In the previous chapter we discussed how flexible the pipeline was when using Unity, and how it is often preferable to not tear through all the art assets before starting on game mechanics and scripting. However, in the 3D art asset pipeline, there are some fairly clearly delineated steps that need to be followed.

UV layout is the second step of these strict guidelines. Generally, it is important to make sure the geometry (model) is just as it should be before starting your UV layout. If a UV layout is created, and then the geometry is changed, the UV layouts have to be redone.

Similarly, the UVs need to be complete before extensive work is done on texture creation. Since UVs define how a texture is “attached” to a geometric form, if UVs are changed (or the form), the texture will no longer lay appropriately across it.

So, although there is almost always some adjustments as the UV is laid out (stray polygons that aren’t needed are found or additional geometry is
needed), the more effectively this particular step of the scenography creation process (i.e., modeling, UV layout, texturing) is completed, the more efficient the process will be.

**UVs**

The process of creating and editing UVs can be one of the most frustrating in all of 3D. There is little intuitive about them. Unfortunately, understanding UVs and how to manipulate and utilize UV space is one of the most important aspects to visually believable projects—and absolutely critical to effective games.

So what are UVs? Well, they really aren’t so much of a *what* as a *where*. UVs are coordinates. They are involved in the complex problem of how to take a 2D texture and wrap it around a 3D form. Think of the problem as much like that of cartographers attempting to create a map of the earth. How do you take a sphere and flatten it out into a flat shape on a page?

**UV mapping** is the process of taking three-dimensional shapes, and *unfolding* or *unwrapping* them into a two-dimensional representation where a texture can be painted on them (Figure 3.1). UV coordinates on a surface are the locations at which to “pin” the textures. Once these locations are pinned down, even if the surface flexes or distorts, the texture will flex and distort with it.

At the end of the day, UV mapping is what helps a texture look “right” on a surface. A poorly laid out UV map yields unsightly stretching and pinching of the texture that is applied to it. A well-executed UV map produces believable surfaces with textures that belie the usually simple geometry beneath it. In games especially, where textures do much of the visual work, a good UV map is critical to visual success.

In the following tutorial, we will be looking at three ways to create UV maps. Through this process, it will be important that we understand texture space and how it relates to 3D space. We will look at the UV Texture Editor within Maya and how it allows for the manipulation of UVs. We will look at Maya’s automatic mapping options, its unfolding capabilities, and manual mapping and manipulation of UVs.
Exploring the UV Texture Editor

When any of Maya’s primitives are created, these primitives include a default UV layout. For instance, go to Create>Polygon Primitives>Cube. With this cube selected, choose Window>UV Texture Editor (Figure 3.2). In the UV Texture Editor, the six-sided cube will appear unfolded. All the faces are laid out flat in the top (1,1) quadrant of the UV Texture Editor space. By default, the background here will be a medium gray color; this is actually the texture that is applied to the cube. Within the UV Texture Editor, right-click and hold, and a hotbox will be presented that allows for choosing which type of component to select. The regular Edge, Vertex, and Face are presented, as well as UV.

After choosing a component type from the hotbox, these components are now selectable within the UV Texture Editor. At first blush it seems that the only difference between Vertex and UV is the color of the component when it is selected (vertices highlight yellow and UVs always highlight green). This is because Maya’s primitives have their UVs located at the vertices of the form. However, don’t confuse the two. Vertices and UVs are one-dimensional—they have no geometry of their own—but they serve much different functions.

In fact, once faces or edges are extruded new vertices will always be created along with the faces created (there cannot be faces without the vertices); however, these new vertices will not necessarily have new UVs attached to them. Further, although UVs are visible in 3D space (i.e., the View panels), they can only be altered—moved, rotated or scaled—in 2D space, within the UV Texture Editor.

This transition from 2D texture space to 3D space is often a hard thing to visualize and even harder to explain. However, it becomes much easier to comprehend once it is seen in action. Because of this we won’t spend too much time pounding on the theory further. Let’s start seeing it in action.
Tutorial 3.1: Game Level UV Layout, Tools, and Techniques

In this tutorial, we will continue working on the EntryWay we began earlier. Although we won’t explicitly map every object, the techniques covered here are extensible to all the objects in this scene, and indeed into all the scenes in the rest of the game.

