

Title of Seminar: Polygon Texturing, Lighting, and Shading with Maya
Presenter: Rob Ormond

OVERVIEW

Polygon Texturing, UV layout, lighting and shading have always walked a fine line between efficient use of texture space, proper layout and the quality of the final result. The mastery of this discipline has required a fine balance between these requirements. This seminar will cover many aspects of texturing Polygons within Maya. It begins by going over some standard processes and techniques and moves on to more complex issues and workflows. The result will be more confidence and understanding of this specific discipline.

I. LAYING OUT UVS FOR EFFICIENT USE OF TEXTURE SPACE

The placement of Polygon UVs within the 0 to 1 texture space is a key consideration when texturing polygonal geometry in Maya. The wide assortment of techniques to achieve these results within Maya allow for a variety of workflows and approaches. The first section of this discussion will cover topics related to this aspect of polygon texturing.

Topology and UVs

Topology of polygon geometry

The topology of polygon geometry will have a major impact on the ability to properly apply texture maps.

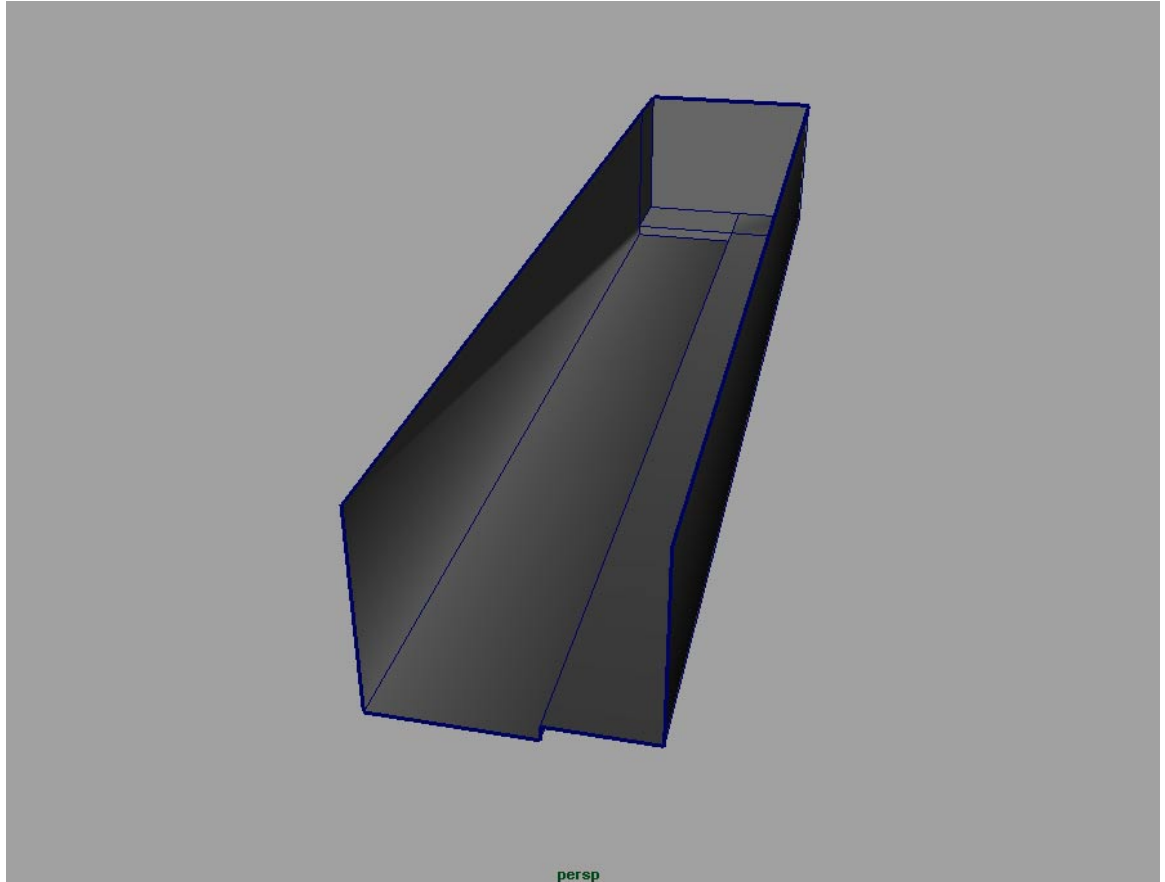
A key consideration before texture mapping a piece of polygonal geometry is how the topology of the geometry is laid out. It is possible to build something that is very efficient with a minimum number of polygons, however this does not mean that a texture map will appear correctly when applied. This is most apparent with tiling or repeating textures. In order for a texture to properly tile across a polygon surface, the topology should generally be consistent and made up of equally sized faces (per texture area).

Tiling texture maps

- **File > Open Scene > 01_basealley.mb**

In this scene you will see a very simple model of an alleyway. It has been constructed out of a single polygon mesh with a very minimal

polygon count. This does not mean that it is ideal for texturing. Due to the large and long area of the walls going down the alley a single texture map is not suitable. A much more appropriate approach would be to use tiling texture maps.



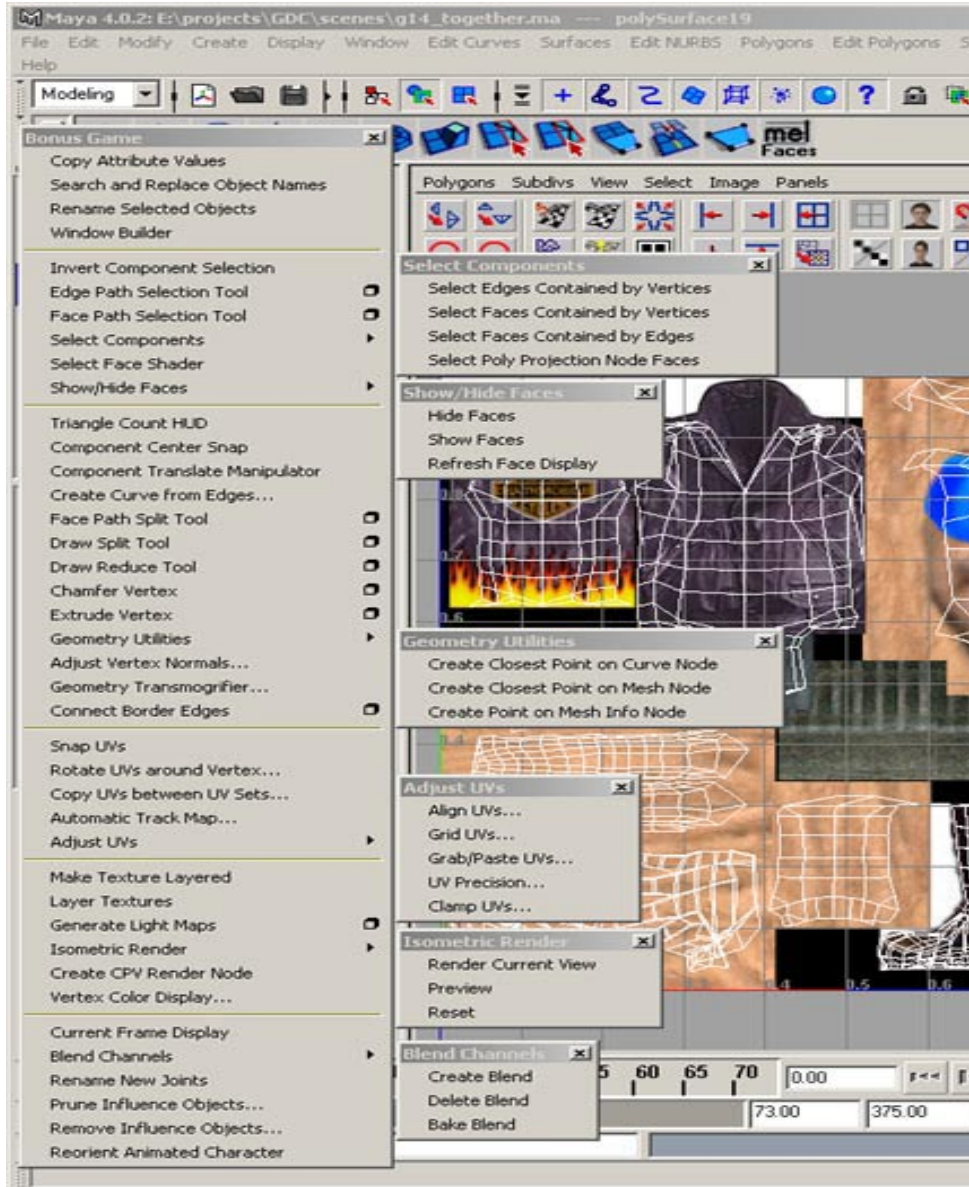
An alleyway

- **Press “6”** to go into textured mode.

You will see that the texture maps that have been applied to the various faces of the polygon mesh do not display correctly. This is due to the fact that they have been built to repeat or tile across faces of approximately the same size. It becomes apparent that the geometry in its current state will not be acceptable for texturing in this way.

The “Bonus Game” package

Before continuing, it is necessary to install the “Bonus Game” Package. This is a collection of Games-related scripts and plugins covering all aspects of Maya, including selection, modeling, texturing and skinning provided by Alias|Wavefront.



The "Bonus Game" package

Included with this seminar is the "bonusGame402.zip" file.

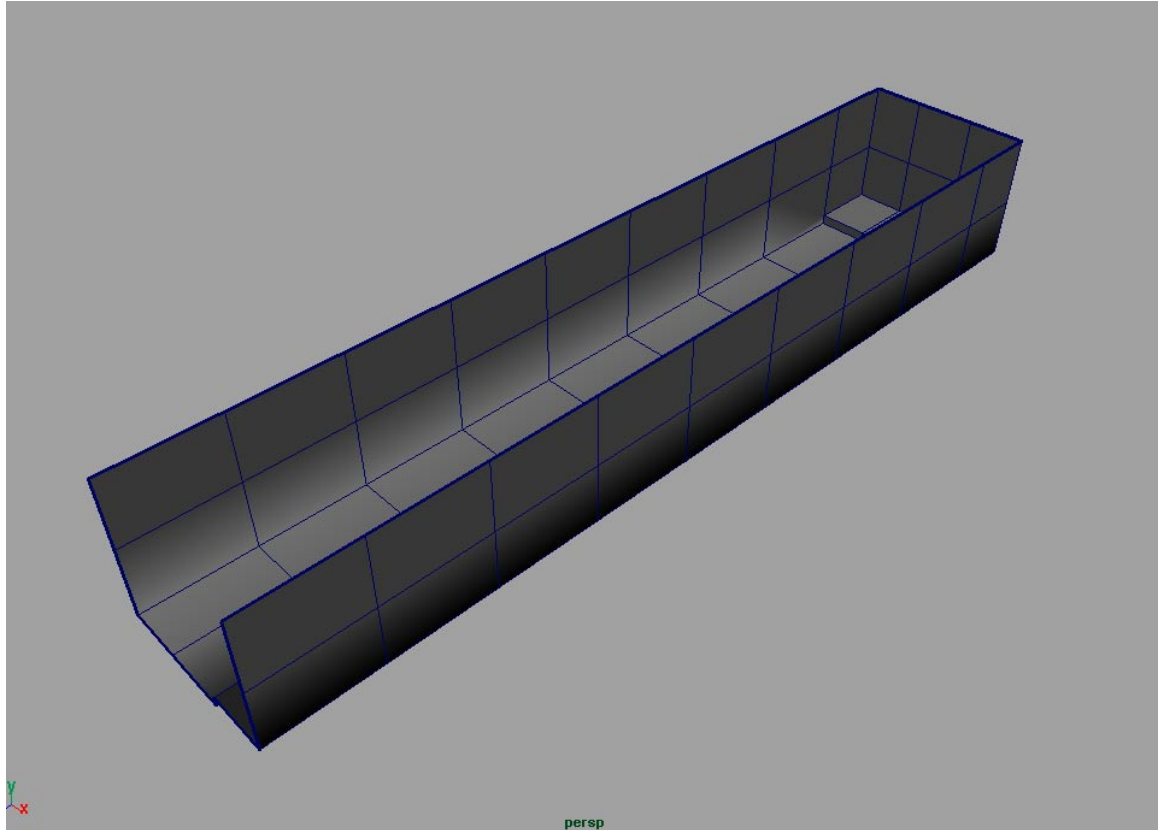
Add faces to the geometry

We will now add faces to the geometry to make it more appropriate for texture mapping.

