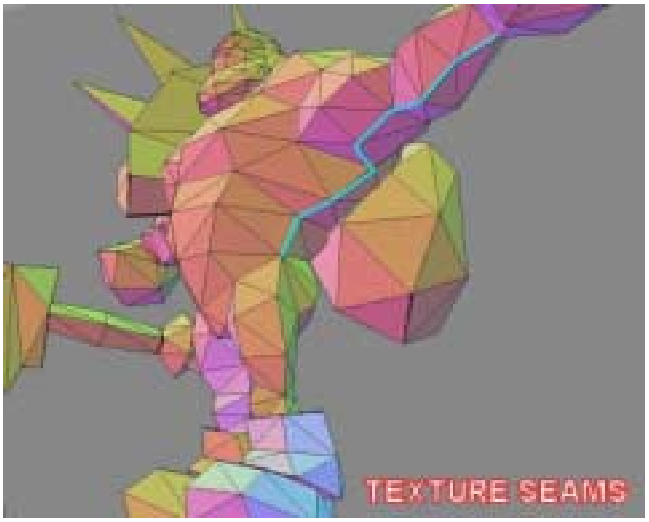


Pixel density and Stretching

One of the main things to keep an eye out for with your mapping is that you keep a consistent aspect ration for the pixel size in the texture mapping. Nothing will annoy a texture artist more then warped mapping. So keep a look out for areas where the texture gets stretched out or skewed. The last thing the texture artist needs to worry about is correcting his texture to counter the warped mapping. It's also considered wasteful when you use more pixel area then required.

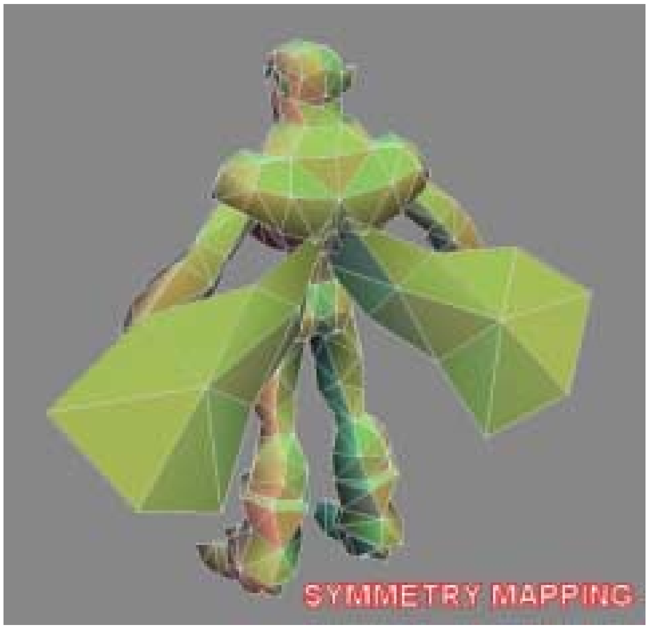
Mapping Seams

Seams, the artifact produced by cuts in the mapping because of the shape of a mesh… you can't avoid them but you can minimize there effects and the amount of work it will take the texture artist to fix them. Simply aligning the vertices of the seam with the corresponding connection in the mapping on either the horizontal or vertical plane of the texture coordinates. This way the pixels align on one of the axis when the texture artist goes in to fix and hide the seams. For technical objects its easier to get away with seams since they always tend to be quite fragmented mapping and the nature of the object allows it. But for organic meshes you really want to minimize the amount of seams as much as possible with some good continuous mapping, which will cover a little later in this discussion. The other solution is a good 3d paint program like Deep paint 3d.



Symmetry mapping and Overlaps

In some cases you will encounter models that are the same on both sides. A lot of characters are this way usually. So you can already cut the mapping time in half by just mapping one side of the model and then mirroring it afterwards. Another optimization that you can use it overlapping mapping coordinates. Say you have a character with lots off bloody spikes protruding from his back. You can map all those spikes to just use the same texture by overlapping them all onto the same area. For a real-time character this saves you a lot of unneeded texture resolution to be used somewhere else, instead of mapping each spike individual.



