Chapter 8. Finalizing the Model

In this chapter, we will cover the following recipes:

- Creating shape keys
- Assigning drivers to the shape keys
- Setting movement limit constraints
- Transferring the eyeball rotation to the eyelids
- Detailing the Armor by using the Curve from Mesh tool

Introduction

In this chapter, we'll see how to create and add **shape keys** (the Blender term for **morphing**) to the model, to create facial expressions for the **Gidiosaurus** and to add shape modifications in a non-destructive way to the model.

Then, we'll see how to set a limit to the **Armature** bones' rotation using constraints and how to slightly transfer a portion of the rotation movement of the **eyeballs** to the covering **eyelids**.

Last, we'll add some detail to the **Armor** by quickly adding **rivets** through a simple and effective technique.

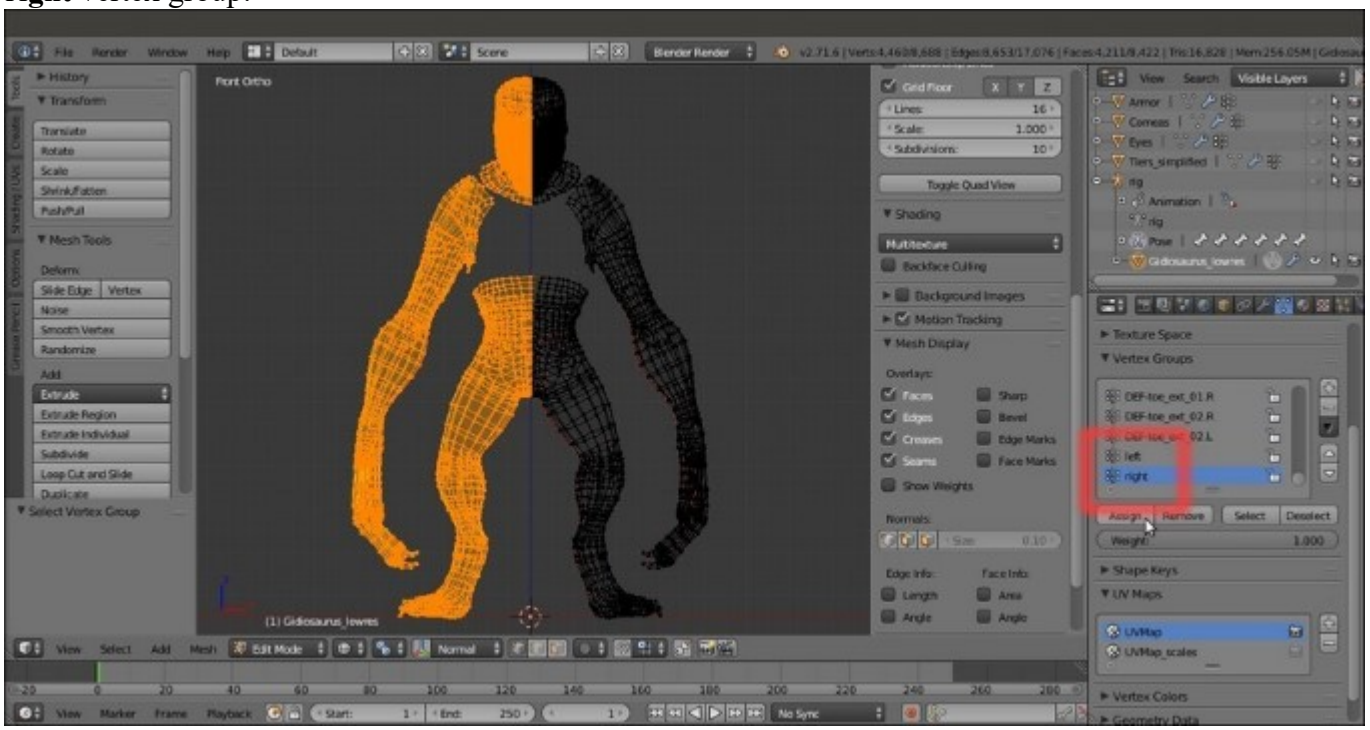
Creating shape keys

In this recipe, we'll set the **shape keys** to create (even if limited) **facial expressions** and to fake the stretching and the contracting effect of the character's **arm muscles**, and we'll add some more shape keys to fix issues in the character's shape.

Getting ready

First, let's prepare a bit the scene and the model:

1. Start Blender and load the `Gidiosaurus_skinning_rigify.blend` file, which is the same as the `Gidiosaurus_skinning_03.blend` file but with the **rig** created by the **Rigify** add-on (and later edited to add the other bones exactly as explained in the last chapter's recipes).
2. In the **Outliner**, click on the respective eye icons to hide the **Armor**, **Eyes**, and **Tiers_simplified** objects and the **Armature**, whose name in this case is **rig**.
3. Select the **Gidiosaurus_lowres** object and press the *1* key on the numpad to go into the **Front** view.
4. Press *Z* to go into the **Wireframe** viewport shading mode and the *5* key on the numpad to switch to the **Ortho** view if it is not already.
5. Enter **Edit Mode** and box-select the **left half** of the mesh vertices (including the **middle vertical edge-loop**) and in the **Vertex Groups** subpanel under the **Object Data** window, create a new vertex group; rename it **left** and assign the selected vertices a weight of **1.000**.
6. Deselect everything and repeat this process for the **right half** of the mesh's vertices to create the **right** vertex group:



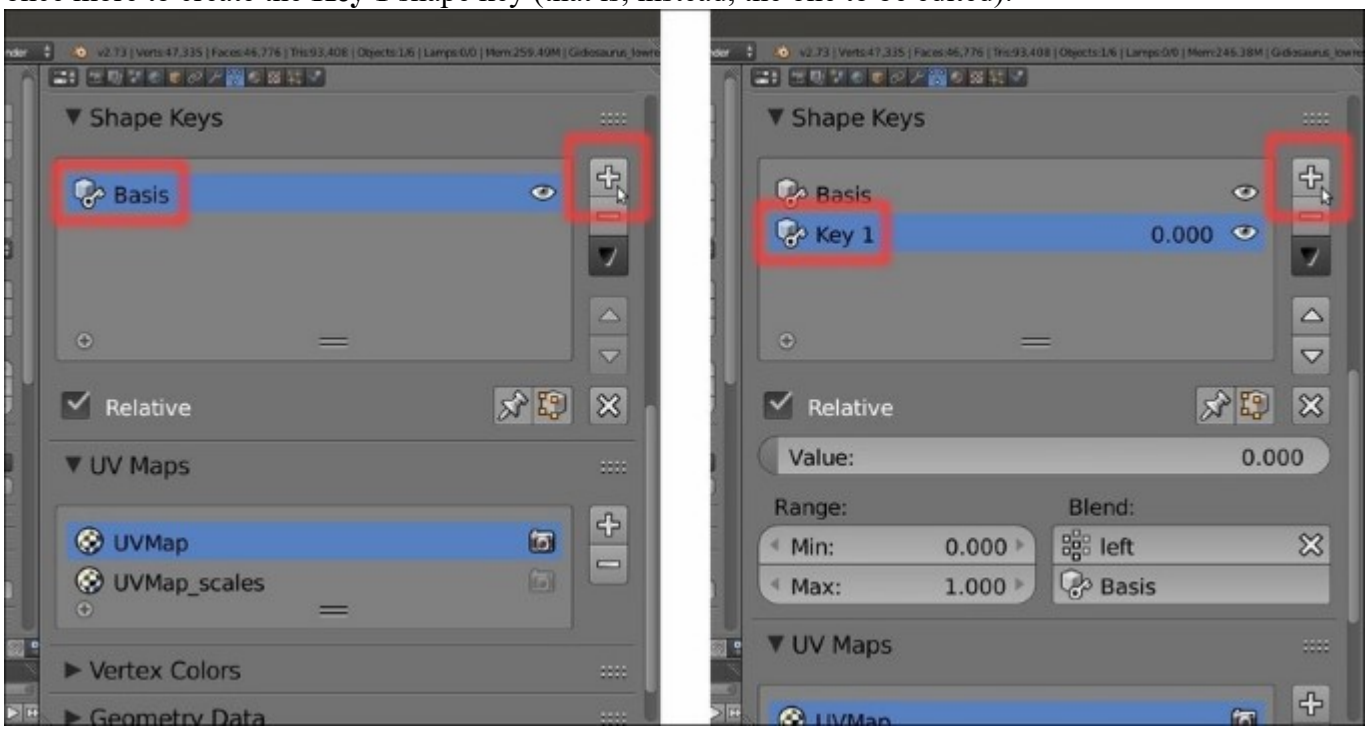
Assigning the mesh's right side vertices to the "right" vertex group

7. Save the file as `Gidiosaurus_shapekeys.blend`.

How to do it…

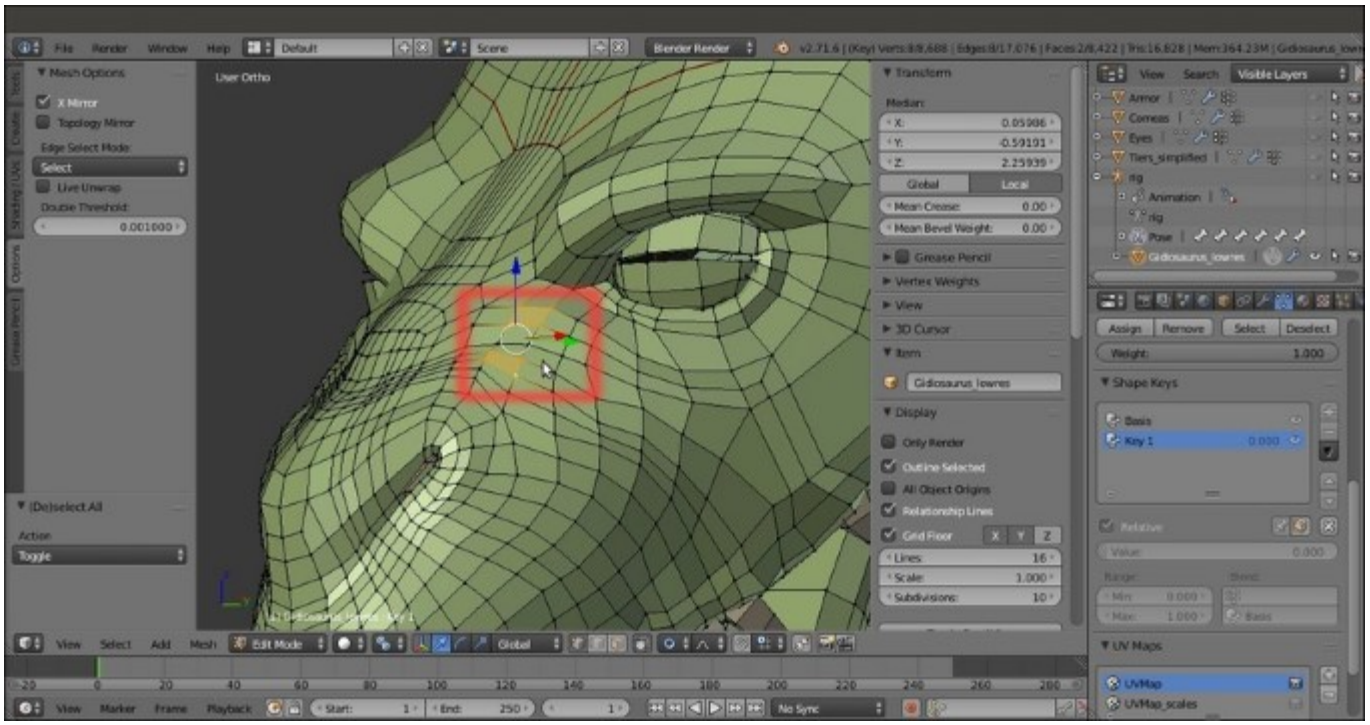
Let's start now by creating the **facial expressions** shape keys:

1. Exit **Edit Mode** and expand the **Shape Keys** subpanel under the **Object Data** window; click on the + icon button to the top left to create the **Basis** shape key (that mustn't be edited), then click once more to create the **Key 1** shape key (that is, instead, the one to be edited):



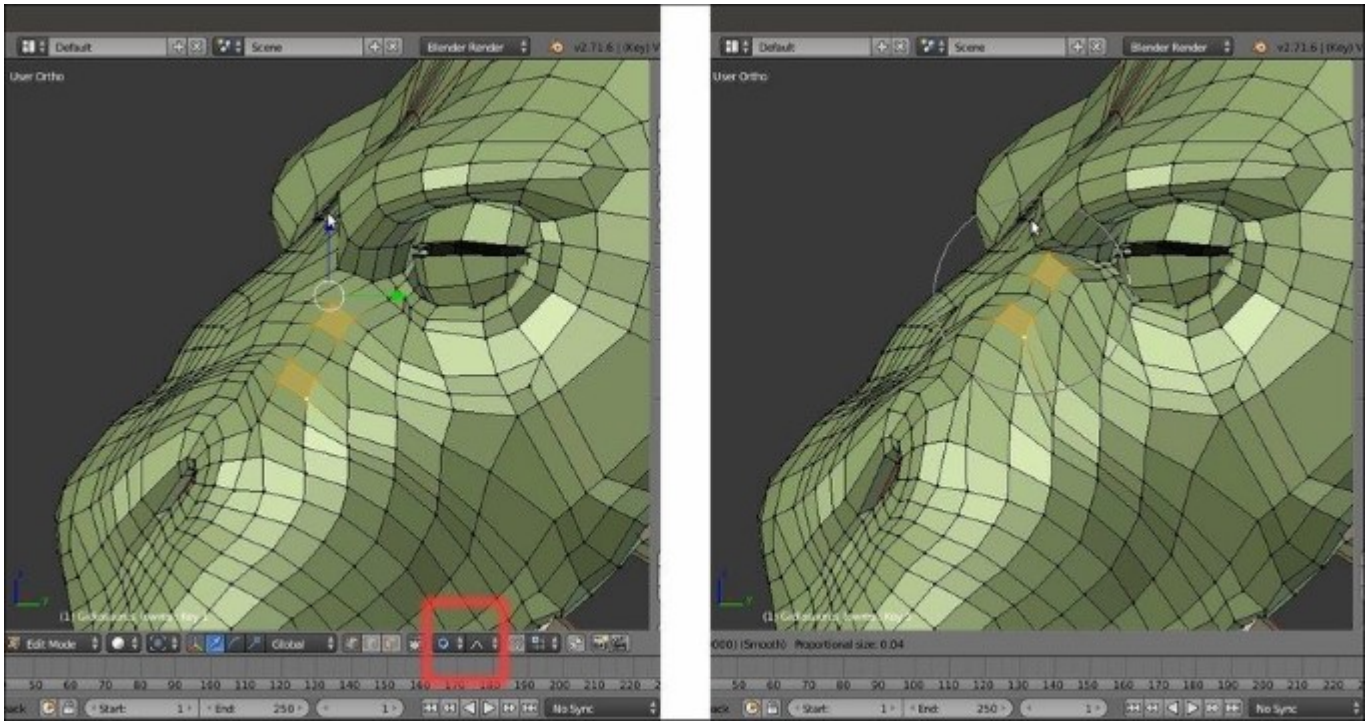
Creating the Basis and the first shape key

2. Be sure that the **X Mirror** item in the **Mesh Options** tab under the **Tool Shelf** is activated and click on the **PET** (**Proportional Editing Tool**) button in the 3D viewport toolbar (or activate it by pressing the *O* key).
3. Zoom to the **Gidiosaurus** head and again in **Edit Mode**, select some of the vertices at the center of the **snout** area, as indicated in the following screenshot:



Selecting the vertices

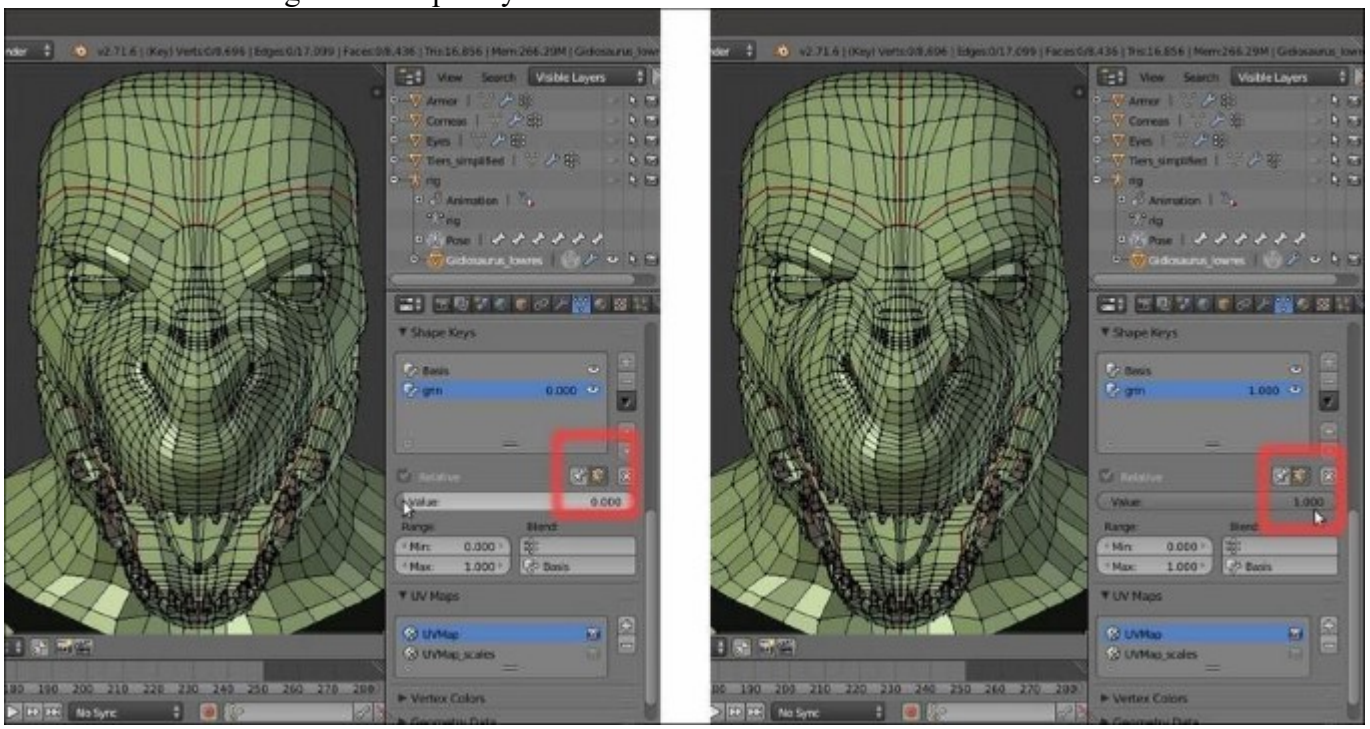
4. Move them upwards and towards the **eye**, set the amount for the **PET** smoothing by scrolling the mouse wheel:



