chain.inc v1.1

a "plug-in" for POV 3

By Rob Antonishen

Intro:
Someone on the IRTC asked "hey, how do I make a chain look like it is hanging?" and I opened my big mouth and rambled a bunch of stuff about how it was really quite simple, and how you just used the formula for a parabola (y=x^2) and scaled it around, and took the derivative at any point to determine the slope and that would tell you where the links were....yaaa-dee-yaaa-deee-yaaa.

So my bluff was called when some people expressed an interest in this. I started doing what I had suggested, and quick realized that to do this analytically would require solving a couple of fourth degree polynomials, and as I hadn't cracked a book since University, I figured, "Heck! POV 3 has looping structures, why not solve the equations the same way my trusty calculator does, through iteration?" And it actually ended up being simpler than that. If you are interested in the details, drop me an E-mail, otherwise I won't bore you. And anyway, the code is pretty well documented.

After posting my original .inc file, a number of people informed me that a hanging chain doesn't actually follow a parabola, as I has stated (hey, Galileo thought so too...) but follows a catenary (y = a cosh(x/a)). So, the include file now supports both curves. (For a sample graph of the difference, see *error.gif.*

Using the *chain.inc* file
A quick summary. To make the thing generic, I needed to standardize what a link can be. Simply put, a link is a declared object where the pivot points (NOT END POINTS) are at <0,0,0> and <1,0,0>. See the examples in *link1.pov*. If no object link is declared, the program defaults to a unit length cylinder with a half blue/half white pattern, and a radius of 0.2. For an example see *link_sample.gif and link_samp2.gif* included in the .zip

Here are more required variables, along with the defaults if not declared prior to including *chain.inc*:

```
//          link_length = float, the link scaling factor, default=0.1
//          link_rotation = float, the degrees to rotate each link, default=90
//          link_error =float, error allowed in iteration, default=0.01
```

```
//          chain_start = vector, one anchor point, default<-1,1,0>
//          chain_end = vector, the other anchor point, default<1,1,0>
//          chain_scale = float, controls the chain tension default=1
//          chain_debug = boolean, turns on/off endpoint markers, default=off
//          chain_type = int, 1=parabola or 2=catenary, default 1 for now
```

Of note: The link_rotation variable determines the rotation of each successive link, so using the link shown at left above, you would ideally use a value of 90. I like to use anything in the range of 85-95 to add a twist to the chain. See some of the sample images included.

Also of note: The length of hang in the chain is determined by chain_scale. Start with a value of 1, and increase it to make a longer chain, decrease to make it shorter. It must be positive.

Making it look real:
A couple of points I found while playing around:
first set the endpoints, the x_axis rotation of your starting link (by rotating within the declaration of link), link_length, and rotation. Then adjust the chain_scale to get the droop you want. You will have to do some fine tuning of the chain_scale value to get the last link to end where you want it to. Setting chain_debug=on will put red and green markers at the endpoints, so you can see how close you are. Lastly, tweak the link_rotation value to get the last link rotated the way you want it.

Hints:
The links all come back as separate objects. If you want, you can group them, so they can be reused as done for the title image see *chaind1.pov.*
The low point of the curve is available in the variable chain_low, and can be used to "hang" objects off of the chain. I may add a variable to simulate a mass at this point, but I haven't gotten that far yet.

HELP!
If you have any questions, comments, suggestions, or just want to tell me about the great images you created using this include file, drop me a line:

rant_on@geocities.com

or visit my homepage: **http://www.geocities.com/SoHo/7865**

EOF