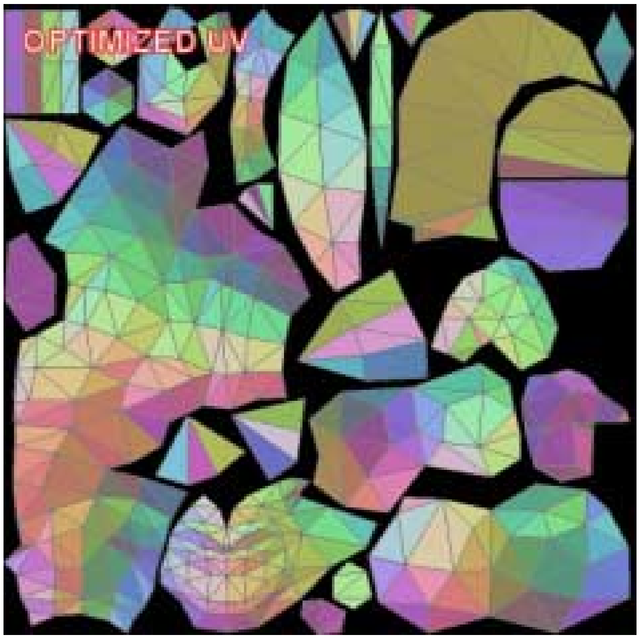
Tiling textures

Ah yes tiling graphics have always been one of the quickest ways to save memory and mapping space. For example the repeating pattern of a rubber hose or the treads on a tank. Their tiling nature allows you to just use a small texture area and then repeat it with tiling. That way you get a lot more pixel density then normal while still achieving the same effect. Artists that work with terrain use tiling textures like no other, but they use blend maps to hide the tiling pattern by mixing other tiling texture over each other. We will cover this more in depth later in the discussion.

Advanced

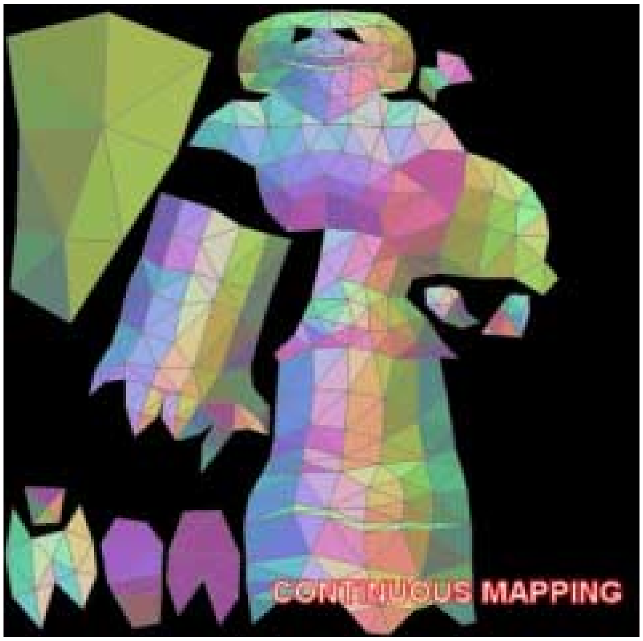
Optimized UV layouts

Now this subject is a little more relevant for real-time characters. In a nutshell don’t waste any space in your texture mapping. Since the entire texture gets loaded into memory it’s a good idea to use as much of the texture area as possible. So… scale rotate and move around those UV mapped vertex bits until you can’t save another ounce of space. Think of it like a giant puzzle with no picture to work with. Your only job at this stage is to maximize pixel resolution and at the same time have as little unused space as possible. A good UV mapped model can use up to 90 percent of the final picture. If you notice parts on a model that rarely get seen then scale the UV mapped space down to 50 percent its original size and cram it in somewhere with the rest of the mess.



