POSER® Custom Model Maker's Guide

by
Dave Hill

Building Bat-Guy*

*resemblance to any 60-year-old nocturnal caped crusading copyrighted crime-buster is purely coincidental.Honest.

1. Three Steps

Completing any custom model for use by Poser involves three basic steps. The first step is creating or modifying our model's geometry with a modelling program. We've created our base model in the previous section, but we're still some ways off from a finished superhero.

Once the model's done, the second step is modifying the resource file for the model. We'll want to remove references or function calls for any parts removed from the geometry file. (If, for example, we wanted to create a one-armed man, we'd have to remove all references to the missing arm.) We'll also want to remove references for any non-moving parts. And we'll want to create references and function calls to any new parts or props added to the file (as we'll be seeing in our treatment of Bat-Guy's belt). We'll want to adjust any parameters for parts which may have changed proportion or position, so that the kinematic links behave properly. And we'll need to redefine the colors and textures used by the model. This second step will involve return trips to both the modelling program and Poser as we tweak both the model and the resource file in order to get things right.

The third step is the relatively simple step of adding the new figure to the Poser Figure library, based on the finished geometry, but reproportioned for a more heroic physique. A brand new resource file will be created by Poser, but all of its geometry references will come from our finished model. This is what actually what happens when you save a figure to the Poser library; a new resource file is created, whose parameters are altered according to the changes in the figure, but the base geometry remains the same.

2. batGuy.obj

So let's get ready to finish up on our model. First we need to make copies of our My Hero project files and rename them for the Bat-Guy project.

Create a new folder in the Geometries folder and call it `BatGuy`. This is where we'll be storing our new geometry file. Make copies of the Ray Dream native geometry file `myHero.rds` and OBJ format file `myHero.obj` and name them `batGuy.rds` and `batGuy.obj` respectively. Move these into the `BatGuy` folder. Make a copy of the file `My Hero.cr2` and name it `BatGuy.cr2`.

Open the `BatGuy.cr2` in a text editor. Replace all occurrences (there are two) of

```
figureResFile :Runtime:Geometries:myHero:myHero.obj
```

with

```
figureResFile :Runtime:Geometries:BatGuy:batGuy.obj
```

As in the previous section, *when I refer to opening* `batGuy.obj`*, I mean opening*

the native file batGuy.rds. *When I refer to saving* batGuy.obj, *I mean saving the native file* batGuy.rds *and exporting the OBJ copy* batGuy.obj. (Just trying to save myself some typing ...)

3. Time to Get Serious

Bat-Guy's not normally known as a joker, so we're going to use Poser's facial expressions controls for the head to put a more characteristic disapproving frown on his face. Then we'll export the model as a Wavefront OBJ file.

Open Poser and load the **Nude Male** from the **Figure** library. Load the expression **Upset** from the **Faces** library. Refer to **Figure 1**. Select the head if it is not currently selected and open the **Parameter Dials** window. Exaggerate the knit of the brows by entering the parameter values shown in **Figure 2**. We want to exaggerate the brows to make the mask look as if the brows were molded on.

Select **Export** from the **File** menu and save the figure in the OBJ format. Make sure the option **Export body parts as groups** is selected. Name the file heroFrown.obj. Depending on the speed of your computer, you may have to wait five or more minutes for the export process to complete. Take the time to reacquaint yourself with your family.

4. Creating the Mask Object

Here we're going to create a duplicate of the head object, which will be the basis for Bat-Guy's mask. The other copy will be used for the part of Bat-Guy's lower face that's not hidden by the mask.

Launch Ray Dream and open heroFrown.obj, making sure **One for each OBJ group** is selected for the option **Create Meshform Objects**. Select all body parts but the head and delete them. Duplicate the head. Select one copy and rename it face. Scale the other copy **105%.** This will be our mask object. Jump into it. The Meshform Modeller will ask you if you wish to create a new master for the object; select **Yes** (the default value) and name the new master head. Resave the file as a new file named batHead.obj.

Because the head is symmetrical, we need to work on only one half; when we're done, we'll duplicate that half with symmetry and weld the two sections back into one. Switch to the **Front** (isometric) view. Select the **Marquee** tool and drag a selection box around the right half of the head. Make sure all vertices up to but **not** including those vertices along the vertical axis of symmetry are selected. Refer to **Figure 3**. Delete the right half of the mask.

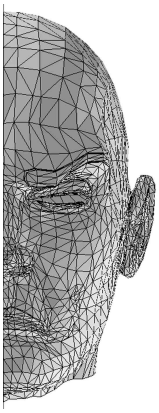


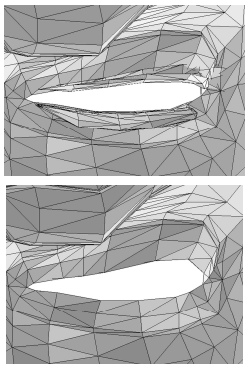
Figure 3: When half a head's better than one.

5. Lashes & Eyes

Now we're going to delete the eyelashes and create a new object for the left eye, based on the edges lining the eye opening.

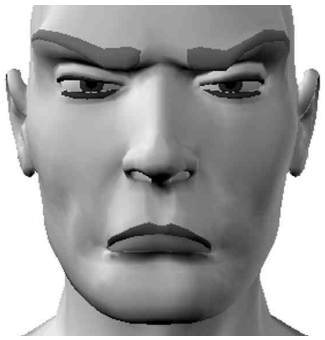
Zoom in tight on the eye opening. Select the polygons making up the lashes. Refer to **Figures 4-5**. Be very careful; these polygons are small and intersect the eye opening in many places. We needn't select them all at once. Delete them.

Select all vertices around the edge of the eye opening. Duplicate them. Invert the selection so that the rest of the head is selected. Hide the selection—all that should remain is the ring of duplicate vertices of the eye opening. Refer to **Figures 6-8**. Link the vertices on the edge of the upper eyelid to those on the lower eyelid. Select everything. Fill the resulting lattice to create polygons. Select everything and name the polygons leftEye. Unhide the rest of the head.



Figures 4-5: My Hero's lashes, plucked.

Figures 1-2: Go Ahead, Make My Day

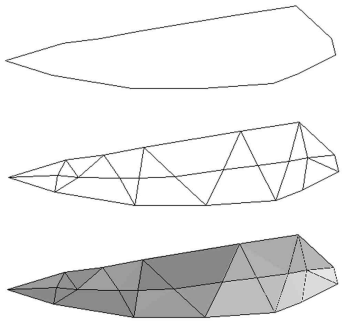


A slight change of expression for "My Hero"...

Parameter Dial values for "Make My Day"

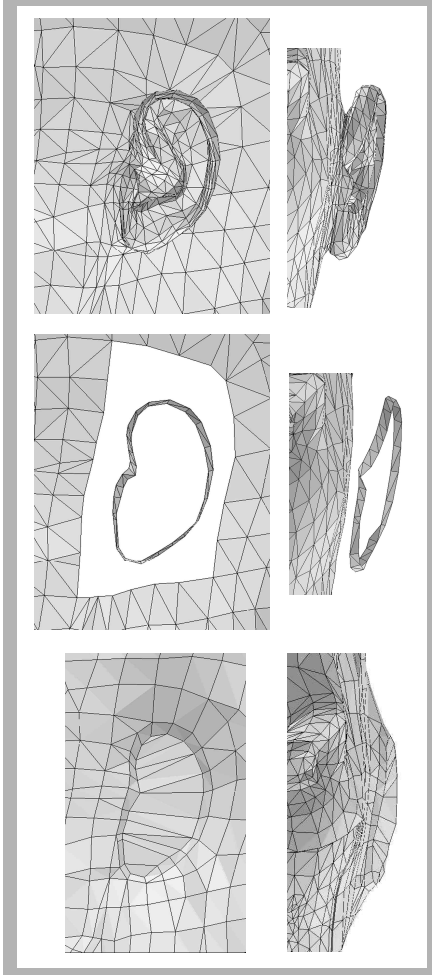
Frown	0.72
Mouth F	-0.76
Mouth M	1.56
rBrowDown	0.68
lBrowDown	0.68
rBrowUp	-0.22
lBrowUp	-0.22
Worry Right	-1.50
Worry Left	-1.50
Blink Right	0.50
Blink Left	0.50

Figures 6-8: The Eyes Have It



Left eye under construction

Figures 9-14: The Natural Look



Exit the Meshform Modeller. Duplicate the head. Jump into one copy, select the left eye we just created and delete it. Exit the Meshform Modeller. Jump into the other copy. The Modeller will ask you if you wish to create a new master for the object; select yes (the default value) and name the new master **leftEye**. Select the left eye, then invert the selection. Delete the rest of the head. Exit the Meshform Modeller and save `batHead.obj`.

6. Ear-Under-Mask

How we approach the ear comes down to a matter of taste. The simplest and fastest way of dealing with it is to delete the ear completely and fill the resulting hole. But because I wanted a more natural look, I decided to go for the invariably more time-consuming "ear-under-mask" look. This involves essentially rebuilding the entire area of the mask around the ear.

Open `batHead.obj` and jump into the head. Switch to the **Right** (isometric) view. Very carefully select polygons in and around the ear, referring to **Figures 9-14**. We needn't select them all at once. Delete them. When all have been deleted, save our work.