**Step 1:** Create a checkered pattern. In Photoshop, create an image that is 32x32 pixels. Create guides horizontally and vertically in the middle of the canvas. Fill the top left and right with black. Select all, choose Edit>Define Pattern. Name the pattern.

**Step 2:** Create a checkered base texture. Create a new image that is 1024x1024. Choose Edit>Fill. In the Fill dialog box, choose Use:Pattern and pick the previously created checkered pattern and click OK. The results should look like Figure 3.3. Pretty exciting, eh?

**Why?**

Why a checkered pattern? Why not just use the checkered texture that’s canned in Maya? Why 1024x1024? All good questions. As we attempt to lay out the UV map, it will be important to know that the mapping we have created does not cause stretched or pinched textures. Further, sometimes we want to know how much texture is actually on any one part of the mesh. By using a checkered texture, we can have a quick reference of how the texture is mapped across the surface, and how we are distributing the texture in 3D space. The reason to build your own instead of using Maya is that the Maya checkerboard is set at the number of checks across and down. By creating our own, we can create a more densely populated collection of checks. The file is 1024x1024 because we want to get in the habit of building by **power of two** when working with texture (much more about this later).
**Step 3:** Save as Checkerboard.psd in the `sourceimages` folder of the project file.

**Why?**
The `sourceimages` folder is where Maya goes to look for textures when a new material is created. There are lots of other seductively logical folders (images, textures, etc.), but don’t succumb. The only folder that matters for saving textures within Maya is `sourceimages`.

**Creating and Applying New Material**

**Step 4:** Create and apply a new material on a column. Right-click and hold on the column that we previously beveled. Choose Assign New Material. In the Assign New Material window, click Lambert. This should open a new lambert material in the Attributes Editor. If it does not, press Ctrl-A, and the Attribute Editor will show the attributes of this new material.

**Why?**
Why a lambert? Lamberts are matte materials that Maya draws really quickly and well. More importantly for us, when Unity initially brings in the Maya-created objects, everything looks like a lambert. Any other attributes present in other materials (specular in Phong, etc.) has to be redefined in Unity anyway. So spending a lot of time tweaking settings in Maya is wasted since it has to be redone. For this reason, lamberts are quick, easy, and predictable ways to get textures applied and to see the general look of the scene.

**Step 5:** Name the material. Name the material. Name the material. Did I mention to name the material? Name the material EntryWay_Column_Mat.

**Why?**
Beginning modelers or beginning game builders often skip effective naming and pay the horrible price later. When moving assets between applications (like from Maya to Unity), assets will be tied together differently, and sometimes in the production process materials will become disconnected from objects. If there are 100 materials named lambertx, you will be immensely slowed in your work. Take just a little time to name materials what they are, and you will save many hours later. Especially if you ever hope to work with others as part of a team—effective naming is critical to smooth work flows and keeping your job.

**Step 6:** Create a Render Node for the Color attribute. In the Attribute Editor, at the far right of the Color channel is a little button that (ironically) looks like a checkerboard. Click this to bring up the Create Render Node window. Tell Maya to import a file to define the color attribute of the material by clicking the File button.
Step 7: Choose Checkerboard.psd to define the color attribute. After following the previous step, the Attribute Editor will change to display a file1 node with an Image Name input field. Click the folder icon next to that input field, and you should be taken to the sourceimages folder of the project. Double-click Checkerboard.psd.

Step 8: View the scene in Textured mode. Do this by pressing 6 on the keyboard. This will show the scene with any applied textures and should yield a strange result as seen in Figure 3.4.

Why?
So what are we seeing here? Think way back to when we first began building this column; the shape we began with was a cube. The parts of the column that have texture are the original faces of that cube. The polygons we extruded off of that shape are gray; they have no UV coordinates, and thus have no location for the texture to be “pinned” to. Further, the texture that is applied to the main shaft of the column is stretched and not efficiently organized.

Warnings and Pitfalls
If the folder next to the Image Name input field is clicked and you are not taken to the sourceimages folder of your project, stop. It means the project is not set right, and if you start maneuvering through file trees to find the right file, you could break the project on any other machine than the machine you are working on. If you find yourself in another location than what you anticipate, stop, save the file, and go back and set your project (File>Set Project).