- Select the *alley* geometry
- Select **Bonus Game > Face Path Split** and use the default settings.

With the tool active select one of the border edges running down the length of the alley. You will see that the faces split all the way across to the other wall down the middle of that face. Continue splitting in this manner until you have around 8 sections running down the length of the alley, 2 sections running from the ground to the top of the wall and 3 sections running the width of the alley.

It may also be necessary to use the **Edit Polygon > Split Polygon** tool to split in areas where you want to split just certain faces.



Faces of the alleyway

You will find that the horizontal and vertical rows of edges may not be perfectly aligned and that the faces may not be all square and the same size.

- In component mode select vertices. Use grid snap and the move tool to align the rows of vertices to match the above image. The important factor is that the faces of areas that will have the same tiled texture map are the same size.

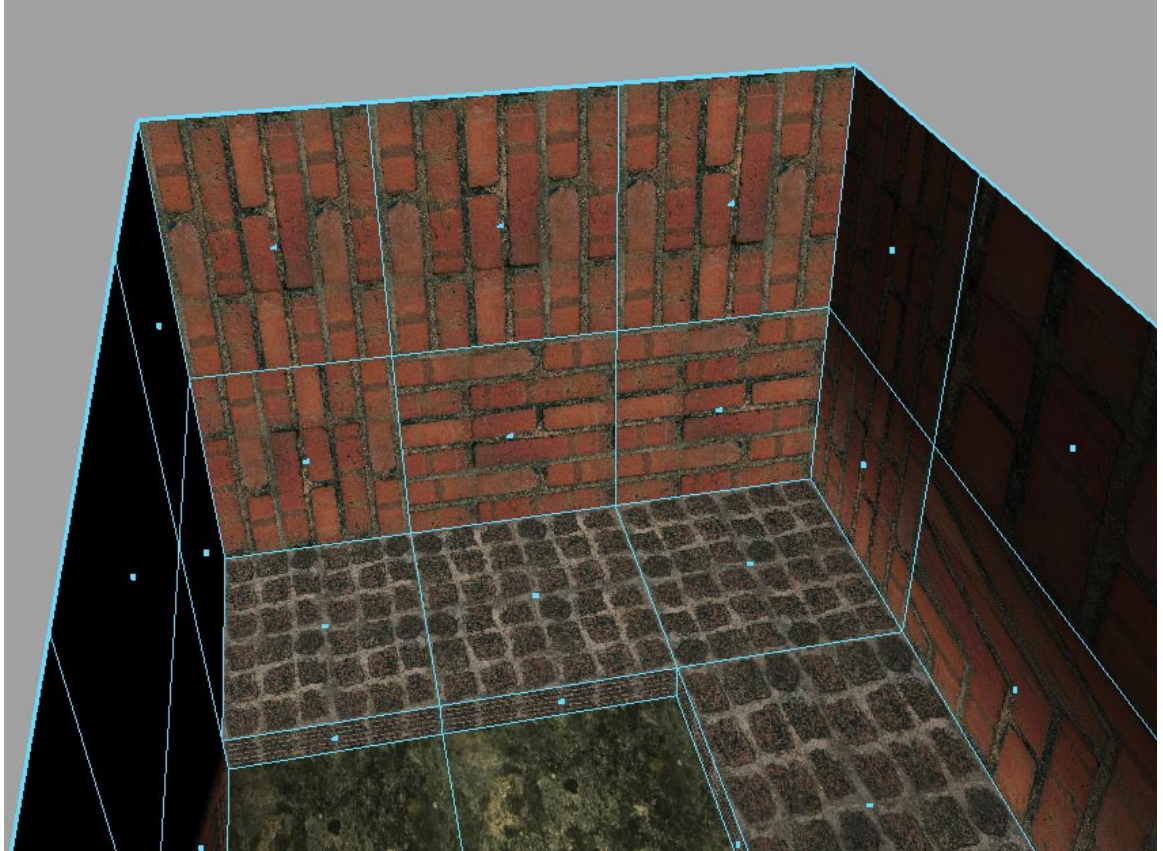
Note: if in certain circumstances it is not possible to get all the faces the same size, don't worry; lighting and shading techniques can be used to hide problem areas.

Unitize UVs tool

We are now ready to look at getting the UVs for the alley set up for texturing. A tiled texture map has been created so that it properly repeats over the 0 to 1 texture space. The edges of the U and V directions will be exactly the same on the image. This allows for the illusion (when repeated) that the texture map is one continuous texture when it is in fact a much smaller texture repeated many times. Following this logic, the UV area on the geometry that we want the texture map to reside on should fill the 0 to 1 UV texture space completely. In other words, we want to map each face to fill the 0 to 1 range. This is simple to achieve using the Unitize UVs tool.

- In component mode, select all the faces of the alley geometry.
- Select **Edit Polygons>Texture>Unitize UVs**.

This will map each face to fit in the 0 to 1 texture space. Make sure that the faces each have the appropriate texture map (shader) applied. You will notice that some texture maps are rotated incorrectly. We will now rotate them in the UV texture editor to correct this.

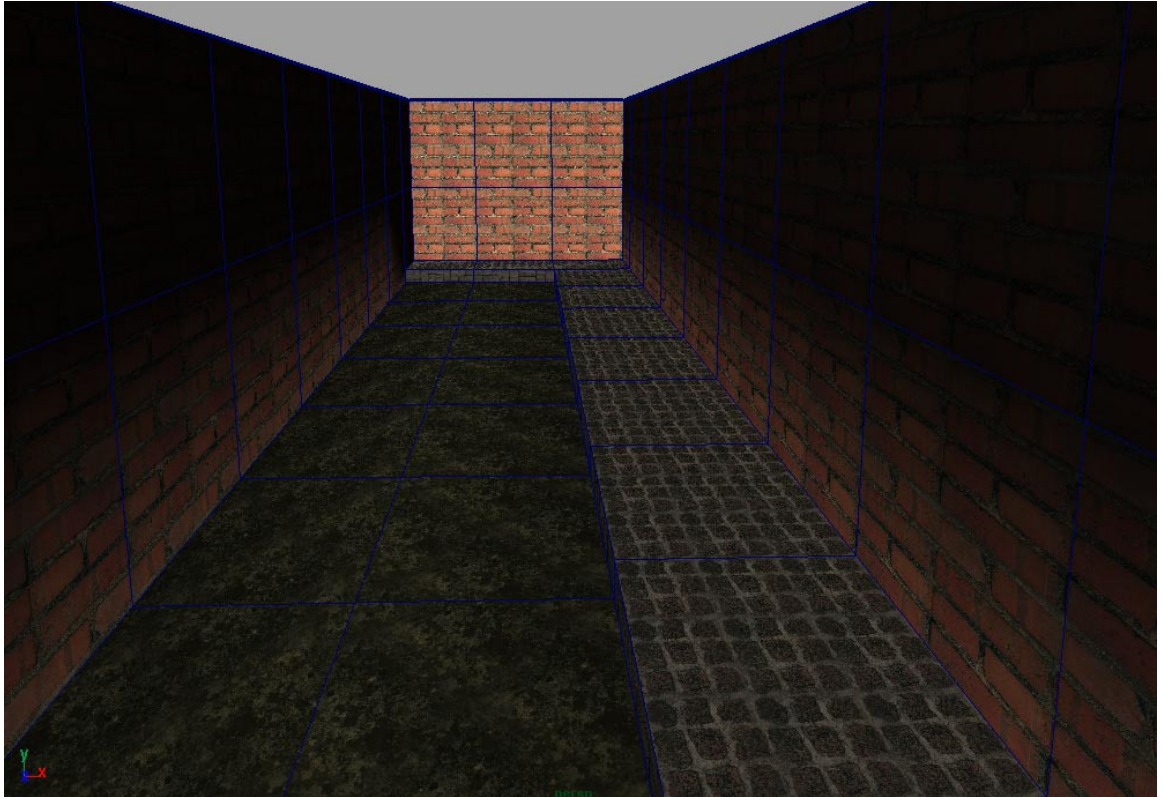


Unitized UVs

- Open **Windows > UV Texture Editor**
- Back in the perspective window select a face with an improperly rotated texture on it.
- Switch focus back to the UV texture editor window; you now have a face selected. In order to rotate the texture you need to rotate the UVs (most likely 90 degrees). You will now switch your selection from the “face” to the UVs associated with that face. When dealing with UV manipulation the ability to quickly switch selections from one selected component type to another associated component is very important.
- Within the UV texture editor press and hold the **ctrl** key and the **right mouse button**. This will open a special marking menu, which allows for you to quickly change what you have selected. Drag to the “**to UV**” item. You will now have the four UVs selected associated with that face.
- Within the **UV Texture Editor > Polygons > Rotate UVs** and set the rotate angle option to the appropriate amount (-90 or 90

degrees). The texture map should now appear proper.

- **Repeat** for all faces that require it. You can select multiple faces at once in order to do this. Also look out for any faces with more than 4 sides or 4 UVs that may require special considerations (manual adjustments).
- **File > Open File > 02_splitalley.mb** to see an example of the work up to this point in time.

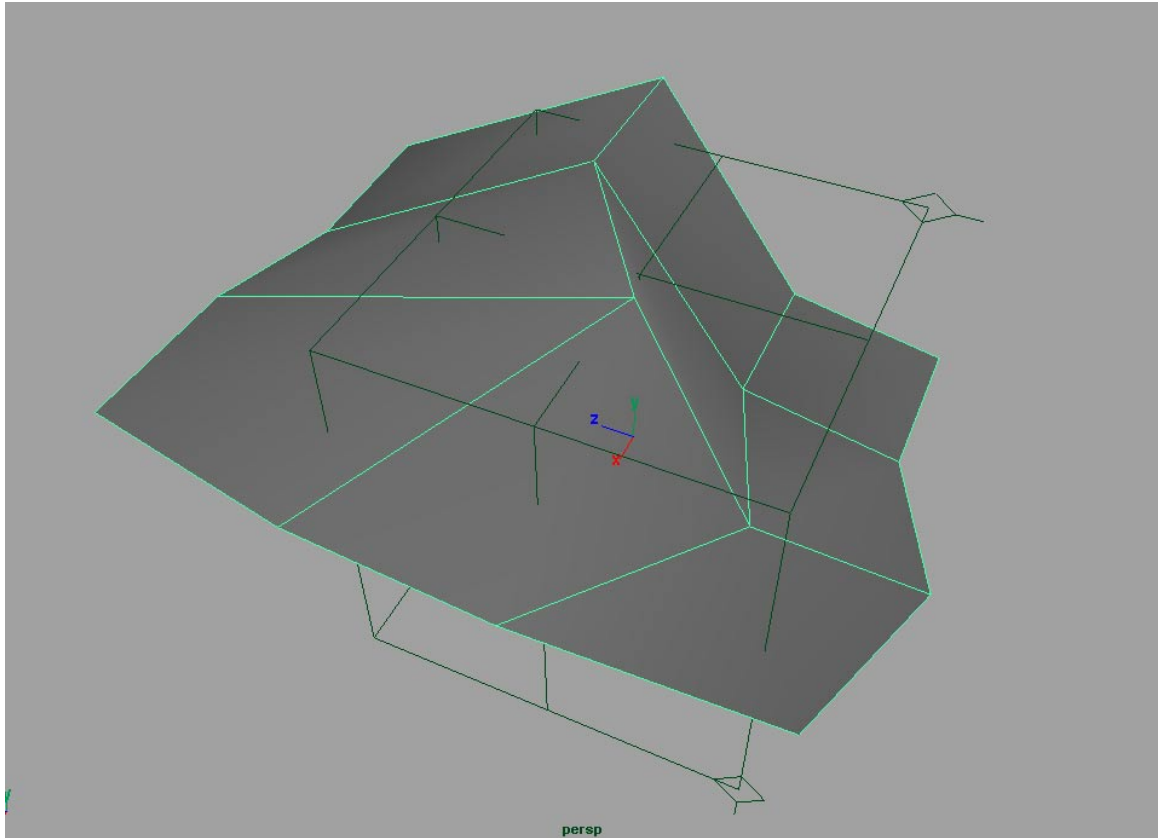


Fixed UVs

Using displacement maps to tessellate geometry

Thinking about the above conditions and considerations for repeating texture maps across faces an interesting idea is to use displacement maps and 3d projection nodes to split and add detail into polygon meshes.