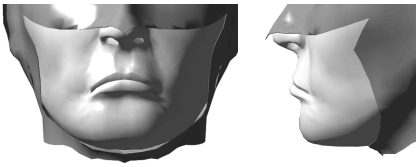
The remaining strip will need some minor adjustments. For example, we should move the vertices comprising the head-side of the strip somewhat outward in order to flatten its slope. Again, refer to **Figures 9-14**.

Rebuild the wireframe around the strip, as we did for the eye, link by link, polygon by polygon, attaching it once again to the head. Fill all polygons and save our work.

7. Mask

Now we're going to cut out the area of the mask where Bat-Guy's lower face is exposed. A number of different techniques can be used to achieve this result.

If Ray Dream had Inspire 3d's **Drill** command, this would be relatively easy; we'd simply draw the curve of the mask opening or import an illustrator eps file of the curve and project it onto the mask in its front orientation. See **Figure 17**. We'd then use the Drill to punch the shape out of the mask.

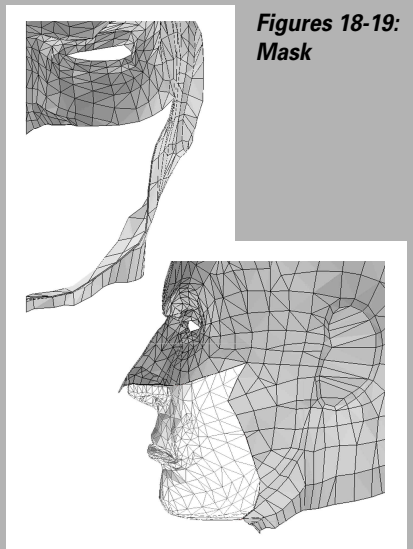


Figures 15-16: Exposed area of mask

At the time I didn't have Inspire 3d and its Drill command, so I used its closest analog; **Boolean Operators**. Ray Dream's Boolean tools can get somewhat messy, requiring an investment in cleanup time. Open the Meshform Modeller and switch to the **Front** (isometric) view. Use the **Polyline** tool to trace the outline of the mask opening into position over the head's lower face. Make sure to close the polyline by clicking on the first vertex. Switch to the **Right** view. If the polyline is not to the left of the head, select it and drag it to the left, placing it in front of the head.

Use the polyline tool to draw a single two-vertex line for our sweep path. Put one vertex in the same plane as the polyline; put the other anywhere behind the jawline, but not so far back to intersect the rear of head. Keep the shift key pressed to constrain the line horizontally. With the outline selected, select **Extrude** from the Selection menu. In the **Extrude Options** dialog, specify the depth of the straight extrusion and click **OK**.

Figures 18-19: Mask



Select both the extruded shape and the head. Select **Boolean Operations** from the Polymesh menu. Enable **Auto** at the bottom of the **Properties** palette. With Auto enabled, you can immediately view the difference between the two subtraction operations, **Subtract A from B** and **Subtract B from A**. Enable the operation that subtracts the extruded shape from the head. We'll probably need to clean up or even reconstruct the opening's edges now.

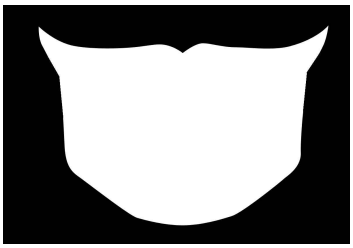


Figure 17: Outline of mask opening

The last method for creating the opening is the most time-consuming and masochis-

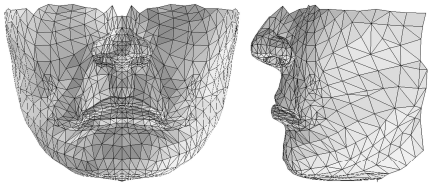
tic, but also the most precise—create the mask outline path directly on the head by selecting vertices and linking or unlinking them, as well as subdividing lines and polygons as necessary. Refer to Figures 18-19. This is the method I in fact used. After working our way from the tip of the nose, across the cheek, down and under the jaw, select all vertices within the outline shape, detach and delete them.

Some optional modifications on the opening include:

- Deleting some of the polygons forming the tight space between the nostril and cheek; reconstructing the area so that the mask stretches directly across the space.
- Pulling out the facets on the tip of the nosepiece so that it overhangs the nose.
- Constructing a rim on the opening so that the edge looks three-dimensional and not paper-thin, and more importantly, renders as two-sided.

8. Face

Remember the duplicate head object we named `face`? Good. Jump into it and carefully delete the polygons that will be hidden by the mask. Leave enough of the lower face and neck beyond the edges of the mask so that there are no apparent "holes" peeking between the face and the mask. Select all the remaining polygons and name them `face`. Make sure the part's master name is also `face`.



Figures 20-21: Polygons left for face

9. Bat-ears

Because at the start of this project I was more adept at manipulating Ray Dream's Freeform Modeller than pushing and pulling, linking and subdividing lines and vertices, I used that tool to quickly construct the curving, pyramidal, back-swept and organic-looking bat-ear I intended to join to the head using the Boolean Operators. Cleaning up after the operation, as well as finessing and finagling the results proved to be far more time consuming than it was worth. (Though I am nonetheless satisfied with the results.) See Figures 22-24.

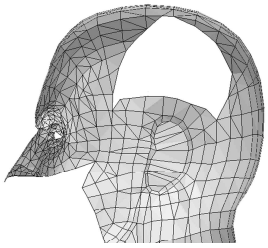


Figure nn:
Initial Bat-ear opening

After quickly constructing the scalloped fins decorating Bat-Guy's gloves by pulling out vertices on the forearm, I've concluded that this is the method of choice for constructing the bat-ears. See Figures 25-28. Simply put: the target area is opened or emptied; one or more links are created to span the area and subdivided; a vertex is pulled out to form the apex; more links created; the shaped is progressively refined through successive subdivision. Skip ahead to the batfin section for details to see how that method may be adapted to creating our bat-ear.

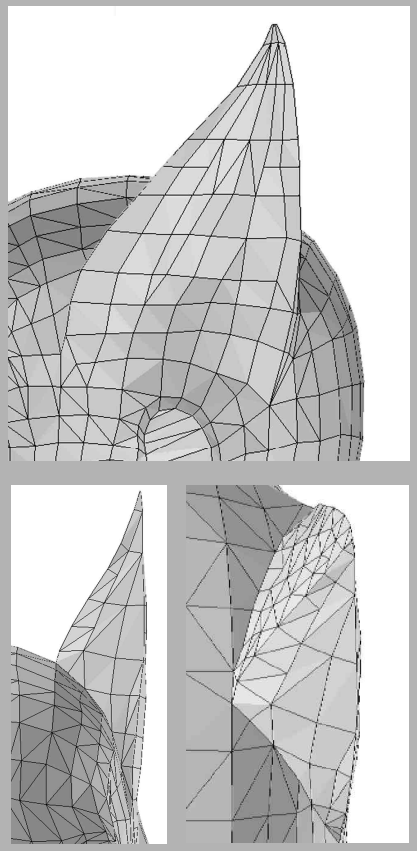
10. Finishing the Head

All that's left to complete the head is to create the right half and alter the neck so that it joins seamlessly to the mask.

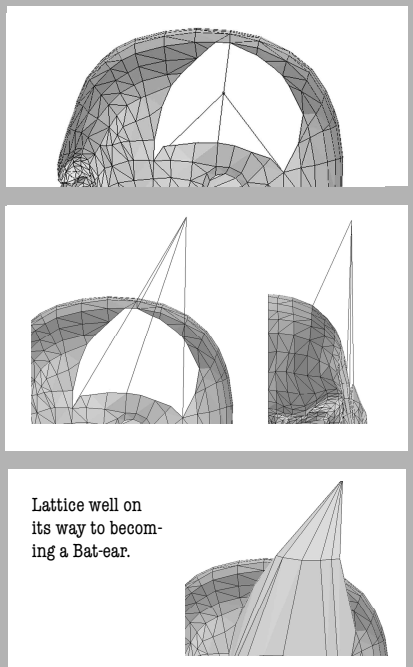
Jump into the head object. Switch to Front view. Select the face object. (Double-click on any polygon to to automatically select the entire mesh.) Select Hide Vertices from the View menu.

Select everything, which should include the left half of our mask object and the left eye. Select Rotate Drawing Plane Left from the View menu to bring the drawing plane parallel to the head's axis of symmetry. Select Duplicate with Symmetry from the Edit menu.

Figures 22-24: A Painstaking Bat-ear

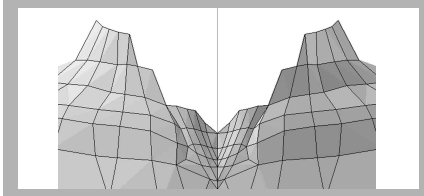


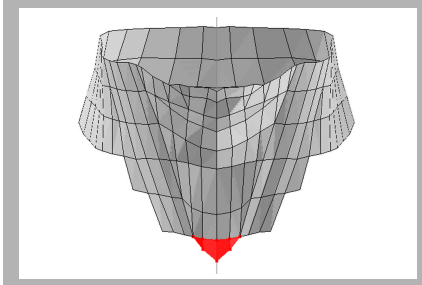
Figures 25-28: A Better Bat-ear

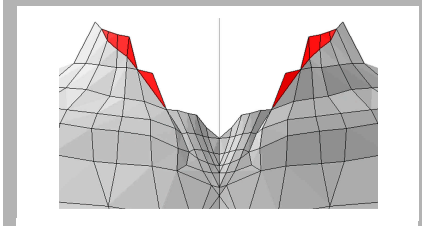


Lattice well on its way to becoming a Bat-ear.