Moving the selected vertices slightly backwards and upwards

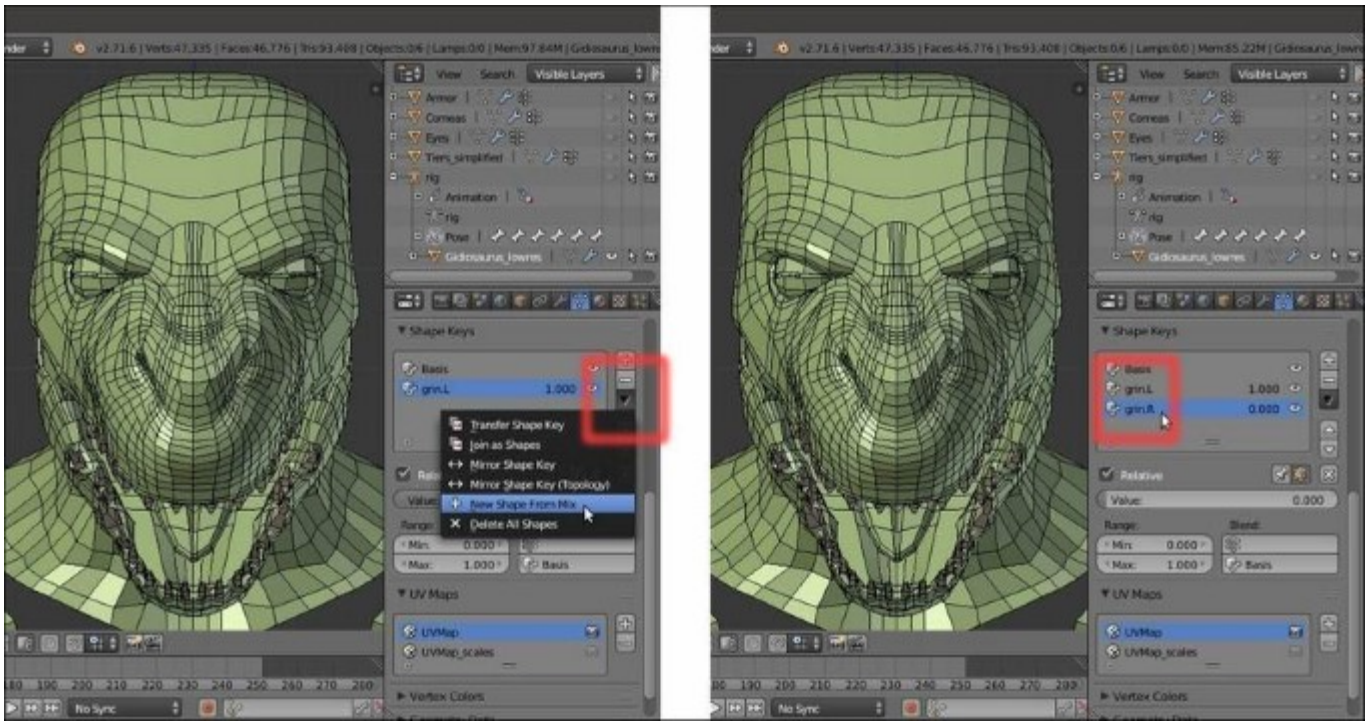
Note that we selected two faces instead of the folder edges, because the muscular *scrunching* involves both movement of the skin and a slight folding of the skin as well; the middle edge does not move as much as the selected faces due to **PET** falloff, hence creating a very slight scrunch and a more *naturalistic* skin sliding.

5. If necessary, adjust the position of the single vertices, maybe also disabling the **PET**.
6. Go to the **Shape Keys** subpanel and rename the **Key 1** shape key as **grin**.
7. Click on the *Apply shape key in edit mode* button located right above the **Value** slider to enable it; go into the **Front** view and by moving the **Value** slider from **0.000** to **1.000** and back, check for the correct working of the shape key on both the sides of the character:



Checking how the "grin" shape key works

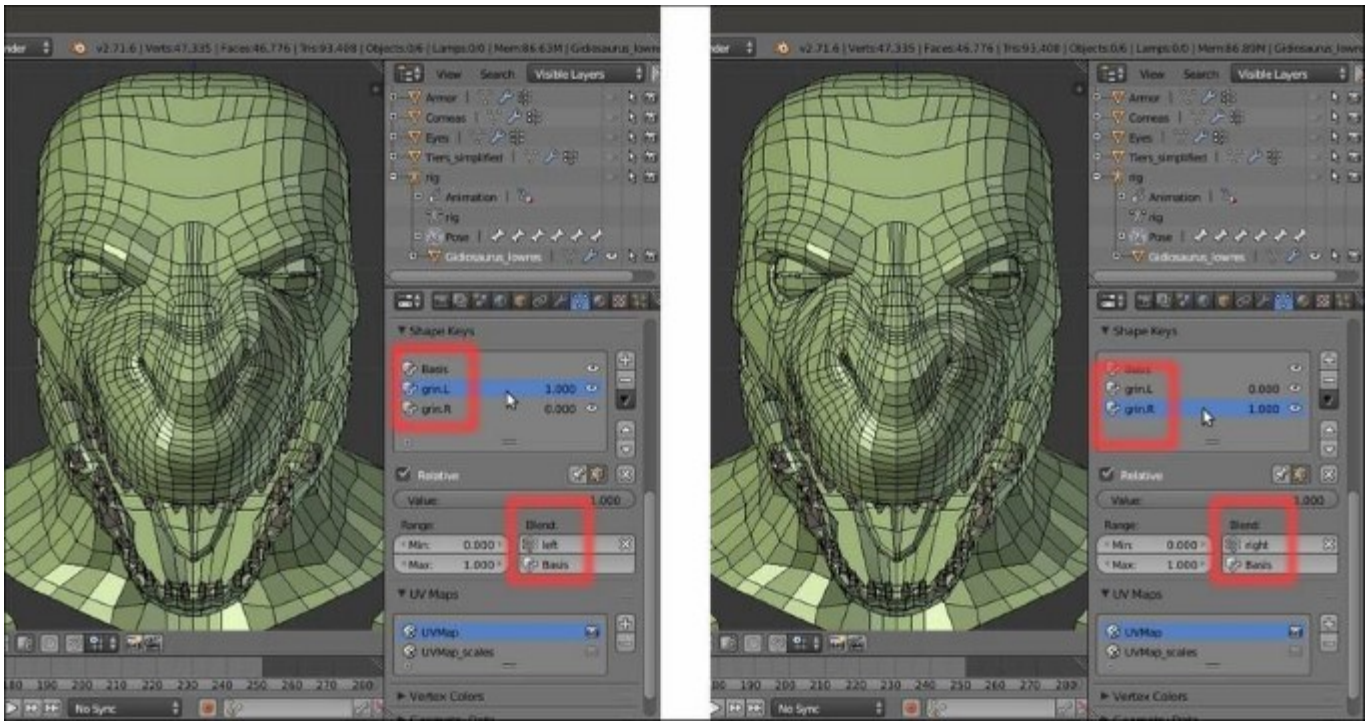
8. Go out of **Edit Mode** and just to have better visibility of the shape key modifications, go to the **Object** window to enable the **Wire** item in the **Display** subpanel.
9. Go back to the **Object Data** window and click on the button that has an icon of a downward pointing arrow (*Shape keys specials*); from the pop-up menu select the **New Shape from Mix** item: this adds a new shape key made by the sum of all the active shape keys. In this case, it is just a perfect copy of the sole **grin** shape key (the **Basis** shape key is, well, just the base starting position of the vertices in the mesh). Rename the two shape keys as **grin.L** and **grin.R**.



Copying the "grin" shape key to a new one

10. Click on the **grin.L** shape key, set the **Value** slider back to **0.000**, and then click on the *Vertex weight group* slot, the one under the **Blend** item and above the **Basis** one, and select the **left** vertex group.
11. Repeat for the **grin.R** shape key by selecting the **right** vertex group, and then once again set the **Value** slider to **1.000** and ensure it works correctly.

    Each shape key now works only on the respective side, according to the selected vertex group:

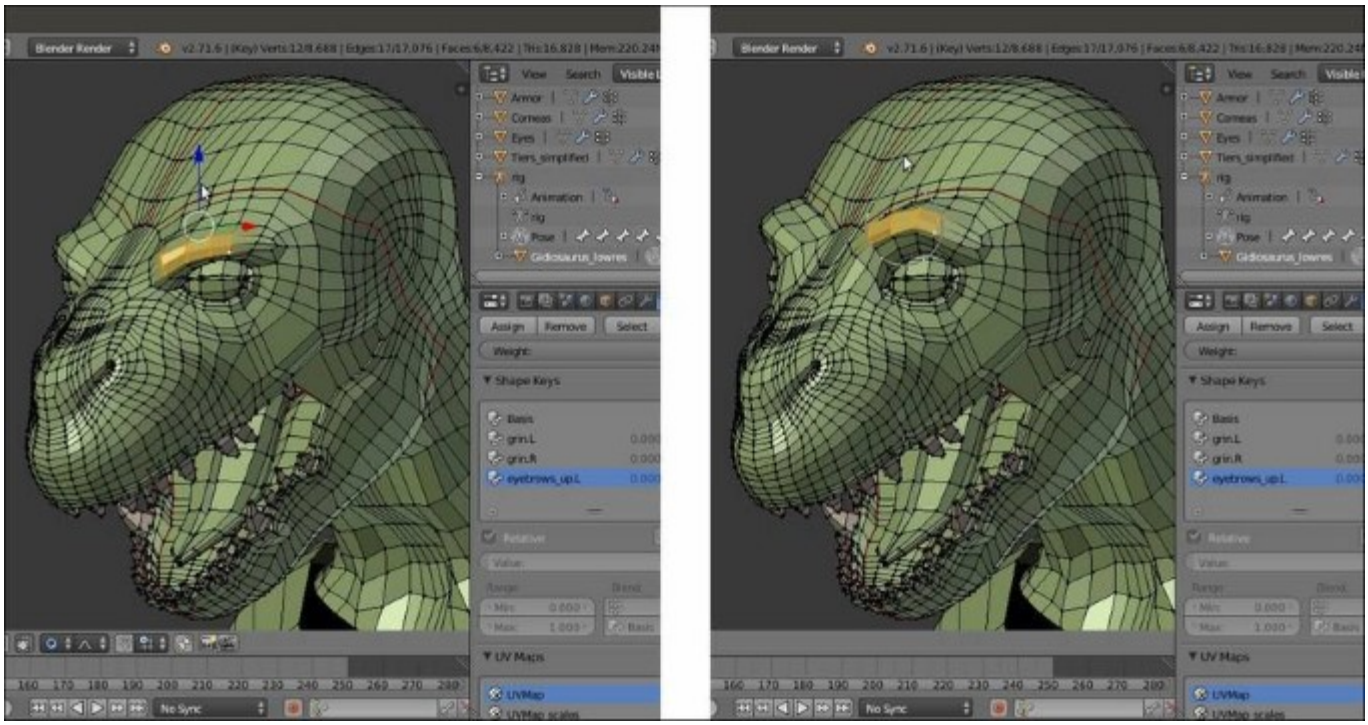


The two "grin" shape keys for the right and the left sides

This was for the **grin** expression; now we need to add at least two or three more kinds of shape keys, namely: two for the **eyebrows** (up and down) and one for the **nostrils**, multiplied for each side.

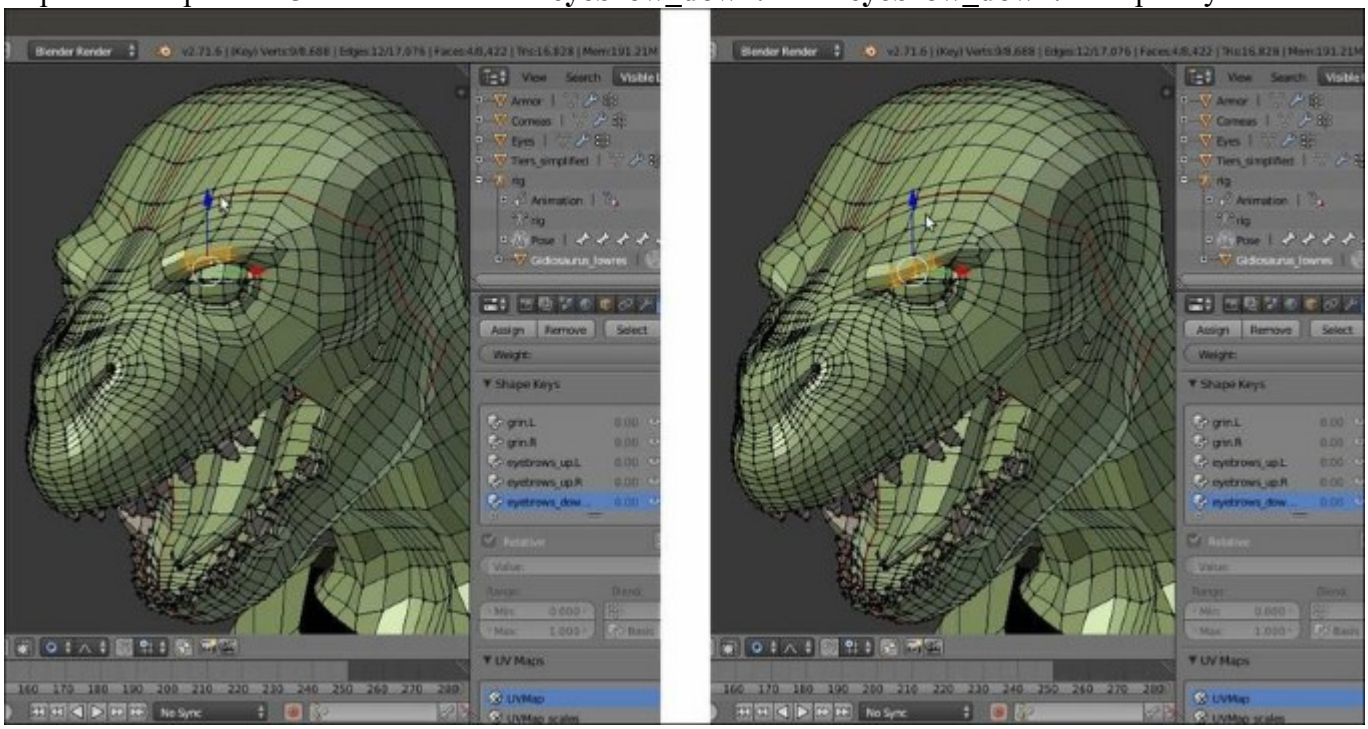
This means **six** more shape keys in total, but as you have seen, the procedure is quite quick and simple.

12. Set the **Value** slider for the **grin.L** and **grin.R** shape keys back to **0.000** and click on the + icon button to add a new shape key.
13. Rename it **eyebrow_up.L** and enter **Edit Mode**; grab some vertices on the left eyebrow and, still with the **PET** activated, move them upward; you can use the mouse wheel to set the influence of the **PET**:



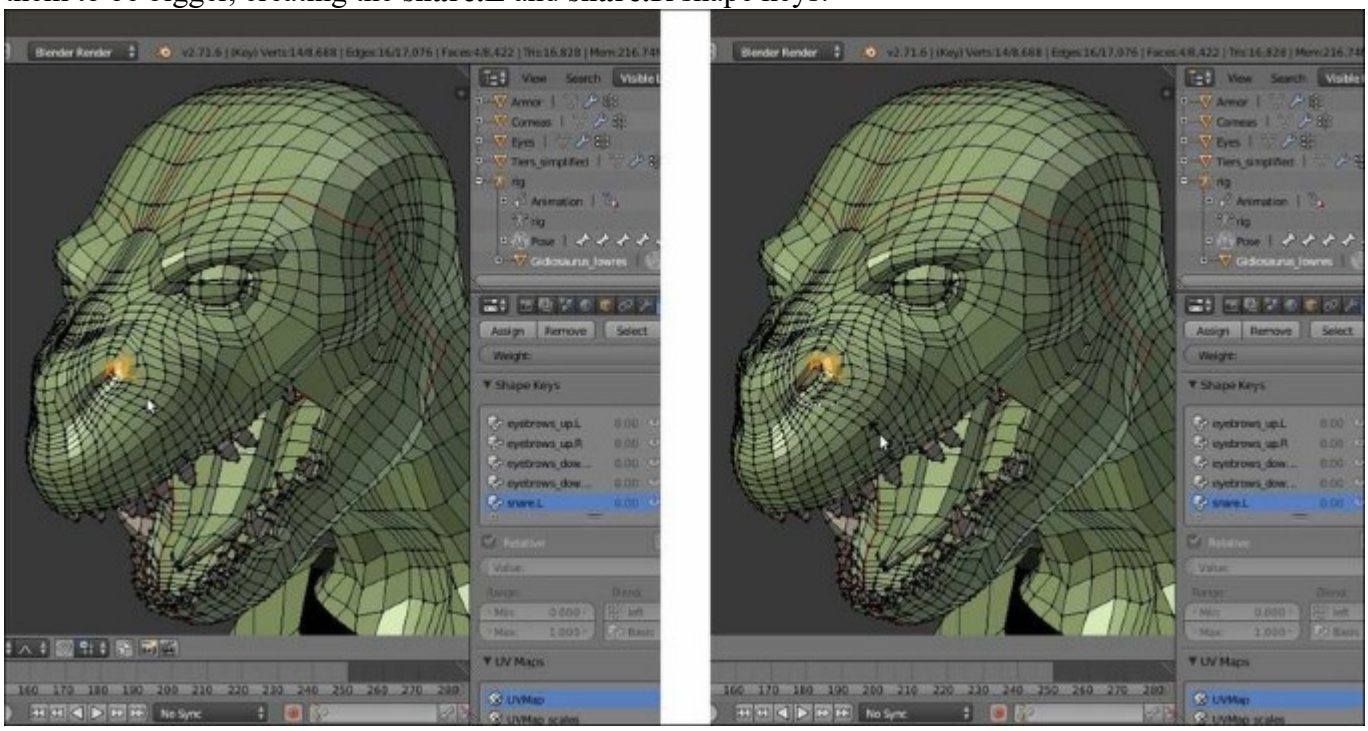
Moving the eyebrow upward

14. Repeat the steps from 9 to 11 to create the **eyebrow_up.R** shape key.
15. Repeat the steps from 3 to 11 to create the **eyebrow_down.L** and **eyebrow_down.R** shape keys:



Moving the eyebrow downward

16. Finally repeat step 3 to step 11, this time selecting the vertices around the **nostrils** and scaling them to be bigger, creating the **snare.L** and **snare.R** shape keys:

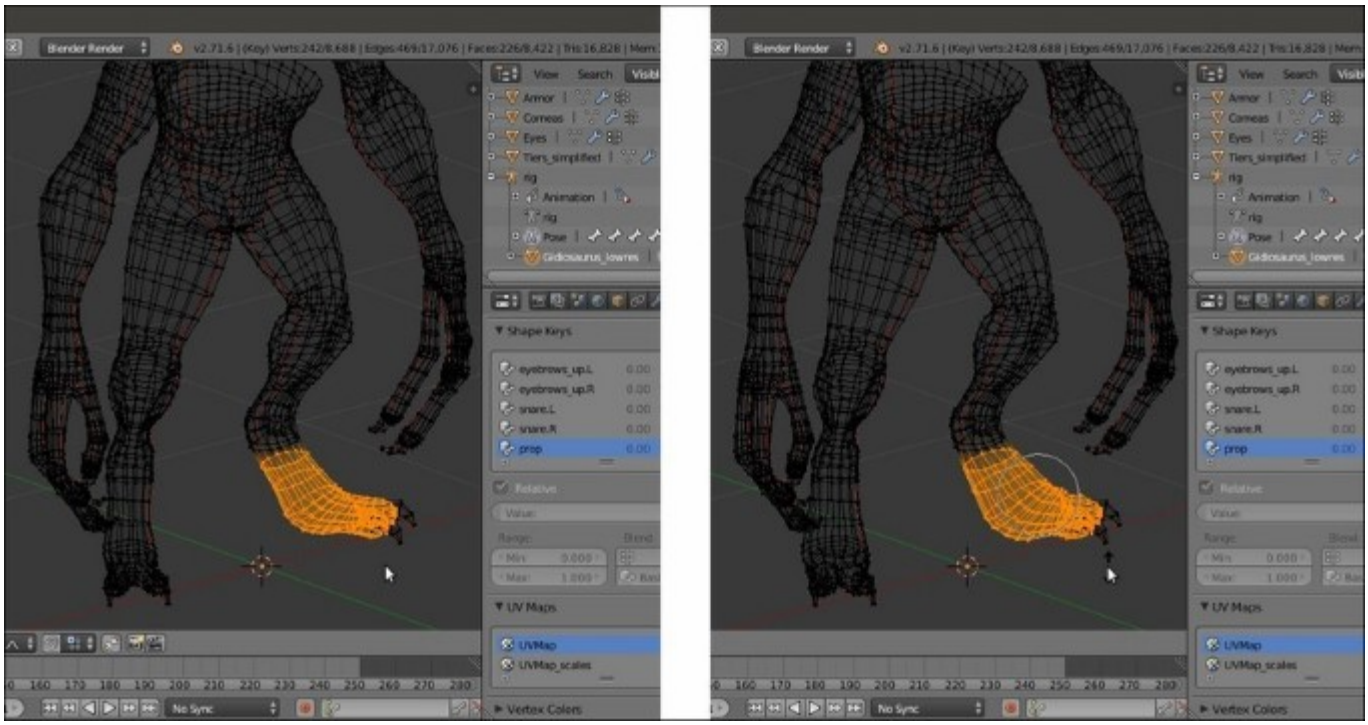


The nostril flaring

Note that, when naming the shape key for the enlargement of the nostrils, I erroneously wrote **snare**; it should have been something like *snarl* or *flaring*, but in the end it's just a naming convention and therefore, this little mistake doesn't pose a real problem.

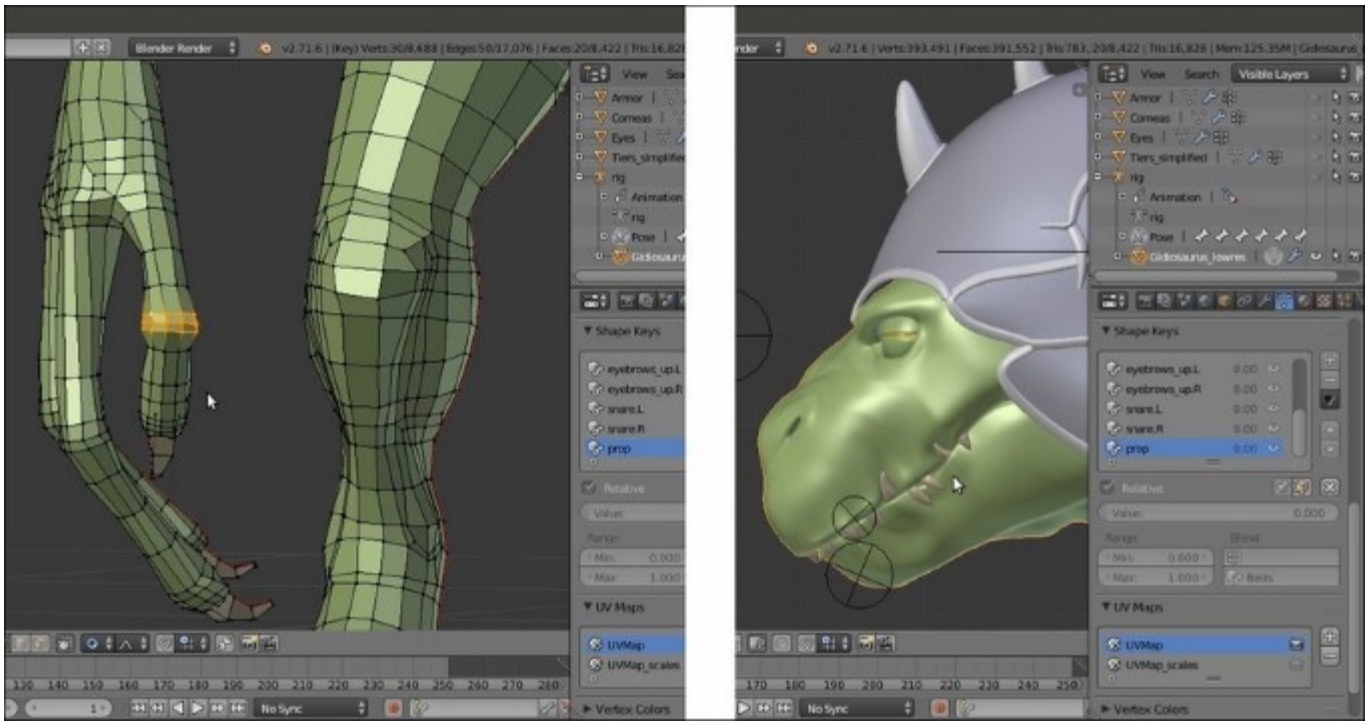
We are done with the facial expressions; now let's add one more shape key to enhance some of the body features of the **Gidiosaurus** a bit; these are not meant to be animated during the animation, but are simply a way to apply non-destructive modifications to the model.

17. Add a new shape key and rename it **prop** (for *proportions*).
18. In **Edit Mode**, select the vertices of the **left foot**, excluding the **feet talons**, and press *Alt* + *S* to scale them *on their normals*; if you are using the **PET**, just be sure to be in the **Connected** mode (so that the **PET** has influence only on the vertices connected to the selected ones, otherwise the unselected **feet talon** vertices will also be modified):



Modifying the feet proportions

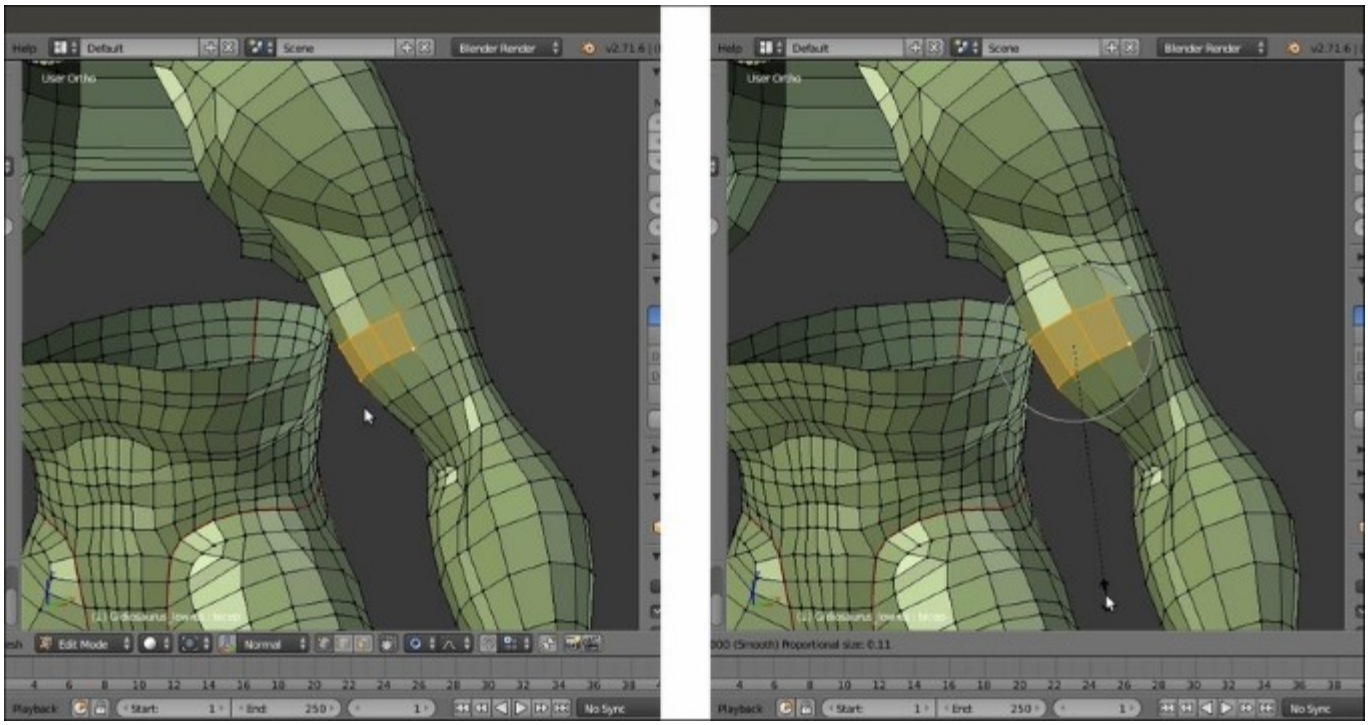
19. Disable the **PET** and adjust the transition between the scaled vertices and the surrounding ones, the area between the two **toes**, and so on.
20. If you wish, you may also make additional modifications; in my case, besides the bigger **feet**, I simply enhanced the **knuckles** on the **hands** and at the **fingers'** joints. Also, I tweaked the rim shape of the upper and bottom borders of the **mandibles** a bit:



Enhancing some of the character's features

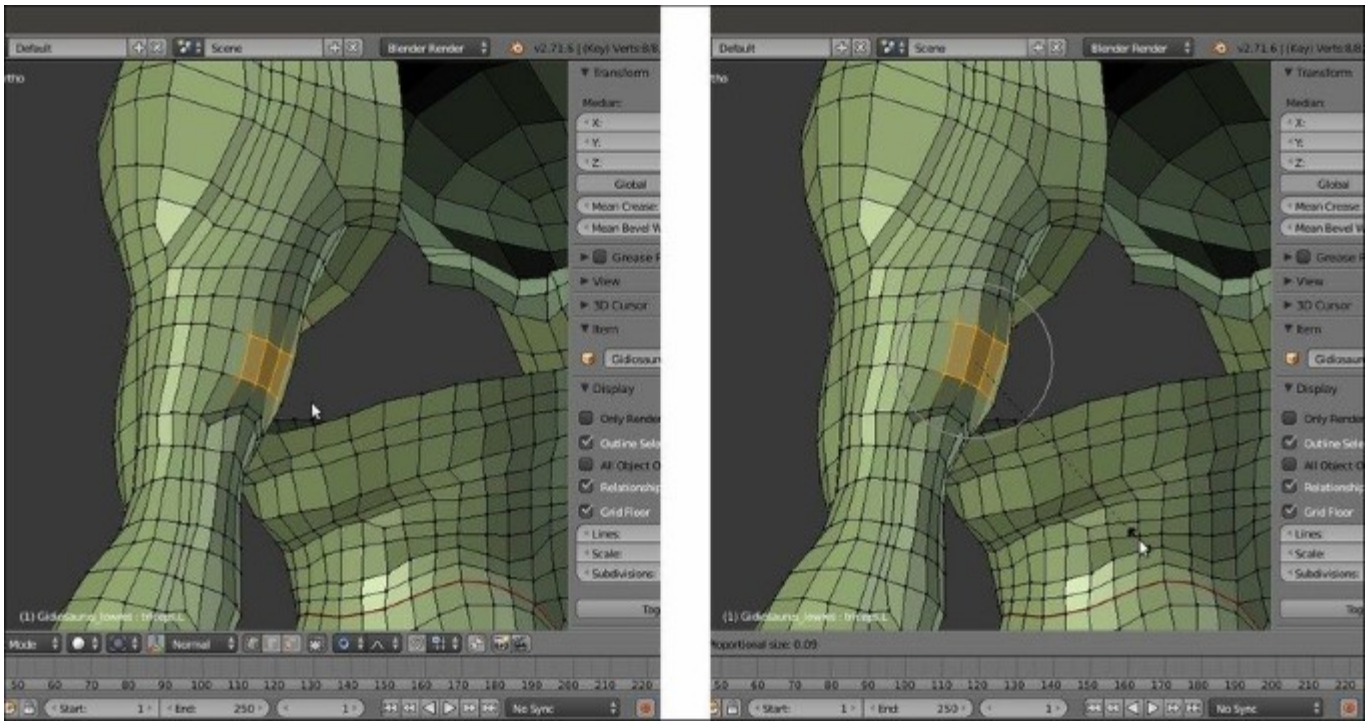
21. When you are done, set the **Value** slider of the **prop** shape key to **1.000**.

    Now, let's add a couple of shape keys to mimic the movement of the main muscles of the arms, specifically of the **biceps** and of the **triceps** muscles:
22. Add a new shape key, then go to the **Gidiosaurus** mesh; select some of the vertices in the middle area of the **bicep** muscle and after enabling the **PET** again, move them forward and also scale them to be bigger, to make the muscle *grow*:



Making the bicep muscle grow

23. Rename the shape key **bicep.L**, and then repeat the steps from 9 to 11 to create the **bicep.R** shape key.
24. Add a new shape key and repeat everything by selecting vertices on the back of the arm, in the **triceps** area, to create the **triceps.L** and the **triceps.R** shape keys.



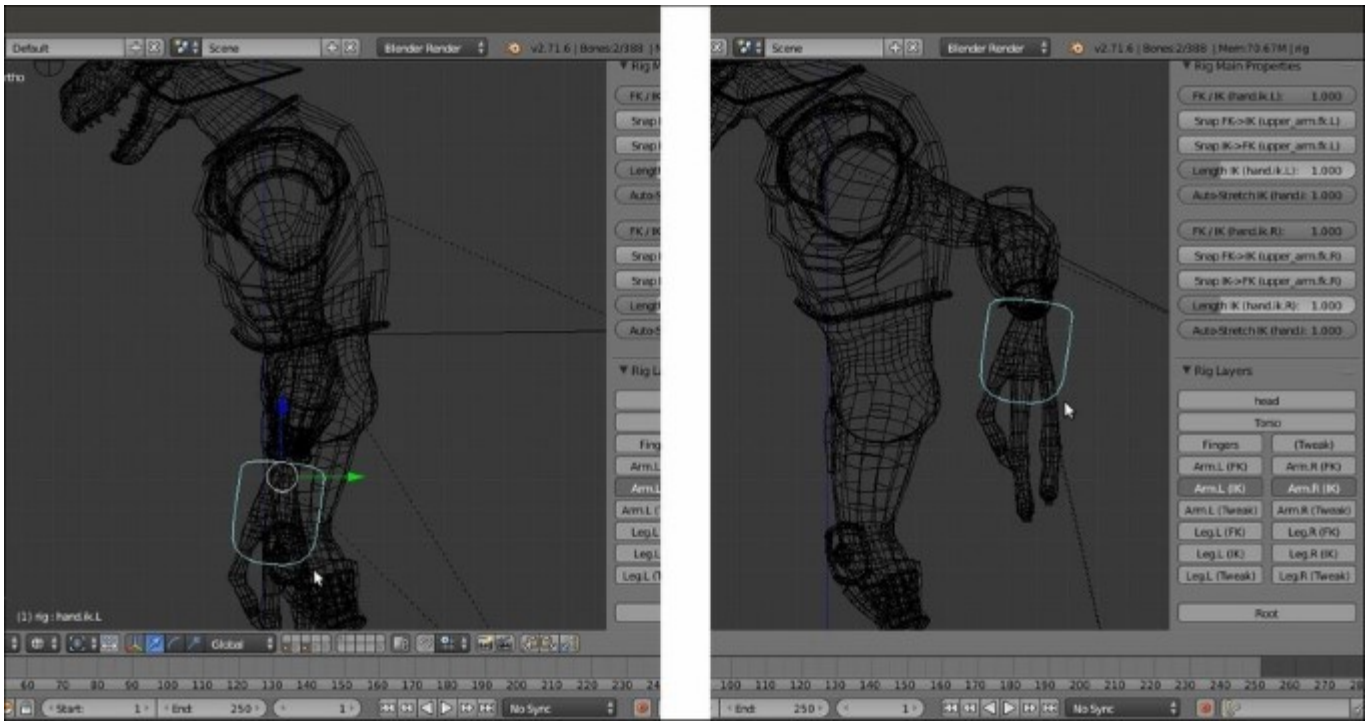
Making the triceps muscle grow

25. Leave the values of these last four shape keys as **0.000** and exit **Edit Mode**.

    More shape keys could be added to simulate a complete muscle system, but in our case we stop here with the **Gidiosaurus** mesh; now let's concentrate on the **Armor**.
26. Go to the **Outliner** and unhide both **Armor** and **rig**.
27. Select the **rig** and then press *N* to call the **Properties** 3D view sidepanel; scroll to the bottom and first go to the **Rig Main Properties** subpanel to set both the slider for **FK/IK (hand.ik.L)** and **(hand.ik.R)** to **1.000**, then go down to the **Rig Layers** subpanel and deselect everything except for the **Arm.L (IK)** and **Arm.R (IK)** buttons.

