Animation Tutorial

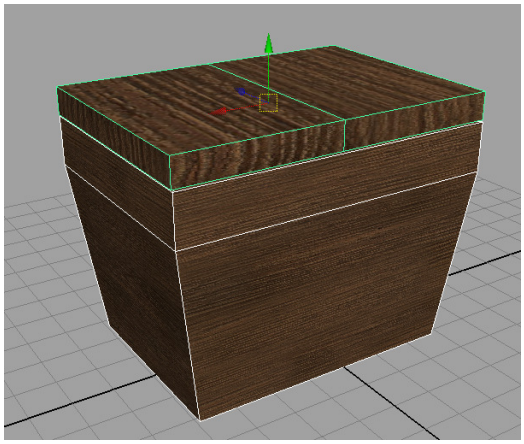
Introduction

The last week before play-testing begins should be focused on presentation. Hopefully, the outcome for this module will be a complete game that you can include in your portfolio – so it needs to look fairly decent.

This week as a minimum you need to have successfully exported a 3D asset with some animation. I'll talk you through adding animation to the chest object (the one I created in week 6). Ideally, you should create your own asset and add animation to it. I am happy for you to use the 3D asset that you created before, or add some animation to another asset you have found. This includes character animations (you can look later in this tutorial for more tips on this). As an alternative I have given you the option of extending the chest object in this tutorial.

Chest GameObject

I'm going to extend my original 'box' 3D asset into a chest. I'll need to write some UnrealScript so that it behaves like a code. I won't focus on the UnrealScript too much here – I'll leave that to you ☺



Our chest could be used in the game to store another GameObject. Our players would need to open the chest in order to get at the item inside.

The chest may prove a very useful GameObject in itself though. The Famous Five may need to solve a puzzle, or collect keys in order to be able to get at the goodness within.

My chest needs a number of simple animations in my total conversion:

- Opening Box
- Closing Box
- Attract the player (box wobbles a bit) (can only do this if we're in closed mode)

If the player now has the right key, perhaps we should wobble the chest slightly to remind our player that they are now able to interact with it.

It is possible to do our 'attract' animation UnrealScript . It might even be better in some respects – since the same piece of code could be used to 'wobble' any objects. The benefit of doing it using an animation though is that we get artist input. The artist can create all manner of 'attract' animations depending on the original object (not just a wobble).

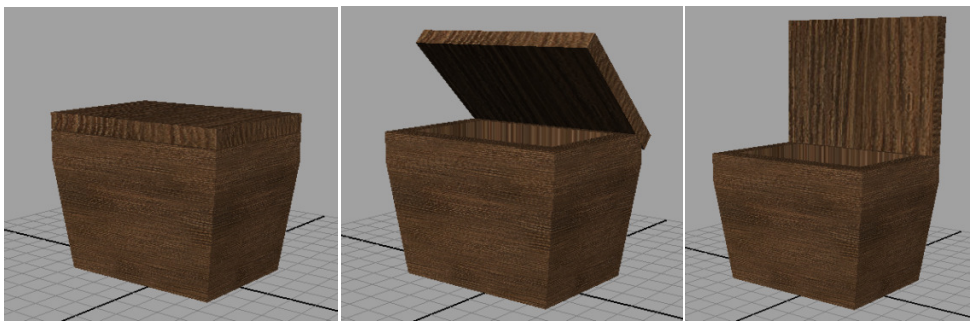
Typically in games we would have a corresponding animation for our player; our player should appear to lift his arms in an opening manoeuvre while the chest's 'BoxOpen' animation plays. We won't worry about this too much at that point. It's left as an exercise to the reader ☺

Getting Started

I've taken my original 'box' model and extended it - I've called it 'Box with Lid' or 'chest'. It's just a 'polygon cube' stretched out and moved to form the lid (see image above)

There's not a lot to animation in Maya. The Maya introduction animation video tells you pretty much all you need: 'Press **s** to set a key-frame'.