Figures 30-32: Maskline







Refining the neckline:
The idea is to swap the red vertices between the two parts.

Select the new right eye and name the mesh `rightEye`.

Now we need to make the two mask halves one solid mesh. Select the line of vertices lying on the axis of symmetry and only those vertices. The easiest way to do this is to drag a selection marquee around the points from top to bottom. Select **Weld...** from the **Selection** menu. The default tolerance should be fine. Click **OK**. Check our results front and back. If any polygons became unfilled or collapsed, undo the operation (**Undo** on the **Edit** menu or **command-Z** on the keyboard). Try increasing the view magnification and reselecting the vertices—it's important to select only the vertices on the line of symmetry. And we might try lowering the tolerance value. If we cannot weld the points without collapsing polygons, we'll have to just reconstruct those areas.

When done welding, select the entire mask polymesh and name it `head`. Jump out of the Meshform Modeller, select the head and check both the object name and the master name. Change both to `head` if they are not. Copy the head to memory and save `batHead.obj`.

Now on to the neck. Open `batGuy.obj`. Delete the `head`, `leftEye` and `rightEye`. Paste the head we copied from `batHead.obj` into `batGuy.obj`. If necessary, align the head as closely to the neck as we can without disturbing the neck. Select both the head and neck and group them. Export the group as `batHeadNeck.obj`.

Open `batHeadNeck.obj`, making sure **One** is selected for the option **Create Meshform Objects**. Jump into the object. Select the mask mesh and invert the selection. Select **Hide Vertices** from the View menu; this should hide the face, eyes and neck.

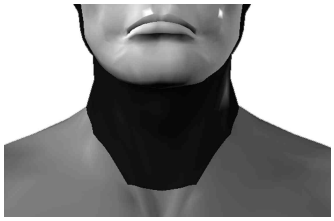
Select all vertices along the edge of the mask that meets the neck. Select Duplicate from the Edit menu. Select invert and delete the mask object. Select **Reveal Hidden Vertices** from the View menu. Delete both eyes.

Rebuild the edge of the neck that meets the mask by systematically deleting the vertices on that edge and linking the remaining neck object to the corresponding vertices on the edge we duplicated from the mask. Refer to **Figures 18-20 of Tutorial 1** for an illustration of the technique. When done, select the entire neck mesh and name it `neck`. Jump out of the Meshform Modeller, select the `neck` and check both the object name and the master name. Change both to `neck` if they are not. Finally copy the neck into memory.

Open `batGuy.obj`. Ungroup the head and neck. Select the neck and delete it. Paste the new neck object from memory into place. Save `batGuy.obj`.

Neckline. The seam between the neck and the chest is not a smooth edge, as it should be at the edge of Bat-Guy's mask (see **Figure 29**). Instead, it's a jagged meeting of polygon corners. We'll need to redefine the polygons making up that edge. We'll create a new edge by subdividing a few of the polygons on the front of the neck opening on the chest object and renaming a few on the neck object.

Figure 29:
Neckline for Bat-Guy's cowl



Launch Ray Dream and open `batGuy.obj`. Select the neck object and jump into it. Select the polygons making up the triangular area at the bottom front of the neck. See **Figure 31**. Rename them `chest` and detach them from the polymesh (**Selection** menu, **Detach Polygons**). Jump out.

Select the chest object and jump in. Imagine a pair of lines originating from the corners of the triangular space (where the detached polygons we renamed `chest` will fit), spanning the five polygons on the edge of the neck opening, and ending at the top of that opening. See **Figure 32**. Create two new edges based on these two lines by subdividing the polygons these lines bisect.

Select the new edge polygons, name them `neck` and detach them (**Selection** menu, **Detach Polygons**). Invert the selection and name the rest of the polygons `chest`. Jump out.

Select the neck and the chest and group them. Export the group as an OBJ file and name it `neckChest.obj`. Open `neckChest.obj`, making sure **One for each OBJ group** is selected for the option **Create Meshform Objects**. Duplicate the combined neck and chest object. Make sure one copy is named `neck` and the other `chest`.

Jump into the neck. Select the all the chest polygons, including the ones formerly belonging to the neck. Delete them. Jump into the chest. Select the all the neck polygons, including the ones formerly belonging to the chest. Delete them. Jump out.

Select the neck and the chest, group them and copy the group to memory. Switch to the Perspective window for `batGuy.obj`. Delete the neck and the chest. Paste the new neck and chest objects from memory.

Eyes. Now we'll take each eye polymesh out of the head object and make it a separate part. Duplicate the head. Name the copy `rightEye`. Jump into rightEye. Double-click on the `rightEye` mesh to select all its polygons. Invert the selection and delete the everything, leaving only the `rightEye` mesh. Jump out. Make sure the part's master name is changed to rightEye.

Duplicate the head a second time. Name the copy `leftEye`. Jump into `leftEye` and delete the everything, just as we did for `rightEye`, leaving only the `leftEye` mesh. Jump out. Make sure the part's master name is changed to `leftEye`.

Now to remove the eye meshes from the head. Jump into the head. Select each eye mesh and delete it. Jump out.

Save `batGuy.obj`.

11. Boots

Boots are easy. Just create a series of vertices around the shin, in order to separate the shin facets from the boot facets.

Open `batGuy.obj`. Jump into the right shin, named `rShin`. Carefully create a continuous series of new vertices, by subdivision and linking of edges, around the shin, in the shape of a boot edge, rising to a peak in the front, while falling to a cleft under the calf. See Figures 33-36.

Select all polygons of below and including the new edge, comprising the boot, and name the polygons `rBoot`. Detach the selected polygons from the shin mesh (Selection menu, Detach Polygons). Invert the selection and name these polygons `rShin`. Jump out.

Now to make each boot a separate part. Duplicate `rShin`. Name the copy `rBoot`. Jump into `rBoot`. Double-click on the `rBoot` mesh to select all its polygons. Invert the selection and delete the everything, leaving only the `rBoot` mesh. Jump out. Make sure the part's master name is also changed to `rBoot`.

Now we need to delete the boot facets from `rShin`. Jump into `rShin.` Double-click `rShin` mesh to select all its polygons. Invert the selection and delete the everything, leaving only the `rShin` mesh. Jump out.

Repeat the above operations on `lShin`.

Save `batGuy.obj`.

12. Gloves

Just like our boots. Just create a series of vertices around the forearm, in order to separate the forearm facets from the glove facets.

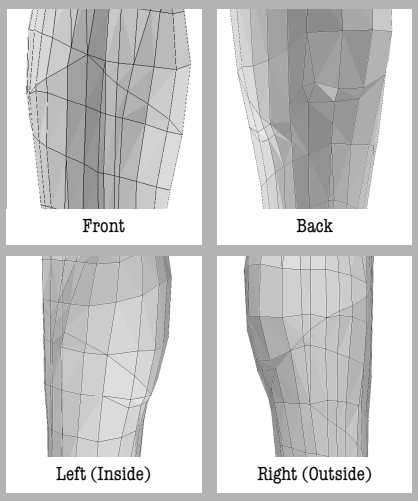
Open `batGuy.obj`. Jump into the right forearm, named `rForeArm`. Carefully create a continuous series of new vertices, by subdivision and linking of edges, around the forearm, in the shape of a glove edge, rising to a peak at the elbow, while falling to a cleft in front. See Figures 37-40.

Select all polygons of below and including the new edge, comprising the glove, and name the polygons `rGlove`. Detach the polygons from the rest of the forearm mesh (Selection menu, Detach Polygons). Invert the selection. Name these polygons `rForearm`. Jump out.

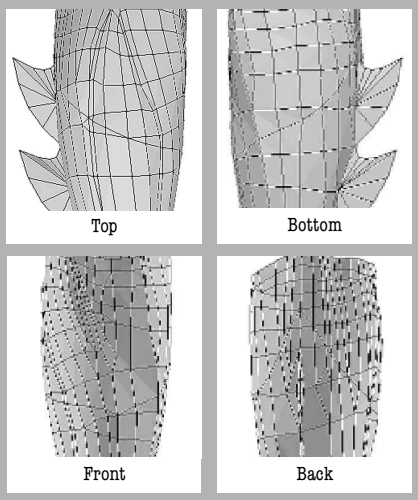
Now to make each glove a separate part. Duplicate `rForeArm`. Name the copy `rGlove`. Jump into `rGlove`. Double-click on the `rGlove` mesh to select all its polygons. Invert the selection and delete the everything, leaving only the `rGlove` mesh. Jump out. Make sure the part's master name is changed to `rGlove`.