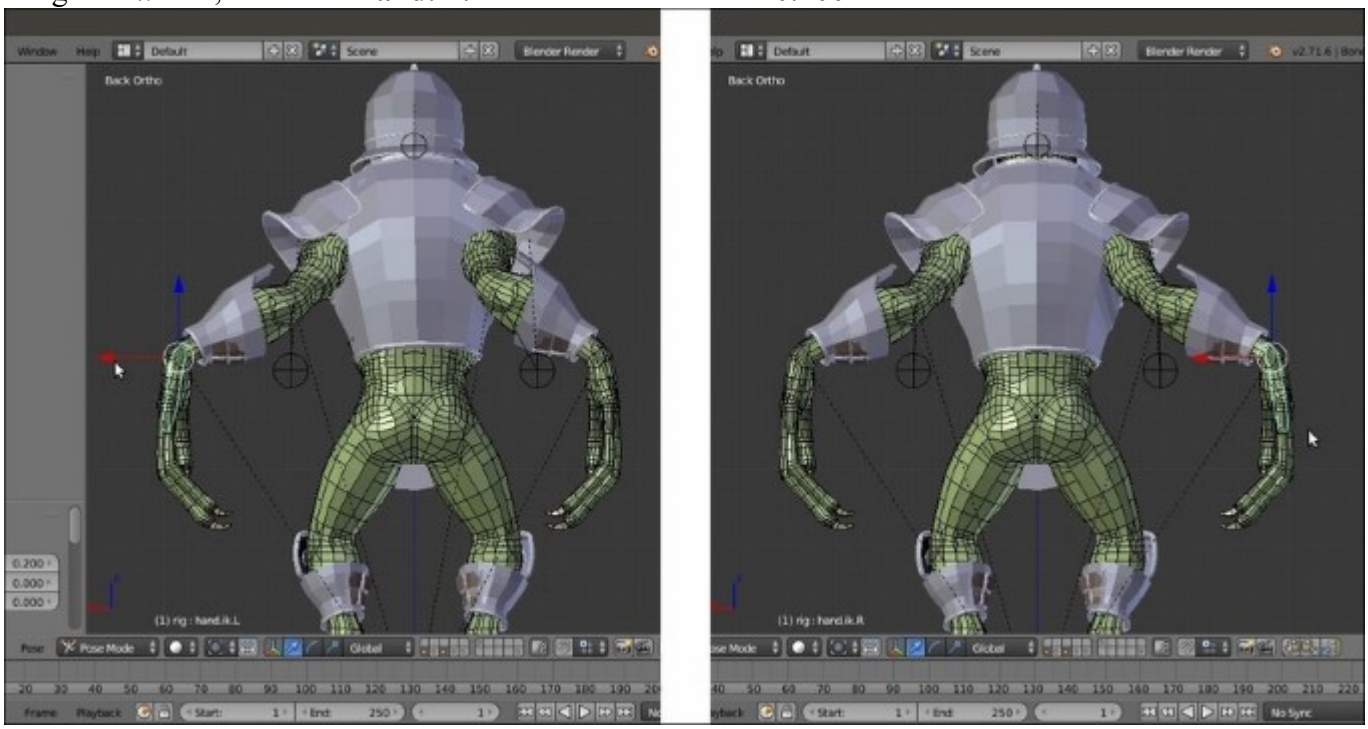
    ## Note

    Note that if the **FK/IK sliders** don't appear, it's because you have to select one of the hand (**ik** or **fk**) bones in the viewport first.
28. Go to the 3D viewport and select the **hand.ik.L** and **hand.ik.R** handle bones, go into the **Side** view and move them towards the upper back.



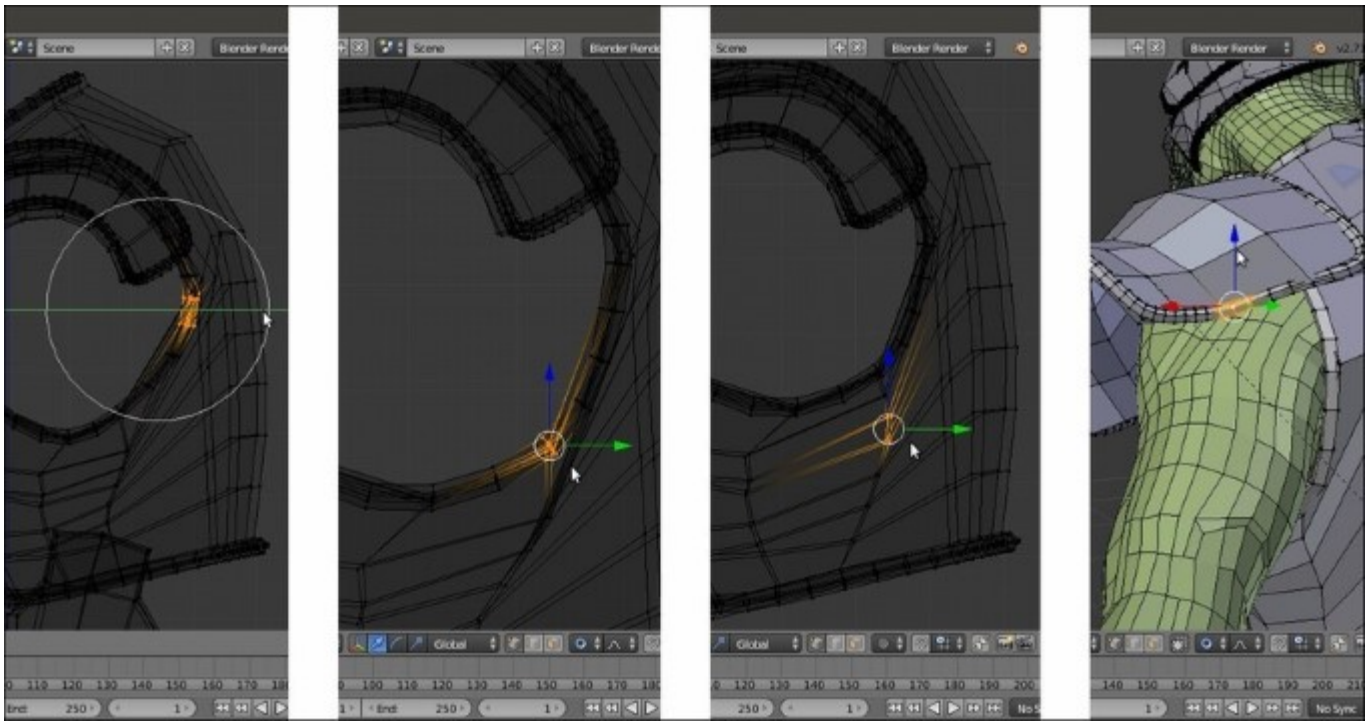
Moving the arm control bones backward using the Inverse Kinematics

29. Now press *Ctrl* + numpad *1* to go in **Back** view and move the **hand.ik.L** bone to **0.200** along the global *x* axis; select the **hand.ik.R** bone and move it to **-0.200**:



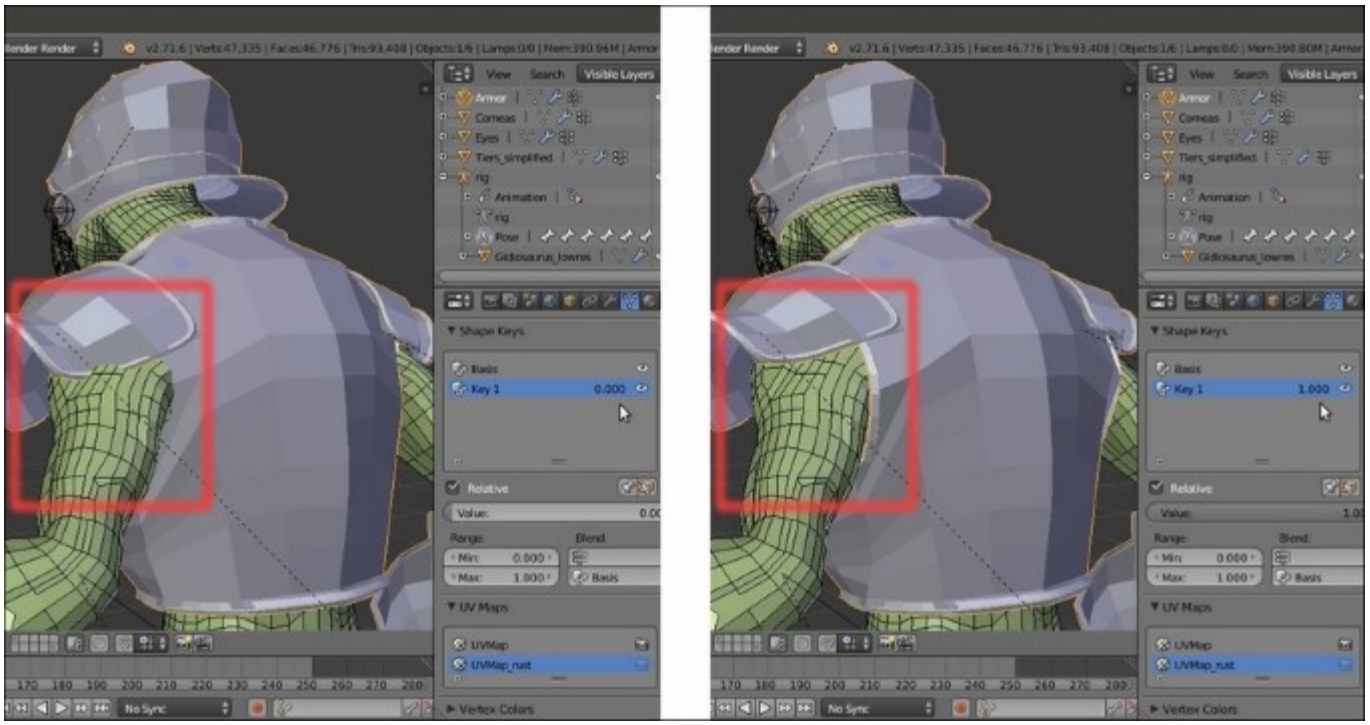
Adjusting the lateral position of the arms

30. Now select the **Armor** object and add the **Basis** shape key in the **Shape Keys** subpanel under the **Object Data** window, and then add **Key 1**.
31. Enter **Edit Mode** and press the slash key (/) on the numpad to go in **Local** view; this way only the selected objects are visible in the viewport, in our case only the **Armor**. Using the **Proportional Editing** mode, start to enlarge the back section of the opening for the **arms**, adjust the position of the surrounding vertices as required, and also raise the back vertices of the **spaulders** a bit, to avoid interpenetration with the borders of the **chest plate** and the **shoulder** as well.
32. Press the numpad slash (/) key again to go out of the **Local** view and check for the correction with the **Gidiosaurus** mesh:



Editing the position of the Armor vertices for a new shape key

33. When you are done, just exit **Edit Mode** and set the **Value** slider of the **Key 1** shape key for the **Armor** to **1.000**:



The shape key working as a fix for the Armor

34. Rename the **Key 1** shape key as **Armor_fix** and save the file.

How it works…

Although technically there are no differences, we could say that we created three different types of **shape key**:

- One type to fix shape errors or make improvements in the mesh, for example, with the **prop** and the **Armor_fix** shape keys
- The second type to modify the mesh only at certain established moments during the animation process, in our case just to animate facial expressions
- The third type to simulate muscle movements in the character

Shape keys work in **linear space**; that means that it's not possible to make vertices rotate around a pivot through a shape key, but only to move them *from point A to point B*. That's why we didn't use shape keys for stuff like the **eyelid** movements, for example, or the opening/closing of the **jaw**, but only for actions including muscles sliding above the bones such as the **eyebrows**, the **grin**, and the **nostrils**, as well as the **bicep/triceps** movements.

Thanks to shape keys, we also made *last minute* modifications and improvements to the **Gidiosaurus** mesh and to the **Armor**; being included inside a shape key, all these modifications are *non-destructive* and can be turned on or off at will, or their influence can be set at an intermediate strength value.

When modifying a mesh using a shape key, be careful not to change too much of the mutual proportions of articulated parts of a *to-be-deformed* mesh; for example, it's usually problematic to scale a whole part such as the **hands** or the **head** of a character, both smaller or bigger, unless you also scale the corresponding bones of the rig and the joints' position accordingly as well.

Beyond a certain threshold, the bones of the **fingers**, or of the **eyes** and **jaw**, start to be *out of register* compared to the respective mesh's edge-loops and you'll have to fix this by re-positioning the joints of the **Armature's** bones (just in case, remember: always do this in **Edit Mode**).

In our example, with the **prop** shape key, we just restricted ourselves to enhance the hands' **knuckles** and to make stronger **feet** by simply making the vertices' positions grow in the direction of their normals.

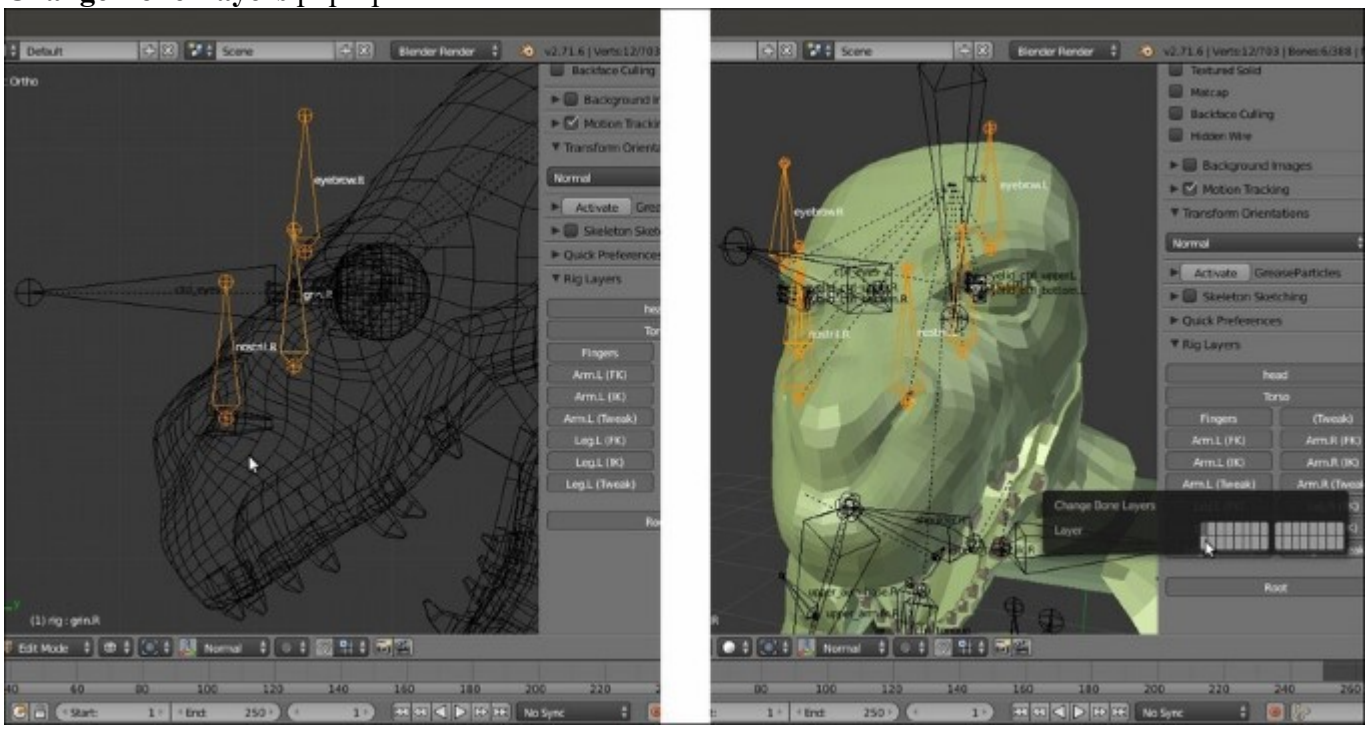
Assigning drivers to the shape keys

In the previous recipe, we created three different types of **shape keys**. Besides the *fixing* shape keys, that have a fixed value (no pun intended), we now need a way to set the amount of influence of the other two types of shape keys, facial expressions, and the muscle movements during the animation. This is accomplished by setting **drivers**, though with different kinds of controls.

Getting ready

Start from the previously saved `Gidiosaurus_shapekeys.blend` file:

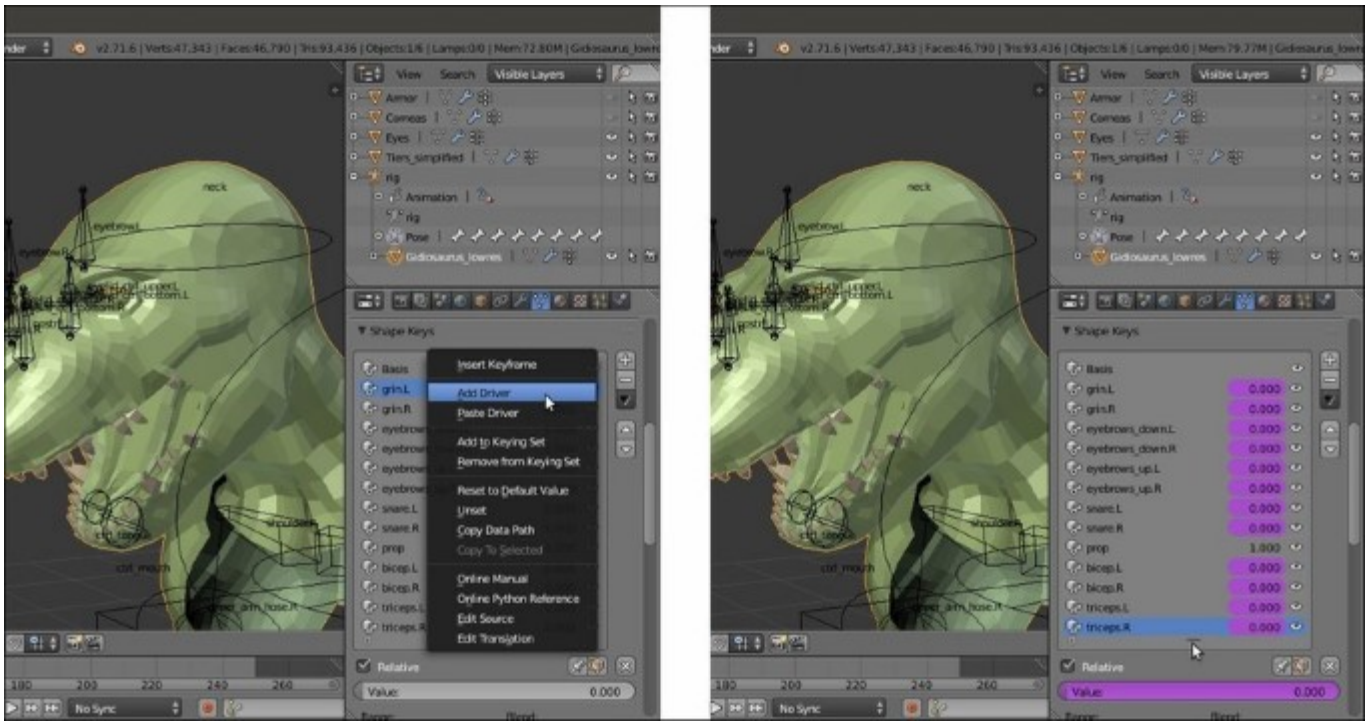
1. Go to the **Outliner** and hide the **Armor** object.
2. Select the **Armature** rig, switch to the **Octahedral** bones draw mode, and press *Tab* to enter **Edit Mode**; zoom to the character **head** and add **six** bones located as follows: **two** bones close to both the right and the left **eyebrows**, **two** bones close to both the sides of the **grin snout** area, and **two** bones close to the **nostrils**. Enable the **Names** item in the **Skeleton** subpanel under the **Object Data** window and rename the bones accordingly and with the correct **.L** or **.R** suffix, then be sure to have them located on the first bone layer by pressing the *M* key to call the **Change Bone Layers** pop-up.