Why?
The first way we are going to look at UV mapping is to start with Automatic Mapping. Generally, anytime software does something “automatically” be a little wary—this is no exception. However, Automatic
Mapping does create UVs for all the geometry we have, and once we have those UVs we can fix the map into what we need.

**Step 9:** Use Automatic Mapping on the column. With the column selected, choose Polygons>Create UVs>Automatic Mapping (Figure 3.5). In the UV Texture Editor window, choose Image>Display Image (to turn off the checkerboard), and see the shapes that represent the column.

![Figure 3.5 Results of Automatic Mapping](image)

**Why?**
At first glance all is well—that was amazingly easy. But take a look at the UV Texture Editor (Window>UV Texture Editor) and you’ll notice a couple of important things. In the UV Texture Editor window choose Image>Display Image (to turn the background checkerboard off), and see the shapes that represent the faces of the column. First, notice that there is a lot of the texture space that we are not making use of. In the game, we are paying for that space whether we use it or not, and it’s very inefficient to leave so much of that texture space unused. Second, notice that each of the sides of the column are separate. This means that the texture won’t crawl around the corners or the column—there will be a seam across every corner of our form.

Seams are inevitable. If a form is to be unwrapped, there has to be a seam somewhere, but we want to minimize those. To do this, we will be cutting up some of this map and reassembling (via sewing) other parts.

**Step 10:** Cut the heads off the columns. In the UV Texture Editor, right-click and choose Edge from the hotbox. Select the edges (there are really three there, although they are all in a line) shown in Figure 3.6. In the Editor window choose Polygons>Cut UV Edges.
Why?
We are cutting the heads off so we can sew up the side edges of the shaft. In the game, we will see much more of the shaft than the capital of the columns. So making sure there is only one seam on the shaft makes for a better experience for the game player.

**Step 11:** Move the head away via UVs. Right-click again and choose UV from the hotbox. Marquee select a few UVs (that are part of the column capital - maybe the top right corner of UVs) and then Ctrl-right-click-hold (I know it’s a lot) and choose To Shell (Figure 3.7). This will select the shell of UVs that make up the top of the column. Move them aside. Moving a UV shell can be done with the regular Move tool (press W to activate it) or in the Editor window choose Tool>Move UV Shell Tool and then move them aside (anywhere really).

Why?
What is a shell? Shells of UVs are shared UVs that share UVs, or edges. Because we cut the edges at the bottom of the column capital, the edges there were no longer shared, so when we selected a shell, it stopped there.
Step 12: Repeat steps 10 and 11 for the other sides of the column. Keep in mind that this all takes place in the UV Texture Editor. Figure 3.8 shows the capital chunks all separated.

Sewing Shells

Step 13: Sew the edges of the shaft. In the UV Texture Editor, right-click and choose Edge from the hotbox. Click (don’t marquee select), on any one of the outside edges of the shaft sections of the column. Notice that although an edge was clicked, a second edge will highlight somewhere in the UV Texture Editor. This is because it is the same edge but is split (which is why we’d have a seam).

With this single edge selected twice, choose Polygons>Move and Sew UV Edges. One shell will slide over to the other and the split edge will become one (Figure 3.9).
Why?
Sewing these edges up mean that the texture will be seamless as it crawls around that corner. For a better understanding of what this does, select the shell that is the newly sewn together shaft segment and use the Move tool to move this shell in the UV Texture Editor (we're moving the UVs through texture space here), and look at what happens in the View panel. See that along the edge that was just sewed up, the texture crawls across the corner without a problem, but on the other three corners there is a visual disconnect.

Step 14: Repeat for all the outside vertical edges of the column shaft. The final shell should appear like Figure 3.10.

Why?
Notice that there will still be one seam on either outside edge of the shell. This is OK and inevitable. As we finally position the columns we may look at ways to rotate this away from where the player will see it.

Step 15: Repeat the process for the column base. However, this time, instead of sewing up the vertical edges, we will sew up the horizontal edges that run across the top of the base. The completed shell will look something like Figure 3.11.
**Why?**

We are looking to reduce seams in places where the player will most likely notice them. For the shaft, it would be easy to see the vertical seams, so we take care to get rid of those. For the base, where we would be standing above it and looking down, the player would most likely notice seams as the form moved from the top to the side, so we get rid of those seams and leave the vertical seams by the ground.