Now we need to delete the glove facets from `rForeArm`. Jump into `rForeArm.` Double-click `rForeArm` mesh to select all its polygons. Invert the selection

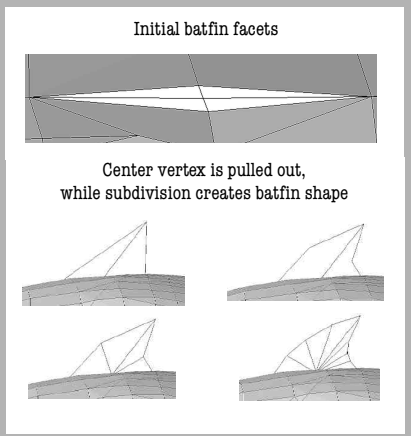
Figures 33-36: Bat Boots (Right Shin)



Figures 37-40: Bat Glove (Right Forearm)



Figures 41-45: Glove Fins



and delete the everything, leaving only the `rForeArm` mesh. Jump out.

Repeat the above operations on `lForeArm`.

Save `batGuy.obj`.

Glove Fins. We'll create the fins decorating the back of Bat-Guy's gloves by making a diamond-shaped group of facets on the glove, and pulling those vertices out from the arm. By subdividing the extruded shape, we'll refine the fins' curvature.

Open `batGuy.obj`. Jump into the right glove, named `rGlove`. Along the median stretching from the back of the wrist to the peak of the glove, measure an area of a third of that length. This should cover the span of two polygons. By subdividing the two links on either side of the median, create the initial diamond-shaped facets. See **Figure 46**.

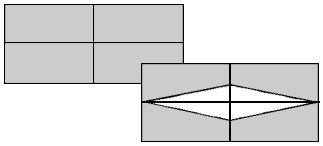


Figure 46: Making the initial batfin facets

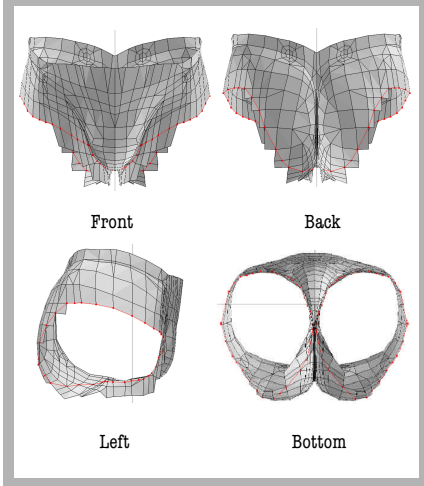
Pull the center vertex out from the forearm's surface, until we have something like a lop-sided tent. Successively subdivide the tent's front and back edges, until a curvature is defined. After each subdivision, link the new vertices to the initial batfin vertices on the surface of the forearm, top and bottom. We'll need to adjust the position of the new vertices to refine the fin's curvature. See **Figures 41-45**. Lastly, fill the polygons.

Repeat the above operations twice more to create the other two fins on the forearm. Select all polygons and name them `rGlove`.

Jump into `lGlove` and complete its fins. Select all polygons and name them `lGlove`.

Save `batGuy.obj`.

Figures 47-50: Bat Shorts



13. Shorts

Here we'll create a series of vertices around the crotch, in order to separate the shorts facets from the hip facets.

Open `batGuy.obj`. Jump into the hip. Carefully create a continuous series of new vertices, by subdivision and linking of edges, around the outside of the hip and under the crotch, in order to define the "panty line". Be especially careful under the crotch, where access is awkward and the polygons small and crowded. See **Figures 43-46**.

Select all polygons of above and including the new edge, comprising the shorts, and name the polygons `shorts`. Detach the polygons from the mesh (**Selection** menu, **Detach Polygons**). Invert the selection. Name the selected polygons `hip`.

Now to make the shorts a separate part. Duplicate `hip`. Name the copy `shorts`. Jump into `shorts`. Double-click on the `shorts` mesh to select all its polygons. Invert the selection and delete the everything, leaving only the `shorts` mesh. Jump out. Make sure the part's master name is changed to `shorts`.

Save `batGuy.obj`.

14. Chest Insignia

Bat-Guy's chest insignia need not be modelled; a simple texture map of proper resolution can suffice. This section therefore can be considered optional. I chose the modelling route because I didn't like the size of the map needed to create a sharp-looking symbol. (Remember, in Poser all texture maps cover the entire figure.) I also wanted to ensure that the coloring of the insignia was consistent with that of the mask, boots and gloves.

Modelling the insignia involves cutting its shape from a copy of the chest part using a Boolean operation. This ensures that the symbol exactly follows the contours of the pectorals. We then extrude it slightly and group the symbol with the chest.

Because I don't like the "raggedness" of Ray Dream's Boolean tools, I opted to use the smoother **Solid Drill** tool of Inspire 3d (which I'd purchased by this time) to punch out the symbol.

(Those of you familiar with Inspire 3d might be tempted to use the related **Stencil** tool to cut the symbol outline into the chest while retaining the chest polygons; don't bother—the complexity of polygons created at the symbol's edge will produce on both pieces an unsightly "buckling", which no amount of cleanup can fix. Because buckling, which can be unavoidable, increases with the complexity of the outline used for cutting, I suggest using line segments rather than curves to define the outline.)

Launch Ray Dream and open `batGuy.obj`. Select the chest. Choose **Export ...** from the **File** menu and save the part as a Wavefront OBJ named `chest.obj`.

Launch Inspire 3d Modeller and load object `chest.obj`. In another layer load your Adobe Illustrator outline for the insignia using the **LW_Illustrator_Import** plugin (**Objects** tab, **Custom**). See **Figure 51**. You'll probably need to move (**Modify** tab, **Move**) and resize the outline (**Modify** tab, **Size**). Place the symbol in front of the chest. Do not move or resize the chest.

Extrude the insignia (**Multiply** tab, **Extrude**) along the z axis—use the side view window—but do not make it so deep as to penetrate the back of the chest. Rotate (**Modify** tab, **Rotate**) the extruded form so that front and back faces of the insignia are parallel to plane of the pectorals.

Figure 51: Chest Insignia

Because the **Drill** tool subtracts the object in the background layer from the object in the foreground, we need to assign layers. Select the top half of the **Layer button** for the layer containing the chest to bring it to the foreground. Now **Shift-select** the bottom half of the **Layer button** for the layer containing the symbol to make it visible in the background. The symbol wireframe should be black and the chest wireframe white. See **Figure 47**.

Select the Solid Drill command (**Multiply** tab, **S Drill**). Set the **Axis** to **Z** and use the **Core** option. Click **OK**. The only remaining facets of the chest should be the symbol's. See **Figure 48-49**.

Export the symbol as a Wavefront OBJ named `batSymbol.obj` by using the **IN_Translator3D-Export** plugin (**Tools** tab, **Custom**).

Launch Ray Dream and open `batSymbol.obj`. Jump into the symbol. Select all polygons and choose **Extrude** from the **Selection** menu. Enter `0.003` as the **Length** in inches and click **OK**. The symbol is now three-dimensional. See **Figure 50**.

Select all polygons and name them `batSymbol`. Jump out of the symbol and copy it to memory.

Open `batGuy.obj`. Paste the symbol from memory into the file. **Shift-select** the symbol and the chest and group the two. Export the group as a Wavefront OBJ named `group.obj`. Open `group.obj`, making sure **One** is selected for the option **Create Meshform Objects.** Jump into the chest if you wish to finesse the position of the symbol—I moved the symbol back into the chest slightly—otherwise simply copy the object to memory.

Bring the `batGuy.obj` window to the foreground. Select the chest object and delete it. Paste the new chest object into its place.

Save `batGuy.obj`.

15. Utility Belt

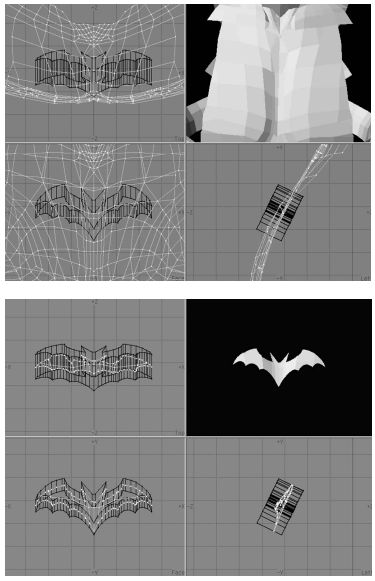
Lastly, we come to Bat-Guy's most famous accessory, his utility belt. Once constructed, the belt will be made an inseparable part of the hip object.

Launch Ray Dream and open `batGuy.obj`. Copy the hip object into memory. Open a new file, paste the hip object into it. Save the file as `batBelt.rds`, then jump into the hip.

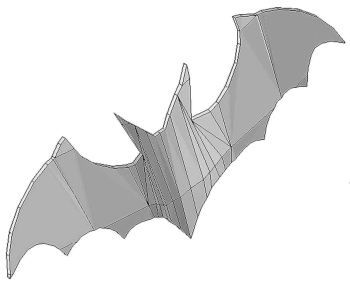
Select all edges bordering the hip object where it meets the abdomen and duplicate them twice. Select one of these copies and move it up 0.5 inches (**Selection** menu, **Move**, **dz: `0.5` in.**). Select the entire hip mesh and hide the selection leaving the two copies visible.