The new bones for the shape key drivers

3. Exit **Edit Mode** and select the **Gidiosaurus** mesh.
4. Go to the **Shape Keys** subpanel under the **Object Data** window and expand the list window by left-clicking on the = icon at the bottom and dragging it downward.

5. Now right-click on the value (**0.000**) at the right side of the name of the first shape key (**grin.L**) and from the pop-up panel, select the **Add Driver** item; the value is enhanced, in violet, to show that now it has a **driver** associated.
6. Repeat the same for all the shape keys in the list except for the **prop** one, which has a fixed value of **1.000**:



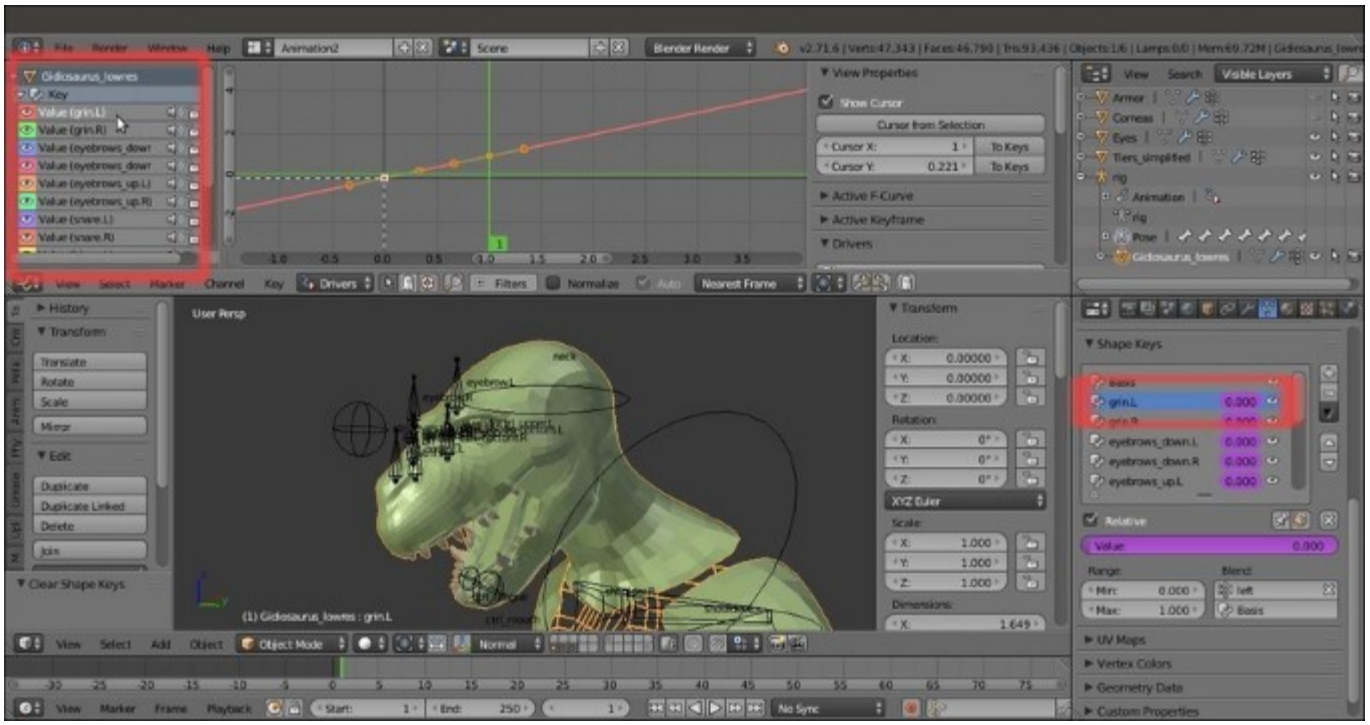
The shape keys list showing they have drivers

7. Split the 3D viewport horizontally into two parts, change the upper part into a **Graph Editor** window, or simply switch the screen to the **Animation** layer (in the **two** files provided with the cookbook, there are actually two prepared animation screens, **Animation1** and **Animation2**). Click on the *Editing context being displayed* button in the toolbar of the **Graph Editor** window and change it from **F-Curves** to **Drivers**.

How to do it…

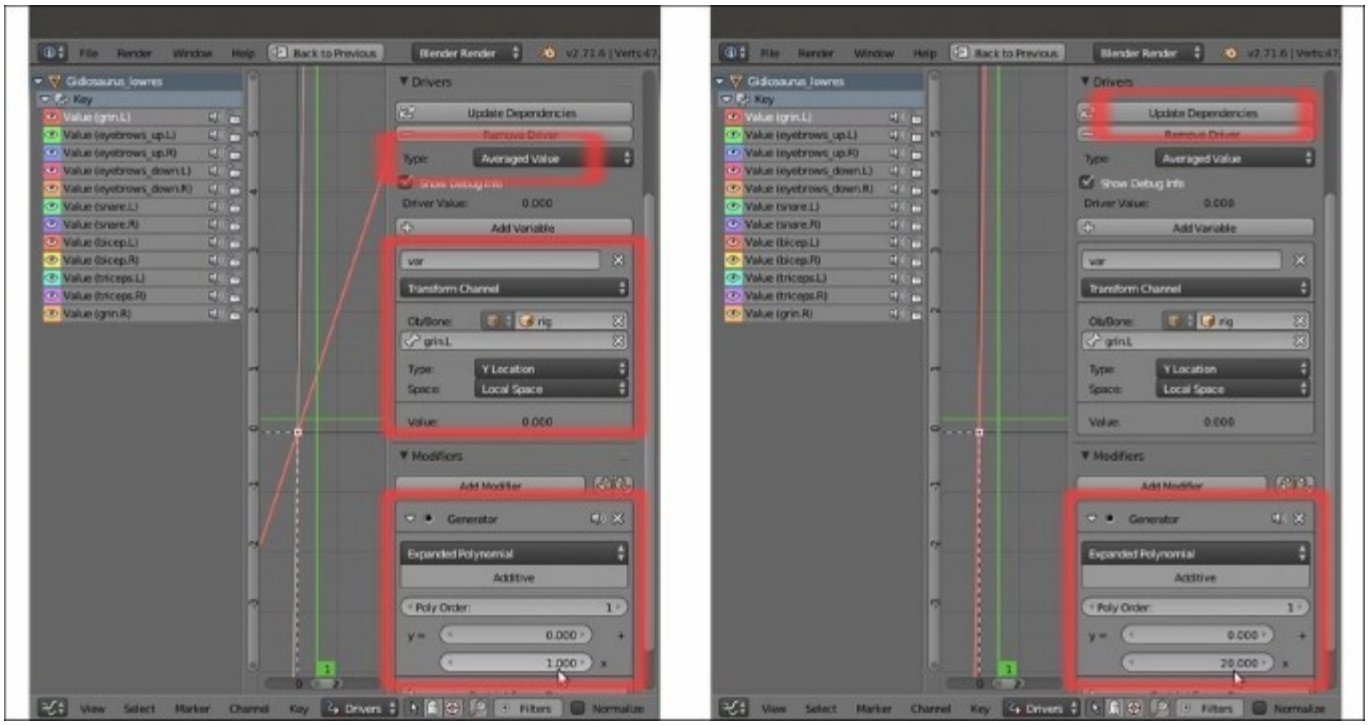
Let's start with the **expressions shape keys**:

1. If not already present, press the *N* key to open the **Properties** sidepanel of the **Graph Editor** window, then click on the **Value (grin.L)** top item in the drivers list at the top-left of the screen:



The Graph Editor window, at the top of the screen, with the driver f-curve

2. Go to the **Properties** panel of the **Graph Editor** and, by scrolling down, find the **Drivers** subpanel. In the *Driver* **Type** slot, switch from the **Scripted Expression** item to the **Averaged Value** one.
3. In the **Ob/Bone** slot, select **rig** and in the under slot (*Name of PoseBone to use as target*), select the **grin.L** bone.
4. Going downward, in the *Variable* **Type** slot, select the **Y Location** item and in **Space**, select **Local Space**.
5. Click on the **Update Dependencies** button at the top of the **Drivers** subpanel (the **Update Dependencies** function works particularly for **Scripted Expression**; it is quite important to use it to refresh the new setups each time).
6. Go even further down and click on the **Add Modifier** button in the **Modifier** subpanel; from the **Add F-Curve Modifier** pop-up menu, select the **Generator** item.
7. In the *Coefficient for polynomial* – **x** slot, change the value **1.000** to the value **20.000** (this is to re-map the declivity of the **f-curve** and therefore the speed of the corresponding shape key):

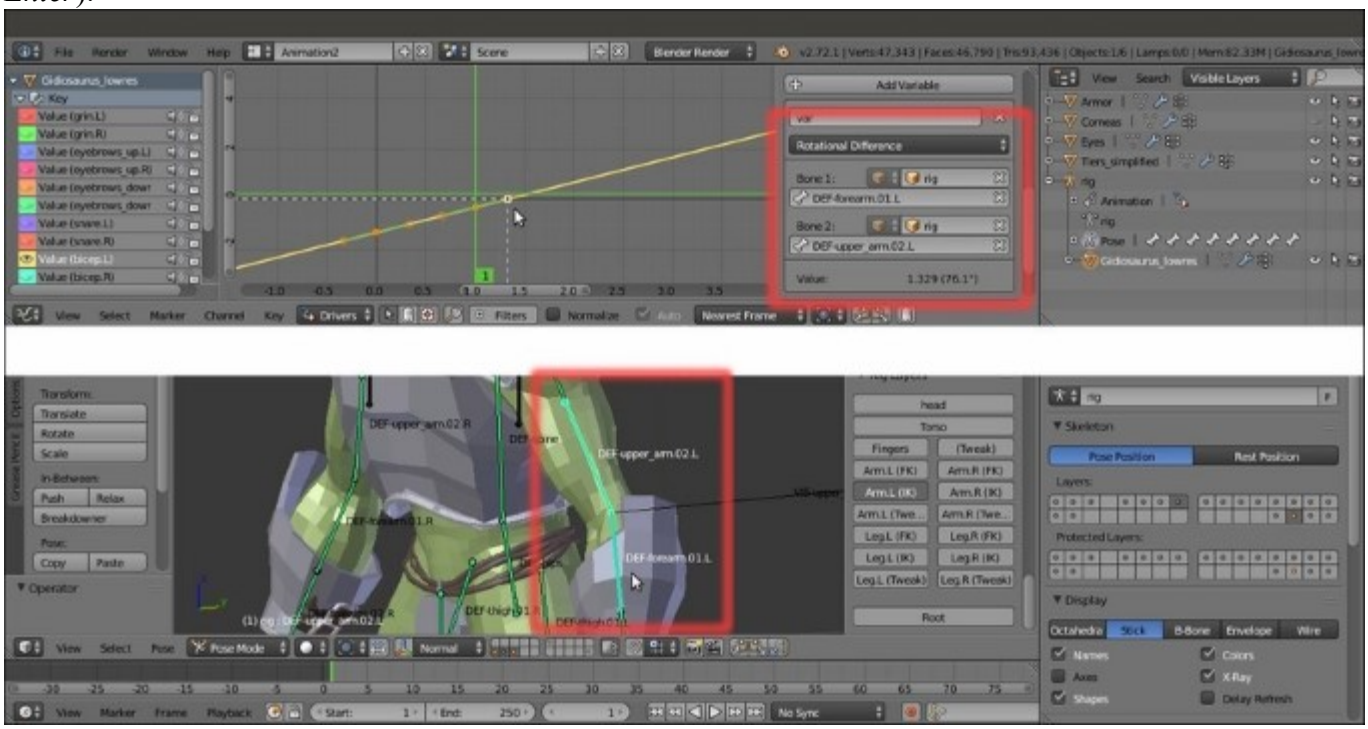


The N Properties Graph Editor sidepanel for the selected driver

8. Now select the **grin.L** bone and in **Pose Mode**, move it upward to see the **grin.L** shape keys being animated on the character's **snout**.
9. Go to the **Shape Keys** subpanel and right-click on the value to the right side of the **grin.L** shape; from the pop-up menu, select the **Copy Driver** item.
10. Select the **grin.R** shape key and right-click on the value to the right; from the pop-up menu, select the **Paste Driver** item.
11. Go to the **Animation** screen and switch the **grin.L** to the **grin.R** bone in the **Ob/Bone** field under the **Drivers** subpanel.
12. Copy and paste the drivers for the **eyebrows_up.L** and **eyebrows_up.R** shape keys, then replace the driver bones names in the **Ob/Bone** field under the **Drivers** subpanel.
13. Go to the **Shape Keys** subpanel under the **Object Data** window and set the **Max** value under the **Range** item to **0.600** for both the **eyebrows_up.L** and **eyebrows_up.R** shape keys; this is to limit the movement of the shape keys to avoid any intersection with the character's **helm**.
14. Copy and paste the drivers for the **eyebrows_down.L** and **eyebrows_down.R** shape keys. This time, leave the same driver bone names and instead change the value of the *Coefficient for polynomial* – **x** to negative and **-20.000** to *invert* the direction of the **f-curve**.
15. Repeat the procedure for the **snare.L** and **snare.R** shape keys, this time switching the *Variable* **Type** from **Y Location** to **X Location** and assigning a negative **-20.000** value to the **snare.L** driver and a positive **20.000** value to the **snare.R** one.

    At this point, all that is left is to assign automatic drivers for the *shape keys to stretch and grow muscles* we created for the character's **arms**.

16. Click on the **Value (bicep.L)** item in the drivers window at the left top of the **Graph Editor** and then go to the **Properties** panel on the right and then to the **Drivers** subpanel. In the *Driver* **Type** slot, select the **Averaged Value** item again; in the *Variable* **Type** slot, switch to **Rotational Difference**.
17. In the **Bone1** slot, select **rig** and in the slot below (*Name of PoseBone to use as target*), select the **DEF-forearm.01.L** bone; in the **Bone2** slot, select **rig** again, and then **DEF-upperarm.02.L**.
18. In the **Graph Editor**, click on a point of the **f-curve** to select it and then press the *L* key to select all the points of the **f-curve**; move them downward, on the *y* axis, by **-1.400** (*G* | *Y* | **-1.4** | *Enter*).



The triceps Rotational Difference driver

19. Copy and paste the driver to the **bicep.R** shape key, then change the **.L** suffixes of the bones to the **.R** ones.
20. Copy the **bicep.L** driver and paste it to the **triceps.L** shape key; click on the **Add Modifier** button under the **Modifier** subpanel; and from the **Add F-Curve Modifier** pop-up menu, select the **Generator** item.
21. In the *Coefficient for polynomial* – **y** slot, write the value **2.300** and in the **x** slot, write the value **-1.000** (remember that all these values in the recipe are not universal and are valid just for this **Gidiosaurus** model in this particular setup; the drivers values could change from character to character, so always test them on your model).
22. Copy and paste to the **triceps.R** shape key, and change the suffixes of the bones.
23. Click on the **Update Dependencies** button at the top of the **Drivers** subpanel and save the file.

How it works…

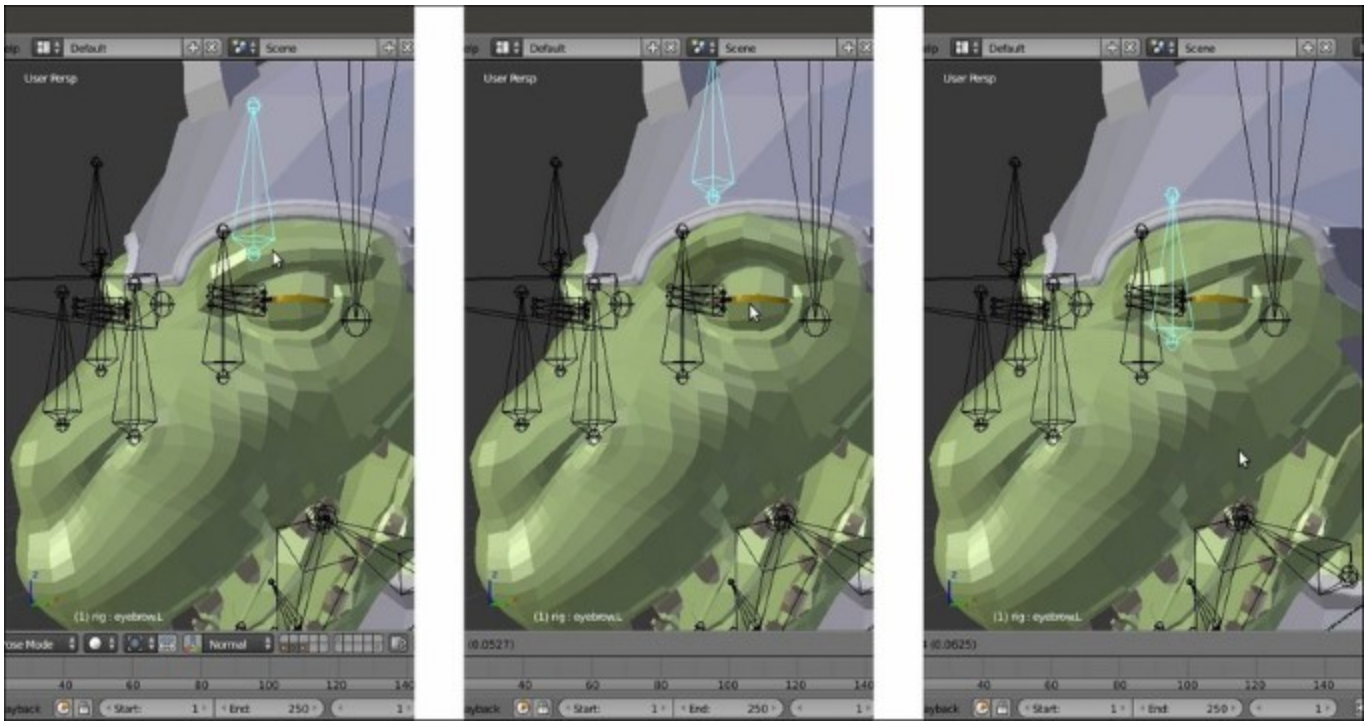
Drivers assigned to bones as controllers for the shape keys are not only an effective way to create a device for animation but also a mandatory technique in the Blender pipeline workflow, where a character is usually linked into the scene from a different file and the rig gets **proxified** (we'll see how to do this in the next chapter). The only possible way to have access to the shape keys in a linked character is through the **drivers** and the **rig**.

As you probably already know, shape keys are often used not only for **facial expressions** but also to mimic the stretching and the growing of the body's **muscles** according to the movement of a character's **limbs**. In this case, their influence is automatically driven by the rotation of the respective bones through the **Rotational Difference** drivers that, as the name itself says, base their influence on the difference of rotation between two bones; more precisely, on the **angle** between them.

