David J. Sushil davidjsushil@gmail.com

Animating Characters in Torque

One of the most difficult tasks when working with the Torque Game Engine is the creation and implementation of custom characters. This tutorial is a starting point for students who wish to develop their own characters. It is not intended as a definitive guide to character development – other references exist which provide a more in depth exploration.

Building the Model

The first task in crafting a custom character is building the model. When doing so, the following heuristics are important to keep in mind:

1. The character should always face towards the positive y-axis
2. The character should be built using Torque's world unit scale
3. The character must be an editable mesh

As with all asset modeling, beginning with a clear goal in mind is vitally important to building a successful character. It is recommended that students build a "rough draft" model first, and having perfected the process, go on to build the model a second time for use with Torque.

The Body

OK, let's begin with a simple enough character – a Scuttlebot. Scuttlebots resemble horseshoe crabs. They are four-legged robots with a tail and a carapace-like frame. They are also fairly easy to model in 3D Studio Max, so let's get down to it.

Start by clicking the Box tool in the Create panel. Using the Keyboard Entry rollout, create a box that is 1 x 1 x 1.

In Torque world units, this is about three cubic feet. It's incredibly tiny. In fact, it's so tiny that 3D Studio Max has trouble working with models this small! You will notice undesirable behavior – mostly interface inconsistencies – quite often as we work with character models at this scale. This leaves us wondering, why don't we just build the model at whatever size we want, and then scale it down later? Unfortunately, doing so can cause behavior that is even worse than.

Click on the Modify panel, and name your box "scuttlebot". Under the Parameters rollout, give the box 3 height, 3 length, and 3 width segments. This will allow us to modify the shape of the box. Convert our scuttlebot to an editable mesh by right-clicking on it, and selecting Editable Mesh from the Convert submenu. You should have a model that looks like Figure 1.

Now we can begin deforming the shape to our liking. Before we do so, make note of the world axis in your Perspective viewport. The side of the box that is facing towards the positive y-axis is the front of our model. It's important that we keep track of where our model is facing, because Torque doesn't like objects that have been rotated *post hoc*.

Open the Editable Mesh modifier and select the Vertex tool. Grab the top-most level of vertices, and – using your Select and Uniform Scale tool – scale them all inward. Having done so, select only the top-most, outer rim of vertices and pull them down slightly. Finally, grab the top-most corner vertices, and scale them all inward a second time. Following these steps will create a round dome on top of our model, as pictured in Figure 2.

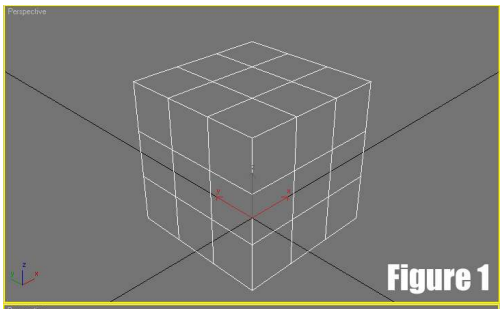


Figure 1

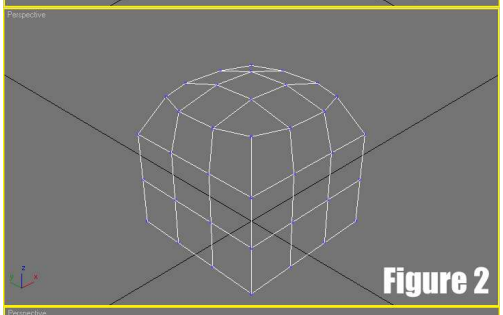


Figure 2

Next, we want to create flares around the bottom of the scuttlebot, in order to protect its legs. Working with the bottom-most layer of vertices, grab all four corners, and scale them outward slightly. You should create flares similar to those in Figure 3.

We're almost done with the main body of our scuttlebot. What remains is the tail, which we will create by extruding a face on the rear of the model.

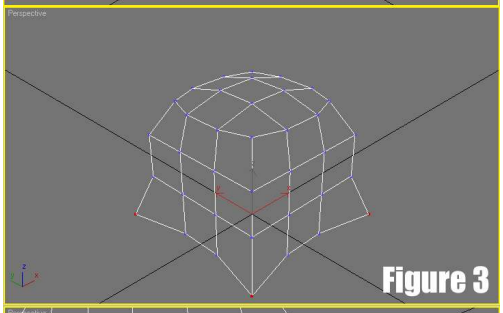


Figure 3

From the Editable Poly modifier, select Polygon. Click the polygon which faces towards the negative y-axis in the middle of the lowest level of polygons. From the Edit Geometry rollout in the Modify panel, use the Extrude tool to create a protrusion from the back of the model, as seen in Figure 4.