- **File > Open File > 03_displacementsplit.mb**



Tessellated geometry

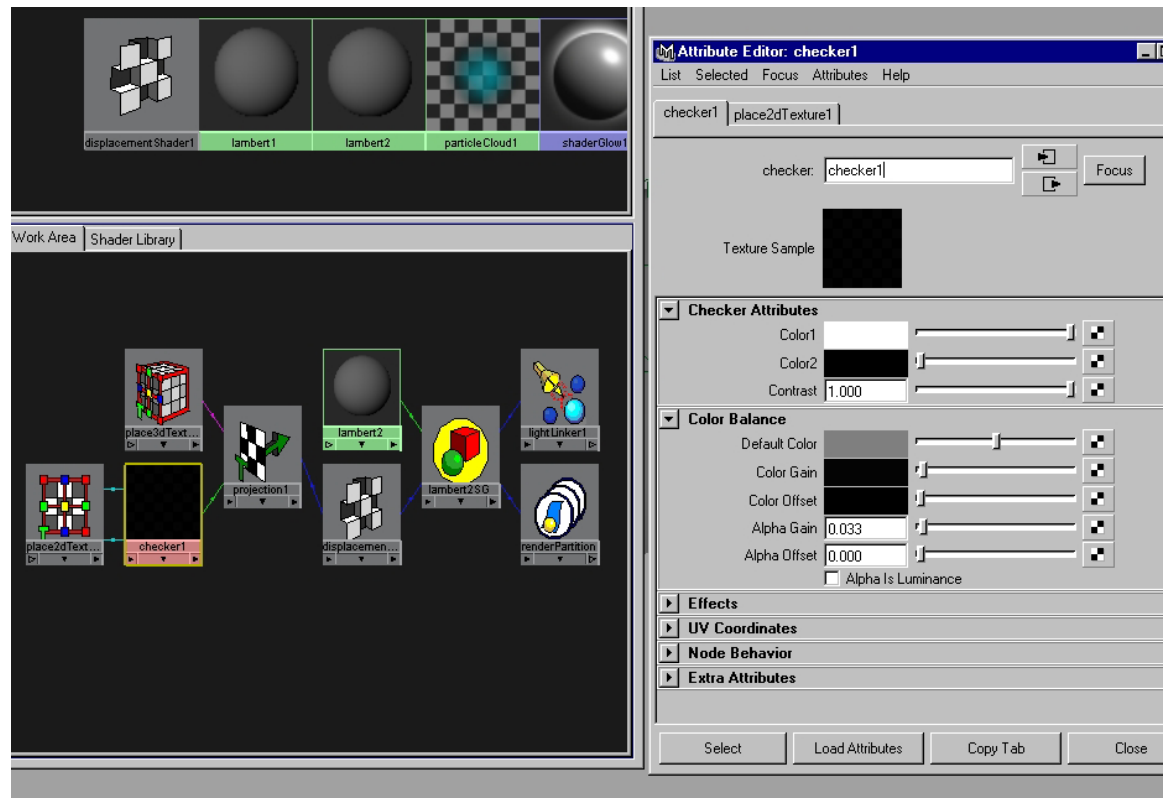
In this scene you will see a simple polygon mesh representing perhaps an outdoor terrain. It consists of 9 faces. If this geometry were to represent a very large area the texture maps needed would have to be extremely high resolution. As discussed before, a more appropriate technique would be to use tileable textures. The previously used technique could be applied, however, we will look at an interesting alternative.

Displace to polygon tool

Displacement maps in Maya are normally associated with Rendering and the ability to create the appearance (in render) of more detail on a piece of geometry than is actually there. Beyond this displacement maps in conjunction with the "displace to polygon" tool can be used to create new geometry based on the displacement map. This is the basic concept that will be used to subdivide this mesh evenly.

If you look at the shader which is on the geometry you will see that a 3d triplanar projector is being used to project a checker onto the geometry through displacement and the color gain attribute on the checker node has been brought down to almost zero (black). Selecting the mesh and exploring the displacement map attributes on the shape node in the attribute editor you will notice that the sample rate has been reduced to 3

and the extra sample rate has been reduced to 0. The result of these adjustments will be that the “displace to polygon” that will be done will have a very minor effect.



- **Select** the mesh in the perspective window
- **Modify > Convert > Displace to polygon**
- **Move** the resulting mesh off to the side in the X-axis

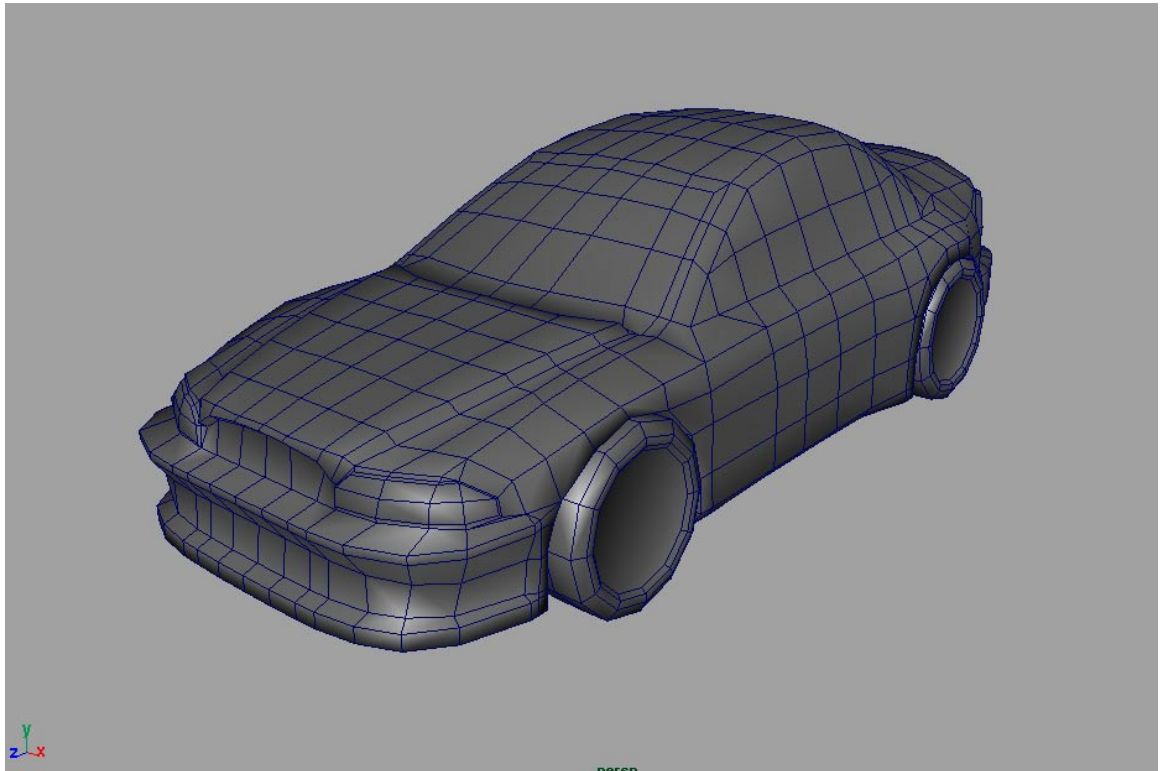
You will see that the results are a nice evenly split mesh. At this point the **Polygons > Quadrangulate** tool can be used to convert the triangular result of the 'displace to polygon' back into quads. This technique can be modified to create more or less faces by playing around with the color gain attribute on the checker and the sample rate attribute on the mesh displacement map attributes. Other texture maps can be used for interesting variations; it's really just a matter of experimentation.

UV Layout

This workflow is generating UV shells and fitting them into the 0 to 1 texture space with maximum use of this space. Some included concepts of importance are texture border location, aligning UVs and square layout.

- **File > Open File > 04_basecar.mb**

This file contains a polygon car, which we will use to explore the ideas around UV layout. Windows>open the UV texture editor and select the car you will see that the UVs are very scrambled. This is expected; it is very rare for a piece of polygonal geometry to have good quality UVs once modeling is finished and if a texture decal map is required work must be done to get to this point.



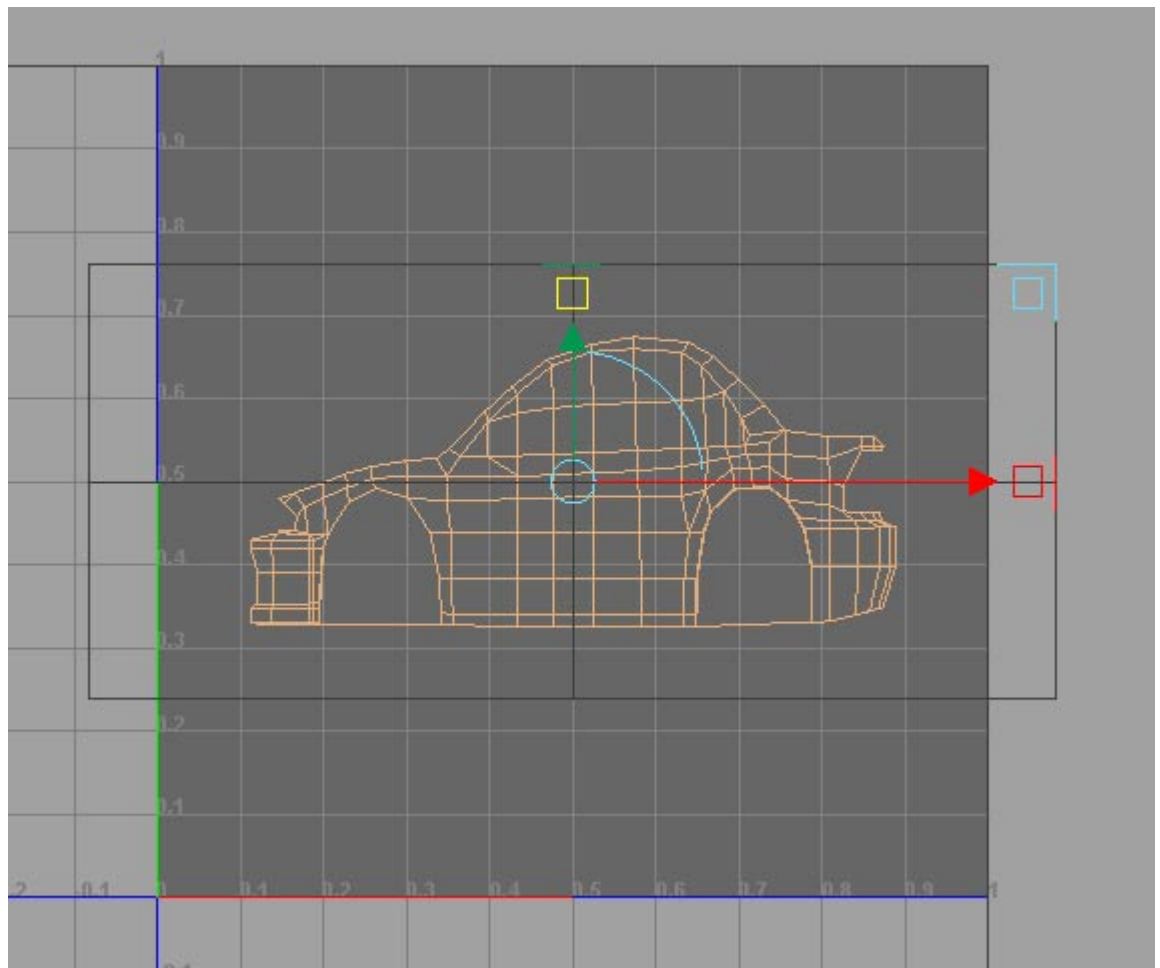
Maya allows for many methods of UV manipulation. Using the polygon projection techniques is one of the best approaches.