Link the two copies, vertex to vertex, then fill the resulting lattice, to create a belt. Select the entire belt, and extrude it to give it depth (**Selection** menu, **Extrude**, **Length: `0.05` in.**). See **Figure 56**. Move the belt down 0.25 inches (**Selection** menu,

Figures 52-54: Chest Insignia

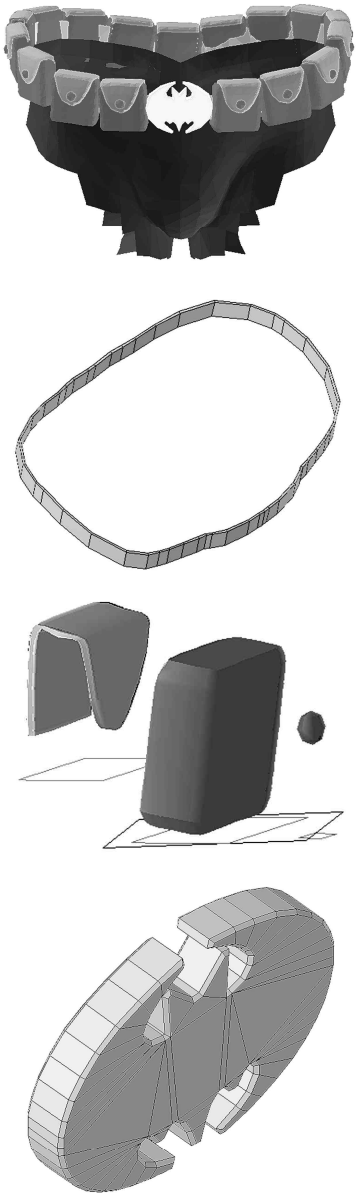


Lightwave's Solid Drill tool in action



Insignia polygons extruded in Ray Dream

Figures 55-58: Belt



Belt and components,
all to become the new hip object

Move, dz: `0.25` in.). Jump out of the hip object.

Let's put together one of the belt's compartments, which we'll duplicate many times and place around the belt. To create the compartment, select the **Free Form** object from the **Tool** bar and drag a new object into the **Perspective** window, which will automatically launch the **Free Form Modeller**. Name the object `pocket`.

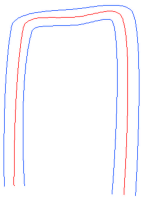


Figure 59:
Ground Plane Sweep Path and Envelopes

Select the **Rounded Rectangle** tool and draw a rectangle on the **Cross-Section** plane. Make the pocket's extrusion envelope symmetrical (**Geometry** menu, **Extrusion Envelope**, **Symmetrical**). Set the extrusion method to translation (**Geometry** menu, **Extrusion Method**, **Translation**). On the **Ground** plane, shorten the **Sweep Path** by moving its endpoint back toward the **Cross-Section** plane.

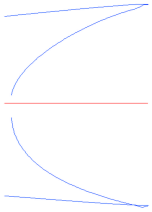


Figure 60:
Side Plane Sweep Path and Envelopes

Select the **Side** plane to make it active. Double-click the **Convert Point** tool to activate the **Add Point** tool. Add one point at near each end of either of the two extrusion envelopes. Because the extrusion method is symmetrical, all envelopes will have two new corresponding points added. Shift-select the two new points on either envelope and move them out, away from the sweep path, in order to create a bevel. You'll probably need to tweak the proportions of the cross-section, the sweep paths and the envelopes. When satisfied with the shape of the pocket, jump out of the Free-Form Modeller.

Create another **Free-Form** object and name it `flap`. Again select the **Rounded Rectangle** tool and create a tall thin shape on the **Cross-Section** plane. Make the flap's extrusion envelope free (**Geometry** menu, **Extrusion Envelope**, **Free**). Set the extrusion method to pipeline (**Geometry** menu, **Extrusion Method**, **Pipeline**).

Add two new points to the **Sweep Path** on the **Ground** plane and reposition the endpoint and the new points so that the path bends back towards the **Cross-Section** plane in a rough inverted-U shape. See **Figure 59**.

Select the endpoint and create a new cross-section (**Sections** menu, **Create**). Move to the new cross-section (**Sections** menu, **Next**). Click on the cross-section shape to select it. Reduce the shape (**Geometry** menu, **Scale**, **Horizontal:** `10%`, **Vertical:** `10%`).

Click the **Side** plane to make it active. Reshape the extrusion envelope to create the flap's tongue shape. See **Figure 60**.

For the flap button, select the **Sphere Primitive** tool and drag a new sphere into the **Perspective** window. Name it `button`. Reduce the size appropriately and flatten it slightly. Now we need to resize and align all three components. See **Figure 57**. When done, group all three; call the group `batpocket`.

Move the group onto the belt, just right of center. We'll need to be conscious of the slope of the hip at that point; we want the pocket parallel to it. When satisfied, duplicate `batpocket`, and place the copy to the right of the original, again remaining conscious of the angle of the belt and the slope of the hip. Continue creating and placing copies (I used 17 in all) until the belt is full, leaving a small buckle-sized space in front. See **Figure 55**.

Save `batBelt.rds`.

The BatBuckle. The buckle I created in Inspire 3d because I wanted to try its **Bevel** tool.

Launch Inspire 3d Modeller and load your Adobe Illustrator outline for the buckle using the **LW_Illustrator_Import** plugin (**Objects** tab, **Custom**). See **Figure 61**. You'll probably need to move (**Modify** tab, **Move**) and resize the outline (**Modify** tab, **Size**).

In the **Polygon Edit** mode, select the object and make sure the surface normals are facing us. If it isn't, flip it. Extrude the outline

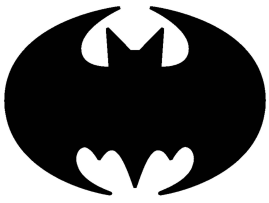


Figure 61:
Belt buckle outline

along the Z axis. Unselect all polygons.

In the **Top** view, lasso the front polygon. Open the **Bevel** requester (**Multiply** tab, **Bevel**). Set the **Inset** and **Shift** values to about 10% of the outline's height. **Edges** should be set to `Inner`. Click **OK**.

Select all polygons and copy them to memory. Paste a new copy into the layer. Rotate the copy 180° on its Y axis. Align the copy along the Z axis so that its back edge meets the back of the original. Unselect all polygons, and in the **Top** view lasso the back edges of both objects. Merge the points (**Tools** tab, **Merge**).

Export the buckle as a Wavefront OBJ named `batBuckle.obj` by using the IN_Translator3D-Export plugin (**Tools** tab, **Custom**).

Putting it all together. Launch Ray Dream and open `batBuckle.obj`, making sure **One** is selected for the option **Create Meshform Objects**. Select the buckle and copy it to memory. Open `batBelt.rds` and paste the buckle into it. Resize and align the buckle to fit the space left for it on the belt. Close `batBuckle.obj`. Save `batBelt.rds.`

Delete the hip object. Resave the file as a Wavefront OBJ named `batBelt.obj`. Close `batBelt.rds` (don't bother saving it) and open `batBelt.obj`, making sure **One** is selected for the option **Create Meshform Objects.** Jump into the object, select all polygons, and name them `belt.` Jump out.

Copy the belt to memory. **Open** `batGuy.obj` and paste the belt into place. Save `batGuy.obj.` Close `batBelt.obj`. Select the belt and hip and group them. Export the group as a Wavefront OBJ, named `batBelt.obj`, copying over the previous version. Reopen `batBelt.obj` and copy the object to memory. Make `batGuy.obj` the active window and delete the belt/hip group. Paste the new belt/hip object into its place. Rename the object `hip`. Be sure to also rename the master to this new object `hip.` Save `batGuy.obj.`

■

16. Editing batGuy.obj and BatGuy.cr2

Now that the modelling is essentially done, the next step is to open the geometry file and modify the grouping information for our parts. We'll also open up the resource file, `BatGuy.cr2` and remove any references or function calls for any parts we've deleted from the geometry file or rendered non-moving. And we'll create references and function calls to any new parts or props added to the file.

We'll also adjust any parameters for parts which may have changed proportion or position, so that the kinematic links behave properly. And we'll need to redefine the color and textures used by the model. This step will most likely involve return trips to both the modelling program and Poser as we tweak both the model and the resource file in order to get things right. To keep this section from getting too long, I'll presume we got the modelling right the first time and are otherwise satisfied with the way the model looks (a big presumption, I know), and simply point out the likeliest pitfalls along the way.

First let's make a checklist of the model's parts. (You'll need to return to your modelling program for this step.) Note that the parts do not have to appear in this or any particular order. If your Bat-Guy's model parts and their names do not correspond to the listing below, you will have problems using the model with the resource file.