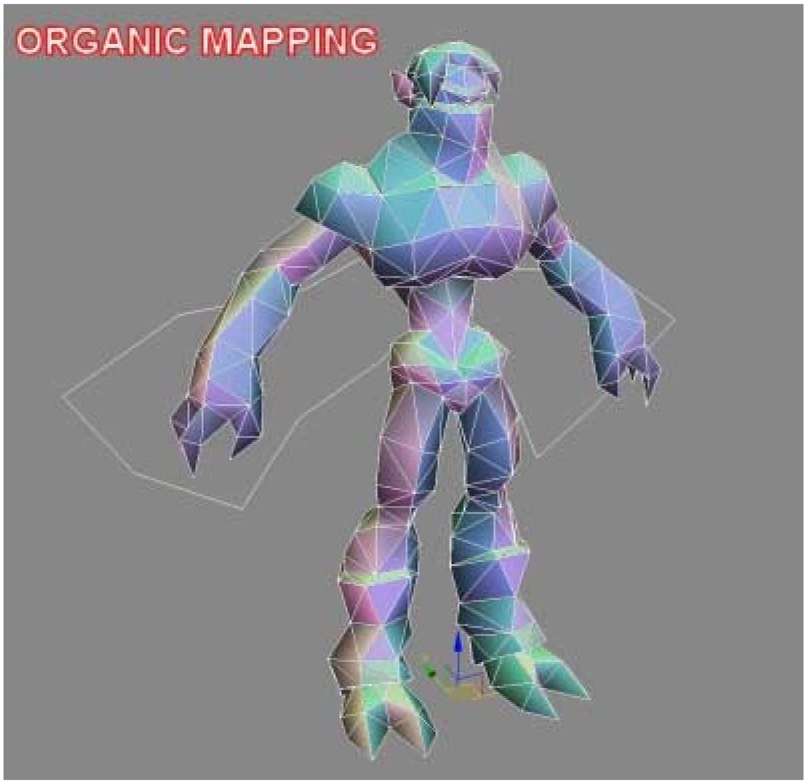
Continuous mapping

Continuous texture mapping is pretty much used only for organic creatures. The concept is basically you trying to link all the relevant body parts mapping so as to reduce the amount of seams. It involves some clever warping and arrangements with the UV mapped bits for the mesh you're attempting to map. In the picture below you can see that the entire body, legs and head consist of one single mapped area and I used symmetry as well to save on texture space for the character. This gives the texture artists less seems to deal with and just makes his life easier in general. So in short the more parts you can link together the better. Trying to do the same on mechanical parts is almost impossible without warping the texture though.



Organic model mapping

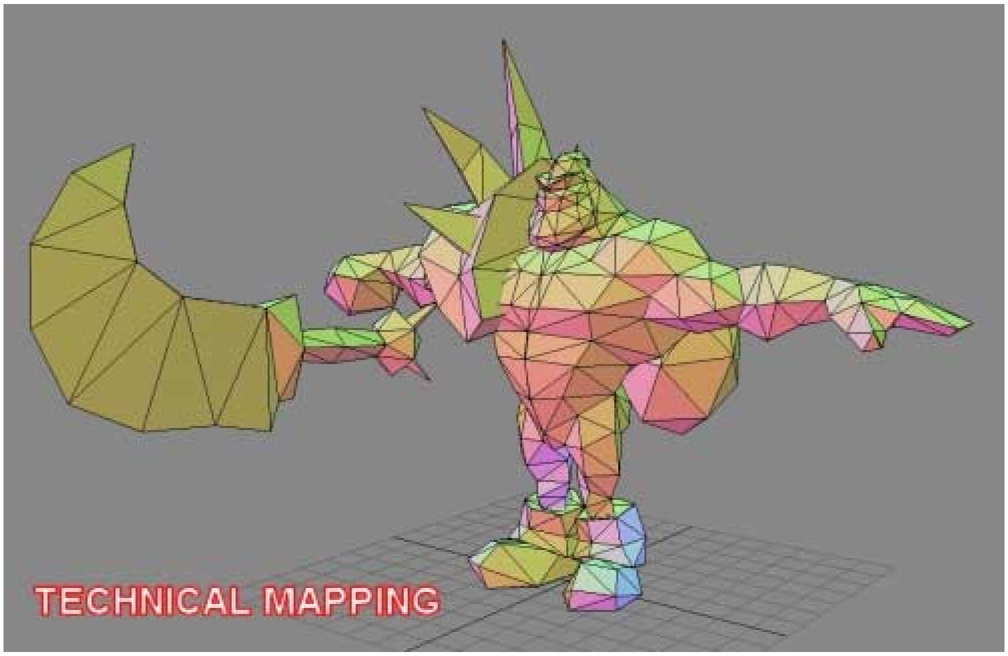
Creating texture maps for organic models is a lot easier in some respects but it does come with its own pitfalls. For one you're bound to encounter some of the more crazy shapes when mapping an organic creature. But the blessing is that you can get away with a lot more warping in the mapping though because of the curvy nature. Continuous mapping is definitely one of the key things you should do when going about mapping your organic shapes. Keep an eye on your seams count and look for ways to just unify everything into a continuous map. While still using as much of the texture space as possible. It's almost an art from in itself.



Technical model mapping

Technical objects are actually pretty simple to create mapping for, but the art of it comes from being able to organize and optimize the layout since the opposite can be says for the poor texture artist. All he gets is a giant color puzzle where he'll have to spend hours figuring out where every part really goes.