- **Edit > Quick Selection Sets**

You will see a bunch of quick selection sets of faces on the main body of the car have been created. This helps with organization before doing projections on the various parts of the car. We want to layout the UVs so that a decal sheet (a single image for all aspects of the texture) can be created. One projection will not result in UVs that can be used for this. A much better approach is multiple projections. If you pick the different quick selection sets you will notice that they have been organized to break the body up into specific projection areas. Once this is done it is simply a matter of selecting the selection sets and using **Edit Polygons > Texture**, and the appropriate projection (mapping).

Generating UV shells

- Edit > Quick Selection Sets > *car_sides*. This will select the faces on the two sides of the car body.
- **Edit Polygons > Texture > Planar Mapping – options** and set to project on the X-axis.
- **Window > UV Texture Editor**
- Within the UV texture editor window, use the manipulator to **scale** the side UVs down and make them more proportional (for less distortion).



Scale the UVs

- **Move** the UVs for the side of the car down and out of the 0 to 1 range.
- **Select** a single UV on the right side of the car and in the UV texture editor window hold ctrl and hold down the right mouse button. This will open up a very handy marking menu, which allows for quick selection and manipulation of UVs. Go to the “to shell” item. Now all the UVs for that UV texture shell will be selected. Go back to the marking menu and go the item “to face”.
- (Within the UV texture editor) **Polygons > Flip UVs – options** and set to flip vertically.
- **Move** this shell as required in the UV texture editor so the two sides are stacked in a vertical row. Leave some room between them for the roof of the car.

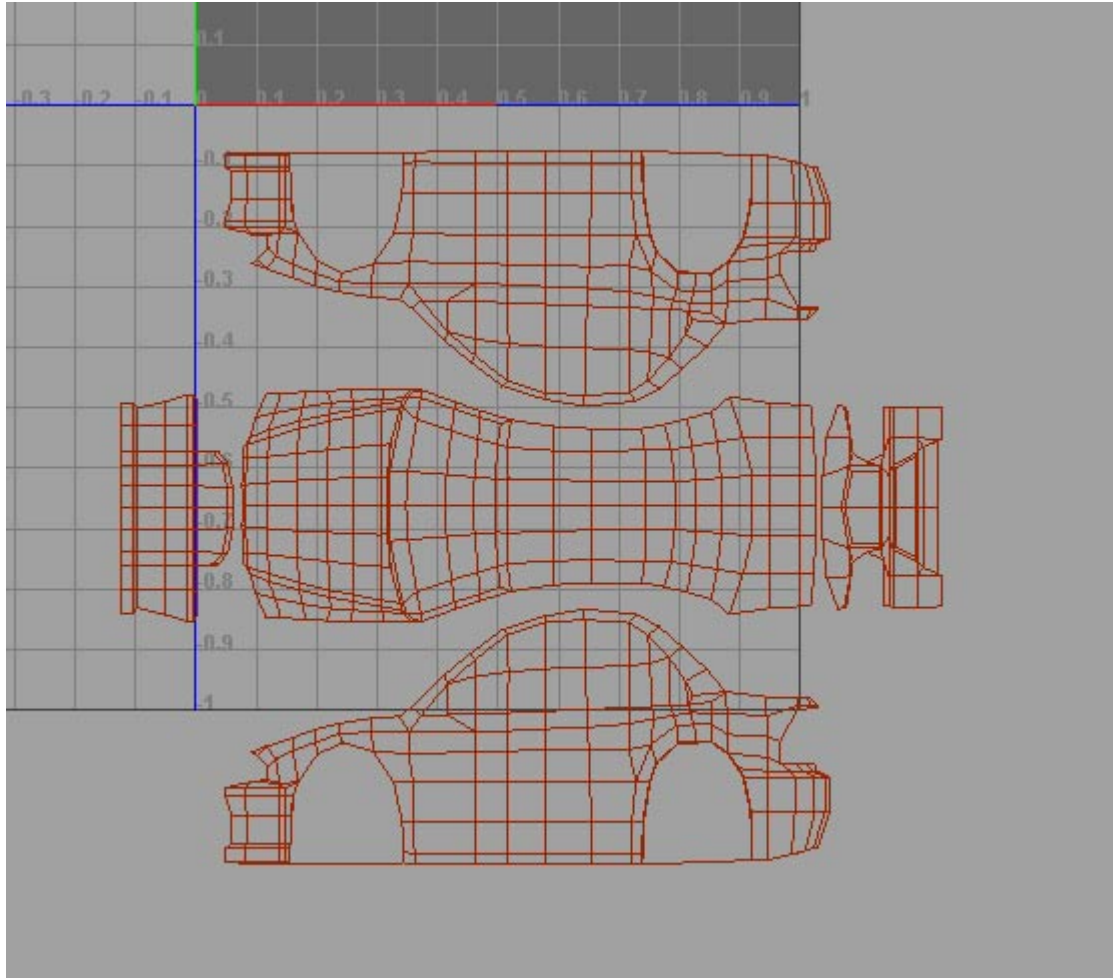
Continue using the same technique for the following quick selection sets:

Car_back

Car_front

Car_top

Use the transform tools in the UV texture editor window until the following layout is achieved.



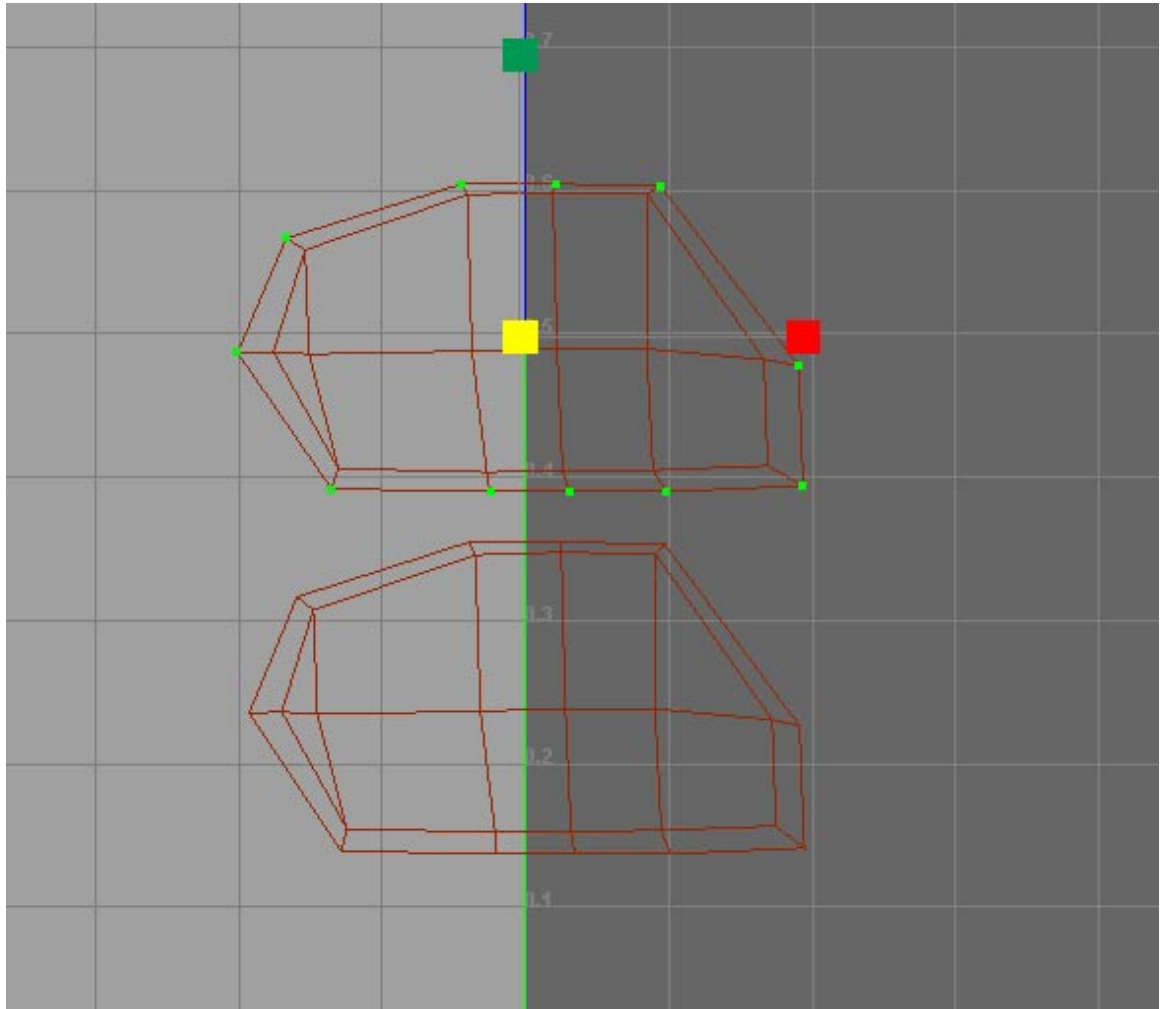
Desired layout

Continue working with the remaining quick selection sets. The front and rear lights can be planar projected or as an alternative, a cylindrical projection can be used.

- **Edit > Quick Selection Sets > *car_backlights***
- **Edit Polygons > Textures > Cylindrical Map**
- In the UV texture editor adjust the projection manipulator in order to shape the UV shells appropriately.

The two rear lights are going to have the same texture map on them. We can therefore place the two UV shells on top of each other.

- **Select** one of the two rear light shells and flip it to match the other in the UV texture editor. The cylindrical projection may have overlapped some of the UVs. Select and manually move any UVs that require adjustment.



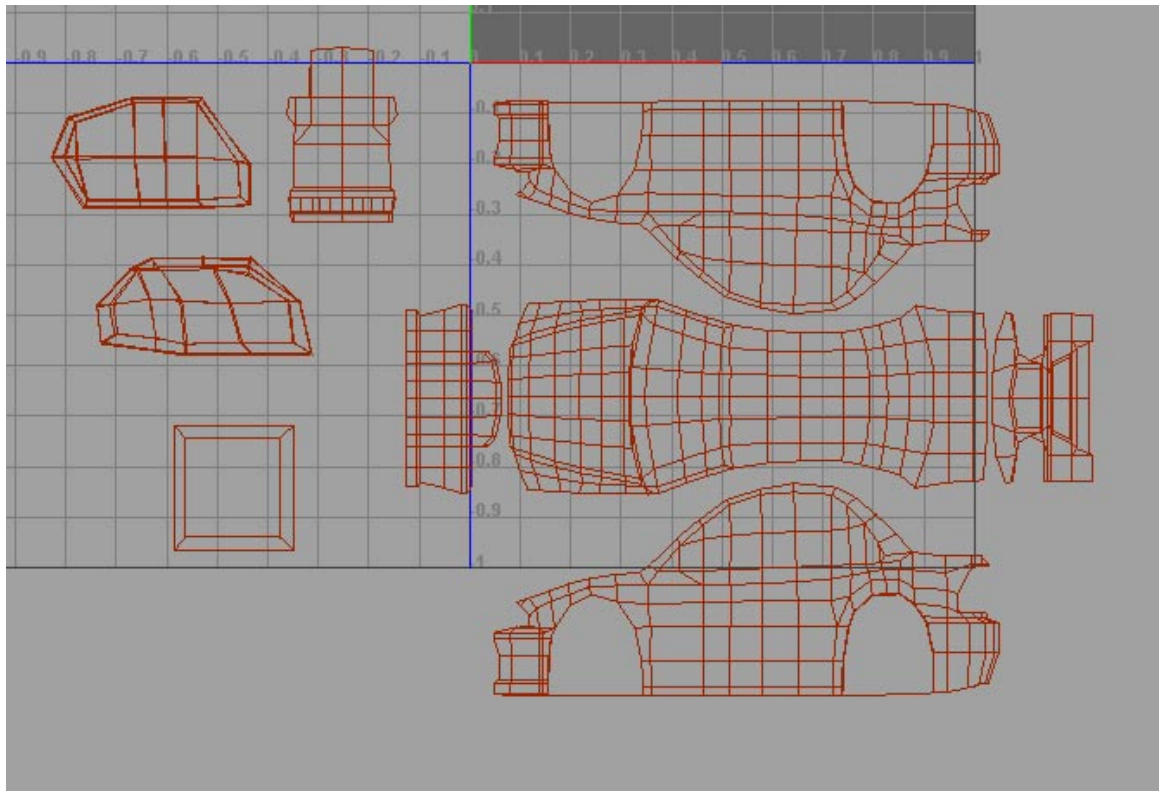
Place the rear lights together

- **Place** the two UV shells on top of each other.