(Note to Ray Dream users: By selecting and dragging part names in the ***Timeline*** *window, you can change the order in which the parts are listed, the significance of which will become apparent when we move on to coloring and texturing the model.)*

Bat-Guy Parts:

rThumb1, rThumb2, rThumb3, lThumb1, lThumb2, lThumb3
rIndex1, rIndex2, rIndex3, lIndex1, lIndex2, lIndex3
rMid1, rMid2, rMid3, lMid1, lMid2, lMid3
rRing1, rRing2, rRing3, lRing1, lRing2, lRing3
rPinky1, rPinky2, rPinky3, lPinky1, lPinky2, lPinky3
rHand, lHand, rFoot, rToe, lFoot, lToe
rThigh, lThigh, rShin, rBoot, lShin, lBoot
rForeArm, rGlove, lForeArm, lGlove
rShoulder, lShoulder, rCollar, lCollar

abdomen, chest, batSymbol
head, neck face, rightEye, leftEye
hip, shorts, belt

Grouping parts in batGuy.obj. Open up batGuy.obj in your favorite text editor. Wavefront geometry files are segregated into distinct sections, arranged by command function. (For a more complete outline of this file format, see *Appendix nn: Wavefront OBJ File Format*.) Go to the start of the last section, which defines how the model's facets are grouped into parts. This section begins with the "`g`" (for "group") commands, which denote specific groups, or parts. Each new "`g`" command defines the succeeding list of facets as a single group, and remains in effect until the next "`g`" command.

If, for example, the first part to be defined were the first part from our list above, the section would start with the first command

```
g rThumb1
```

which is followed by a list of coordinates. Our task here is to regroup the model's pieces into parts Poser is ready to use.

(Note: Get used to typing these commands. Each time we make a change to batGuy.obj in Ray Dream or any other modeller and resave the file, our revisions will be lost, making it necessary to type every command over again. I would highly suggest getting a text editor that will allow you to run grep-style batch search-and replace editing, like BBEdit for the Macintosh. This task will become especially tedious when we move on to texturing our model.)

Face. Since Bat-Guy's lower face (`face`) is a static part of the head object and not an independent animatable entity, we're going to declare those facets as part of the head.

Find the command line `g face`. This is the line that identifies the facets that comprise the face. Replace this line with the following two lines:

```
g head
#face
```

The first line assigns the succeeding facets to the head, while the second line is a comment line. All lines beginning with the pound character (#) are ignored. This second line is for our own benefit so we know which facets belong to the face.

Eyes. Since Bat-Guy's eyes (`rightEye`, `leftEye`) are now static parts of the head object and no longer independent animatable entities, as in regular Poser 3 figures, we're going to declare those facets also as parts of the head.

Find the line `g leftEye` and replace it with the lines:

```
g head
#leftEye
```

Find `g rightEye` and replace it with:

```
g head
#rightEye
```

Gloves. Bat-Guy's gloves (`rGlove`, `lGlove`) are parts of the forearm objects (`rForeArm`, `lForeArm`). Find the lines `g rGlove` and `g lGlove` and replace them with the lines:

```
g rForeArm
#rGlove

g lForeArm
#lGlove
```

Boots. Like his gloves, Bat-Guy's boots (`rBoot`, `lBoot`) are parts of the shin objects (`rShin`, `lShin`). Find the lines `g rShin` and `g lShin` and replace them with the lines:

```
g rShin
#rBoot

g lShin
#lBoot
```

Shorts. Bat-Guy's shorts (`shorts`) are part of the hip object (`hip`). Find the line `g`

shorts and replace it with the lines:

```
g hip
#shorts
```

Chest Insignia. The chest insignia (batSymbol) becomes part of the chest. Find the line g batSymbol and replace it with the lines:

```
g chest
#batSymbol
```

Save batGuy.obj.

We'll also open up the resource file, BatGuy.cr2 and remove any references or function calls for any parts we've deleted from the geometry file or rendered non-moving. And we'll create references and function calls to any new parts or props added to the file.

Editing BatGuy.cr2. Now to open up the resource file, BatGuy.cr2 and remove any references or function calls for any parts we've deleted from the geometry file or rendered non-moving. And we'll create references and function calls to any new parts or props added to the file.

As mentioned when we opened up the geometry file, batGuy.obj, the eyes are now static parts of the head object and no longer independent animatable entities, as in regular Poser 3 figures. We'll need to delete all references to them from the Poser resource file BatGuy.cr2.

Delete object declarations actor leftEye and actor rightEye (5 lines each)

```
actor leftEye:2
    {
        storageOffset 0 0 0
        geomHandlerGeom 13 leftEye
    }

actor rightEye:2
    {
        storageOffset 0 0 0
        geomHandlerGeom 13 rightEye
    }
```

Delete all of the two parameters settings functions actor leftEye:2 and actor rightEye (302 lines each)

```
actor leftEye:2
    {
        name    GetStringRes(1024,101)
        statement
        statement ...
        customMaterial   0
    }

actor rightEye:2
    {
        name    GetStringRes(1024,102)
        statement
        statement ...
        customMaterial   0
    }
```

Delete grouping functions addChild (2 lines each)

```
addChild    leftEye:2
    head:2

addChild    rightEye:2
    head:2
```

Save BatGuy.cr2.

Utility Belt. Bat-Guy's belt is the one new part that we'll be adding to the resource file. This is not a required step; in fact, we could have simply saved the belt as a separate file, imported it into Poser as a prop, and locked it about Bat-Guy's waist. While this step is far more complex, it does save us the effort of importing the belt object every time we wanted to use the Bat-Guy figure. Besides, this is the only section where you'll learn how to add a new part to the resource file itself.

Adding that part involves generating a new resource file for the model and copying the functions referencing the new part from this file into `BatGuy.cr2`. Why not simply use the new resource file? Because the resource file generated by Poser will have none of the complicated joint parameter values or the spherical falloff zones that control how the parts move. We'd have to literally enter by hand dozens of values for each of our model's parts—51 in all.

(It took me the better part of two weeks just to document all those values, which can be found in *Appendix A: Nude Male Joint Parameters* and *Appendix B: Spherical Falloff Zones*. If you wish, you can use these Appendices to enter these values into our new model. If you don't have two weeks to spare, read on.)

Instead, once we've copied the functions for the belt into `BatGuy.cr2`, we'll open the figure in Poser, and tweak the belt's joint parameter values and the spherical falloff zones to our satisfaction. Then we'll save the tweaked figure to the library as a new figure, creating yet another resource file. From that tweaked resource file we'll copy the tweaked values into `BatGuy.cr2`, making them permanent. Confused? Admittedly, it sounds more complicated than it is in actual practice.

Our Temporary Model. For the purposes of just generating functions for our belt, `batGuy.obj` is just a little too complicated; it now has too many nonstandard parts—such as the gloves, boots, shorts, etc—that we don't need to deal with for this task. We can go back to our simpler base model and create a temporary model by copying into memory the belt object from `batGuy.obj` and pasting it in place in the model myHero.obj. Save the model as `tempHero.obj`.

Creating Hierarchy Files. In order to generate brand new resource files for custom figures, Poser needs to import a hierarchy file for the new model. This is a text file created by the user. As explained in Metacreations' *Poser 3 Advanced Techniques Guide*, which does a good job covering this topic, "the hierarchy file describes how the groups in a figure are arranged and outlines the parent-child relationships between groups. Once created it can be imported into Poser 3 where it is converted to create a new figure file in the New Figures library."

The hierarchy file establishes four important things:

- The geometry file to use. (In this case, `tempHero.obj`.)
- The hierarchical connections (parent-child relationships) of the body parts.
- The desired rotation order of each body part.
- A specification of each inverse kinematics chain to be included in the figure.

For creating non-human figures, this is a necessary beginning in the process. But for our purposes, since we're modelling human figures, there's no need to create a hierarchy file to generate the resource file, and you've probably noticed by now that we've haven't bothered to do so. We've simply utilized the files supplied with Poser.

But we do need to generate resource file functions for the belt. In the hierarchy of parts, place the belt on level number 2, below the hip, at the same level of the abdomen and thighs. We'll call our file `tempHero.phi`. It can be found in *Appendix nn: Hierarchy File tempHero.phi*.

Generating the Resource File. Once the hierarchy file is complete, we can import it into Poser, which converts it into a resource file. Select **Convert Hier File** from the **File** menu. In the **Open** dialog box locate `tempHero.phi` and click **OK**. The conversion will take several minutes when the geometry file is as large as `tempHero.obj`. Provided there are no errors, Poser will prompt you for a name for the new figure. Enter `tempHero`. A new Poser 3 figure is now added to the **New Figure** library and its resource file `tempHero.cr2` will be found in

```
Poser:Runtime:libraries:characters:New Figures:tempHero.cr2
```

Copying the Belt Functions. Using our text editor, copy the following functions from the resource file `tempHero.cr2` into their respective places in the Bat-Guy resource file `BatGuy.cr2`:

Object declaration: (5 lines)

```
actor belt
    {
        storageOffset 0 0 0
        geomHandlerGeom 13 belt
    }
```

Joint parameter subroutines from the parameter settings function `actor hip`: (82 lines)

```
twistY belt_twisty
    {
        name    belt_twisty
        statement
        statement ...
        calcWeights
    }
jointZ belt_jointz
    {
        name    belt_jointz
        statement
        statement ...
        calcWeights
    }
jointX belt_jointx
    {
        name    belt_jointx
        statement
        statement ...
        calcWeights
    }
```

More joint parameter subroutines from the parameter settings function `actor hip`: (19 lines)

```
smoothScaleY belt_smooY
    {
        name    belt_smooY
        statement
        statement ...
        calcWeights
    }
```

Main parameter settings function `actor belt`: (387 lines)

```
actor belt
    {
        name    belt
        statement
        statement ...
        customMaterial   0
    }
```

Grouping function `addChild:` (2 lines)

```
addChild    belt
    hip
```

Model stitching function `weld:` (2 lines)

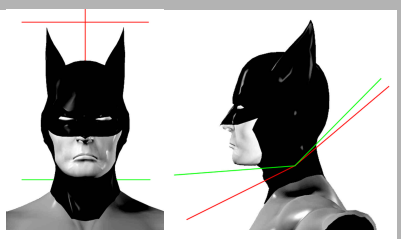
```
weld    belt
    hip
```

Save `BatGuy.cr2`.