There are some simple rules to follow when mapping a technical shape. NO distortion on the mapping, this is a technical shape so keep your mapping as true to the original polygons as possible. Pixel aspect ration is also very important. Nobody wants to be painting tons of little rivets and bolts and then notice that their being squashed or stretched on the final mesh. Orientation, make sure that if the texture artist needs to write logos on the metal plates that you keep those polygons at 90 degree angles. The last thing they want to do is rotate the logos at odd angles to make it actually show up vertical or horizontal on the final mesh. And last but not least make it easy to figure out. It has to be a functional map. So it's got to use its space effectively while still being comprehensible for the texture artist. It really is a balancing act.

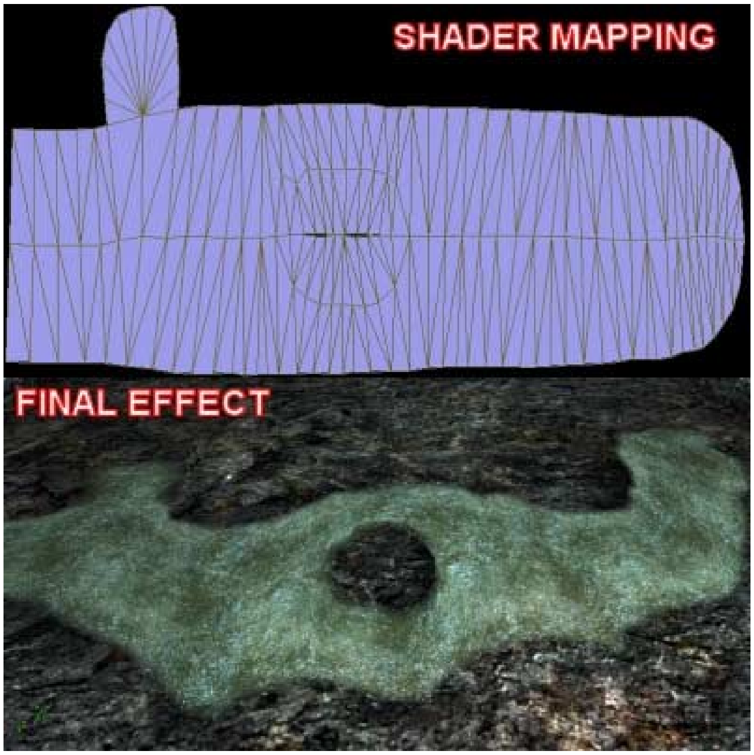


UV Mapping with shader FX in mind

Shaders have definitely been a welcome change to the world of gaming. Carmack really introduced the concept and his artists took full advantage of it. Quake3’s characters and environments benefited from this feature a lot. From cheating effects like cascading waterfalls to the patters off energy running through the outer layers of a mechs armor plating in Zone of the Enders, its opened up a wide new range of visual possibilities for artists everywhere. And all you have to do to achieve these effects is by following some basic mapping rules.

Say you wanted to create a flowing river that actually wraps around objects that get in its way? All you have to do is decide which direction the water will be flowing from and then orientate and distort your mapping to help achieve this. The picture blow illustrates the Mapping and final look of the effect. And all I did was straightened out the more curved areas so that when the shader is applies that it will look like its actually going around obstructions.

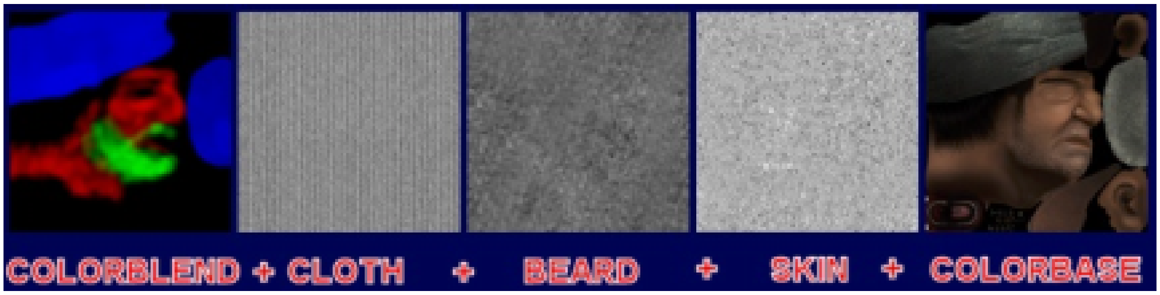
The same rules apply for something like tank treads or bellowing smoke cheated with polygons instead of particles “used to great effect in Freedom Fighters” Its just a simple scrolling texture with the models mapping orientated to take advantage of it.