You may notice a very small offset between some of the UVs. If you wish to have the UVs matching exactly you can use one of the bonus game scripts to do this.

- **Select** all the UV for both rear lights
- **Bonus Game > Adjust UVs > Grid UVs – options** and set grid U and V to 256 or 128 and move UVs to pixel center. This may not give you the results you're looking for all the time but it can help.

Repeat the same process for the front lights as you used for the rear lights. The side vents and the car bottom are the two remaining sets of faces that need to be dealt with. A planar will work fine for these areas. The two side vents can be placed on top of each other like the lights.



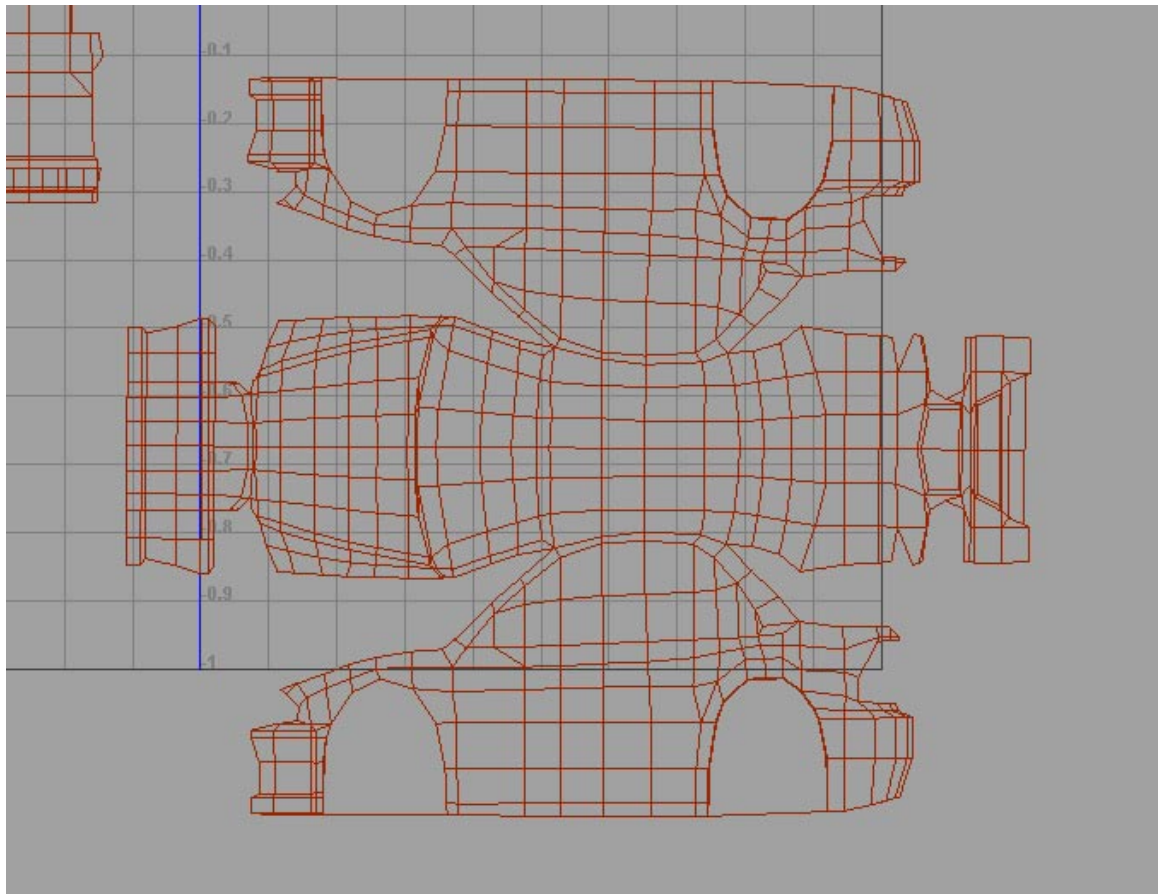
Shells generated

Wherever possible it is always better to have less UV shells. It makes it a lot easier to deal with things and even more important, it will reduce the number of texture border edges. Texture border edges are the visible edge where a UV shell ends. They are noticeable on the geometry because it can be hard to match the texture color exactly at two related UV shell edges. If the number of these edges can be reduced, the final result will be improved.

We will connect the roof to the two sides and the hood and trunk to the front and back UV shells.

- In the UV texture editor **move** and **scale** the four UV shells we are dealing with as close to each other as possible.
- **Select** the four edges along the roof edge on one of the two sides of the car. You will notice that the related edge will also highlight.
- **Edit Polygons > Texture > Move and Sew UVs**. The two shells have now connected.

Repeat for the other roof edge and the four edges at the front and back of the car until the following has been achieved.

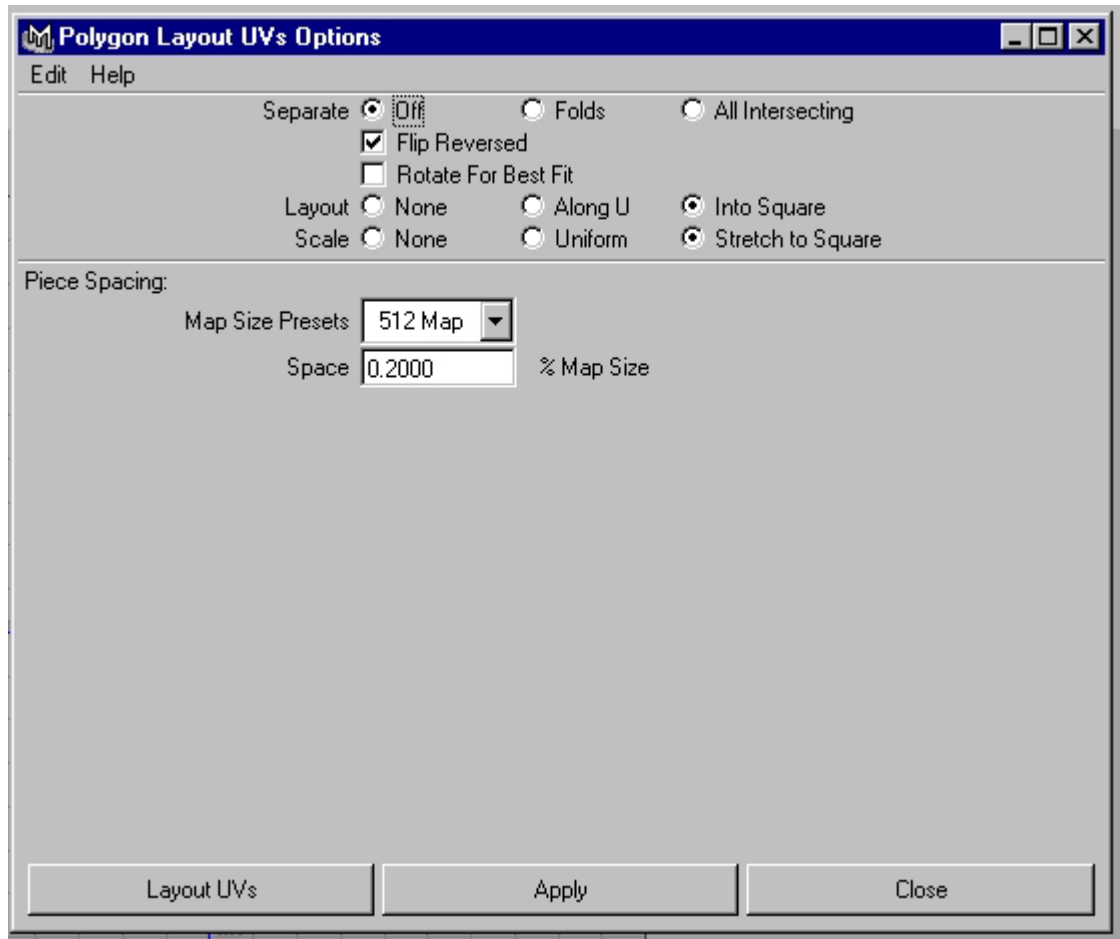


Sew the shells together

Setup for a decal sheet

With the desired UV shells created we now need to move them back into the 0 to 1 space.

- **Select** the main UV shell and then switch selection to faces.
- **Edit Polygons > Texture > Layout UVs – options** and set the following:



Lavout UVs options

Layout UVs allows you to move that shell back into the 0 to 1 range scaling it so it stretches to the whole range. The map size presets options (512/0.2 %) will leave a single pixel offset from the edge of the range. This prevents any texture bleeding.

- **Scale** and **move** the remaining UV shells into the four corners of the 0 to 1 range. Once this is done a UV snap shot can be taken from within the UV texture editor window.
- **Polygons > UV Snap Shot**. Make sure to set the image size to 512 x 512.