Checking Our Work. We've finally reached a stage where we can check our model and its viability as a Poser figure. If are files are in the proper places, loading the Bat-Guy figure should be as simple as double-clicking its icon in the **New Figures** library. The first time a new figure is loaded, Poser takes a few minutes to read the geometry file. Once the figure's loaded, any problems with the model's geometry should be immediately evident. Since we haven't yet defined color or textures for the figure, this lack should not be understood as a problem. Even if the model looks okay, put it through some of Poser's poses from its library, both mild and extreme. Some problems we may encounter might be:

- **Missing parts:** Parts are missing or only partially visible, which suggests two possible causes. First, the part's normals may not be correctly oriented; simply open the geometry file, select the affected part, and reverse the normal. The second cause may be a typo in our geometry file.

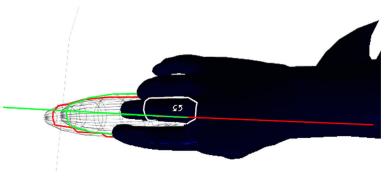
Figures 63-64: Joint Parameters (Head)



Center Bend

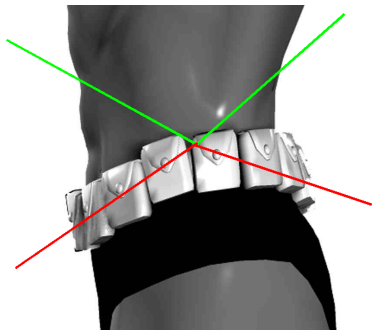
Endpoints and Joint Angles

Figure 65:
Spherical Falloff Zones (Fingers)



Zones for Bending:
Make sure each finger joint is properly contained.

Figure 66:
Joint Parameters (Abdomen)



Joint Angles for Bending:
Make sure abdomen doesn't violate belt when bending.

For instance, failing to properly group either or both eyes with the head (see our grouping revisions made to the geometry file, at the start of this section) will result in the eye(s) not being displayed in Poser because our resource file `BatGuy.cr2` contains no functions referencing the eye.

- **Split seams:** The model's seams are visible and may even gape. In extreme cases, the model seems to have been exploded. One or more factors may be at work. First, the model's parts may have moved during the modelling process; open the geometry file and check each part's position, size and orientation against the original poser nude male geometry.

  Second, the edges of model's parts may not be properly matched. Working on two parts at a time, try this, especially if there's no discernable gap in the parts when view in our modelling program: group the affected parts and them. Then, after opening the group as a single mesh, try using the **Align** command (**Polymesh** menu, **Adjust**) in Ray Dream's **Meshform Modeller**. If the seams persist, we'll have to recreate the affected edges following the seamless edge technique outlined at the end of Tutorial 1.

- **Faulty IK:** Poses result in unpredictable positions and/or the model stretches like Elastic Man. If the poses are badly off, either the parts have move from their default positions, or we have serious problems with our resource file, though such problems may result from only a few typos. But given the size of the resource file (39120 lines!), typo-hunting may prove impractical. Check all our revisions made to the resource file; if that fails to turn up discrepancies, we may need to scrap it and create a new one from a copy of `MyHero.cr2`. If the problems are very mild, then we may only need tweak the model's joint parameters and/or spherical falloff zones, tasks covered in the very next section.

(Note: Any changes saved to ***`batGuy.obj`*** *will result in the loss of all grouping revisions outlined at the beginning of this section. They will all have to be retyped.)*

Tweaking Bat-Guy. Bat-Guy should require tweaks in only three areas, if we're lucky: the head, the fingers, and the belt, abdomen and hip. Select **Joint Parameters** from the Window menu. Some of our tweaks will be entered using this window, shown in **Figure 62**. Tweaks to the model's spherical falloff zones will be entered in the main program window, using Poser interactive editing tools. This is the only method available for manipulating the zones. Once finished, we'll save the edited model as a new library figure. Lastly, we'll open the new resource file with our text editor, copy the changes, and paste them into `batGuy.cr2` to make the changes permanent.

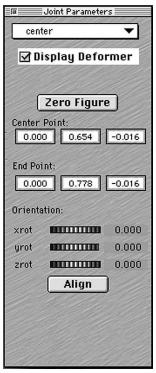


Figure 62:
Joint Parameters window

Head. Bat-Guy's bat-ears give his head different proportions than the standard nude male, so the parameters controlling the head's endpoint need to be updated. Select **Front Camera** (**Display** menu, **Camera View**) and move in close on the head. Select the head. Select **Center** from the pulldown menu at the top of the **Joint Parameters** window if it is not already selected.

The coordinates for the head's centerpoint and endpoint will be displayed, and their locations on the model are represented by red and green crossbars. See **Figure 63**. The crossbars may be directly manipulated with the same tools used to manipulate the figures. Drag the red crossbar from the top of Bat-Guy's skull to the top of his bat-ears. Alternatively, we can type the new endpoint coordinate (`0.00`, `0.778`, `-0.016`) into the fields of the **Joint Parameters** window.

Given the slightly larger proportions of the mask (105% of the head, remember?), its joint angles controlling bending (forward and backward rotation) may need changing. If bending the head backwards causes the space between the chin and the bottom of the cowl opening to stretch unnaturally and/or the base of the skull to fold or crumple unnaturally, these angles need to be increased.

Select **Bend** from the pulldown menu at the top of the **Joint Parameters** window to display the inner (green) and outer (red) joint angles. See **Figure 64**. Increase the size of the angles by pulling the endpoints out. Alternatively, we can type the following new angles into the fields of the **Joint Parameters** window:

Static A: 206.549 Dynamic B: 184.634 Dynamic C: 45.767 Static D: 40.545

Fingers. Switch to the Top Camera (Display menu, Camera View) and close in on either hand. From the Hands library load the Fist. If the sides of the fingers do not follow the rest of the hands into its curl, our spherical falloff zones need to be tweaked.

Carefully note which finger joints are affected and select one. Select Bend from the pulldown menu at the top of the Joint Parameters window to display the inner (green) and outer (red) falloff zones as well as the inner (green) and outer (red) joint angles. See Figure 65.

Using Poser's interactive editing tools, move the zones so that they completely encompass the affected joint. The zones are somewhat clumsy to manipulate—and it is too easy to mistakenly select the wrong joint or zone—but the zone's placement doesn't need to be precise. Repeat the process for each affected joint.

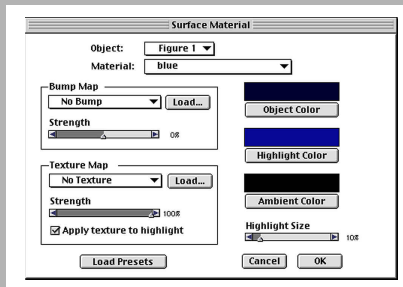
Belt, Abdomen and Hip. First we want to lock the belt to the hip. Select the hip. Select Center from the pulldown menu at the top of the Joint Parameters window. Jot down the coordinates of the centerpoint and endpoint. (The impatient may simply look up the values in *Appendix nn: Joint Parameters: Nude Male*.) Select the belt. Enter the coordinates we took from the hip as the belt's coordinates.

Switch to the Left Camera (Display menu, Camera View). Select the abdomen and bend it forward and backward. Does the abdomen unnaturally violate the volume of the belt? That is, does the abdomen "swallow" the belt when bent over it? If so, we'll need to tweak its joint angles. Decrease the size of the static angles (red) by pulling the endpoints up. See Figure 66. Alternatively, we can type the following new static angles into the fields of the Joint Parameters window:

Static A: 225.41 Static D: 325.735

Hopefully these should be the only tweaks we'll need to make, but be sure to put the model through a large variety of poses and view it from all angles. Most problems can be solved by adjusting the joint angles or moving the spherical falloff zones. When done tweaking, add the tweaked figure to the New Figures library as `BatGuyTemp`. Quit Poser (don't bother saving the file). Delete the original resource file `BatGuy.cr2` from the `New Figures` folder on your hard drive and rename the new resource file `BatGuyTemp.cr2` to `BatGuy.cr2`. With the modelling complete and the joint parameters adjusted we're now ready to paint our hero.

Figure 67: Surface Material Dialog Box



17. Colors & Textures

Painting Bat-Guy is actually a collaborative effort between the resource file, which defines the color components and the locations of texture maps, and the geometry file, which contains the commands directing which of the defined colors and textures are applied to which facets.