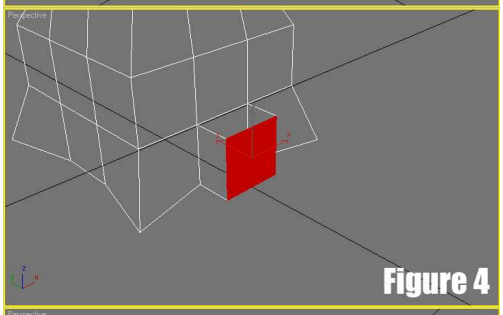


Figure 4

Now, use your Select and Uniform Scale tool to make the tail begin to taper. Use your Select and Rotate tool to make the extruded polygon face upwards slightly. When you have done so, it's time to extrude another polygon from the end of the tail. Again, scale and rotate this new face. Finally, extrude a third polygon, and scale it so that it appears to come to a point. Figure 5 demonstrates how your tail should look when you have finished.

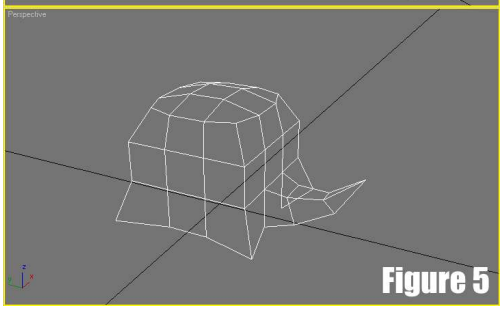


Figure 5

Finally, let's merge the vertices at the tip of our tail, to avoid problems with texturing later on. We do so by selecting Vertex from the Editable Mesh modifier. Select all four vertices at the tip of the tail, and then click the Selected button in the Weld section of the Edit Geometry rollout panel.

Congratulations! You've created a scuttlebot body. I wish I could say the worst is behind us, but sadly, it is not. Let's take a moment to save our work, lest disaster strike us unprepared.

The Legs

Our scuttlebot has four cylindrical legs which move in a shuffling sort of way. In part due to their inconspicuous placement under the model, we can skimp a bit on precise animation. This is, after all, more a tutorial on interfacing with Torque than it is a treatise on picture-perfect animation.

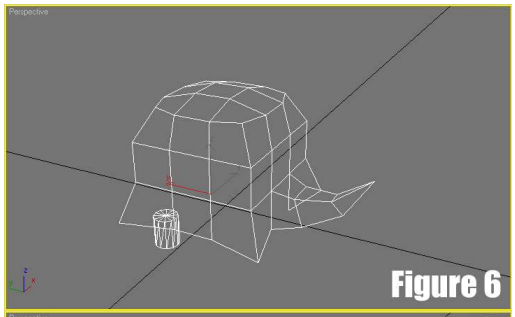


Figure 6

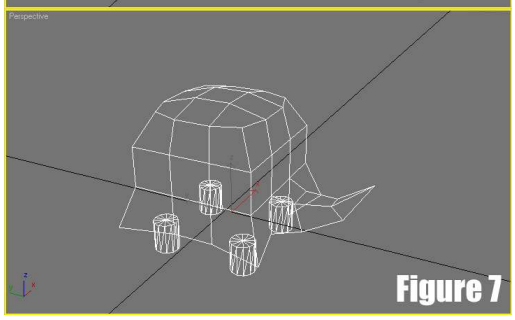


Figure 7

Begin by creating a cylinder that is 0.25 units high, with a radius of 0.1. You can do so by clicking on the Create panel and choosing the Cylinder tool. I recommend using the Keyboard Entry method of creation – creating it by hand at this scale in 3D Studio Max can be problematic. Under the Modify panel, give the cylinder 1 height segment and 12 sides. By default, the cylinder will be placed dead-center in your model. Using your Select and Move tool, position the cylinder roughly where the front-right leg should be. The numerical position of this front-right leg is (-0.35, 0.35, -0.25), if you're keeping score. Check out Figure 6 for a reference.