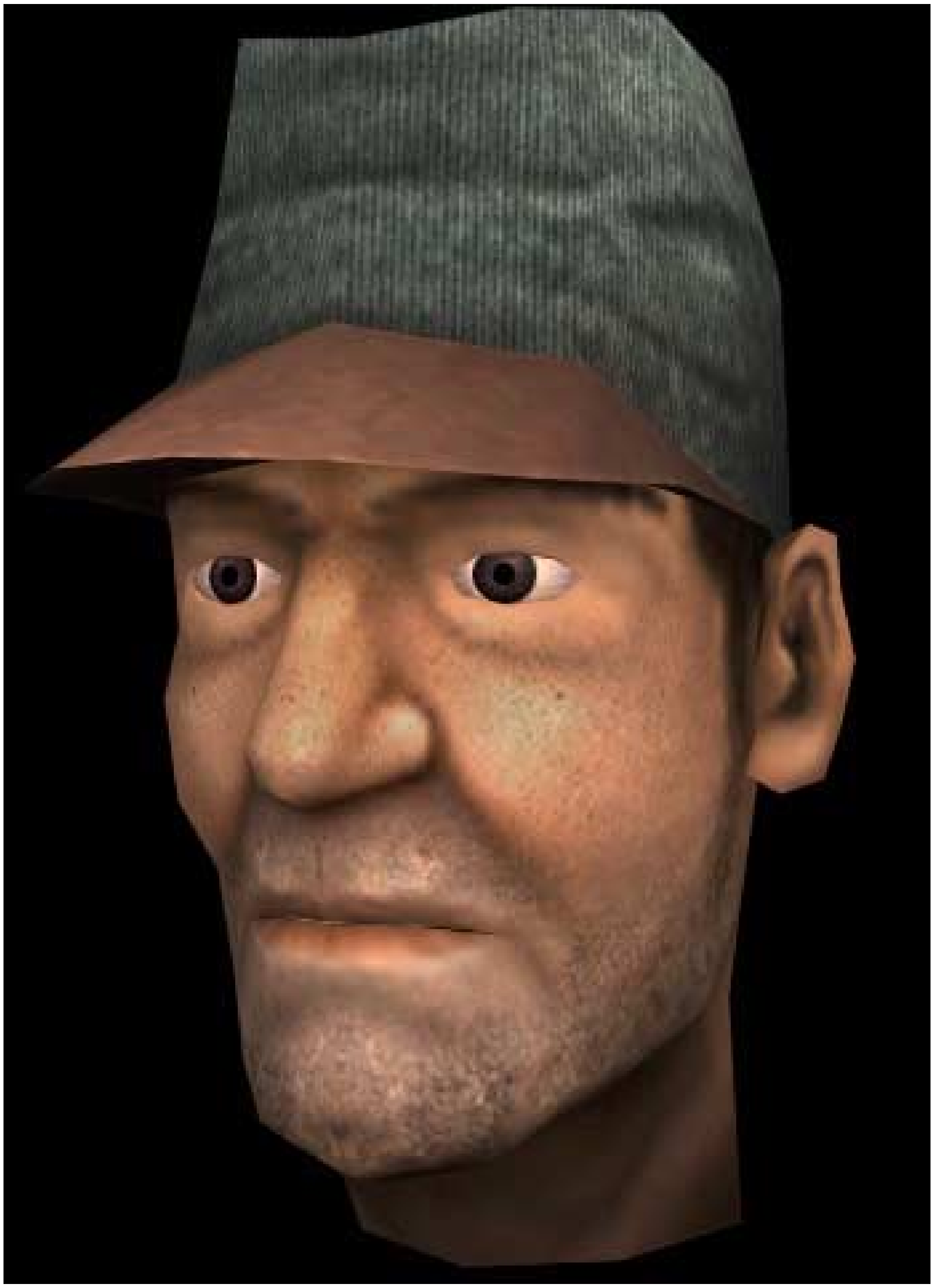
<u>Blended Detail textures with layered materials</u>

Ok everyone here knows what detail textures are right? If not then get our from whatever rock you've been hiding. Ok basically a detail texture is a texture that's blended over another texture depending on your distance from the object. The advantage or this is that you can cheat a lot of micro details without actually having to use a massive image. So if you have a brick wall at 256x256 and then another texture at 128x128. The smaller one will be used for the detail texture. Usually when you start getting really close to the wall it would start looking pretty blurry. But when used in combination with another texture that fades in when you get closer, it will give the illusion of a greater amount of detail then there actually is.