**Step 16:** Repeat the process for column capital shells (Figure 3.12).

---

**Why?**

Why Move and Sew UV Edges and not just Sew UV Edges? Part of what the Automatic Mapping did for us was to evenly distribute our UVs so the checkers were indeed square. When Move and Sew UV Edges is used, Maya moves the entire shell and thus keeps the relative position of the UVs constant. This means the texture is not distorted in the process. Sew UV Edges moves just the edge, and in the process distorts the texture across the surface.
**Creating Games with Unity and Maya**

**Tips and Tricks**

Often, when working with objects that have had things like bevel applied to them, they will end up with little shards of polys floating around in the texture space. Keep your eye open for these, and look at where they could be resewn (via Move and Sew UV Edges) back onto larger chunks.

---

**Step 17:** Arrange the shells to better take advantage of the space (Figure 3.13).

---

**Warnings and Pitfalls**

Notice that in Figure 3.13 the shells have been scaled (the shaft now vertically fills the quadrant). Scaling the UVs is alright and in fact a good thing to do—UVs that use more of the texture space are UVs that allow a texture to show up higher fidelity in the game. However, in this case it is important that if any part of the UV is scaled, that all of the UVs for that object are scaled as well. This isn’t always the case, but in this case where we will be using a painted cement texture we don’t want to have the tactile parts of the texture (pocks and bumps) to be smaller in some places than others. Keeping the relative size of the UV shells will ensure that the surfaces get the right amount of texture space.

---

**Why?**

“Hey, wait a minute!” you may be saying, “that doesn’t look like you’re using very much of the texture space. I see a whole lot of empty space.” And you’re right. There is a whole lot of empty space left in this UV map. There are two things we could do with that space. One, we could resize our UV shells to better take advantage of that space—but if we did that we would need to construct our texture equally distorted to match the distortion in texture space. This can be harder than it sounds.

The second alternative is to simply use that space to hold the UVs of another object. This is the idea of **atlasing**, meaning that a **texture atlas** can be created that holds the texture information of multiple objects. A texture atlas is really just like any texture—only there are multiple objects’ texture information crammed into one file. This is actually a very useful optimization technique because the video card then simply draws the same texture on the same shader multiple times to represent a lot of
different objects on the screen. It takes much less of the video memory, and dramatically reduces the draw calls. Since in my scene I have other things that are attached to the columns (ladders and frames), I am going to place the UVs for those objects in the spaces of the quadrant that are empty, and then assign the same texture to all these objects. The idea of Texture Atlases are so powerful that we will be visiting them again.

Finally, when the UV was done for this level, the UV map for the single mesh EntryWay_Columns (which was actually the result of lots of combines) looked like Figure 3.14.

Further Optimization

**Step 18:** Duplicate, place, and combine this UVed column. Delete the old duplicates of the column and replace them with this one. When they are all placed, select all of them and combine them (Polygons>Mesh>Combine). Name this new polygon shape EntryWay_Columns. Finally, to make sure the manipulator handle for this new group makes sense, select Modify>Center Pivot.

**Step 19:** Repeat this process for other squarish shapes. Using the objects in my scene this included the beams above the columns, the roof, and various other obviously square structures. Those shapes that were UV mapped using this technique are shown checkered in Figure 3.15.
**Tips and Tricks**

Remember to create a new material for each object or groups of objects (that are going to share the same material). Although all of them will share the same Checkerboard.psd file as their color texture, we want to be sure they have their own unique material assigned to them.

**Tips and Tricks**

A nice thing about using the same checkerboard for all the UV mapping is that it gives a quick look at how much texture space each object will have. Knowing this helps show how high fidelity the texture will likely appear in a game. Notice in Figure 3.15 that objects that are close to the ground or on the ground have small checkers. This means they have a lot of texture space, which means there will be a lot of pixels used to describe this surface. Things that are far away from the player in the game (like the beams overhead) have larger checks because they won’t need quite as much texture since they won’t be subject to quite so close a scrutiny.

**Maya’s Unfold UV via Smooth UV Tool**

**Why?**

Automatic mapping works great if working with cubes or objects that are largely cubic in shape. But a scene full of crates makes for a pretty boring scene, and ultimately more organic shapes become important to most scenes that have any variation. Maya’s Unfold UV tool is a relative newcomer to the Maya toolset (at least in the configuration it currently uses). This kind of technique is available with other tools (see the free and powerful Roadkill [http://www.pullin-shapes.co.uk/page8.htm](http://www.pullin-shapes.co.uk/page8.htm)), but is nice to have available in Maya. In the next few steps we will be looking at how Unfold works within Maya in a simple shape. However, this tool is especially powerful for things like faces and other complex organic shapes, so we will be visiting it again.