Next, we need to clone this leg three times, positioning each clone where a remaining leg should be. We can do so by clicking on the Select and Move tool, holding down the shift key, and clicking and dragging on our original leg. Doing so pulls out a cloned copy of the leg. You will be presented with a Clone Options dialog box – simply make sure the Object is set to Copy, and click OK. The remaining legs should be positioned according to the following coordinates:

- (0.35, 0.35, -0.25)
- (0.35, -0.35, -0.25)
- (-0.35, -0.35, -0.25)

At this point, your model will look similar to Figure 7. If so, we can attach the legs to the body. Do so by first selecting your scuttlebot body. From the Modify panel, click Attach List, located in the Edit Geometry rollout. Select all the cylinder objects from the list – these are our legs – and click the Attach button. Now, our four leg cylinders have been merged into the body mesh, and our scuttlebot is one solid piece.

If you'd like, you can take the time to further refine your scuttlebot model, but for our purposes, it's lookin' good, and we can move on to animating the object.

Adding Animation

Torque has some nifty animation options available. First and foremost, if you're animating a humanoid character, there's a better than average chance you don't even need to worry about animation. The reason for this is that Torque's poster boy and stock character, Kork, comes with preset animations stored outside of his .DTS model file. These stock animations can be utilized by any biped character that possesses a similar bone structure to Kork.

Our scuttlebot, on the other hand, is not a biped (it's a quadruped), so we need to create our own custom animations. We begin by setting up a skeleton.

Creating the Bone Structure

The cornerstone of good animation is a well-crafted bone structure. We will create one by accessing the Bone Tools dialog, available from the Character menu in 3D Studio Max 8, and the Animation menu in 3D Studio Max 9.

The Bone Tools dialog will allow us to create individual bones in a chain, as well as to link chains together to form a skeleton. It's important to always start a chain from the center of the character and work your out – for example, start with the shoulder and create out towards the fingertips, not the other way around. Doing so makes the difference between the dog wagging the tail and the tail wagging the dog. Our scuttlebot's pelvis will be the root bone for our entire model – everything else stems from this point.

Let's begin by creating the bone chain that begins with the pelvis and works its way out towards the head. We will need to create six bones altogether – the pelvis, three vertebrae, a head, and a final bone called "eye", which we will discuss shortly. To create this chain, click on the Create Bones button in our Bone Tools dialog. Now, because we're working on such a small scale, our bones will be enormous if we begin creating them now. First, from the Create panel, set the Bone Object Width and Height to 0.1 each. That way they will fit the size of our model properly.

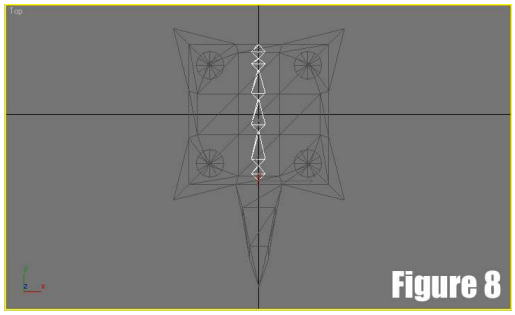


Figure 8

With the Create Bones tool selected, position your cursor in the Top viewport, just where our scuttlebot's hips would be, and click one time. This sets the base for our pelvis. Clicking a second time will end the pelvis bone and begin the first vertebra. Make the pelvis just a square block. Each vertebra you set should be longer than the pelvis, however the head should be about the same size. Finally, once you have set the base of the eye bone, right click to end the chain. Your bone chain should resemble the one in Figure 8.

While naming your bones for custom animation is not necessary, it's still good practice. Someday you might want to borrow Kork's animations (or better yet, have your own custom characters borrow animations from each other), and it's important to rehearse the

conventions Torque expects for such an occasion. The names of bones in Torque are borrowed from 3D Studio Max's Biped character, so a handy reference is available at any time from the Create panel's System's mode. In general, each bone begins with a Bip01 prefix, such as "Bip01 Pelvis". Bip01 is always followed with a space (one of the few times a space is allowed to be exported from a 3D Studio Max model to Torque), and then the name of the bone. Repetitive bones, such as vertebrae, follow a pattern like this:

1. Bip01 Spine
2. Bip01 Spine1
3. Bip01 Spine2
4. Bip01 Spine*N* (where *N* is the total number of bones in the chain, minus one)

The only bones that don't follow this naming convention are special objects, three of which we'll address in a moment. Let's go ahead and name these bones, like so:

1. Bip01 Pelvis
2. Bip01 Spine
3. Bip01 Spine1
4. Bip01 Spine2
5. Bip01 Head
6. eye

That last one isn't a mistake. Eye is a special bone which Torque uses to control the placement of the camera when in first-person mode. By linking eye to our Bip01 Head bone, we can make sure that when the character moves its head, the camera moves with it.