Ok now that we have that explained lets get into another method for its use. This will require some extra programming in creating a pixel shader for your game engine allowing you to use up to 3 different detail textures at the same time on a model using either vertex colors or and extra texture to define the blends. You'll be using the RGB color values in the picture to define the blends. Each color value coincides with one of the detail textures that you will define in the games material system. This also allows for smooth blends from one detail texture to another because it checks at the RGB level. Black of course won't show up at all. Here is a color example and the final results when using the detail textures in conjunction with a color map. It demonstrates the final results well enough, no fantastic example but hey ☺ it works.



And the final Blend of the face itself.



Renier’s Texture Mapping Process

And now I will show you my own method that I’ve developed for UV mapping objects in 3d Studio Max. It takes a little work to get use to it but once you do, I assure you you’ll never map objects the same way ever again. Think of it like actually skinning an animal. Removing the skin and laying it out on the ground for optimal painting. What I do is I make a clone of the original object and then take it apart manually in max. Then organize it and apply mapping to it, saving that out to a separate file. After that I select the original object and load the mapping from the cloned version and it’s done. But let me explain it in more detail.

Things to keep track off

Same Polygon count

The polygon count of your object needs to stay the same as the original from which you cloned it. When doing the final welding make sure you don’t delete or merge any polygons by accident. Otherwise your mapping will fail to load and you need to do it all over again. Save regularly and keep an eye on that polygon count every time before you save.

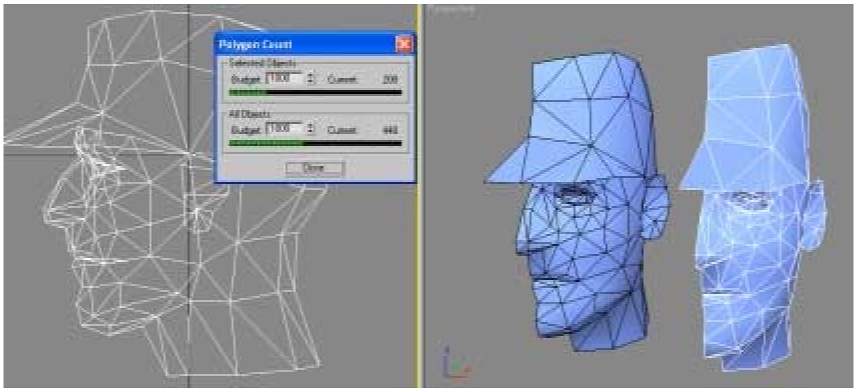
Detach to same object

When breaking up the model into the desired sections that you want to map make sure that you detach it to the same object. Otherwise you are removing polygons and the objects mapping will become invalid and you wont be able to load it on the original mesh that you cloned from. Keep an eye on these and you will be fine.

Mapping by hand

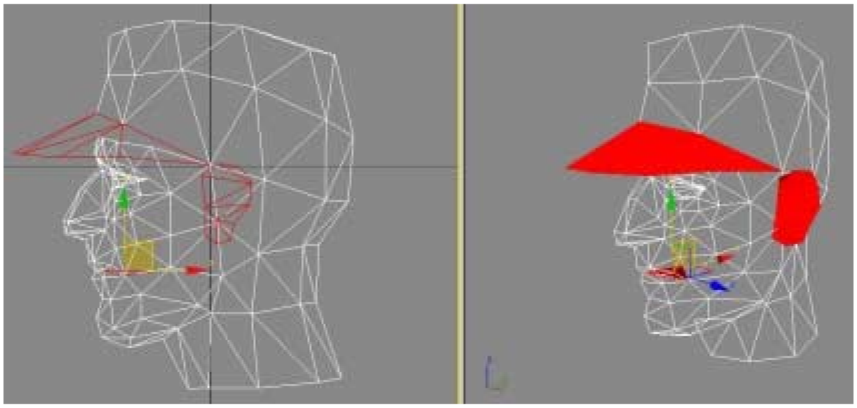
Clone object

Ok now the first step is to create a clone from the original object. Write down its polygon count and put that on the side for later reference. Now hide the original mesh because you don’t want to edit it, you’re just going to be taking the cloned mesh apart for now and you don’t want to mess up the original.