**Step 20:** Assign new material (with checkerboard as the color texture) to the EntryWay_Earthwall object (if you are building off of the version of the map downloaded from the site; if you are using your own version assign the new material to whatever you are using for the big retaining wall). For a refresher on how to do this, be sure to view steps 4 through 7. Name the new material **EntryWay_EarthWall_Mat**.

**Step 21:** Use Planar Mapping to ensure the entire object has UVs. With the object selected, choose Polygons->Create UVs->Planar Mapping (Options). Change the Fit projection to: setting to **Best Plane**, and press Project (Figure 3.16).
**Why?**

Unfold works great; however, it only unfolds UVs that exist. As we have seen before, when faces or edges are extruded, often the new faces that are created lack UVs. Using Planar Mapping does a few important things for us. First, it ensures every part of a particular mesh has UVs. Second, it ensures all these are part of one shell. Third, by choosing Best Plane in the options, we give Unfold its best chance at providing a distortion-free UV map.

---

**Step 22:** Unfold (interactively) the UVs via the Smooth UV tool. In the UV Texture Editor (Window>UV Texture Editor), right-click the mesh and choose UV from the hotbox. Marquee select all the UVs. Select the Smooth UV tool either from the iconography in the top left of the UV Texture Editor or from Tool>Smooth UV Tool.

**Step 23:** Drag the new Unfold button to unfold the mesh. Look for the new pop-up tools that are surrounded in yellow boxes. Drag the word Unfold and watch as the UVs unfold and reveal a much better distributed checker pattern in the View panel (Figure 3.17).

**Step 24:** Resize (and rotate if needed) the UV Shell to get it to fit into the top-right quadrant.

---

**Why?**

Although UVs do not need to always remain in that top-right quadrant (and in fact in the next setup steps we will be moving out of that space), if the situation calls for a nonrepeating texture, it needs to have it remain within this quadrant. For this earth wall, since we will be seeing all of it at once within the game, we will want to avoid a repeating pattern if we can. Plus, by rotating and scaling these UVs, it allows us further texture space to create another mini-atlas.

---

**Figure 3.16** Results of Planar Mapping (showing parts that didn’t work so well).

---

**Warnings and Pitfalls**

At first blush, all looks well—the entire surface is covered with checks. However, if you look carefully at the checkers you will see that there are some really distorted checkers (especially in the upper-right corner of Figure 3.16). This is happening because Planar Mapping shoots a texture across a surface like a slide projector. As the image is projected against faces that are perpendicular to the projection plane it looks great; but as the form begins to wrap around so that the planes are parallel to the projection, we get stretching. If this is left this way, the rock texture that will be applied will also be deformed as it wraps around the form. Thus, the next steps are important and powerful.
Manual Mapping

Why?
With all the “auto-magic” techniques that are available in Maya, sometimes some manual mapping with some good ol’ fashioned edge sewing is the only way to get a good UV map. If you’re familiar with manual mapping, skip the next few steps, but if you haven’t made UV maps before, the following steps will be of value.

Step 25: Isolate the main archway (EntryWayArch). Select the arch and choose Display>Hide>Hide Unselected Objects.
Step 26: Assign a new material (EntryWay_Arch_Mat) to the arch. As usual, be sure that Checkerboard.psd is the color texture.
Step 27: Select all the faces that make up the front of the arch, and use a Planar Mapping to map these faces. To do this, right-click the archway and choose Face from the hotbox. Select all the faces that are not inside the arch. Select Polygons>Create UVs>Planar Mapping (Options). In the Planar Mapping Options window, check Bounding Box and Z Axis (unless your arch is facing a different direction than mine). Click Project (Figure 3.18).
**Tips and Tricks**
Selecting faces can be a tricky thing. Remember that sometimes it’s easier to select all the faces and deselect the faces not wanted. Sometimes the Paint Selection tool is the best way to go.

**Tips and Tricks**
Notice that after the Planar Projection was created, the checks might not be perfectly square. Also notice that immediately after the Planar Projection, there are some manipulator handles that surround the area just mapped and that these handles allow for the projection to be adjusted so it can be scaled narrower if needed.