Before we proceed, check to make sure that this bone chain is not only centered in the Top viewport, but passing through the middle of the bottom row of polygons in your Left viewport.

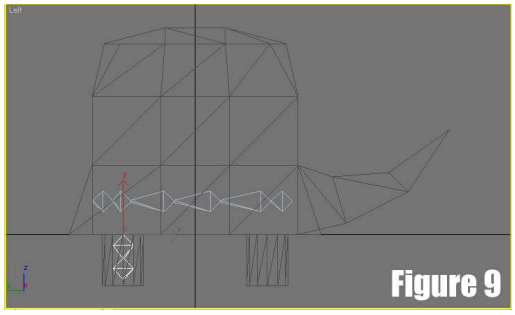


Figure 9

Now, let's set two bones for our scuttlebot's front right leg. These bones are Bip01 R UpperArm and Bip 01 R ForeArm (notice the R qualifier – on the opposite side, we'll use L instead). The first bone will begin at the top of the cylindrical leg and end about halfway down, at which point the second bone begins and continues to the bottom of the cylinder. Be sure to check your Top viewport as well, and position the bones central to the cylinder according to that perspective as well. Use Figure 9 as a reference if you're unsure of how to proceed.

Just like we did with our original leg cylinders, we can clone this bone chain and position the copies in their respective locations. The names of the remaining bones are:

1. Bip01 L UpperArm and Bip01 L ForeArm
2. Bip01 R Calf and Bip01 R Foot
3. Bip01 L Calf and Bip01 L Foot

The final bone chain is the tail. Add three bones, one for each of the extrusions we created. Name them Bip01 Tail, Bip01 Tail1, and Bip01 Tail2, respectively. When you are finished, your bone structure should look like that which is pictured in Figure 10.

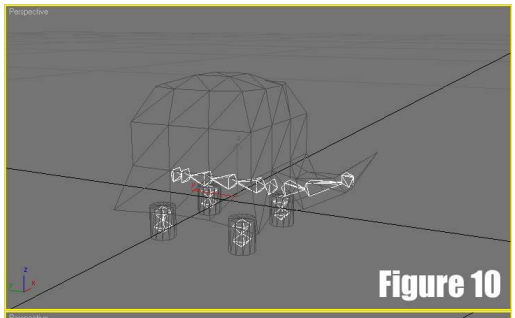


Figure 10

Finally, we must link our bone chains together into one skeleton. We do so by first selecting the Bip01 Pelvis bone. In the Bone Tools dialog, click on the Connect Bones button. Moving the mouse over your active viewport, you will notice a dashed line that follows your cursor around. Click on the Bip01 L Calf bone, and a new bone will appear. Rename the bone Bip01 L Thigh. Reselect the pelvis, and click the Connect Bones button again. This time, click on the Bip01 R Calf bone. Again, a new bone will appear. Rename it Bip01 R Thigh. Repeat this process for the font legs, connecting the Bip01 Spine2 bone to the UpperArm bones. The connecting bones should be renamed Bip01 L Clavicle and Bip01 R Clavicle. Finally, create a Bip01 TailBase bone that connects the pelvis to the tail. Your skeleton will now resemble the one pictured in Figure 11 (I've hidden the model so you can see the entire bone structure).