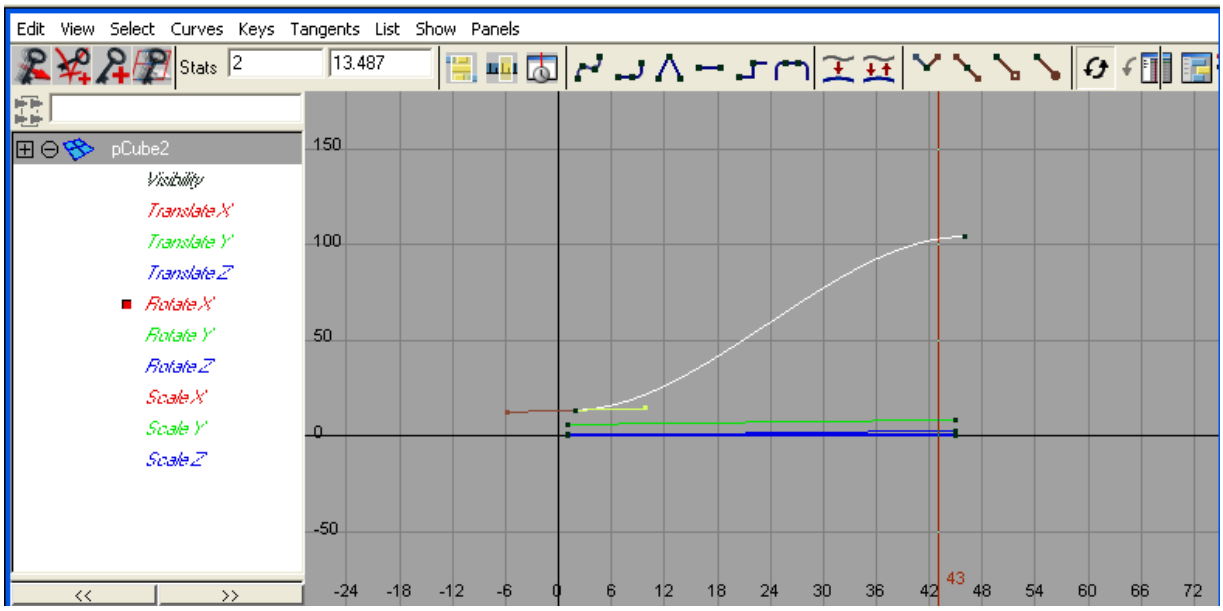
It is simply a case of setting the chest up in the closed position for frame 1 and storing a key-frame. Then doing the same for the closed position later on (I used frame 60). Unreal runs animations at 30fps by default.



1: BoxOpen KeyFrame 2: Interpolated Frame 3: BoxClosed KeyFrame

Once you have a simple animation up and running, you can tweak things by using the GraphEditor. You must have an object (i.e. the lid) selected in the scene to be able to see the animation associated with it.

Once you have it open, you can have a bit of a play: Select the points by left clicking, then move using the middle mouse.



There is loads more help on animation on the Maya website:

`images.autodesk.com/adsk/files/gettingstarted1.pdf`

There are two ways of exporting for UT2K4:

- Vertex Animation

  This simply records the positions of all the vertices from frame to frame. As you can imagine this takes up much more space on disk than the skeletal animation files. It's only really worth it if you need really fine control, e.g. an approximation of cloth blowing in the wind.

- Skeletal Animation

  Skeletal animation stores just the transforms of the joints in the hierarchy. This is much more efficient than Vertex animation, plus it allows the animations to play on any 3D object with the same skeleton. It also allows us to blend animations on top of each other. For example, we could overwrite the head orientation of our character and have them look at a specific item in the gameworld. (Feel free to give this a try in your TC using UnrealScript. If a character is near an object of interest, their head could turn to look at it – overriding the playing animation.)

Vertex Animation Export

So vertex animation is generally a bit rubbish and not really suitable for what we're doing. However, if you want to you're welcome to give this a try. The process is described at:

`udn.epicgames.com/Two/UnrealVertexAnimation.html`

There is support for this in ActorX under Maya, use the command `axvertex`

Skeletal Version

In the lecture we talked about creating a hierarchy and then attaching these bones to the geometry. This version is an improvement on our previous chest as it has bones in it (well actually only 1 bone to start with).

You'll probably want to refer to the Mastering Unreal book. There is an excellent section at the back which describes using Maya with Unreal.

In order to add bones we use the Joint tool.