**Why?**
Yes, you’re right. A flat projection creates some mean distortion as the faces turn the corner to create the relief of the archway. To fix this, use the trick of using the Unfold section of the Smooth UV tool (not Polygons>Unfold).

**Step 28:** Map the UVs of the inner curve of the arch. In Face mode, select the faces that make the curved part of the arch (Figure 3.19).

![FIGURE 3.19 Selected faces for cylindrical mapping.](image)

**Step 29:** Select Polygons>Create UVs>Cylindrical Mapping. Rotate the projection by –90 degrees in X. Press Ctrl-A to bring up the Attributes Editor. There should be a polyCylProj1 tab active. Make sure Projection Attributes is expanded. In the first Rotate input field (the X), enter –90 (Figure 3.20).

**Why?**
Cylindrical projections are like wrapping an object with a tortilla—or more accurately, by default, wrapping half the object with half a tortilla. This half tortilla by default is standing up. In this case, the polygons we are attempting to map would be best wrapped with a tortilla wrapping from the top. By rotating the projection by –90 degrees, the projection (as can be seen by the manipulator) is much closer to the shape of the polygons they are projecting upon. Getting closer.
Step 30: Adjust the manipulator to get the checkers square. It’s a little hard to see this in the screenshots as the manipulator handles are so small, but there will be green handles on the top and bottom edge of the manipulator. Pull one out to lengthen the projection to get the checkers closer to square (Figure 3.21).

Step 31: Adjust the projection center to match the curve of polygons. Figure 3.21 also highlights a little tool on the corner of the manipulator that looks like a little red T. Click this, and the manipulator will change and show a new manipulator handle in the middle of the projection that looks like the Extrude tool’s (it has Translate, Scale, and Manipulator handles). Use the Translate tools to slide the projection down so that the
projection matches more closely the arch. It is right when the checkers at the top of the arch are without distortion (Figure 3.22).

**Step 32:** Project the bottom polygons of the arch. Select the two faces on either side of the bottom of the arch. Choose Polygons>Create UVs>Planar Mapping (Options). In the Planar Mapping Options window, check the Bounding Box and X axis radio buttons and press Project.

**Step 33:** Adjust the manipulator handles to get the checkers to match those in the curved part of the arch (Figure 3.23).

**Step 34:** Arrange the shells so they can all be seen separately in the UV Texture Editor. In the UV Texture Editor, click (not marquee select) any one UV. Ctrl-right-click-hold and choose To Shell from the hotbox. Use the Move tool to move the shell away to a place where it can be worked with. Repeat for each of the shells until all the shells can be seen in distinct shapes (Figure 3.24).
Why?
We are going to sew some of these shells together so that the texture can move uninhibited across the surface. When we used Automatic Mapping the shells were all laid out well for us; but not so when we manually map.

Step 35: Resize shells to appropriate sizes. Earlier we learned that when an edge is selected, if that edge is shared in another shell it will highlight as well. As this happens, it may become clear that earlier estimates of matching checkers in the View panel doesn’t quite hold up here (the edges might not be the appropriate sizes). Move and scale the shells to be appropriately matched (Figure 3.25).

Step 36: Sew arch edges up. Select an edge at the bottom of the arch, which should select the top of one of the bottom polygons and use Polygons>Move and Sew UV Edges. Do this on both ends of the arch (Figure 3.26).
**Tips and Tricks**

Since we projected both sides at once (with a Planar Projection), one face is going to have UVs that are backward—mirrored of what they should be. When edges start to get sewn together, one side will yield a really strange result. Figure 3.27 shows the result with the Toggle UV Shaded display activated. The lavender area shows overlapping UVs. If this happens, Undo back so that the backward shell is disconnected. Select the shell and flip it using the Flip Selected UVs in V Direction button. Then resew and things will behave more predictably.

**Figure 3.27** Problems with sewing due to planar mapping two facing faces, and the tools to fix it.

---

**Tips and Tricks**

That Toggle UV Shaded display tool is very handy. Notice that with it checked you can see if any shell is backward. Generally, all the shells should appear blue; they show up pink if the back of the shell is being shown. In this case, it doesn’t matter if the entire inner arch shell is backward, but if you wish to flip it, use the Flip Selected UVs in V (or U) to get it facing the right way.