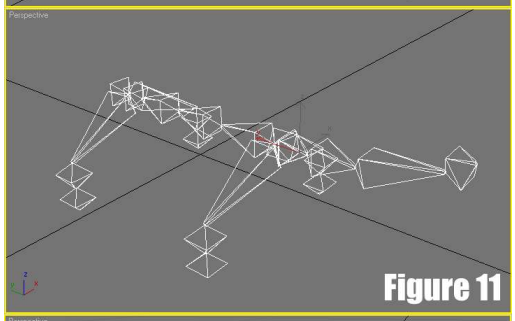


Figure 11

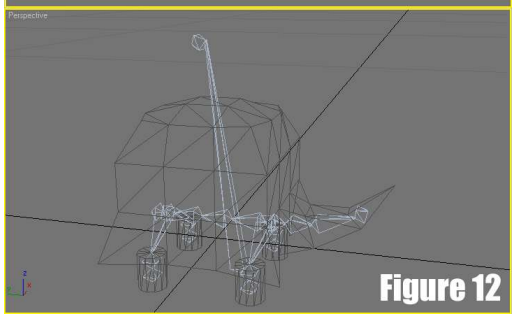


Figure 12

We're almost done with "dem bones." We need only create one more chain, consisting of two bones: unlink and cam. These bones don't animate our character at all. Instead, Torque uses them to position the over-the-shoulder camera when not in first-person mode. To create this chain, click on the Create Bones button, and in your Left viewport, create a bone which begins just before the back legs, and rises up through our scuttlebot and above the shoulders. After setting unlink, right click to set cam. Name both of the bones accordingly. Your finished model will look like the one pictured in Figure 12.

Skinning the Model

Our next step is to add a Skin modifier to our scuttlebot model. Skin will bind the model to the bones, so that whenever a bone is moved, the model will deform according to the movement.

This is a fairly simple procedure. First, click on the scuttlebot model. From the Modify panel, click on the Modifier List, scroll down, and select Skin. From the Parameters rollout, click the button labeled Add, which appears to the right of the word Bones. From the dialog box that appears, select all of the bones which will deform the model. That is, everything except eye, unlink, and cam.

Now, the model acts like skin around the bones. We can test this by clicking on our Bip01 L ForeArm bone, and using the Select and Move tool to reposition it. If our Skin modifier was setup properly, the front left leg should move in unison with the bone.

You may notice that the front right fender around the leg also moves. That's OK for now. This is not a tutorial on animation, so much as it is a tutorial on properly interfacing with Torque. We can always refine the animation later.

PS: Save your work!

Keyframing Animation

With our recently skinned scuttlebot (that sounds gross, doesn't it?), we can begin to craft animation for use with Torque. We will create two animation sequences – a default idle animation, and a walk cycle animation. Let's begin with the more difficult walk cycle.

For simplicity's sake, select the eye, cam, and unlink bones, and hide them from the scene by right-clicking and selecting Hide Selected from the popup menu. Now, we're only working with the bones that deform our model.

To animate our model, we'll need to use the timeline at the bottom of the screen. See the button marked Auto Key? Click it. Our timeline becomes red, which serves as a warning that we're no longer working with a static model, but a dynamic model that changes over time. If you're done animating at any point, click the Auto Key button again to return our timeline to its normal gray color.

Next, click on the Bip01 R ForeArm bone. We're going to animate the bone swinging forward, then back, then forward again. Using your Select and Move tool, position the bone slightly closer to the head of the model, which will deform the front right leg into a forward-stepping position.

When you are satisfied with the placement, click the button below the timeline that looks like a key. Notice how a box has been added to the timeline on frame 0? That's a keyframe. 3D Studio Max and Torque create animation by computing the look of the model based on transitions between keyframes. This saves us time – instead of animating each frame, we animate a few keyframes, and Max and Torque do the rest of the work for us. Fancy that!

Now, we need to drag our timeline slider out to about frame 15. Using your Select and Move tool again, swing that front leg bone into a backwards position. Set the keyframe by clicking the key button below the timeline.

Now, you can test your rudimentary animation by dragging the timeline slider back and forth between frame 0 and 15. If you've followed this tutorial correctly, the leg should swing forward and backward with the slider.

We need to set one final keyframe for this bone at frame 30. This time, swing the bone back into the forward stepping position. It should match frame 0 almost exactly. Set the keyframe, and test the animation by dragging the timeline slider back and forth.

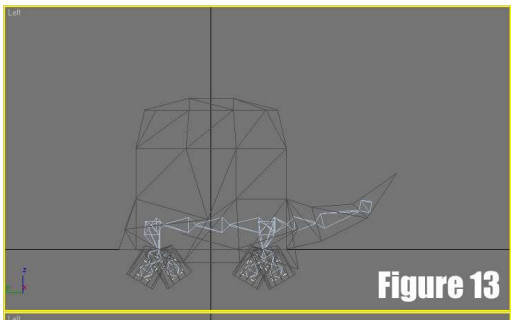


Figure 13

One leg down, three to go! Follow the same procedures to animate the front left leg, back right leg, and back left leg. They should all share keyframes 0, 15, and 30, in order to march in step with each other. Check out Figures 13 through 15 for a play-by-play example of how your model should look at each keyframe.