The **Generator** modifier we added is a multiplier we used to *virtually* modify the slope inclination of the **f-curves** of the drivers. The default inclination of the **f-curve** wasn't enough to fully map the curve itself to a driver bone movement of (almost) just one or two Blender Units (it was too slow, resulting in a required driver movement of several units to have an appreciable effect), so we increased the declivity by a factor of **20.000** to have a faster correspondence.

However, the same modifier was also used to reverse the direction of the **f-curve**, by using a negative value of **-20.000**, for example to drive the downward movement of the **eyebrows**, or to change the location of the curve along the *y* axis so as to tweak the timing of the driver influence, like in the **triceps** shape keys.

Therefore, by copying and pasting a driver and giving an opposite declivity at the slope of the copied one, it is possible to drive two opposite shape keys through the same bone, as for the **eyebrows** shape keys:

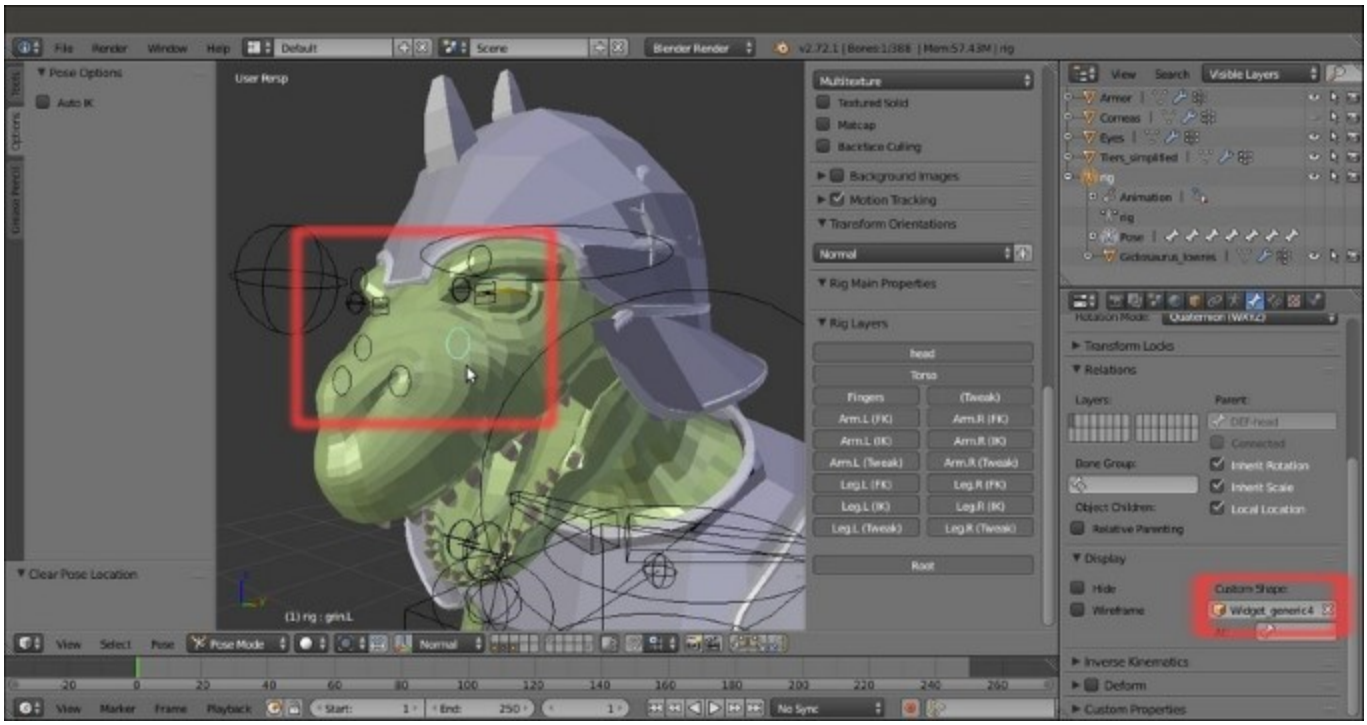


The same bone moving in two opposite directions to drive two opposite shape keys

There's more…

To add shape keys and the respective drivers to the **Gidiosaurus** model, we used the `Gidiosaurus_skinning_rigify.blend` file, with the **rig** created by the **Rigify** addon. The control bones of a **Rigify** rig have pre-made **Custom Shapes** to make their identification and selection easier and are usually located in the last scene layer.

So, for the last step, I just modeled a new simple custom shape, a small **Circle** mesh with **16** vertices. I named it **Widget_generic4** and I assigned it to all the *driver* bones:



The driver bones with the new Custom Shape

See also

- http://www.blender.org/manual/animation/basics/drivers.html

Setting movement limit constraints

Often, it is very useful to put movement limitations on the bones of a rig, for several reasons—usually, to make them easier to work with, but also to establish a maximum range for the rotation of the **limbs** or other parts like in the **mandible** or the **eyelids**.

Two types of limits for the bones are: by the **Transform** locks, and by **bone constraints**.

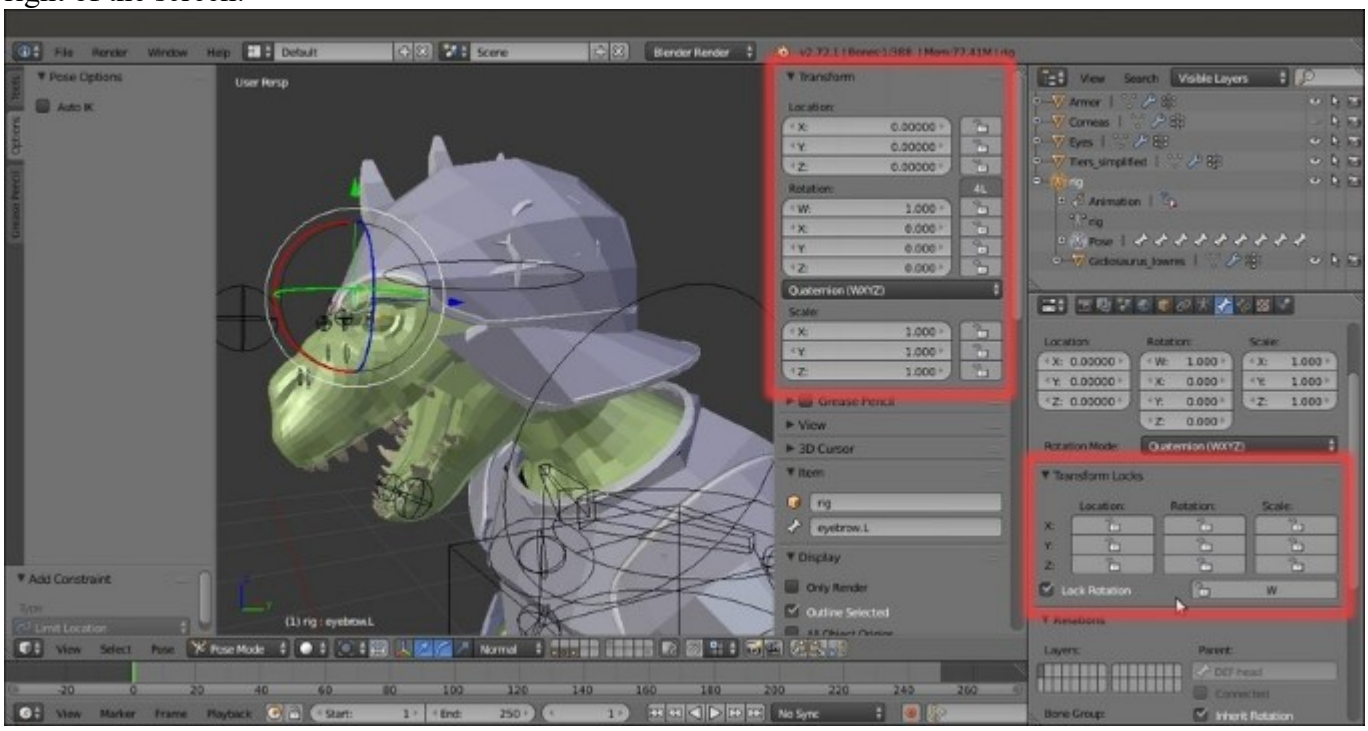
Getting ready

Load the `Gidiosaurus_shapekeys.blend` file, select the **Armature**, and go in **Pose Mode**.

How to do it…

Let's start with the **Transform** locks:

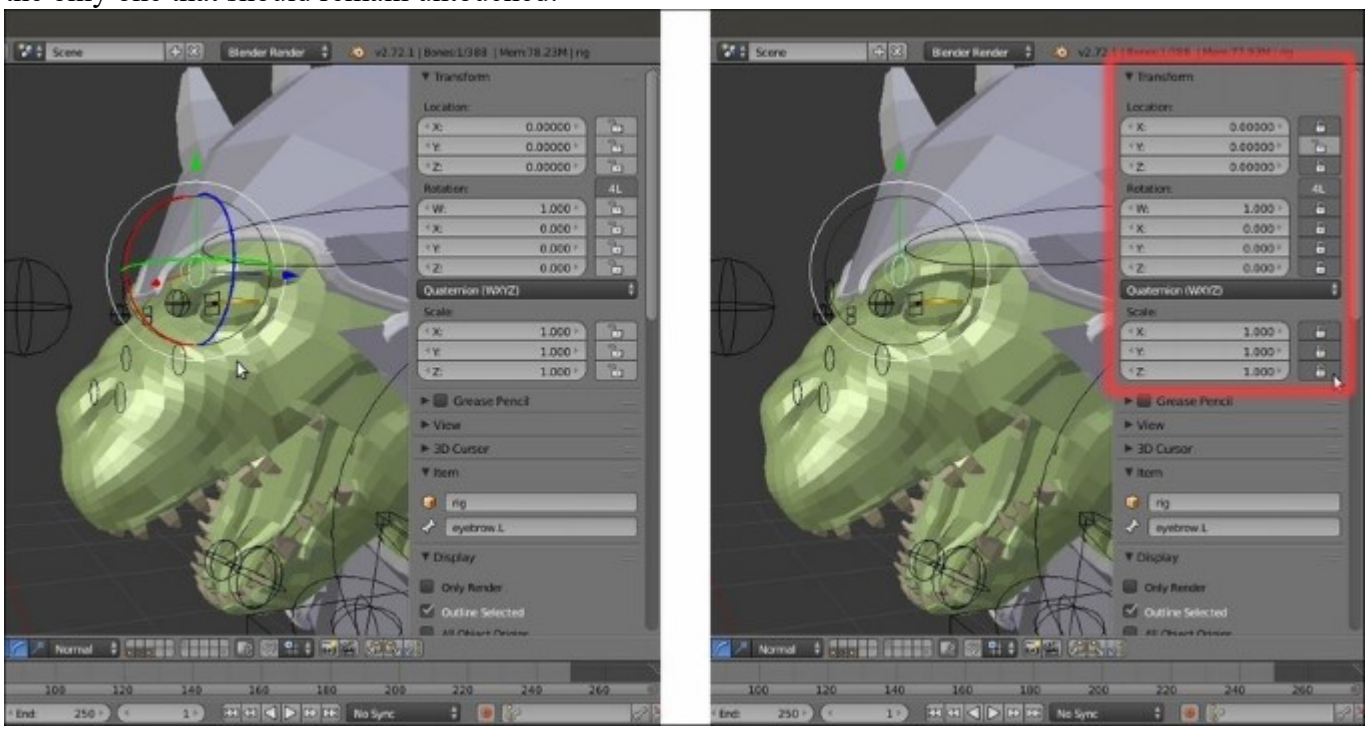
1. Select the **eyebrow.L** bone and if not already present, press the *N* key to call the 3D viewport **Properties** sidepanel. Go to the **Transform** subpanel, which is the first entry at the top, or also to the **Transform Locks** subpanel under the **Bone** window in the main **Properties** panel to the right of the screen:



The Transform subpanel in the N Properties sidepanel and the corresponding Transform Locks subpanel under the main Properties panel

2. Click on the lock icon to the right side of the properties; for this bone (which, if you recall, is the driver control object for the left up and down **eyebrow** shape keys), we want to lock all the

possible transformations *except* for the movement on its *y* axis, so the **Location Y** lock button is the only one that should remain untouched:



Setting the axis Transform locks for the Location, Rotation, and Scale

If you now try to move the **eyebrow.L** bone, you will notice its movement is constrained only to its local *y* axis; the movement is directed by the **Roll** orientation of the bone in **Edit Mode** (and not by the **Normal** item enabled in the **Transform Orientation** button on the viewport toolbar); enable the **Axes** item in the **Display** subpanel under the **Object Data** window to see this.

Having locked the other two axes, it's no longer necessary to use the widget arrow to move the bone on its local *y* axis but it's enough to simply press the *G* key and then move the mouse instead.

And now, let's see limits by **constraints**.

3. Go to the **Bone Constraints** window and assign a **Limit Location** constraint to the **eyebrow.L** bone.
4. Check the **Minimum X** and **Maximum X**, **Minimum Y**, and **Maximum Y**, and **Minimum Z** and **Maximum Z** items. Leave the values for the *x* and *z* axes as they are, change **Minimum Y** to negative **-0.050**, and change **Maximum Y** to positive **0.050** (again, remember that these values are valid just for this file).
5. In the **Convert** slot, change the **Owner Space** item to **Local Space**:

The assigned Limit Location constraint subpanel under the main Properties panel

In Chapter 1, *Modeling the Character's Base Mesh*, we enabled the **Copy Attributes Menu** add-on in **User Preferences** and then we saved the **User Settings**, so I'm taking for granted that you have the script still enabled.

Therefore, we do the following:

6. Select the **eyebrow.R** bone and then *Shift*-select the **eyebrow.L** bone. Press *Ctrl* + *C* and from the **Copy Attributes** pop-up menu, select the **Copy Bone Constraints** item.
7. Select the **grin.L** and **grin.R** bones and then *Shift*-select the **eyebrow.L** bone. Once again, press *Ctrl* + *C* | **Copy Bone Constraints**, and in the two copied constraints, set the **Minimum Y** value to **0.000**.
8. Select the **nostril.L** and **.R** bones and *Shift*-select the **grin.L** bone, then press *Ctrl* + *C* | **Copy Bone Constraints**. This time, set both the **Y** values to **0.000** and **Minimum X** for the **nostril.L** bone to negative **-0.050** and the **Maximum X** for the **nostril.R** bone to positive **0.050**.
9. Save the file.

Several other movement constraints have been added to different bones in the rig, for example the **jaw** bone, or the **eyelid** controllers, but especially to the **eye** bones, to limit the range of possible rotations. To have a look at the various settings, just open the `Gidiosaurus_limits.blend` file.

See also

- http://www.blender.org/manual/animation/techs/object/constraint.html

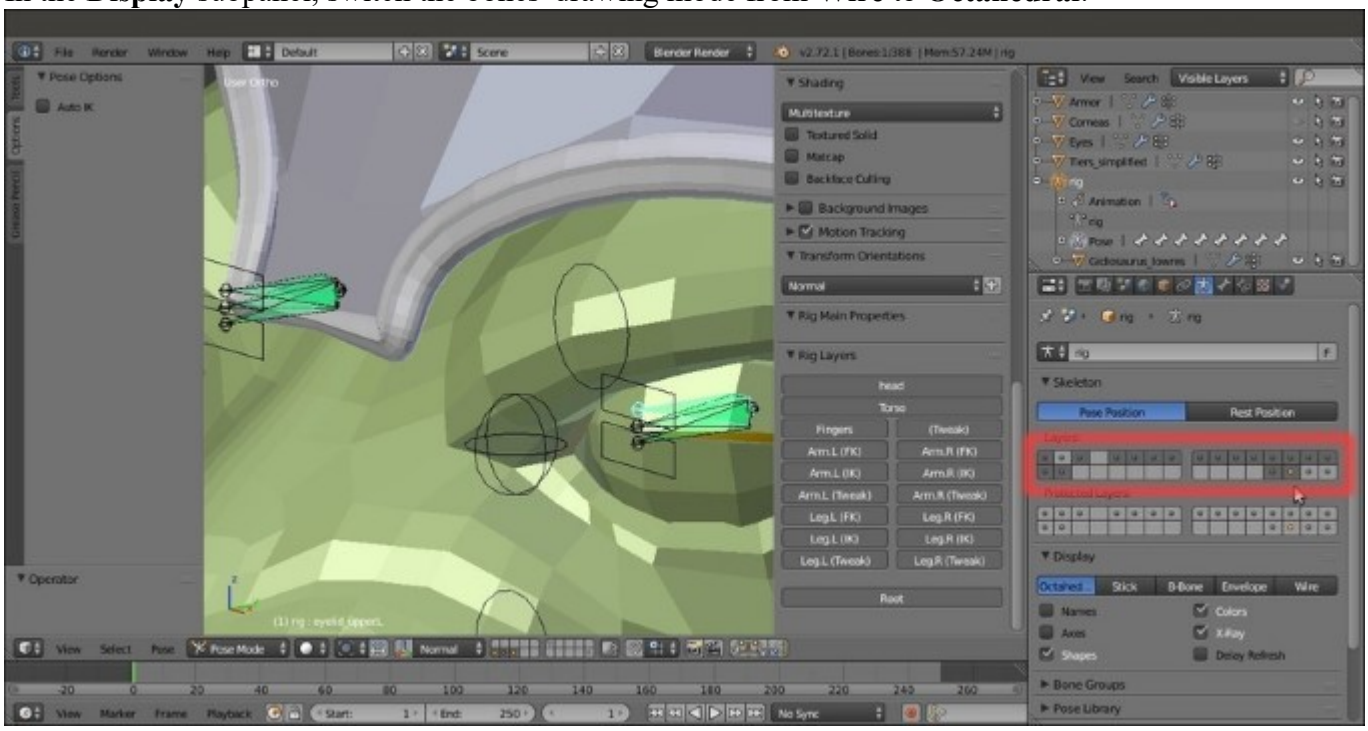
Transferring the eyeball rotation to the eyelids

This is a really simple trick that can add a lot of life to the facial expressions of an animated model, making the **eyelids** follow some of the movement of the **eyeballs**.