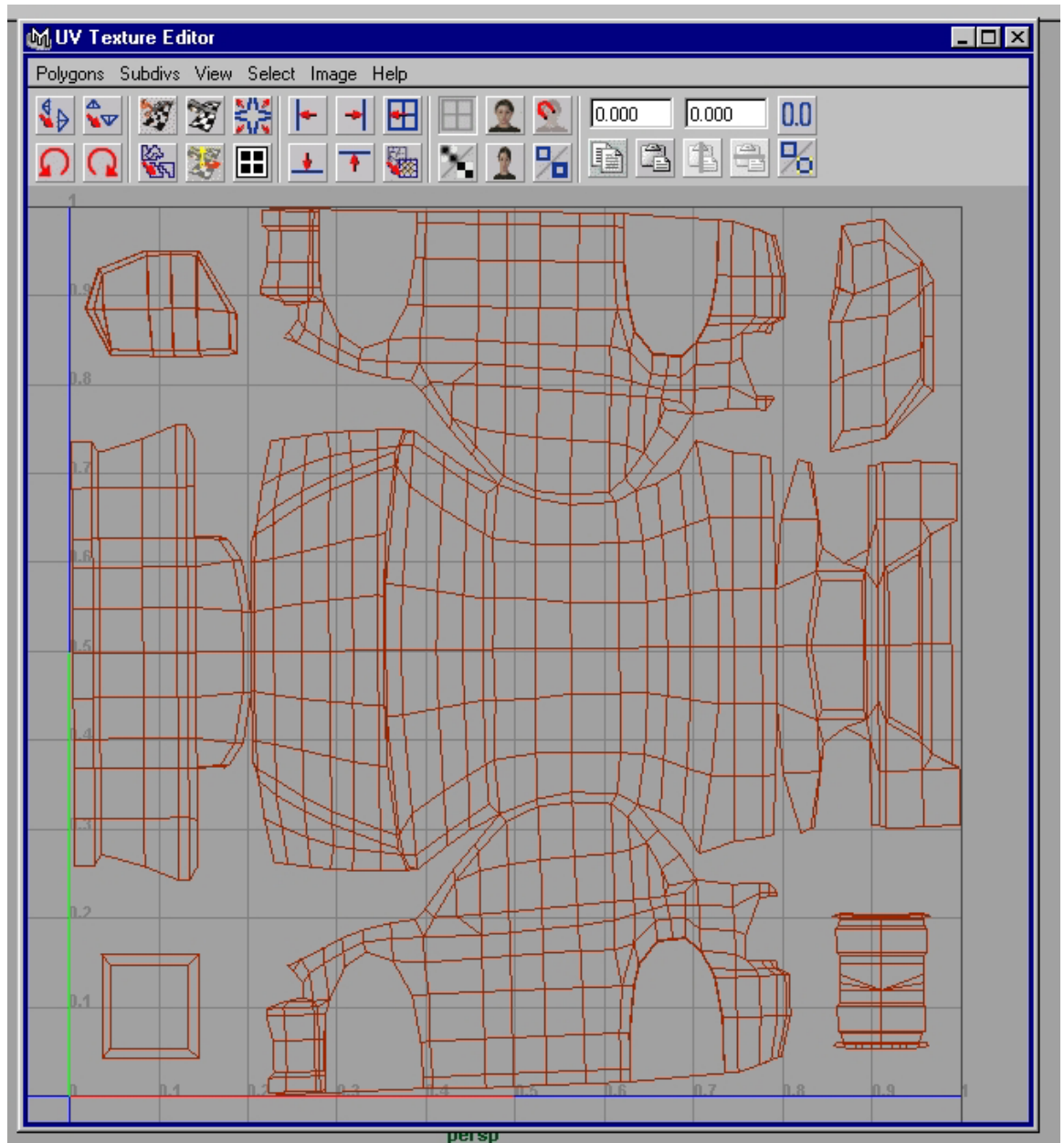
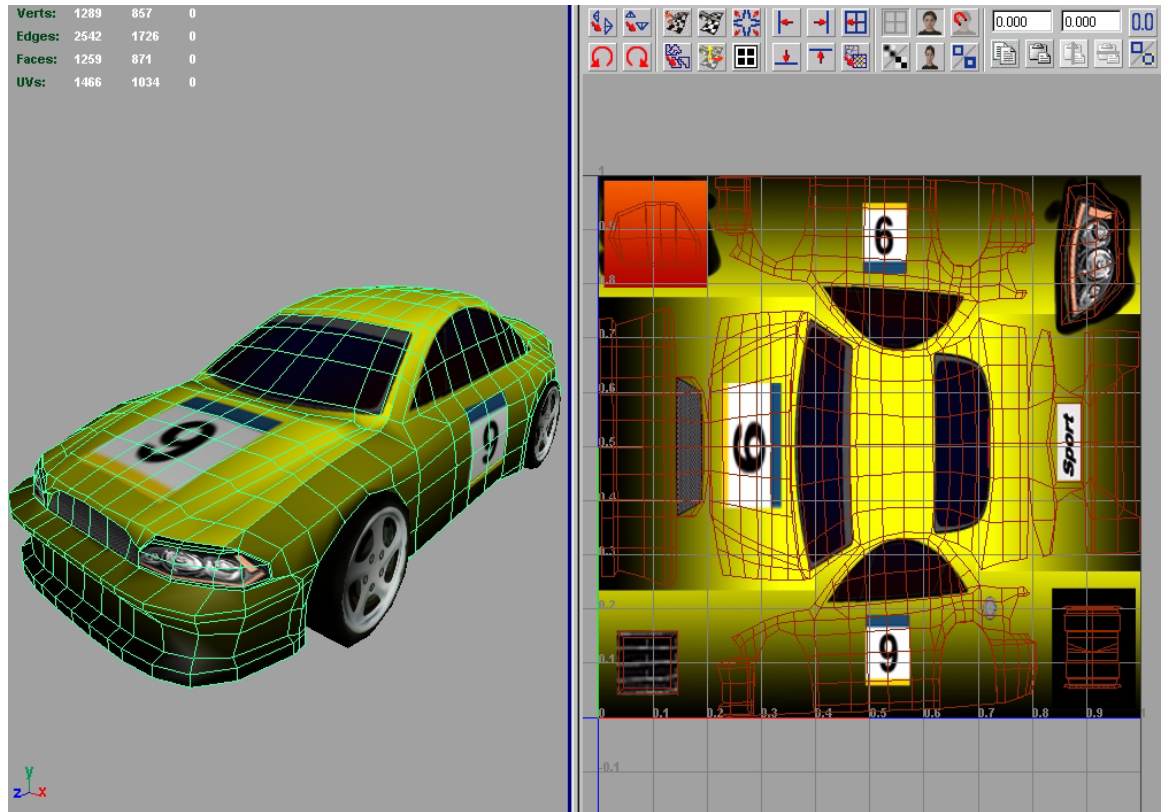


Image map ready

The resulting image file can now be brought into a paint package for texture map creation.

- **File > Open File > 05_finishedcar.mb**

This is a completed version with a decal sheet texture map. The wheels and rims have also been textured. Planar mapping was used to create the UVs and Normalize UVs was used to fill the 0 to 1 range.



Finished car

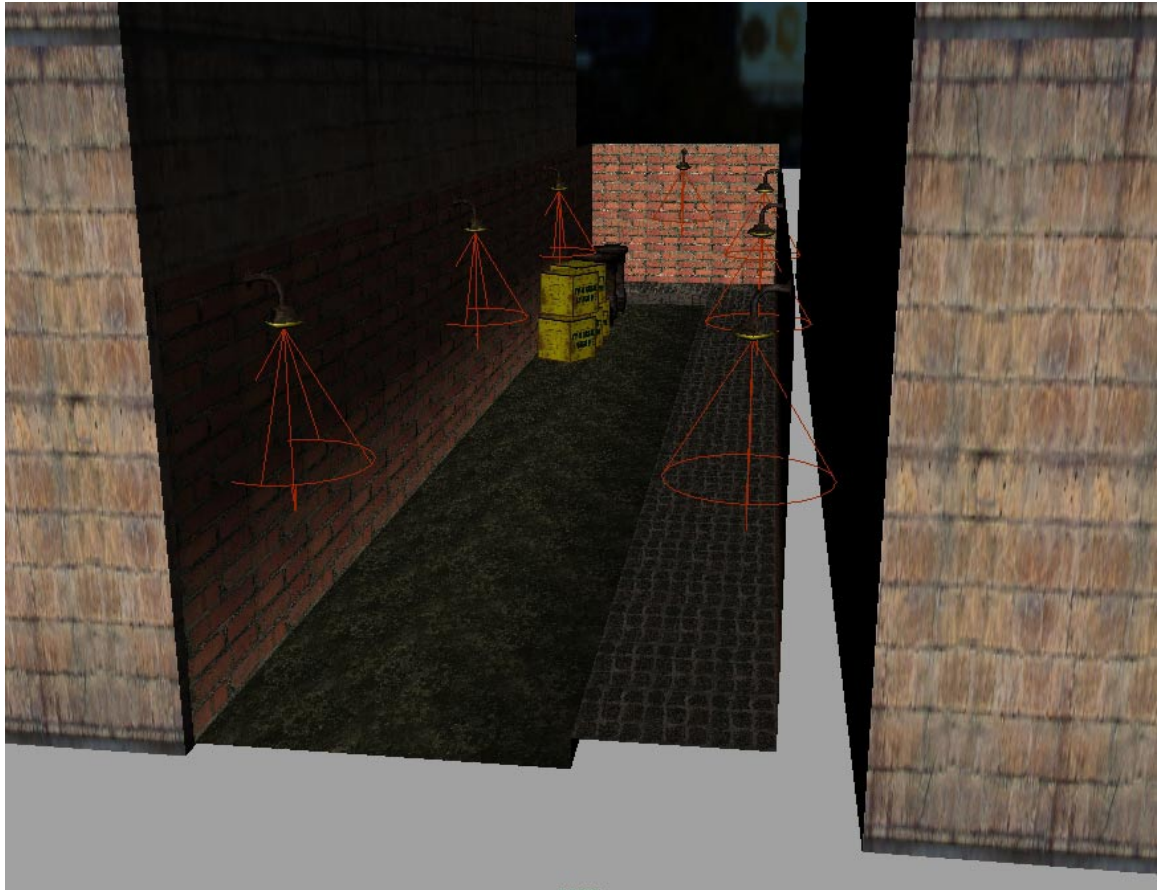
II. USING SHADING TRICKS TO ACHIEVE COLOR AND LIGHTING EFFECTS

Once you have achieved a good UV layout and texture maps have been applied to your geometry lighting and shading will start to come into play. If you are building content for video games the techniques for lighting or faking light starts to become very important. The desired overall ascetic will require a good grasp of the options available in Maya.

Baking lights

Dynamic lighting is a very powerful technique to use in video games. The primary problem is that the amount of dynamic light allowed in the game engine will be limited. This dilemma will require the illusion of lighting to be built into texture maps. Maya can be a great tool to help achieve these desired results.

- **File > Open File > 06_fullalley.mb**



Textured alleyway

Within this scene you will see the original alley that was worked on earlier and some additional elements: some props, lights, and a background cityscape. The lighting within this scene is very static and ideally it should be baked into the texture maps or used in a layered texture situation. Below are the steps to reproduce this.

Creating a light map

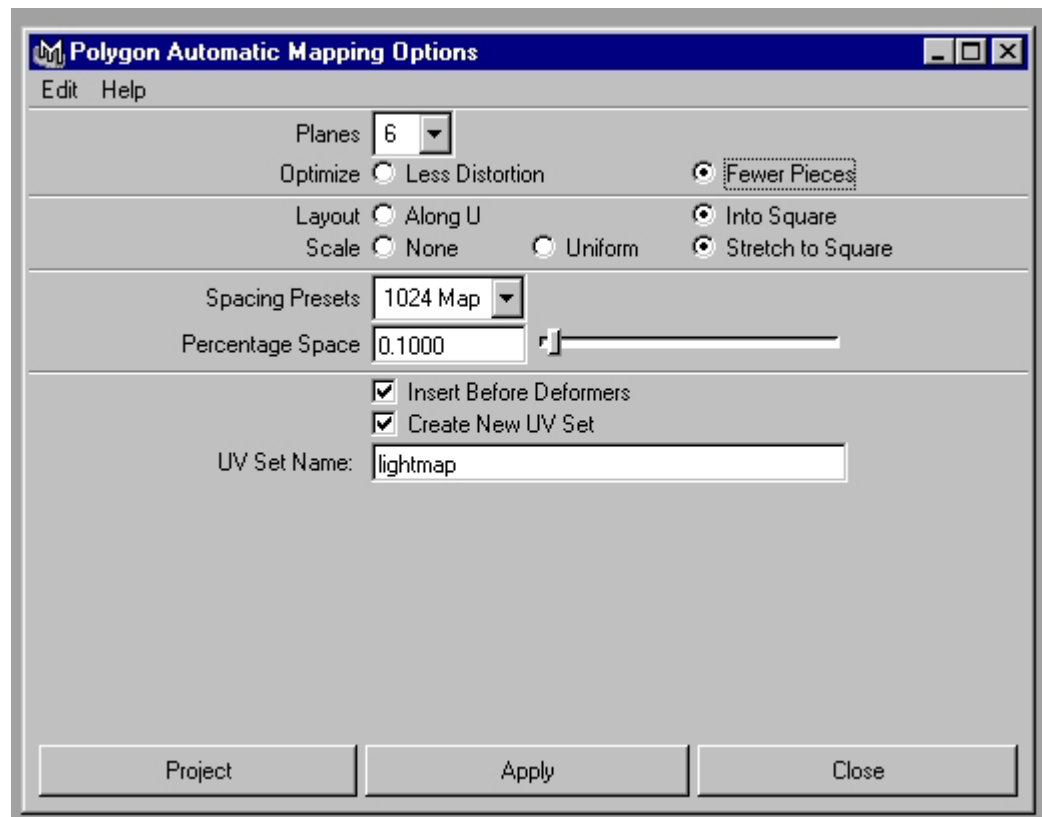
First we need to create multiple UV sets to deal with the various aspects of the layered textures.

- **Select** the alley geometry
- **Edit Polygons > Texture > Copy Current UV Set.** Name the new UV set "*textureUV*"

Now we need to create new UVs that are suitable for a light map. Due to the fact that the texture maps on the geometry are tiled across individual faces it would be impossible to bake the lighting information into these individual tiled textures. A totally separate texture map (light map) must

be created and then layered over the tiled textures using a layered texture. A single light map will be used for this entire piece of geometry and will be layered with three different tiled textures (the walls, the ground, the sidewalk).