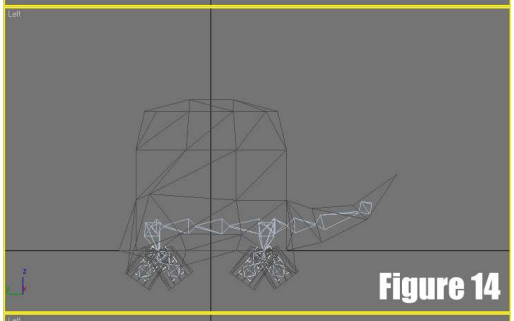


Figure 14

This completes our walk cycle animation. The next item on our to-do list is an idle animation, which is easy enough. Our idle animation will begin on frame 31, and run to frame 41 (10 frames altogether). It will consist of our scuttlebot wagging its tail happily, in anticipation of its next great adventure, scuttling here and there in the ignorant bliss of a robotic consciousness.

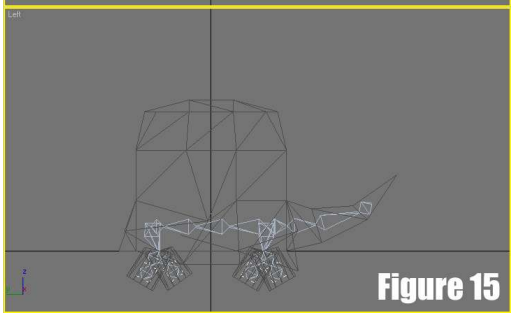


Figure 15

OK, move your timeline slider to frame 31. For each of the four leg bones you worked with in the walk cycle, set a keyframe which positions them in a downward position. Set keyframes at frame 41 for each leg as well. This pins the legs into a standing position immediately after the walk cycle ends and for the duration of the idle cycle.

Now, click on the last tail bone. For frame 31, set a keyframe with the tail off to one side. Set the timeline slider to 36, and set a keyframe with the tail off to the opposite side. Finally, move the slider to frame 41, and set a keyframe which matches the tail's position with that of frame 31.

Be sure that when you're satisfied with your animation, you click the Auto Key button to turn off animation mode in 3D Studio Max.

We're finished with the animation section of this tutorial. Keep in mind, you can add additional animation sequences to the timeline. Just make sure the next one starts on frame 42. In the next section, we'll look at how we interface our 3D Studio Max model with the Torque Game Engine.

Preparing to Export

It's important that we properly prepare our model for use with Torque. There are a handful of hoops to jump through, but once you've practiced the routine a few times,

those jumps will begin to make sense. By the way, whenever Torque starts to make sense, immediately shut down your computer, visit a pleasant park in your city or town, and consider therapy.

Initial Preparations

In order to export your model, you'll need to access the Torque DTS Exporter utility. If you've never used this tool before, I recommend reading the Importing Assets From 3D Studio Max tutorial.

Like every other model, we need to renumber and embed the shape. So make sure you've selected our scuttlebot, and click the Renumber Selection button from the Utilities panel. In the Renumber dialog box, type 2 and click OK. This renames our scuttlebot to "scuttlebot2". Then, click the Embed Shape button, but only one time. This button creates a few helper objects in our scene – Start01, Base01, and Detail2. It otherwise is not apparent that it worked, so check your Select by Name tool to make sure they've been added.

Next, we need to add a bounding box to our scene. Using the Box tool from the Create panel, create a shape that encompasses the entire model. Name this box "bounds". It's important to note that we should not resize bounds using the Select and Uniform Scale tool. Although the size will change in 3D Studio Max, it will not change in Torque. This is because scaling geometry is much like adding a modifier to the object, which Torque doesn't understand. Instead, use the properties in the Parameters rollout, located within the Modify panel, to change the size of our bounding box.