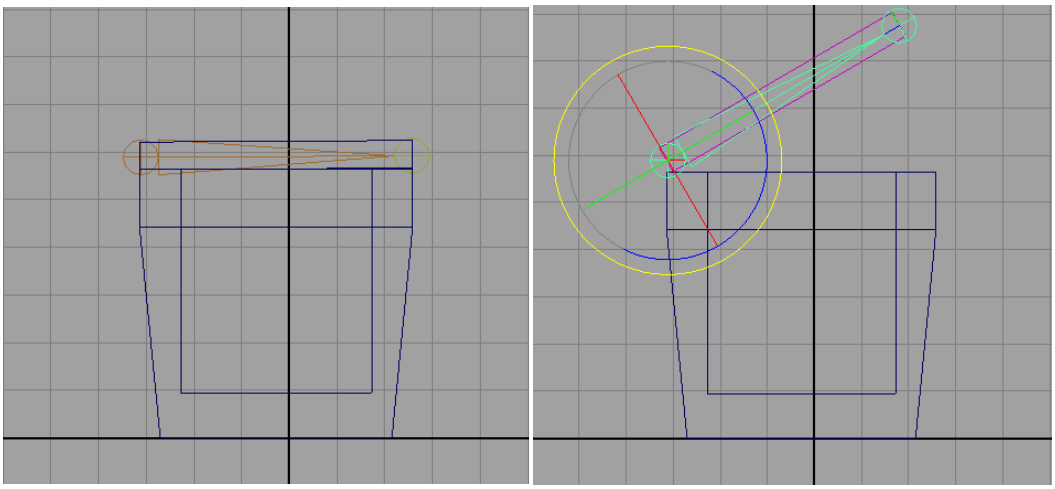
It's probably easiest to do this from the side view. Ensure you are in animation mode (section from the drop-down in the top-right) and then select `Skeleton→Joint` tool to add bones. I have clicked once where the chest hinge will be and once at the other end of the chest lid. You then press enter to exit the Joint tool.

The next step is to link the bones to the mesh (skinning). In this case it is quite straight forward. You select both the joint and the mesh together and then use the Smooth Bind tool:

`Skin→Bind Skin→Smooth Bind`

Note that Unreal doesn't support rigid bind – we must use smooth (but it actually makes no difference)

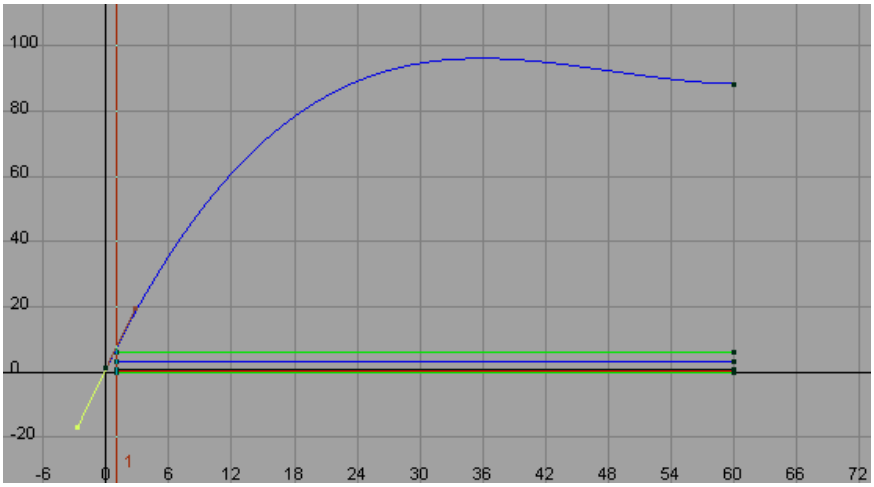
If everything has gone to plan then you should be able to rotate the chest lid, by just moving the hinge joint.



Next step is to add our key frames again:

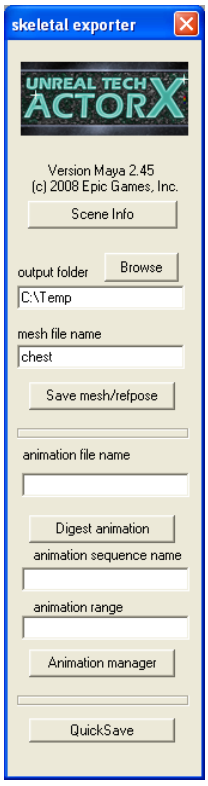
Frame 1:	Lid Closed Position
Frame 60:	Lid Fully Open Position

Again you can tweak the values in the Graph Editor again to make the animation a little more interesting:



Then I then added a final keyframe:

Frame 120: Lid Closed Position



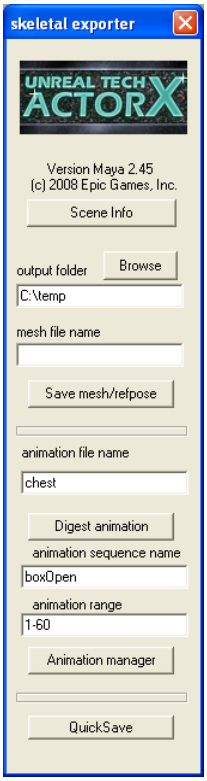
This is a good point to test that your animation is working correctly. What we should have is two animations that play back to back, one of the box opening and one of it closing. We will 'digest' these as two separate animations.