- With the alley still selected, **Edit Polygons > Texture > Automatic Map – options** and set the following:



Polygon Automatic Mapping Options

You will notice that a very high resolution has been set for the “Spacing Presets”. In this case I want a very detailed light map; in most games applications this light map would be too large, a more appropriate size would be 128 or 256. Automatic map uses a multi planar mapping technique if you look at the resulting UV layout in the UV texture editor you will notice that the texture shells that have been created are all square or rectangular and horizontal or vertical with the UV space. This is ideal for light mapping.

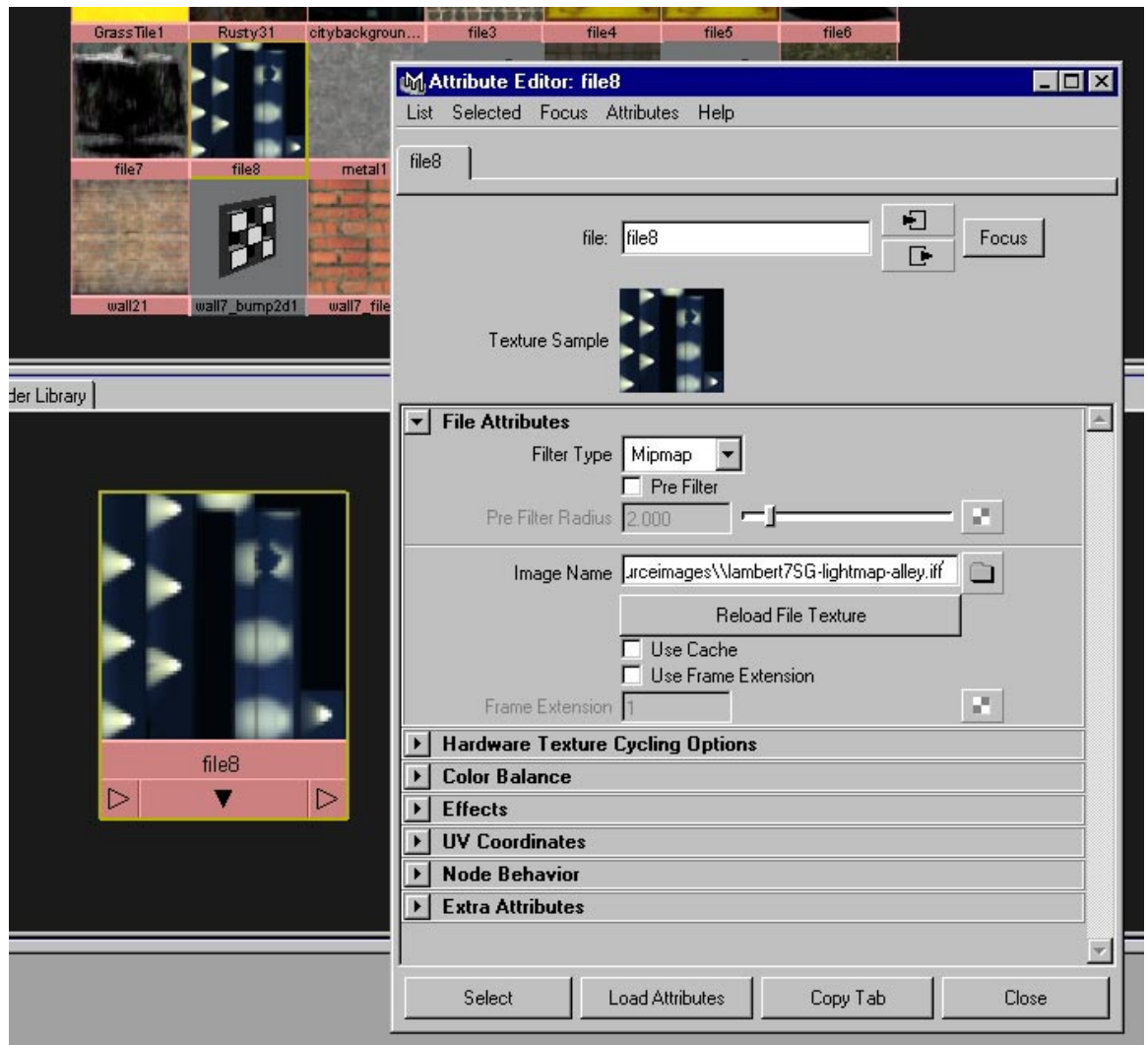
With the UV sets created we are now ready to create the light map. Convert solid texture is used to do this. The first step is to create a plane white Lambert shader this will be our base for the light map.

- **Windows > Render Editors > Hypershade**
- **Create** a new Lambert shader and set the color to full white, rename it “lightslambert” and assign it to the alley geometry.

Convert solid texture from the menu item cannot be used in this case to create the light map. It will convert using the default UV set. In order to get the tool to use the “lightmap” UV set the MEL command must be used.

```
convertSolidTx -antiAlias 0 -sh 1 -alpha 0 -
doubleSided 0 -componentRange 0 -resolutionX 1024 -
resolutionY 1024 -fileFormat "" -uv lightmap
lambert7SG alley;
```

The resulting texture map can now be used in a layered texture.

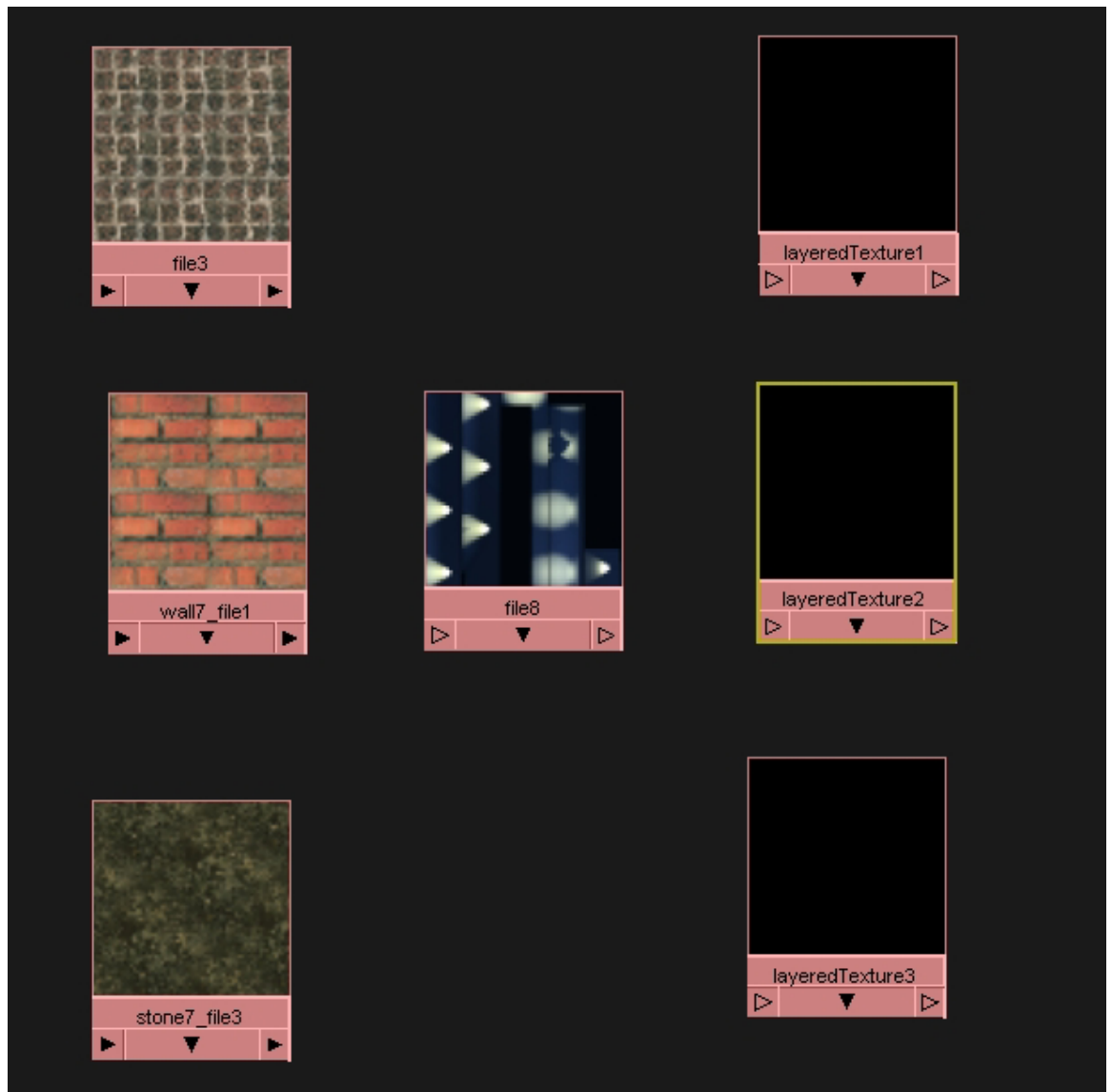


Our light map

Layering over tiled textures using a layered texture

Now layer this light map with the three original texture maps.

- In the hypershade window create three layered texture nodes. Drag them and the three original textures and the light map texture into the working area (of hypershade).

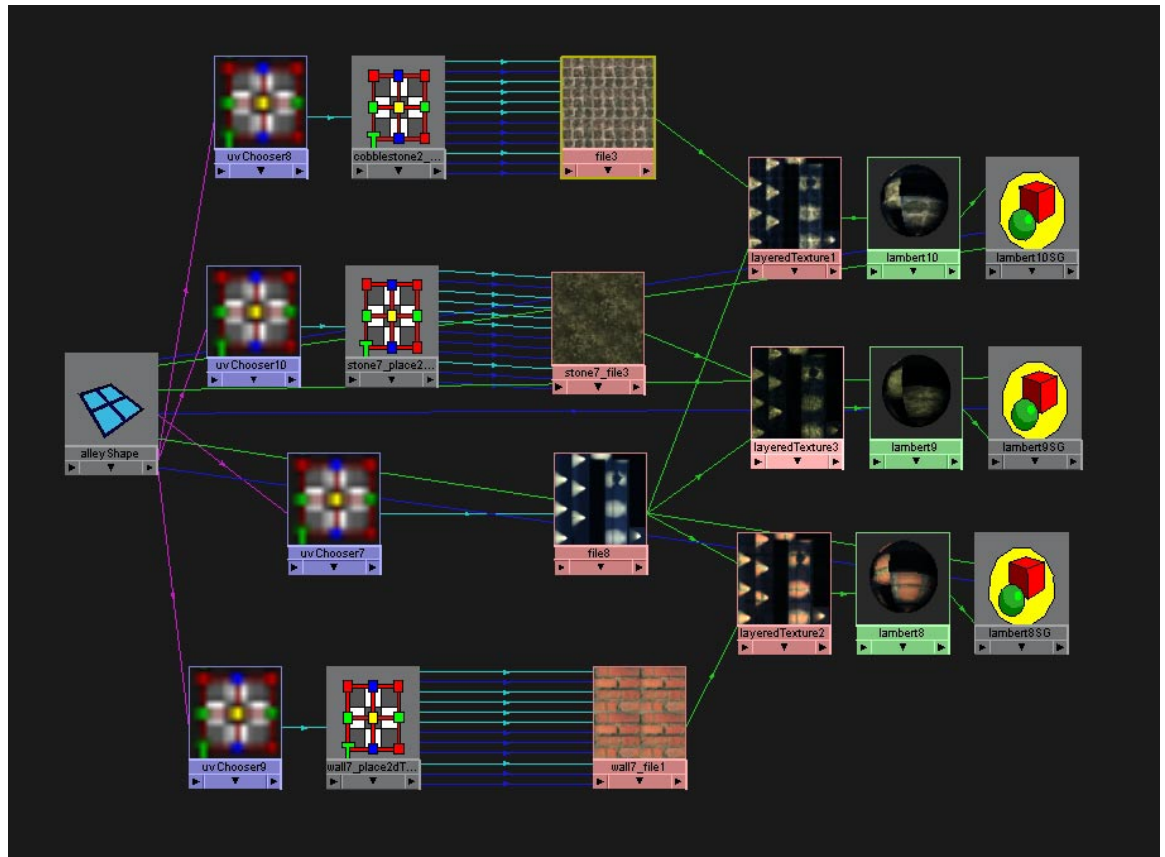


Hypershade window

- Create the shading networks by **connecting** the out color of the three original shaders to the input color 1 of each of the three

layered texture nodes. Connect the light map texture out color to the input color 0 of each of the three layered texture nodes.