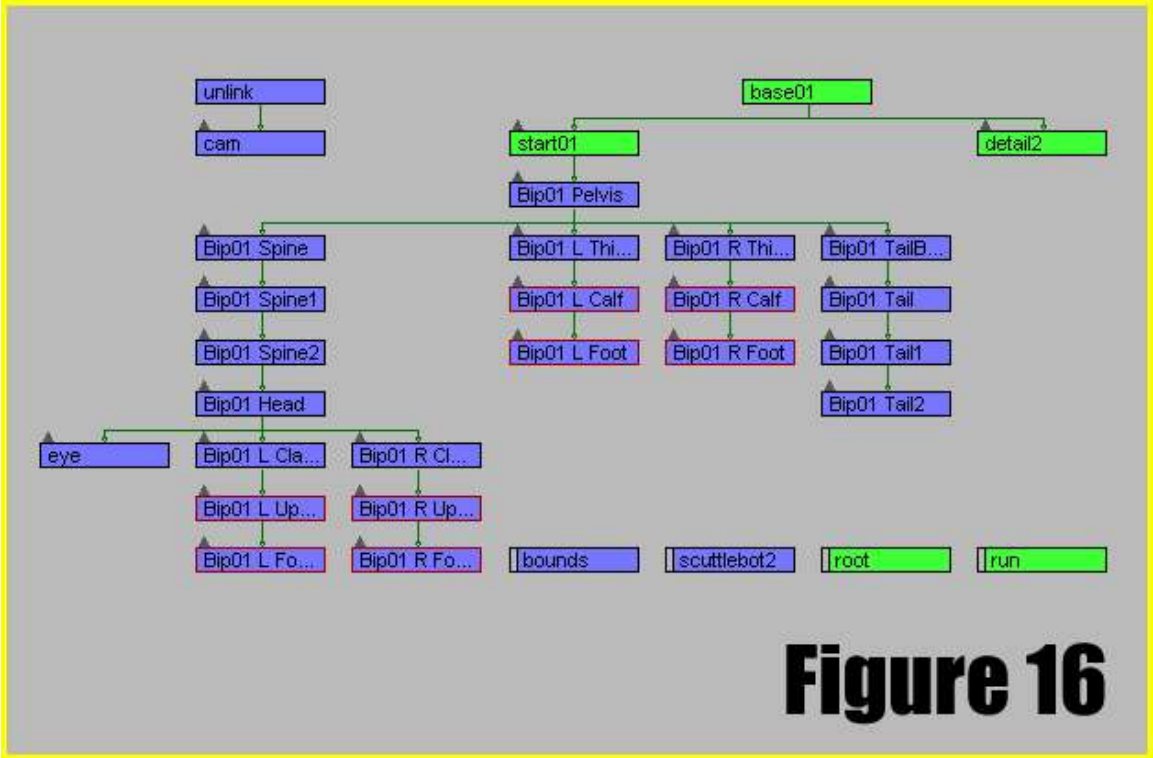
The Schematic View

Open your Schematic View window by clicking on the Schematic View (Open) button in the toolbar – the one that looks like an orange box pointing at a gray box. This window contains a representation of the relationships among objects in our scene. We need to make a few changes to these relationships in order for Torque to be happy.

First, we need to disconnect scuttlebot2 from start01. Normally, this is exactly the way our scene should be setup, however, since we have a bone structure in the mix, we need to make this adjustment. In order to disconnect scuttlebot2, click on its representative box in the Schematic View, and click the Unlink Selected button in the toolbar. Leave scuttlebot2 hanging out on its own – it doesn't need to be connected to anything in our scene (he's a loner, anyhow).

Next, we need to link our Bip01 Pelvis object to start01. In order to do so, click on Bip01 Pelvis, then click the Connect button in the toolbar. Moving your cursor into the Schematic View window, you will notice a dashed line following the mouse. Click on start01 to link the pelvis as its child.

As usual, bounds remains on its own. So too does unlink and cam. Check out Figure 16 for an example of how your Schematic View should look. Don't freak out if you're missing objects called root and run – we'll create those in the next section.



Animation Sequences

Next, we need to add a few helper objects which tell Torque where to locate our animation sequences. These helper objects can be accessed from the Create panel by clicking on the Helpers button (it looks like a white box with a circle in it). From the dropdown list just below the Helpers button, select General DTS Objects. Next, click the Sequence button, and create a sequence objects in the Perspective viewport by clicking and dragging. Name the object "run". Run is a special name that Torque recognizes. Run animation is played – you guessed it – whenever the player causes his avatar to run.

Create a second sequence object and call it "root". Root is the special name Torque understands for idle animation. There is a whole list of these special animation names, and you can of course create your own. It's best to plug in to the engine as often as possible, so utilizing existing names for animations is highly recommended.

