chapter 11
Tail Rigging for *Paul and his Bananas*

Paul the monkey is the star of the upcoming Bishop Animation short film, *Paul and his Bananas* (see Figure 11.1) While this is an educational short film for children, we wanted to put our all into the development of the project. There is a lot of very simple (and quite frankly, not very good) children’s educational content out there, and we wanted our project to be different. Therefore, when it came time to build Paul, we wanted him to be able to do everything a real monkey could do, and use those abilities in the short. After studying monkeys at the zoo for a while, it became apparent that the tail could be a nightmare if not done correctly.

Monkeys use their tails for many different purposes. The tail can lag behind, allowing for some very nice secondary motion. The tail can be moved using forward kinematics, acting like another arm or leg. The tail can also wrap itself around a branch, allowing the monkey to hang and swing from the tail. We needed a tail rig that would allow for all of these functions, as well as combinations of the three (see Figures 11.2 and 11.5).

Here is a little bit of what we did to achieve Paul’s tail rig.

1. First, we created the bones for the tail.
2. We created a joint per every row of geometry (or every two, in order to have a lighter skeleton).
3. We checked out that LRAs (local rotation axis) are all the same.
4. Then we duplicated the whole thing three times, and we named one BIND, second one FK, third one IK, and fourth one FullFK.
5. Next, we created layers and hid all of them except Full FK skeleton for now.
6. We created our control curves for each joint, renamed them CTRL_tail_base, start, mid, end.
7. Then we created a null group and duplicated it, so we had a null group for each controller.
8. Then we point- and orient-constrained the nulls to the joints, one by one, and deleted the constraints, so that they had the same position and rotation information with the joints.
Figure 11.1
Final image of monkey.
“Paul and his Bananas” written and directed by Floyd Bishop
Character Design by Benjamin Plouffe
Character Model created by Tristan Lock
Character Rig by Firat Enderoglu
Character materials and fur by Floyd Bishop
TM & © 2005 Bishop Animation
9. Then we parented the curves under each null group and did freeze transformations on them so that they would get the rotation information from the null group while keeping their rotation values 0.

10. Now we have one curve for each joint positioned at the same place with the same rotation information as the joint. Then we opened the Connection Editor and connected the rotations of the curves to the joints, one by one (we can do the same thing in a dependency graph).

11. Now, we have all the curves that rotate the joints. Starting from the very last curve, we started parenting. We took the null group above the last curve and parented it under the adjacent curve and then took its null group and parented it under an adjacent curve. So it’s like Null1 -> Curve1 <- Null2 fCurve2. By doing that, if we selected all the curves together and rotated them, it would make a curl.

12. And finally, we parented the very top node under Hip Null node and locked/hid translates and scale attrs on the curves. Then we hid FullFK (see Figure 11.4).
Inspired 3D Advanced Rigging and Deformations

This time we turned on the FK tail and did a similar thing without using direct connections.

1. First, we created three or four controllers and placed them with equal spaces along the length of the tail and then did a freeze transformation on them.

2. After that, we opened the Set Driven Key window and took the curve control as the driver and picked the joints that were close to that curve control as driven. Then we set drive the rotations of all those joints to that curve control so that when you rotate the single control, several joints will rotate together.

3. We did the same thing for the other two curves and then parented them under to the first joint that they controlled.

4. Then we made only the IK tail visible and created an IK Handle from the beginning of the tail to close to the end, about 3–4 joints until the end.

5. Next, we created an empty node, pointed- and orient-constrained it to the first joint after the IK part, and deleted constraints.

6. We created a curve and placed it on top of the IK Handle, parented it under that empty node and did freeze transformations, and then in the Connection Editor connected the rotations of that curve to the very first joint's rotations (the one after the IK part). Then we parented the IK Handle under that curve to finish it off (see Figure 11.5).

For the last 3–4 joints left, we used the same method that we used on FullFK tail.

1. We create d3–4 curves and 3–4 empty nodes and then point- and orient-constrained them to the joints and deleted the constraints.

2. We parented the curves under the empty nodes.

3. We did freeze transformations on the curves.

4. We connected the rotations under the Connection Editor or dependency graph.

5. Starting from the last one, we parented the curve's group node under the adjacent curve.

6. This time, we parented the main group node (the one above the curve that controls IK Handle) under the placement node of our full skeleton.
Now that we had all our different tail rigs with all working controllers, the last thing we needed was to have all these rigs control the bind one that we hadn’t touched yet, so we could switch between these rigs easily. We wanted to orient-constrain the BIND skeleton joints to all these rigs.

First, we have FullFK and BIND on our Hypergraph or Outliner. Then we selected the first joint on fullfk and then selected the same joint on bind skeleton and orient-constrained it. We did this for every joint. After that, we did the same thing with IK and FK skeletons. So now we had all the BIND joints orient-constrained, and under the Channel Box we had three attributes to control the weights on each constraint (such as FullFKw0, IKw1, and FKw2). We wanted to create a float attribute on any controller, such as the hips or on the main controller of the character and name it tail switch (or whatever). We restrained that attribute’s values to -5 to 5, or -10 to 10. Then we are opened the Set Driven Key window and loaded that attribute as driver and loaded all the constraints under the Bind skeleton as driven. We keyed on the last three attributes on the constraints (the ones that should have W-and number). See Figure 11.6.

If we wanted -5 FullFK, 0 FK, 5 IK modes, while the driver was at -5, we would adjust the constraint weights on all of them (fkW* and ikW* ones are going to be 0 and fullFkW* will be one) and Set Driven Key them. We will do the same thing for FK and IK modes and that’s it. One last thing we could do is to set drive the control curve’s visibilities to the controller attribute so that when we switch we won’t see the other rig’s controllers (see Figure 11.7).
Additional Characters and Rigging

In addition to Paul the Monkey, character rigging for several other characters was required for this short. Below we look at some of the rigging for the Magicians’ hat. Figure 11.8 shows the final rendered character and the base mesh for the model is shown in Figure 11.9. You can see the exaggerated base pose and the edge flow resulting from using Maya Sub-D surfaces.

Figure 11.8
Final image of magician.
Here is a quick soft body tutorial for the magician’s hat. This same technique can be used for many things like coat tails, long ears, or belt straps on your characters.

1. Create the bones for the hat. Make sure to add an extra joint as a spacer to limit the effect of the spline IK and to allow the hat to be removed, if needed (see Figure 11.10).
2. Create a spline IK from the base to the tip.
3. Now select the spline Ik curve only from the Outliner or Hypergraph.
4. From the Dynamics→Soft/Rigid Bodies→options box set the creation option to Duplicate Make Original Soft and turn on “make non-soft a goal.” This way the spline IK curve will have particles in place of the CVs and those particles will follow the duplicated curves’ CVs.
5. Then create clusters on the non-softbody curve’s CVs for extra control over the animation. You can parent the clusters under extra FK control hierarchy for direct animation control. Additional control over the effect can be hooked up by painting or editing the goalWeight to control the floppiness.
6. With any automated control, hook it up to a custom attribute that can override the dynamics and turn off/on or can be set to some percent in-between blending between the dynamics and the hand key-framed animation (see Figure 11.11).
11. Tail Rigging for Paul and his Bananas

Figure 11.11
Custom attributes setup and the final rig controls.

Figure 11.12 shows the Banana character rig, using the combination of basic rigging controls and mixing in the softbody rig for the banana peels.

Thanks to Bishop Animation Studios for putting together this behind-the-scenes look into their character rigging process. Be sure to check out http://www.bishopanimation.com/ for the latest updates on this film.

Figure 11.12
Additional character rig and skeleton—a quick view inside the Banana character.