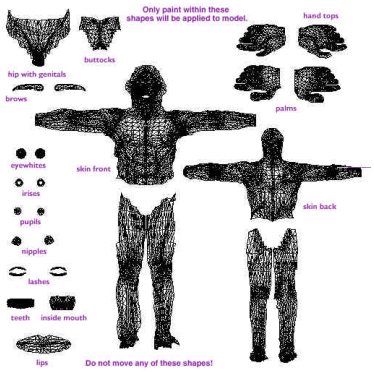
Resource File Color Definitions. The last series of functions found in the resource file define the colors an texture maps used with the model. There are actually two sets of identical functions: the presetM`aterial` function and the `material` function. I presume `presetMaterial` defines the colors and textures the model displays when initially loaded, but I can't quite determine the need for the seemingly redundant second function. Nonetheless, we'll do as the Romans do and define colors for both. Below is the function for a material called `flesh`, which we will use on Bat-Guy's exposed face.

```
material flesh
    {
    KdColor 0.95 0.7616 0.6875 1
    KaColor 0 0 0 1
    KsColor 0.5 0.5 0.5 1
    TextureColor 1 1 1 1
    NsExponent 100
                0 0
    }
```

The first three statements define the surface's three color components in RGB values, using three parameters, each a decimal percentage of 1.00. (I haven't determined what the fourth parameter is, but it is consistently set at `1`.) Sample values follow:

Sample RGB Value	Parameter Values
Black (0,0,0)	0.00 0.00 0.00
White (255,255,255)	1.00 1.00 1.00

Figures 68-69:
Poser 3 Texture Maps (Nude Male)



Wireframe template

Color texture map

50% Gray (127,127,127)	0.50	0.50	0.50
Magenta (212,0,102)	83.14	0.00	0.40

These components will be familiar to any 3d artist. `KdColor` defines the diffuse color component, what we normally consider an object's "color." `KaColor` defines the ambient color component, which reproduces the ubiquitous reflected "daylight." `KsColor` defines the specular highlight color component, for reflected highlights. `NsExponent` defines the size of the specular highlight; it is a percentage expressed as a whole integer.

Of the material components not used in this material, `bumpStrength` is blessedly self-explanatory, `ksIgnoreTexture` defines what percentage of the texture map is applied to the highlight, and `textureMap` designates the pathname of the image file to be used with the material. Like `NsExponent`, `bumpStrength` and `ksIgnoreTexture` are percentages expressed as whole integers; `bumpStrength` ranges from -100 to 100.

The `TextureColor` component and the last line, `0 0`, unfortunately remain a mystery to me.

Each component has an analogous control in Poser's **Surface Material** dialog box (see **Figure 67**), listed below:

Material Function Component	**Surface Material Dialog Input**
KdColor	Object Color
KaColor	Ambient Color
KsColor	Highlight Color
NsExponent*	Highlight Size
bumpStrength	Bump Map: Strength
ksIgnoreTexture*	Texture Map: Apply texture to highlight
textureMap	Texture Map

**The value of this component is actually the inverse of its dialog input analog.*

These are the only material components I've seen used in Poser 3 resource files. However, if Poser's naming conventions are consistent (a *big* if), we might extrapolate the names of the two components that might exist for the only two **Surface Material** dialog controls not covered in the above list:

Material Function Component???	**Surface Material Dialog Input**
bumpMap	Bump Map
textureStrength	Texture Map: Strength

I've never tried using these speculative components; perhaps some of my readers will feel like experimenting with them.

Poser's texture maps are bitmap images (PICT format for Macintosh, TIF or BMP for Windows) whose colors and patterns for each body part must fall within prescribed areas, though the map itself can be proportionately scaled up or down, the only effect being a change in image resolution. See **Figures 68-69**. So if we wish to texture only a single part, we must still use the entire map, which makes them somewhat inefficient.

But in the absence of a designated texture map, Poser will render the figure using the remaining material components, which are sufficient by themselves. Since Bat-Guy's uniform is not made up of highly detailed or wildly flamboyant patterns, but flat colors, maps aren't really necessary. In fact, the reflective, satiny effect on the blue parts of his uniform is *impossible* to duplicate with a texture map!

Another fact which contributed to this movement away from using maps with custom models concerns a quirk that arises when trying to load alternate maps using the **Surface Material** dialog box. The texture loaded is assigned to all parts of the model in an all-or-nothing fashion. Only by directly editing the `textureMap` references in the resource file can we add or change the maps for our model. All other controls of the dialog box work as normal. Listed below are the remaining materials defined for Bat-Guy:

```
material blue
    {
    KdColor 0 0 0.15 1
    KaColor 0 0 0 1
    KsColor 0.05859 0.05859 0.05859 1
    TextureColor 1 1 1 1
    NsExponent 70
                0 0
    }
material grey
```

```
    {
    KdColor 0.45 0.45 0.45 1
    KaColor 0 0 0 1
    KsColor 0.1 0.1 0.1 1
    TextureColor 1 1 1 1
    NsExponent 90
                0 0
    }
material gold
    {
    KdColor 0.99 0.83 0.18 1
    KaColor 0 0 0 1
    KsColor 0.5 0.5 0.5 1
    TextureColor 1 1 1 1
    NsExponent 60
                0 0
    }
material white
    {
    KdColor 1 1 1 1
    KaColor 0 0 0 1
    KsColor 0.5 0.5 0.5 1
    TextureColor 1 1 1 1
    NsExponent 100
                0 0
    }
```

Duplicate these materials for the presetMaterial section and save `BatGuy.cr2`.

Geometry File Commands. The second part of our task involves typing into the geometry file the commands that tell Poser which of the materials defined in the resource file to use for which facets. These commands will follow our grouping commands because we were clever enough to create our model parts according to the arrangement of his costume's elements. For instance, to assign the material `grey` to Bat-Guy's chest, we'd simply find any and all occurrences of the grouping command `g chest` and add the material command line `usemtl grey`, as shown below:

```
g chest
usemtl grey
```

Insert a material command line for each group. That's all there is to it.

And because OBJ file commands stay in effect until a subsequent command supercedes it, the judicious ordering of our groups allow us to issue just one material command at the start of a series of groups in order to assign the same texture to all. Ray Dream's **Timeline** window allows us to drag the hierarchy of model parts into any arbitrary order we wish, and writes out the facets in that order when exporting OBJ files. This makes it easy to group body parts by material, thereby minimizing the the number of commands we'd have to type into the geometry file. Which is a very good thing, considering that every time we need to alter our model and rewrite the OBJ file, all our revisions are lost.

Following is a complete list of our model's parts, organized by material: our **Timeline** window hierarchy list should reflect this order:

Blue: rThumb1, rThumb2, rThumb3, lThumb1, lThumb2, lThumb3, rIndex1, rIndex2, rIndex3, lIndex1, lIndex2, lIndex3, rMid1, rMid2, rMid3, lMid1, lMid2, lMid3, rRing1, rRing2, rRing3, lRing1, lRing2, lRing3, rPinky1, rPinky2, rPinky3, lPinky1, lPinky2, lPinky3, rHand, lHand, rFoot, rToe, lFoot, lToe, rBoot, lBoot, rGlove, lGlove, batSymbol, shorts, head, neck

Grey: rThigh, lThigh, rShin, lShin, rForeArm, lForeArm, rShoulder, lShoulder, rCollar, lCollar, abdomen, chest, hip

White: rightEye, leftEye

Flesh: face

Gold: belt

(While the groups may fall in any order—and within groups, the parts may fall in any order—just remember to keep the parts ordered by group.)

Save `batGuy.obj`.

17. Bulking Up

Finally—the light at the end of the tunnel! We're down to the final refinements,

which will give Bat-Guy the heroic proportions he deserves. Even with his current athletic proportions based on Poser's Nude Male figure, Bat-Guy nonetheless looks a bit undersized and anemic for a superhero. Bulking him up involves only increasing the proportions of his individual parts in Poser, using the Parameter Dials. While his exact proportions are of course a matter of taste, I settled on the following scale:

Body Part	Scale	ScaleX	ScaleY	ScaleZ
head	100	112	108	112
neck	100	125	50	125
chest	100	130	115	125
rCollar, lCollar	100	130	110	125
rShoulder, lShoulder	100	100	125	140
rForeArm, lForeArm	100	100	115	125
rHand, lHand	110	100	100	100
abdomen	100	115	130	120
hip	100	115	100	115
rThigh, lThigh	100	130	106	120
rShin, lShin	100	125	106	115
rFoot, lFoot	115	100	100	100
rToe, lToe	115	100	100	100
belt	100	113	100	115

When finished, add the reproportioned figure to the New Figures library as BatGuyBig. Quit Poser (don't bother saving the file). Remove the original resource file `BatGuy.cr2` from the `New Figures` folder, archive it, and rename the new resource file `BatGuyBig.cr2` to `BatGuy.cr2`. Congratulations! Our superhero model is complete. Go out and get yourself a drink. (Of course, if you really can't think of anything better to do, you might try giving Bat-Guy a cape, or working on morph targets for facial expressions ...)

Here endeth the lesson

■

Other documents and resources in this series include:

Primer on New Models

Preparing a Base Model

Building Bat-Guy

Appendix A: OBJ File Format

Appendix B: Poser Resource File Format

Appendix C: Joint Parameters Tables

Appendix D: Spherical Falloff Zones

Base Model Geometry and Resource Files

Bat-Guy Geometry and Resource Files

Final Touches: Adding the Cape

Cape Geometry & Morph Targets