- **Open the Attribute Editor** and set the blend mode of the light map to “illuminate” within each of the layered texture nodes.
- **Create** three new Lamberts and connect the layered texture to the shader color.



Making connections

- **Assign** the new shaders to the appropriate faces. Note that quick selection sets could have been used to make it easier to do this.

The results at this point will look a little strange. Maya needs to know which UV set is associated with each texture map involved.

- **Windows > Relationship Editors > UV linking > Texture Centric**

With this window open, connect the textures from the left side of the window to the associated UV map on the right side of the window.



The lit alleyway

The alley surface should now appear lit when in shaded mode.

- **File > Open File > [07_alleywithlightmaps.mb](#)** to review a completed version of this process.

Color per vertex

Color per vertex is an excellent way to layer color/lighting effects to achieve mood and give that little extra bit to your texture. It can also be a great way to fake lights and shadows.

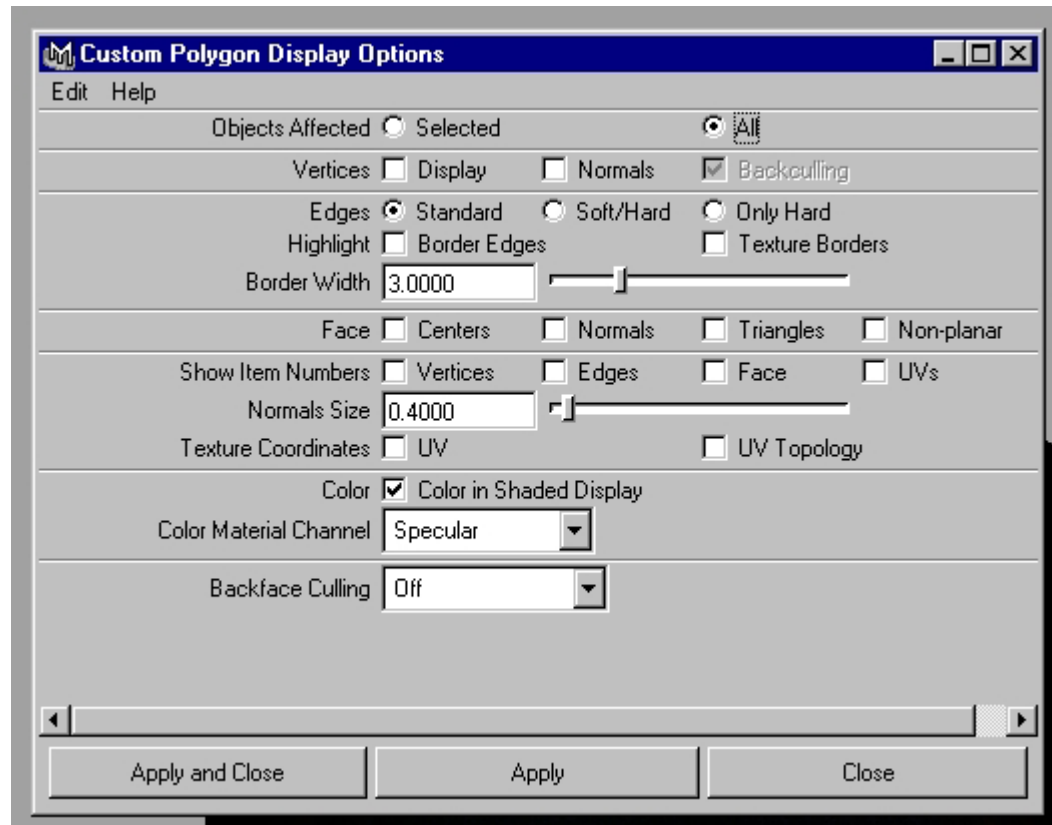
- **File > Open File > [07_alleywithlightmaps.mb](#)**

You will see a series of crates towards the end of the alley. We will look at how color per vertex can be used to add a radioactive glow effect to this area of the scene. Color per vertex gives us the ability to tag individual vertices with RGB color data.

Faking a glow

Before we begin, we should set how Maya is going to display color per vertex for us.

- **Display > Custom Polygon Display - options** and set the following:



Custom Polygon Display Options

The color per vertex will be applied to the specular channel of the materials involved.

- **Select** one of the crates
- **Edit Polygons > Color > Paint Vertex Color Tool**
- Set the options to paint a mid tone green color and using the artisan brush paint the front of the crate.
- Continue painting the crates with this technique and then move on to paint the ground in front of the crates to give the illusion that the crate glow is generating some lighting.



Radioactive crates

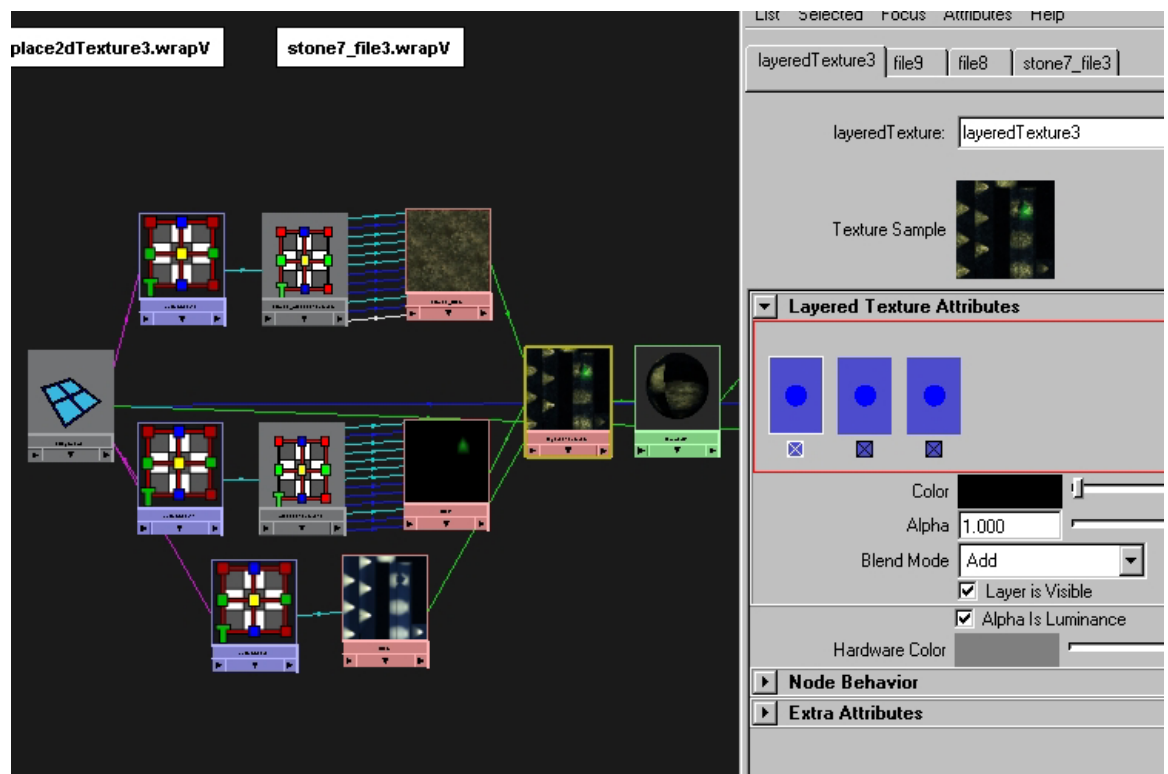
Color per vertex information as a light map

Color per vertex can also be stored as a light map. This is a very powerful way to layer over the current color per vertex in order to free things up to paint more variations over top.

- **Select** the *alley* geometry
- **Right Mouse Button select** over the alley geometry and in the marking menu items go to the “UVsets” item and set the current UV set to “lightmap”.
- **Edit Polygons > Colors > Paint Vertex Colors Tool – options** and go to the “map” tab in the option box. You will see an area for “export attribute map”. Set the map size X and Y resolution to match the original light map created (1024). Set the map name to an appropriate location and name and press the “resave” button. In this case “greenlightover.jpg” is the file created.

The new texture map can be added to the layered texture for the road area of the alley geometry.

- **Windows > Render Editors > Hypershade**
- **Create** a new file texture and assign the “greenlightover.jpg” to it.
- **Drag** the layered texture shader and the new green light texture map into the working area.
- Add the green light texture map to the layered texture and place it at the left side of the layering (the top) set the blend mode to “add”. If you want you can experiment with other blend methods.



Layered texture with color per vertex light map

The green color per vertex information has now been applied as a texture map.

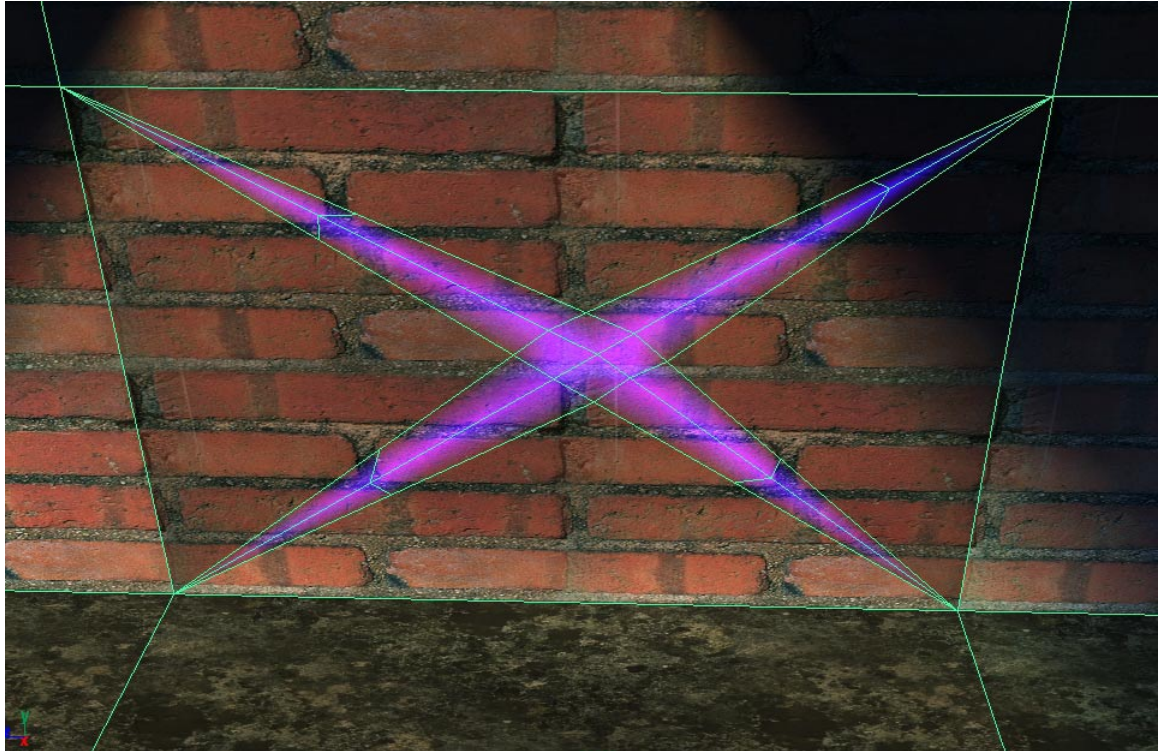
Color per vertex and topology

The results that you achieve with color per vertex are very dependent on the topology of the geometry that you are working with. If your goal is to have a very sparse additional coloring representing a broad area of lighting then a

low polygon count (spread out vertices) may be more appropriate. However if you wish to use color per vertex to create detailed additional shading it may be necessary to add detail into the geometry just to deal with the color per vertex data.

- **File > Open File > 09_alleycpvtag.mb**

You will see an “X” on the alley wall. This color per vertex data is only possible because of the extra faces that have been added in this area.



Special thanks to Cory Mogk, Kevin Pinchbeck, Tim H. Brown, Martin Crawford and Jeremy Keays