---

**Step 37:** Move and scale the shells to fill (but not overflow) the quadrant.

**Why?**

This is a very large object in the scene, and an area where the player may indeed be walking up close. Because of this, we are going to leave this object as one UV map, and not try and force it into one of our texture atlases.

**Step 38:** Make everything visible. Back in Maya’s main interface, select Display> Show> All.

**Floor UV Strategies**

Floors are actually deceptively tricky. It seems like a flat plane would be a really easy thing to texture. However, in most games, the player ends up seeing an awful lot of floor at any one time. It’s not unusual for players to have two-thirds of
their visual space taken up with a floor plane with wide open spaces. So having a floor with a texture that holds up to this scrutiny ends up being very important.

To make this successful, we are going to work with tiling textures. For most of the UV maps we have set up so far, the texture is meant to have a 1:1 ratio. The UVs are all contained within the top-right quadrant of the UV space. This means the column texture is painted and wrapped around the column once. Each pixel is represented once on the surface of the column. But on large surfaces like the dock deck, this technique would yield a very pixely surface and one that would look very poor if the character looked down.

For this technique we will be making UV maps that are much bigger than the top-right quadrant. This means the texture will repeat, or tile across the surface. This also means that the texture can be much higher resolution and will hold up much better in the game.

In order to do this, we can’t be dealing with texture atlases—the entire texture needs to replicated multiple times across a surface—and if there are multiple textures, all of that would show up. So we can either split the dock up into separate objects, each with their own material, or we can apply multiple materials to one object on a per-face basis.

Step 39: Create three new materials in the Hypershade. The Hypershade is available via Window>Rendering Editors>Hypershade (the exact layout of the Hypershade may differ depending on which version of Maya you are using). Figure 3.28 shows Maya 2011’s Hypershade layout. To create a new material here, click the Lambert button in the Create tab. The new material can be worked in the Hypershade, or continue working with it in the Attributes Editor (Ctrl-A). For each, import the same Checkerboard.psd file into the Color channel. Name the first EntryWay_DockTile_Mat, the second EntryWay_DockWhiteCement_Mat, and the third EntryWay_DockCement_Mat.

Warnings and Pitfalls
Don’t be fooled by the gray thumbnails. This is sometimes the drawback of using the very flexible .psd format, but sometimes the previews of the textures are not accurate in the Hypershade. Rest assured (and you can see), these materials indeed have the checkerboard assigned within them.
Why?
Why are we suddenly building a bunch of materials in the Hypershade, when in the past we have built them as we went? Well, creating a material as we assign it works great for situations when one material is all that will be on a given object. However, when assigning multiple materials to one object, it works best if the materials are already constructed.

**Step 40:** Assign EntryWay_DockTile_Mat to the polygons shown in Figure 3.29. To do this first swap to Face mode in the View panel (right-click-hold and select Face from the hotbox). Select the polys shown in Figure 3.29. Open the Hypershade, and right-click and hold on EntryWay_DockTile_Mat and select Assign Material to Selection from the hotbox.

![FIGURE 3.29 Picking the faces to assign EntryWay_DockTile_Mat.](image)

**Tips and Tricks**
When the material names get long, while in the Hypershade, it can be difficult to know which is which depending on the size of your thumbnails. However, as the mouse hovers over any material swatch, a yellow hint box will pop up with the full name.

**Step 41:** Assign EntryWay_DockWhiteCement_Mat to the polygons shown in Figure 3.30. The polygons selected are essentially all the polygons along the inside of the canal, and the polys of the lip. The process of assigning the material to just these polys is the same as in the previous step.

![FIGURE 3.30 Polygons for EntryWay_DockWhiteCement_Mat.](image)

**Step 42:** Finally select all the remaining polygons (and deselect those that we have already assigned materials to) and assign EntryWay_DockCement_Mat to them.
Creating Games with Unity and Maya

Why?
Pretty strange results, no? If you are working with the version of your set that you built from the previous chapters, you are finding that there are lots of polygons that are still gray, and those polygons that have texture are all sorts of screwy. Of course, this is because we have not UV mapped this form, so the only polys that will have any textures are the original faces of the cube we began this form with. In the next few steps we will take control of the UVs again, and give recognizable texture to this form.