Export Process

`.psk`

Firstly, export the `.psk` file. Start ActorX using the command `axmain`. We only need to use the top section of the ActorX window in order to output the `.psk` file.

Just specify the output folder and the filename (it will append the `.psk` for you). Then click the 'Save mesh / refpose' button.



`.psa`

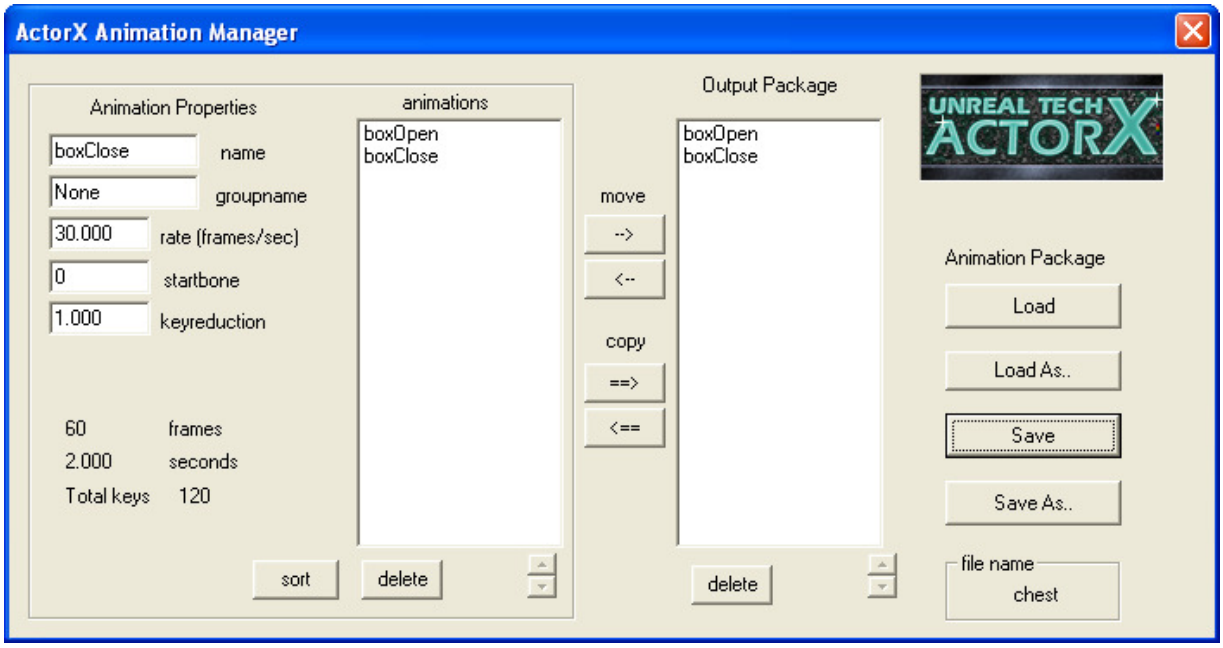
Next we need to digest these two animations (one at a time):

Specify the `.psa` filename, the name of your animation sequences and the frames of the Maya animation we will use. The details in the screenshot show it set up for the first anim. Then click digest. You should be able to see the frames of the Maya animation play back as it gets 'digested'.

Repeat this prcoess for frames 61-120 for boxClose.

Finally, we need to open the Animation Manager in order to save the file.

There are lots of options to support editing an existing `.psa` file. However, all we need to do, is copy our two anims over and then hit the Save button.



Congratulations – that's your first animation exported. Now to get it into your Total Conversion.

Importing your Animation into UnrealEd

For this we'll need:

- `chest.psa`
- `chest.psk`
- `wood02.tga`

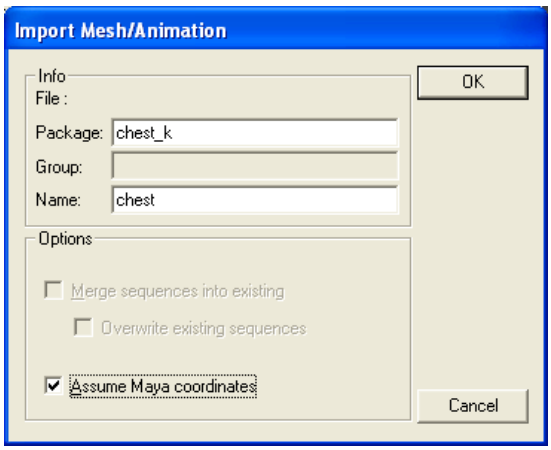
Fire up UnrealEd, add the `wood02.tga` into a texture package of your choosing.