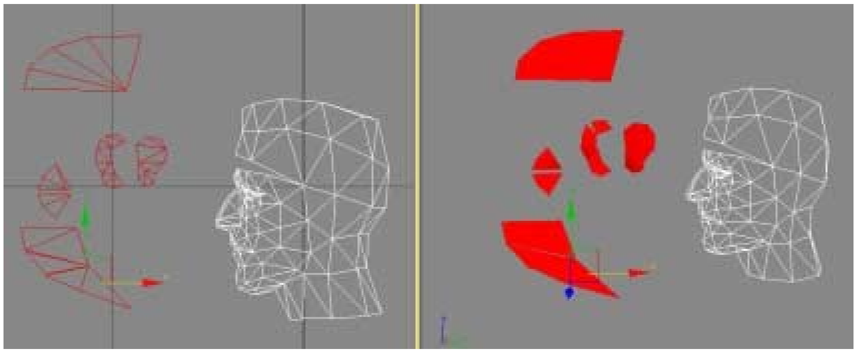


Fragmenting the Object

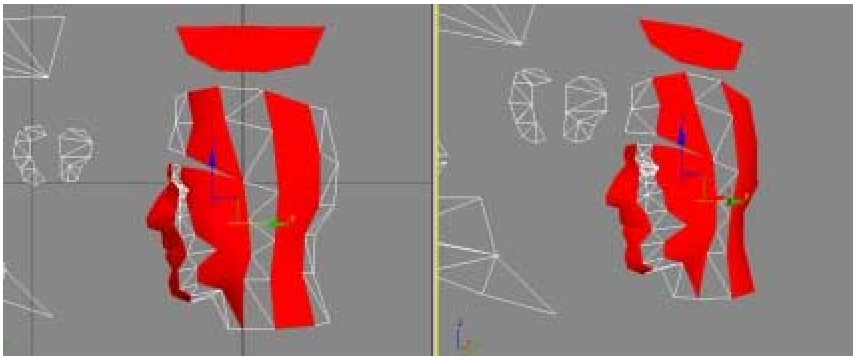
Now before I cloned the mesh I decided to delete the right side of the head. Since it's a symmetrical mesh I had no reason to map both sides at the same time. I can just map one side and the mirror its mapping to the other side. Alright the mesh is ready for mapping. I hid the original and now its time to start taking this sucker apart. I've isolated the ear, inside of the mouth and the front area of his hat to be the first areas to cut as you can see below.



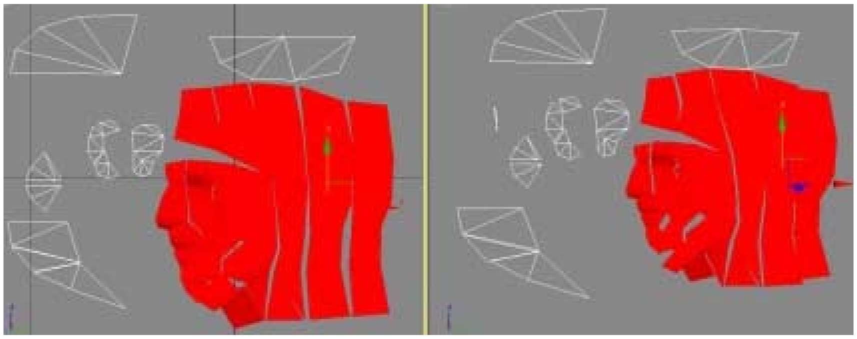
Ok now the beauty of this system is the fact that you have all the modeling tools at your disposal when taking apart the model. Simply pick a viewport that you want your polygons to face and then just start taking the mesh apart, arranging the polygons to face in the desired direction like I did in the picture below.



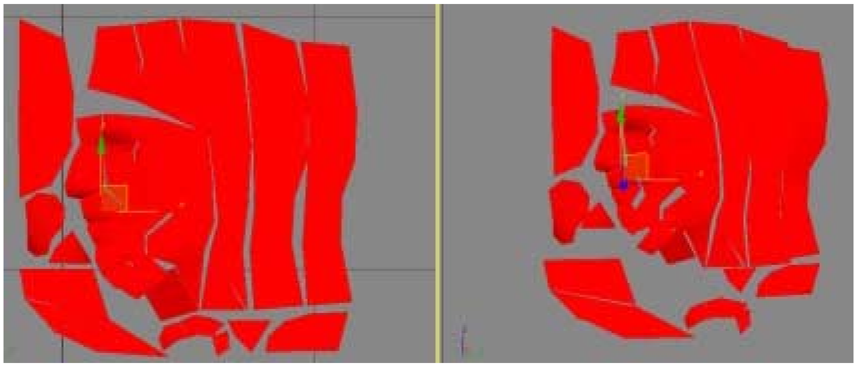
Now as you can see I started detaching the polygons and made them face the viewport as parallel as possible. This way I insure an accurate pixel density for all the objects I map. Make sure you haven't deleted any polygons in the process. Now on to the face, I've highlighted the sections that I plan on breaking apart.