A comprehensive list of the existing animation names is available in the player.cs file, which is located in demo/data/shapes/player/. What follows is a partial list of some of the more common animation names, and a brief description of their purposes:

Name	Description
root	Idle animation – the avatar is not doing anything
run	Moving forward
back	Backpedaling animation (running backwards)
side	Strafing animation
fall	Falling animation
land	Landing animation
jump	Jumping animation when the avatar is moving
standjump	Jumping animation when the avatar is standing still
death*N*	Death animation (replace *N* with a number from 1 to 10)

Once we have created our sequences, we must take one final step to let Torque know where they begin and end on our timeline. We do so by opening the Curve Editor, which is accessed by clicking the Curve Editor (Open) button on the toolbar, which is located left of the Schematic View (Open) button.

With the Curve Editor window open, click the plus sign to the left of the Objects item on the left side of your screen. Locate root, and expand it as well. Within root is an item called Object (Sequence II). Expand it, and click on Sequence Begin/End. Now, we need to add keys on the chart which designate where the root animation begins and ends. To do so, make sure you have the Add Keys button highlighted in the toolbar, and click on the timeline at positions 31 and 41. If you accidentally misplace a key, you can move it by highlighting the Move Keys button.

Next, repeat the above steps with the run animation, but set the beginning and end keys to 0 and 30. When you've done so, you can close the Curve Editor window. We're almost done!

One final note on animation sequences – you need to designate whether your animation loops or whether it plays only once. For example, running is a cyclic sequence, whereas a death animation is non-cyclic. It would look rather silly if our character died over and over again, or if our character ran through one walk cycle and then glided everywhere.

We can designate a sequence as cyclic or non-cyclic by toggling the Cyclic Sequence check box in the Modify panel of any given sequence object. By default, all sequences are cyclic, so we don't have to worry about root or run.

Utilizing the Model in Torque

Finally, we're ready to reap the rewards of our hard work. Before we do, let's make a backup copy of the existing player model – Kork. Navigate to demo/data/shapes/player in your Torque directory. Locate player.dts, and make a copy of the file. That way, we can always bring Kork back into action.

To export your custom character model, click the Whole Shape button in the Utility panel. In the Export DTS window that appears, navigate again to demo/data/shapes/player in your Torque directory, and save your file as player.dts. If all

goes as planned, you should be able to open Torque and immediately see that your player model has replaced Kork. If so, congratulations on building your very own scuttlebot! If not, well, the next section is for you. Good night and good luck…

Troubleshooting

There are any number of ways to screw up a character model in Torque, and if you're reading this section, you most likely discovered at least one of them. Take a deep breath. I'll walk you through some of the more common errors you may encounter.

"Assertion Failed on Skin Object"

Let's start off with an easy one – the ol' "Assertion Failed on Skin Object" error. What Torque means to tell you is that your model is not an editable mesh. It's probably an editable poly. To fix this error, click on your scuttlebot and go to the Modify panel. Delete the Skin modifier on your object by right-clicking on it and hitting Delete. Right click on your scuttlebot and select Convert to Editable Mesh from the Convert To submenu. Add another Skin modifier to the object, and re-add all your bones. Export the model again and the error should not appear.

"Skin Found on Linked Node"

If this error appears, it means you most likely forgot to replace your character model with Bip01 Pelvis as a child of the start01 object in the Schematic View. Re-read the section titled *The Schematic View*. In short, disconnect your character model from start01, and connect the root bone of your skeleton (most likely the pelvis) instead.

Bones are Visible in Torque

Sometimes a character's bones are visible in the engine. This can be remedied one of two ways. First, make sure the bone is properly positioned within the model, and that no parts of the bone are protruding in 3D Studio Max. You can also set the bone Width and Height to smaller values in the Modify panel. If you're still having problems, it's probably because the bone has a trailing number in its name – for example, Bip01 Spine2. If the trailing number (in this case, 2) matches the trailing number of your character model (say, scuttlebot2), then Torque thinks the bone is supposed to be rendered. Rename the bone to "Bip01 SpineTwo", or "Bip01 Spine-2" (negative two). Either way, the bone should stop appearing in the engine.

Character Faces the Wrong Way, or is Rotated

The easiest way to avoid this error is to make sure that you keep track of where your model is facing in 3D Studio Max. By default, all character models should face towards the positive y-axis. Rotating a model into this position later on is dicey at best, so be sure to get it right the first time! This may also occur because you rotated an object in the scene (say, a cylinder for an arm), and then attached the entire model to the rotated object, which may have the effect of rotating the entire model out of whack. Try to attach such objects to the main body, instead of vice-versa.