Step 43: Planar map the tiled part of the dock. Select the two polys that we previously assigned EntryWay_DockTile_Mat to. Choose Polygons>Create UVs>Planar Map (Options). Tick the Bounding Box and Y axis radio buttons and press Project. Use the manipulators to adjust the projection to yield square checkers (Figure 3.31).

Why?
Don’t worry too much about the exact size of these checkers. The UVs that are being edited at this point can be way out of the top-right quadrant of the UV Texture Editor, and in fact once we start building textures, this will undoubtedly change. What we’re interested in is undistorted UV mapping at this point. So as long as the checkers are square, you’re good.

Step 44: UV the white cement parts with automatic mapping. Select the polygons that have previously been assigned EntryWay_DockWhiteCement_Mat to, and UV map these polys with Automatic Mapping (Polygons>Create UVs>Automatic Mapping). Use the projection manipulators (visible in the View panel immediately after Automatic Mapping is chosen) to scale the UVs to ensure square checkers (Figure 3.31).

Tips and Tricks
Be sure you are looking at all sides of this projection. The manipulators for Automatic Mapping look different than those for Planar Mapping, and it will be important to take a close look at all the polygons to get things adjusted appropriately. Remember that sometimes that manipulator is good for gross adjustments, but for any sort of fine adjustment use the UV Texture Editor.
Asset Creation: Maya Scenography UV Mapping

**Tips and Tricks**
The UV Texture Editor can be very powerful, but a little picky. One trick I like to use goes like this: Immediately after using Automatic Mapping, use the Manipulator in the View panel to get a very rough cut of square checkers. Then, without selecting any other object or component, bring up the UV Texture Editor (Window>UV Texture Editor). There notice that only the polygons (faces) just mapped are selected and highlighted. Right-click and select UV from the hotbox, and marquee select around everything visible. As soon as this is done, notice that all the UVs of the form appear (it’ll look a mess), but only the UVs that you just mapped will be selected. Use the Move tool to move these UVs off to the side somewhere you can work with them. Here, you can scale UV shells into submission. Move and Sew UVs if there are easy places to eliminate seams.

**Step 45:** UV map the remaining polygons (those assigned to EntryWay_DockCement_Mat) by either automatic mapping, manually mapping chunks and sewing them together, or some combination of both (Figure 3.32).

![FIGURE 3.32 Finished dock UV Mapped.](image)

**Tips and Tricks**
Again, don’t worry about the absolute size of the checkers. Figure 3.33 shows how my UV map looks at this point. There are three clusters of UV shells (the left is cement, the middle is white cement, and the far right is tile), but they aren’t concerned with living inside the top-right quadrant. Since we will be repeating the texture, we want the shells to be larger than the texture. Remember that although we see all the shells together here, there are actually three materials involved. When we texture (next chapter) we will spend some good time again here understanding how to work with multiple materials on one object.

**Step 46:** Finish UV mapping the scene. At this point there won’t be too much left, but using the techniques covered thus far, you should be able to finish UV mapping the scene so you are ready to texture in the coming chapters.

**Conclusion**
So there you have it. We have looked at all the core ideas to UV mapping. Hopefully this is largely a review of techniques you have worked with previously, but placed within a game design framework. At this point, the
scene will look like a jumble of checkers. Working with this checkerboard texture helps to know what objects have yet to UV, and how the textures are distributing across the objects.

Knowing how to use texture atlases and assigning multiple materials to a single combined object will assist in keeping the draw calls low(er) and keep the scene running smoothly. In the next chapter we will be creating several textureatlases to see how this works; but textureatlases (textures) are only as effective as the UVs that are laid out to take advantage of them. Having control of how to distribute UVs, and how to cut, sew, and position UV edges means you have absolute control over what parts of the texture appears on what parts of the polygons.

To me, UV mapping is work. It has to be done, and it has to be done well. Don’t take shortcuts because good textures are a critical part of the visual impact of games (at least as important as the geometry beneath it), and a good UV map is the way to effectively place and control textures.

The good news is that texturing the scene—the sexy visual part of the process—comes next. And once your UVs are well laid out, the texturing process really does become an immensely fun and rewarding process.

**Homework and Challenges**

- **Challenge 1:** UV Map the Hallway challenge built in the last chapter.
- **Challenge 2:** UV Map all the props built in the Homework and Challenges of the last chapter.