Ok now I proceed to detach them from each other and start orientating them towards the one viewport again. While making sure to get the optimal angle on all of the polygons. Again I take a look at my polycount just to make sure I haven't deleted anything, and then save the scene incase max decides to crash ☺
As you can notice the mesh is really fragmented. But no matter for right now since that will be taking care off at the end.

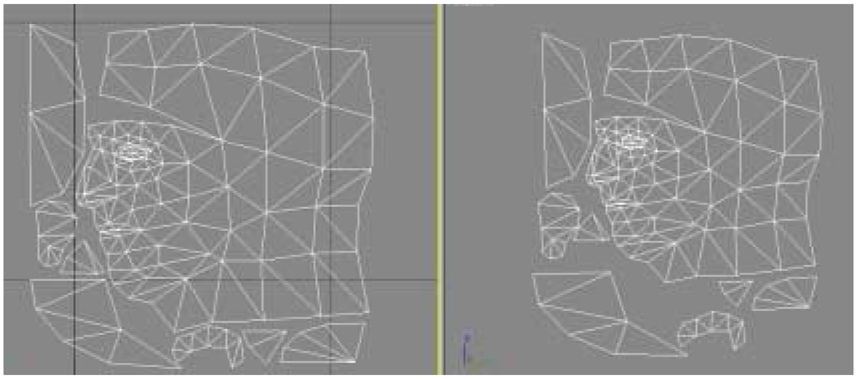


Ok now that everything is detached and laid out nicely we can go about organizing it. I keep to a 1x1 aspect ratio for the layout since this model is meant for real-time gaming. After moving around some polygons here is the end result.



Reassembling the Puzzle

Ok now that it’s all organized its time to weld the polygons back together again. It’s super simple and doesn’t take much time. Just keep and eye on the polycount when merging the vertices. After some welding here is the final result. Now it’s just time to transfer the mapping over to the final object.

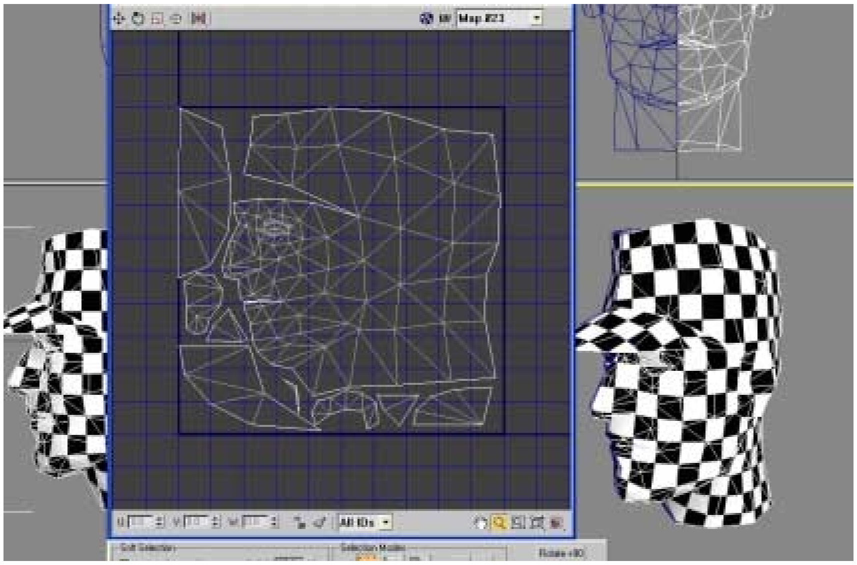


Transferring mapping

Ok select the mesh and then apply the planar map to the model. Now that it has mapping apply the UNWRAP modifier. This is the modifier I use to load and save the mapping of the model. Click and save mapping for the mesh.

Saving and Loading mapping

Ok now unhide the original model and apply another UNWRAP modifier to it. Click on load and then apply the mapping from the cloned model onto the final mesh. Click on Edit mapping to check if the mapping loaded properly.



If all went well then you should have a model with nice accurate mapping. Hope this has been an informative session and that you learned something. Pass on and share the knowledge.

Written by Renier Banninga

Date : February 10th 2004