Getting ready

Following on from the previous recipes, open the `Gidiosaurus_limits.blend` file:

1. If not already selected, select the **Armature** and enter **Pose Mode**.
2. In the **Object Data** window, go to the **Skeleton** subpanel and enable the **30th** bone layer, to show the deforming bones.
3. In the **Display** subpanel, switch the bones' drawing mode from **Wire** to **Octahedral**:



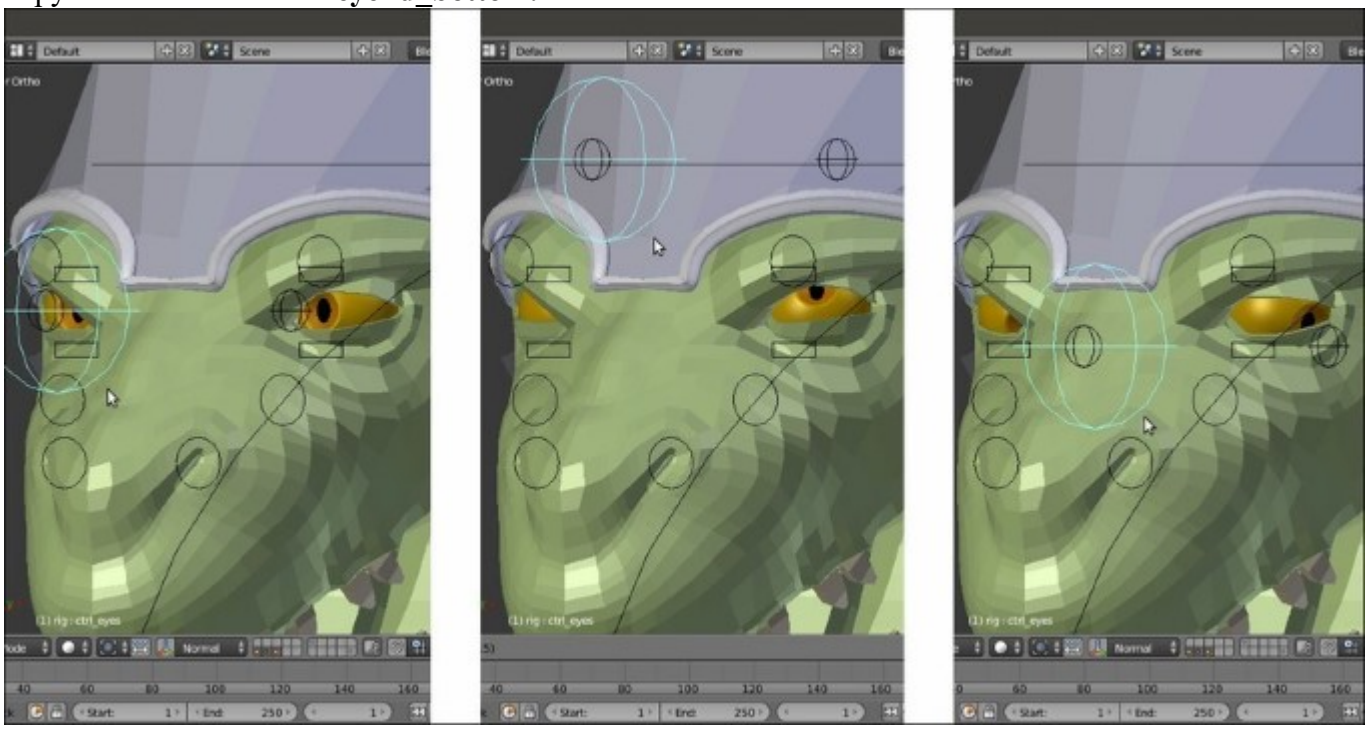
The Skeleton subpanel with the bone layers

How to do it…

Now zoom to the character's **head** and continue with the following steps:

1. Select the **eyelid_upper.L** bone and go to the **Bone Constraints** window; assign a **Copy Rotation** constraint.
2. In the **Target** field, select the **rig** item, and in the **Bone** field, select the **eye.L** bone item. Set **Space** = **Pose Space** to **Pose Space**.
3. Set the **Influence** slider value to **0.300**.

4. Select the **eyelid_bottom.L** bone and *Shift*-select the **eyelid_upper.L** bone, then press *Ctrl* + *C* | **Copy Bone Constraint**.
5. Select the **eyelid_upper.R** bone and repeat the procedure but with **eye.R** as the target bone; copy the constraint to the **eyelid_bottom.R** bone:



The eyelids slightly following the eye movements

6. Save the file.

Detailing the Armor by using the Curve from Mesh tool

In Chapter 3, *Polygonal Modeling of the Character's Accessories*, in the *Using the Mesh to Curve technique to add details* recipe, you already saw how to use this technique as a modeling tool. In this recipe, we'll use the same technique but in the opposite direction—to add **rivets** around the perimeter of the borders of the different **Armor** parts.

Getting ready

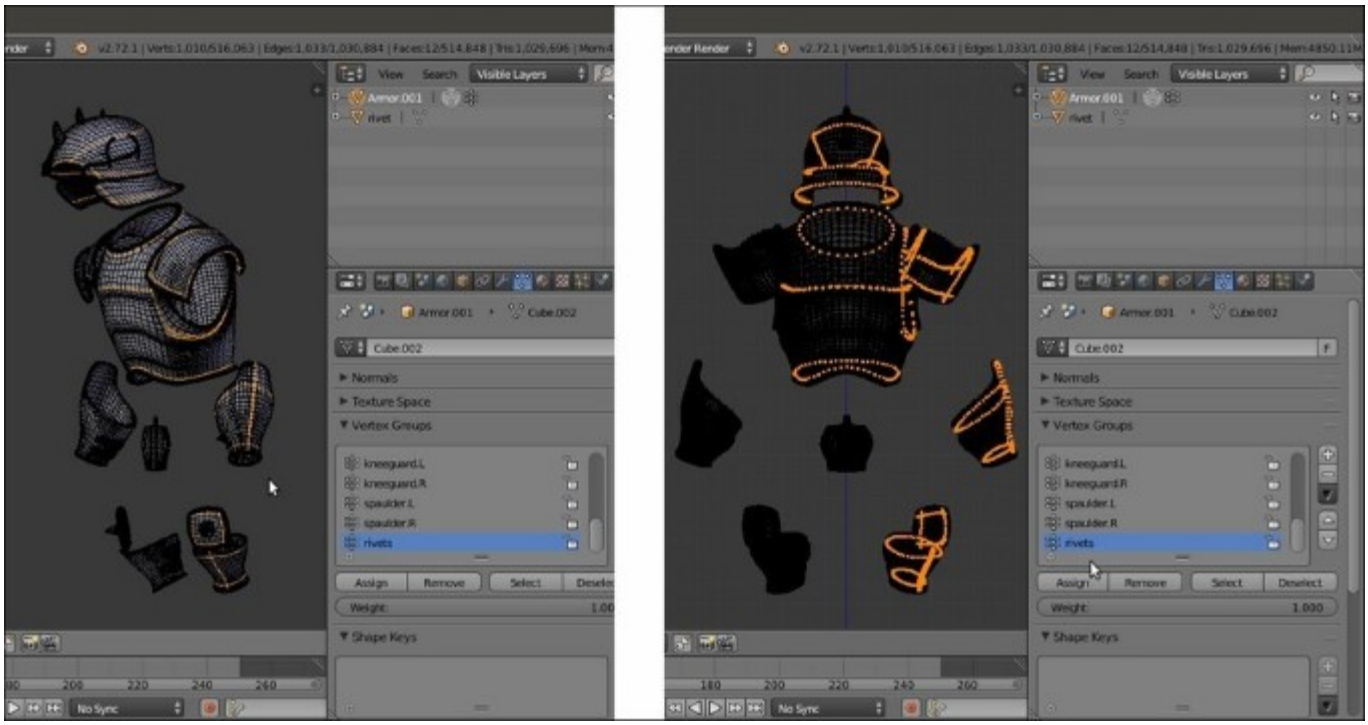
Re-open the `Gidiosaurus_limits.blend` file; the first thing to do is to model a very lowpoly **rivet** object to be duplicated on the **Armor** surface:

1. Switch to an empty scene layer, press *Shift* + *C* to place the **3D Cursor** at the center of the grid, and add a **Cube** primitive mesh. Enter **Edit Mode** and delete the bottom face, then scale the remaining faces by a value of **0.100** twice, then one last time by **0.500**. Move the top face downward to flatten the overall shape a bit and scale the same face by **0.700**.
2. Press *A* to select all the vertices and *W* to choose the **Subdivide Smooth** item from the **Specials** pop-up menu, then delete the middle horizontal edgeloop.
3. Put the pivot on the **3D Cursor** and while still in **Edit Mode**, rotate all the vertices by **90°** on the *x* axis.
4. Select the bottom edgeloop and press *Shift* + *S* | **Cursor to Selected**. Exit **Edit Mode** and click on the **Set Origin** button under the **Tool** tab to select the **Origin to 3D Cursor** item.
5. Click on the **Smooth** button under the **Shading** item and in the **Outliner**, rename the **rivet** object. Once again, place the **3D Cursor** at the center of the grid and the **rivet** at the **Cursor** location; press *Ctrl* + *A* to apply the **Rotation & Scale** option.
6. Enable the scene layer with the **Armor** on it, and in the **Outliner**, hide the **rig**.

How to do it…

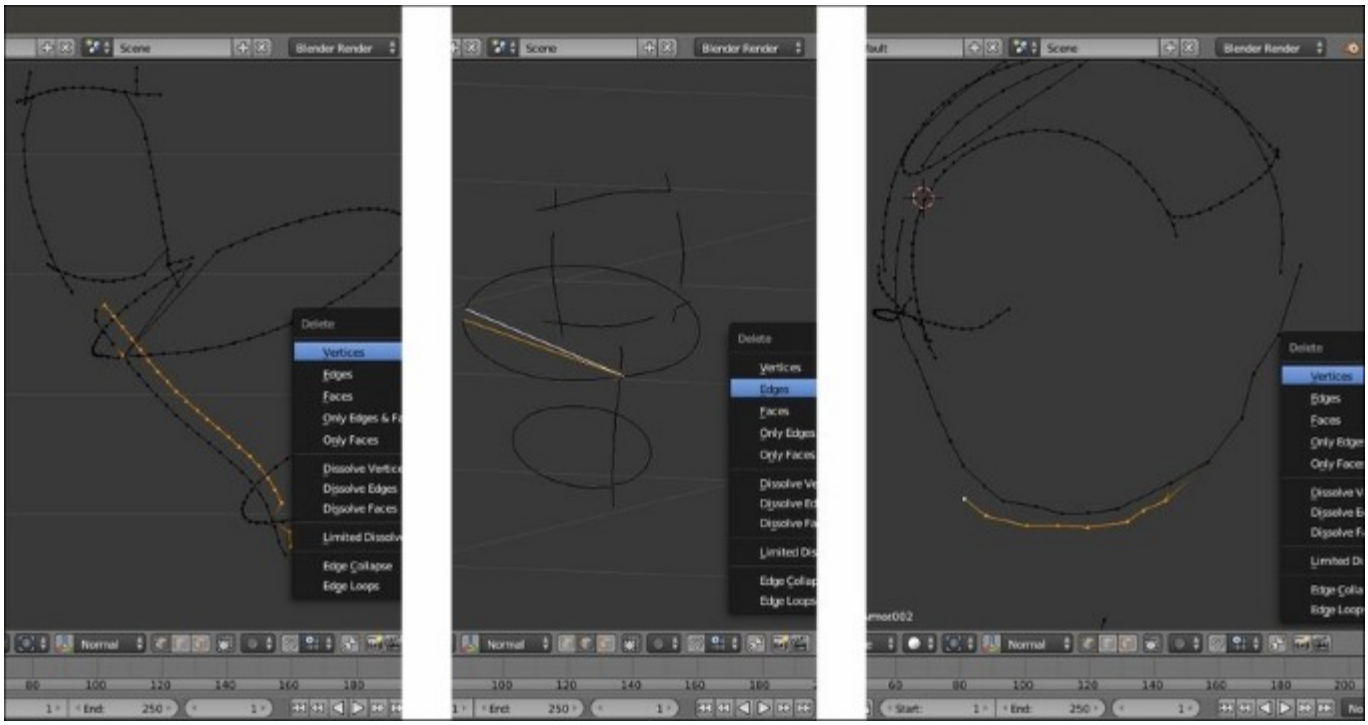
Now, let's create the *guides* to duplicate the rivets on:

1. Select the **Armor** object and press *Shift* + *D* to duplicate it, then place the duplicate **Armor.001** object on the scene layer of the **rivet**. Go to the **Shape Keys** sidepanel under the **Object Data** window and delete the **Armor_fix** first and then the **Basis** shape keys.
2. Go to the **Object Modifiers** window, remove the **Armature** modifier, and apply the **Subdivision Surface** modifier with a **Subdivision** level of **2**.
3. Enter **Edit Mode** and start to select the edgeloops on the different **Armor** parts in areas where you want to add the **rivet** rows (*Alt* + right-click for the first one, then *Alt* + *Shift* + right-click). As usual, it's enough to work only on one half of the mesh:



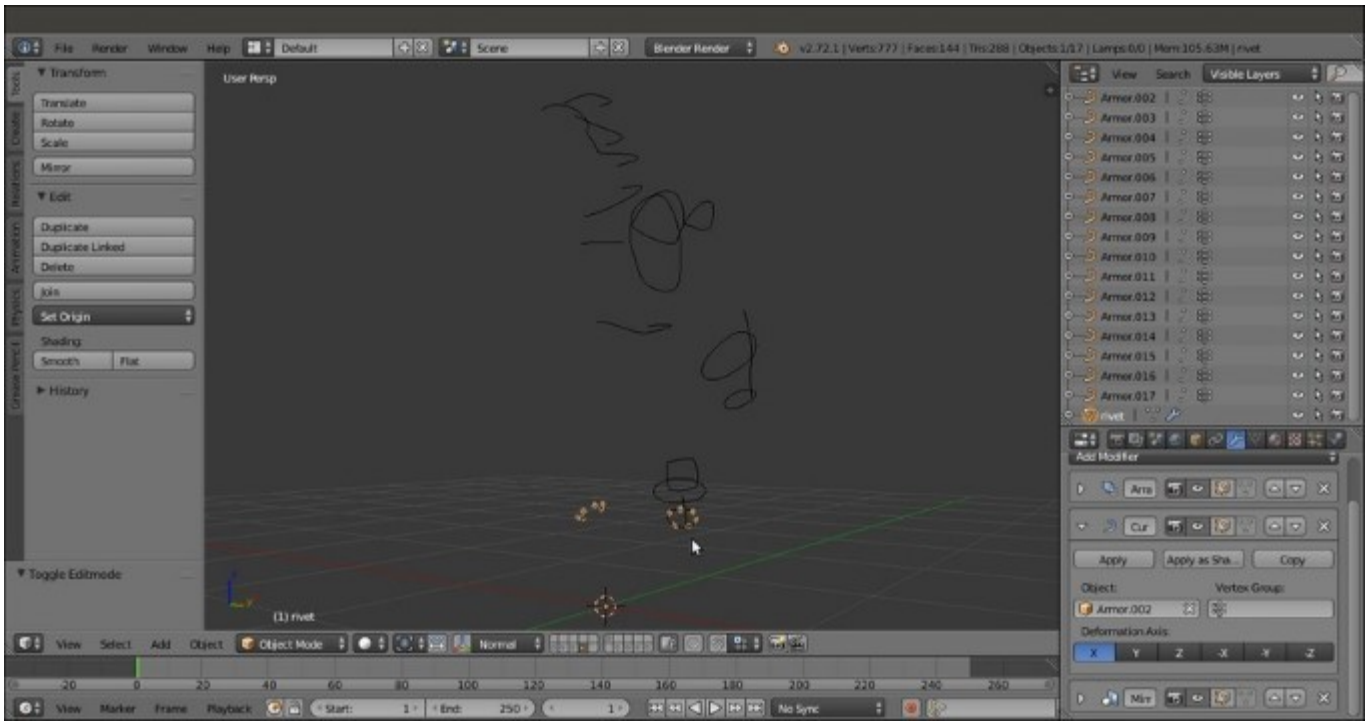
The Armor mesh in Edit Mode with the selected edge-loops

4. Press *Shift* + *D* and soon after, click the right mouse button to duplicate the selected edgeloops without moving them, then press the *P* key to separate them from the **Armor.001** object (in the **Separate** pop-up menu, choose the **Selection** item).
5. Exit **Edit Mode** and delete the **Armor.001** object, or if you don't have problems with big file sizes, move it to a different scene layer to keep it for future refinements. In this case, you can save the edge-loops selection as a vertex group named **rivets**.
6. Select the **Armor.002** object (the duplicated and separated edgeloops) and enter **Edit Mode**; make the necessary adjustments to the edgeloops by deleting the unnecessary vertices, for example the backsides of the plates, and disconnect the welded edgeloops by deleting the common vertices or connecting them where required edges are missing:



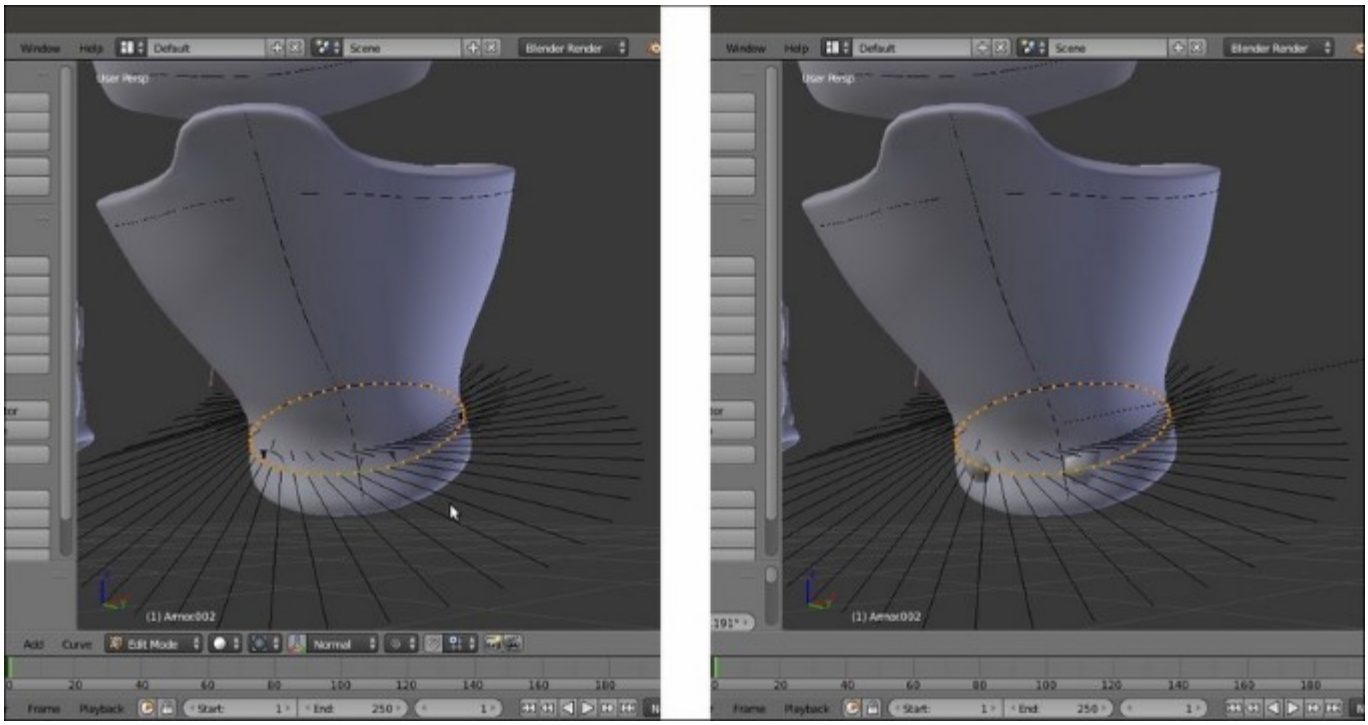
Cleaning the edge-loops of the duplicated Armor.002 mesh