Go to the Animation Browser.

Import your `.psk` file

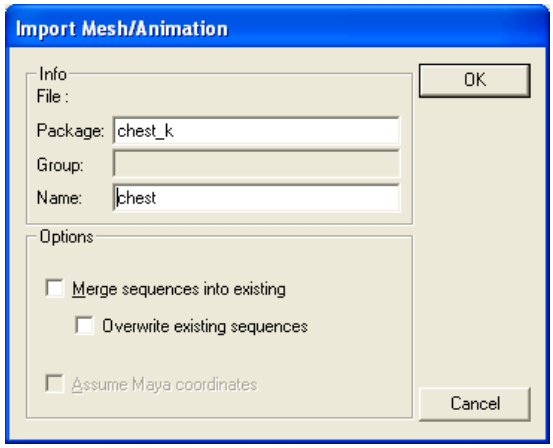
`File`→`Mesh` Import (select your `.psk`)

(Use 'Assume Maya coordinates – otherwise your model will come out on its side)



You should see your model loaded in correctly. You'll probably need to apply the wood texture manually at this point.

This is also the best time to alter the scale / orientation of your model.



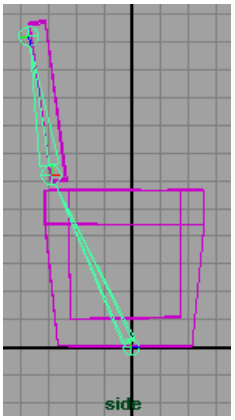
Import your `.psa` file

`File`→`Animation Import` (select your `.psa`)

(as you can see, the two import screens look practically identical)

If all has gone well – you should now see your chest with two animations selected.

Import Problems (and solutions):



I had a few problems with the import stage. The base of my chest stopped appearing or the lid and base moved together when the animations played.

I guessed this is because Unreal expects a single skeleton and also assumes that every part of the mesh is attached to a bone. To combat this I put in an additional base joint (I positioned this in the centre of the base of the chest). I skinned that first bone with the chest base. Then, I made the previous joint a child of this new 'base' joint.

I then played around with the vertex weights (use the 'paint skin weights' tool) so that the chest base was not affected by the lid hinge joint. There may be another way around this – but it worked for me ☺

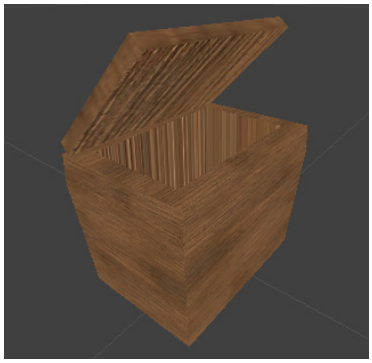
Once you get this working there is one more stage.

Select each animation and choose:

`Edit` →`Linkup Anim and Mesh`

This is a good point to save everything.

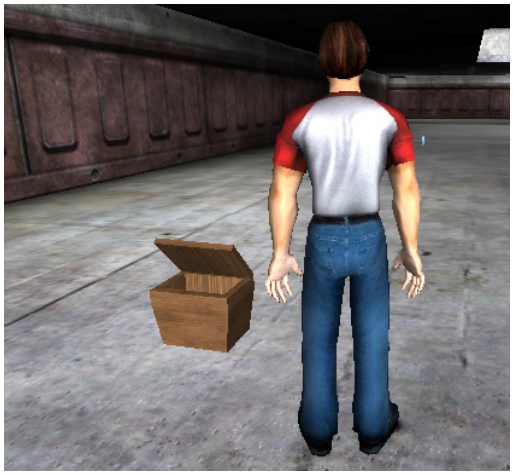
You can check if things are going well, as when you reopen the chest animation package, you'll be able to see the list of animations as soon as it opens. Test your animations at this point:



Use In Your Game

Finally, you need to stick this down in your level. I took the code for existing GameObject code. I deleted my `staticmesh` and then added a `skeletalmesh` to the mesh property. Remember to change the `DrawType` to `MESH`

Finally, you just want to call the `PlayAnim()` function at the appropriate times. It's now about incorporating it into the game and making it a valid GameObject. E.g. the player can only open it once they acquire a crow-bar.



Your deliverable (alternative):

As your 3D Animation asset deliverable, I am happy for you to extend this and include it in your game. However, you need to:

- Improve the model
- Add a lock mechanism
- Add the attract (wobble) animation
- Make it have a valid in-game use