7. Press *A* to select all the vertices and then go to the **Tools** tab under the **Tool Shelf**. Go to the **LoopTools** subpanel and press the **Space** button to evenly space the vertices along the edgeloops.
8. Exit **Edit Mode** and press *Alt* + *C*; in the **Convert to** pop-up menu, select the first item, **Curve from Mesh/Text**. The mesh edgeloops actually get converted into a **Curve** object, as you can see in the **Object Data** window under the main **Properties** panel to the right of the UI. Click on the **Fill** slot to select the **Full** item.
9. Now the tedious part (but not difficult, just a little tedious); in **Edit Mode** again, put the mouse on one of the points and by pressing the *L* key, select each separate part of the **Curve**, then press *P* to separate the whole selected part. This way, you are going to obtain **16** separated **Curve** objects.
10. Select the **rivet** object and go to the **Object Modifiers** window; assign an **Array** modifier with **Fit Type** = **Fit Length**, **Length** = **0.50**, and **Relative Offset X** = **3.000**. Collapse the panel.
11. Assign a **Curve** modifier, then in the **Object** field select the **Armor.002** curve. Leave the panel expanded.
12. Assign a **Mirror** modifier and collapse the panel:



The rivet object instanced on the mirrored curve object

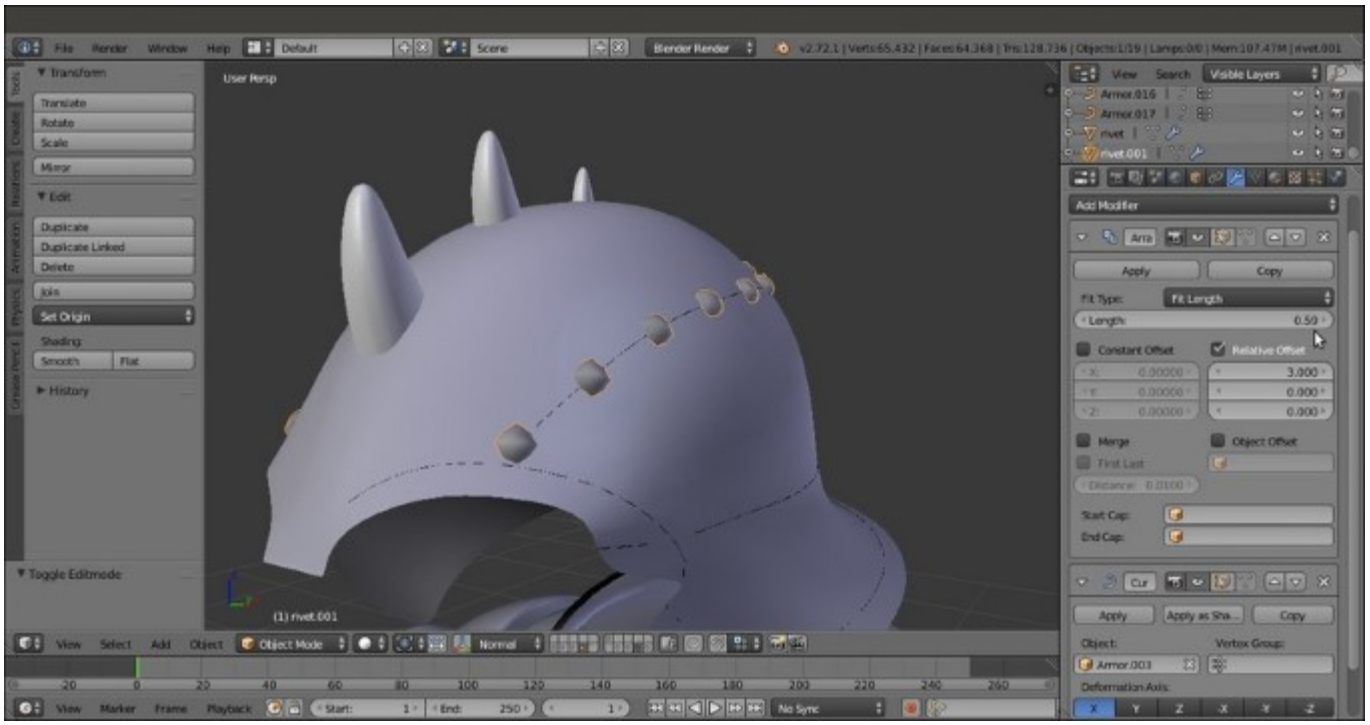
13. In the viewport area, zoom to each curve to check for the correct tilting of the points; if necessary, select the curve, enter **Edit Mode**, select all the points, press *Ctrl* + *T*, and move the mouse to rotate the tilting of the curve's points until the instanced **rivets** are correctly rotated/ aligned with the surface of the main **Armor** mesh:



Tilting the curve's points

If necessary, you can also select individual points of the curve to tweak the orientation of only a part of the instanced rivets, even of single rivets at once; this has been done for part of the **helm** and for the **spaulders**, especially.

14. In the **Outliner**, re-select the **rivet** and press *Shift* + *D* to duplicate it, then in the **Object Modifiers** window, under the **Curve** modifier panel, select the **Armor.003** item in the **Object** field.
15. Once again, zoom to the curve and if necessary, fix the curve tilting and also adjust the **Length** value of the **Array** modifier (for the **Armor.003** curve it has been raised to **0.59**) and the **Relative Offset** value. By selecting all the points and pressing *W*, you can also select the **Switch Direction** item in the **Specials** menu.



The rivets on the helm object

16. Duplicate the rivet and repeat the procedure changing the curve name in the modifier for each curve object and so on. At the end, you should have **16** copies of the rivet as well.

    At this point, if required, we can still make some modification to the rivet mesh; in my case, I just subdivided it a bit more, then deleted some useless edgeloop, made it rounder, and extruded the open side a bit more.

    Now that the rivet is ready, select all the rivet copies (so that the *modified one* is the active object, that is *the last selected*) and press *Ctrl + L* | **Object Data** to share the modifications between them.

    Leaving everything selected, press *U* | **Object & Data** to make them single users again (this is necessary for the next step with the modifiers).
17. When you are done, select all the rivets *one at a time* in the **Outliner** and apply all the **Array** and the **Curve** modifiers.
18. Join all the rivets into a single object (select all and press *Ctrl + J*) and in **Edit Mode**, delete the unnecessary or overlapping ones, keeping only the rivets that really add to the **Armor** look. Then, apply all the **Mirror** modifiers:

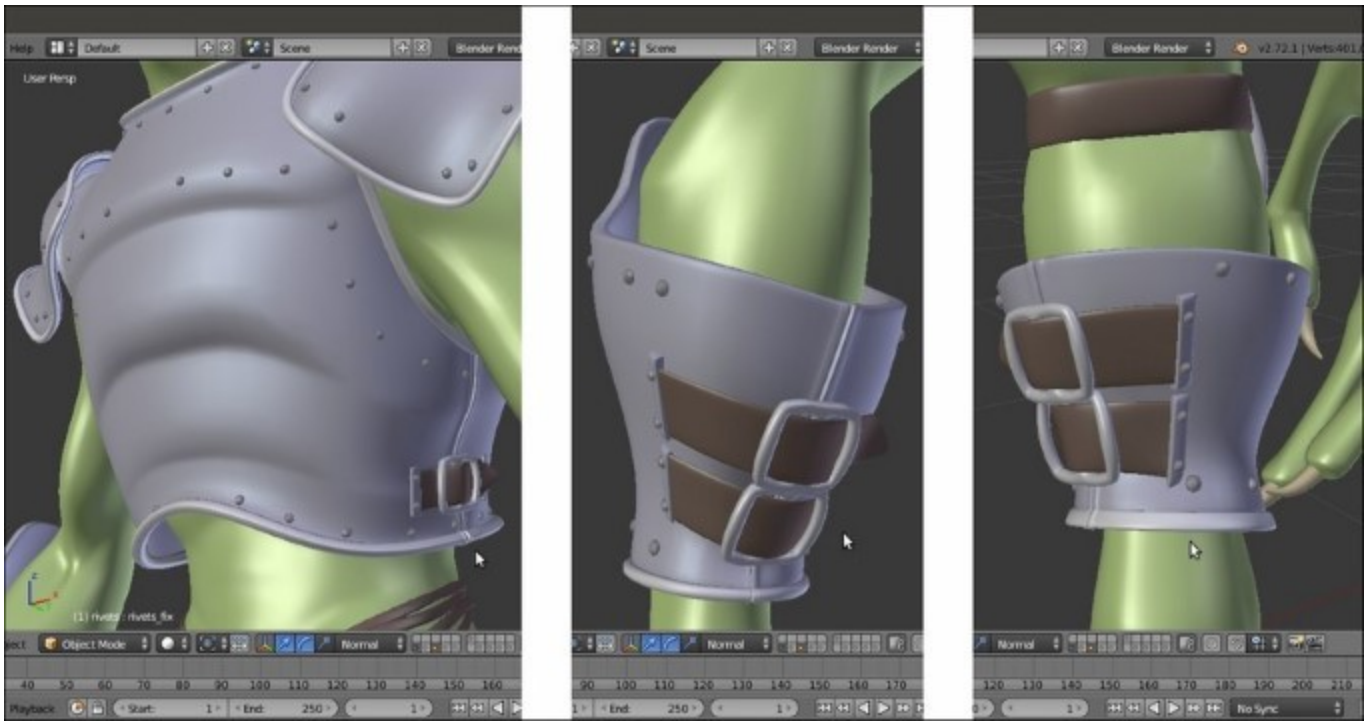
The completed rivets

19. Select the **Armor** object and then *Shift*-select the rivets object, press *Ctrl* + *Tab* to go in **Weight Paint** mode and click on the **Transfer Weights** button under the **Tools** tab.
20. Exit **Weight Paint** mode and assign an **Armature** modifier to the **rivets** object, select **rig** in the **Object** field.
21. Save the file as `Gidiosaurus_final_detailing.blend`.

There's more…

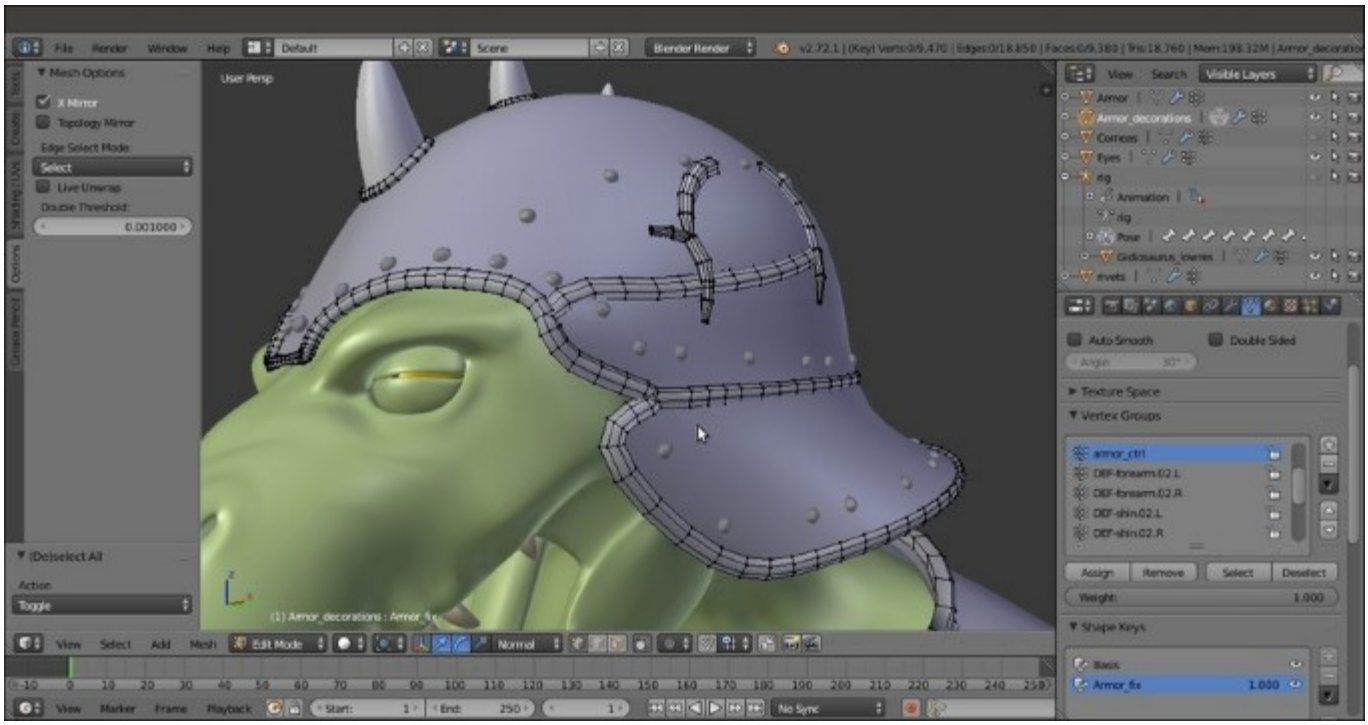
At this point, the **Gidiosaurus** model is ready to be animated, but some minor adjustments are still missing and can be added.

I won't go into the details about these additions, they are all processes you have already seen in the previous chapters and recipes, so this is simply a showcase:



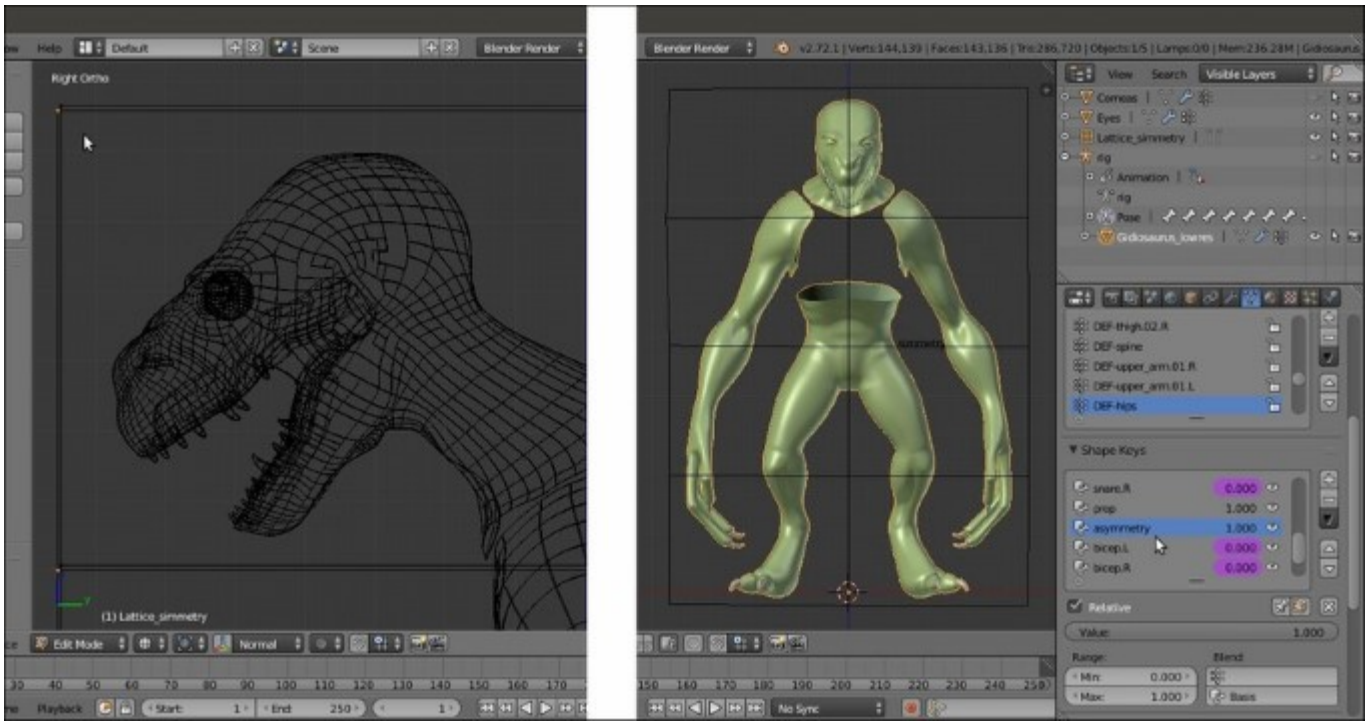
The modeled tiers and the rivets

1. The tier attachments on the **Armor's vambraces** and on the **greaves** have been refined by adding smaller **rivets**, and new tiers have been added to the sides of the **Armor chest plate**. Also, the opening seams in the **Armor** parts have been modeled under each tier location.
2. The **Armor** decorations have been separated as a new object (the **Armor_decorations** item in the **Outliner**) and simplified by deleting as many edgeloops as possible without altering their basic shape:



The simplified decorations in Edit Mode

3. The **Armor's** shape also has been tweaked even further through the **Armor_fix** shape key to adjust some overlaps that were occurring during the movements of the **spaulders** and in the **stomach** area too. The same shape key has been repeated also on the **decorations** and on the **rivets** objects for the areas of interest.
4. A bit of asymmetry has been introduced in the **Gidiosaurus** mesh by assigning a **Lattice** modifier to the character, and slightly modifying the shape on the left side, then applying the modifier as a shape key (the **Apply as Shape Key** button):



The asymmetry lattice

5. Finally, after some test renders, I realized that the **teeth** and the inside of the **mouth** of the **Gidiosaurus** still needed refinements, so I made some more adjustments to the **prop** shape key by making the **teeth** bigger and bolder, and the **inner mouth** more organic-looking and smooth:

The modified teeth and inner mouth

Be aware that almost in every modeled object there is still room for improvement, and that's okay, it's not a sign of a bad job! This sort of improvement is done all the time and is simply part of the working experience.

See also

- http://www.blender.org/manual/modeling/curves/index.html

Chapter 9. Animating the Character

In this chapter, we will cover the following recipes:

- Linking the character and making a proxy
- Creating a simple walk cycle for the character by assigning keys to the bones
- Tweaking the actions in Graph Editor
- Using the Non Linear Action Editor to mix different actions

Introduction

There are literally a *plethora* of tutorials and manuals about animation principles in general, and in Blender in particular, on the Web and in bookstores, so this one is going to be just a very *easy* chapter, mainly about the **technical aspects** of creating a simple animation with the rigged **Gidiosaurus** character, following the most usual pipeline commonly used in Blender (at least for the **open movies**).