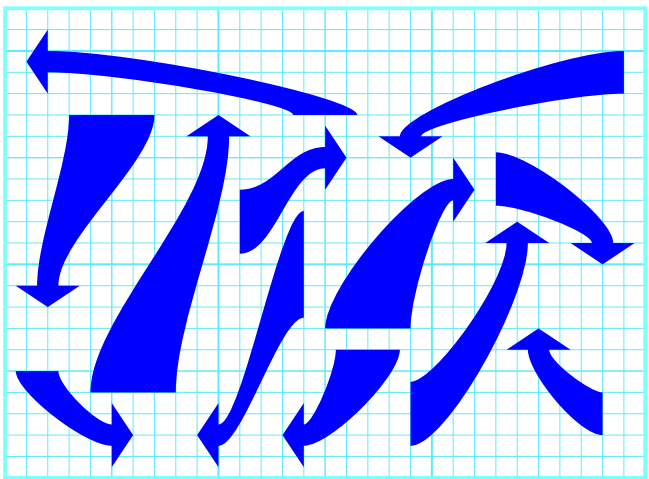
Some "Fat Tail Arrow" Utilities

Don Lancaster
Synergetics, Box 809, Thatcher, AZ 85552

http://www.tinaja.com
don@tinaja.com
(928) 428-4073

Way on back when I was writing the **Micro Cookbook** , I needed some way to add a modest amount of informality to some relatively dry tech info. And came up with what I now call a **fat tail arrow**. Per this group portrait...



In updating some of our **eBooks** to **Level II**, I decided I needed a fairly simple method to programmatically create and enter these arrows. And did so, creating **this file** using my **Gonzo Utilities** and, of course, the **PostScript** language.

The method is based on using single "fairly weak" **cubic splines** for the sides and straight line segments for the ends. It turns out there will be **twelve** possible arrangements of end orientations for fat tailed arrows of this type, so there are twelve different routines. An example routine might be named **fattailne** for an arrow that "comes from" the north and "points to" the east.

Identical data formats are used for each routine, consisting of an eight element array. As in **[tipx tipy tailx taily tipfat tailfat tipdepth tipspread] fattailne**.

As is usual with our **Gonzo Utilities**, you'll first create a standard textfile full of **PostScript** commands. And then send it to Acrobat Distiller by **Using Distiller as a General Purpose PostScript Computer** to generate a .PDF file.

Here's a sample fattail routine…

```
/fattailne { gsave aload pop /tipspread exch store /tipdepth
exch store /tailfat exch store /tipfat exch store /taily exch
store /tailx exch store /tipy exch store /tipx exch store

/enthu 0.28 store

/deltaxn tipx tipdepth sub tailx tailfat 2 div add sub enthu mul store
/deltaxs tipx tipdepth sub tailx tailfat 2 div sub sub enthu mul store
/deltaye taily tipy tipfat 2 div sub sub enthu mul store
/deltayw taily  tipy tipfat 2 div add sub enthu mul store

tipx tipy mt
tipx tipdepth sub tipy tipspread 2 div add lineto
tipspread 2 div tipfat 2 div sub pd

tipx tipdepth sub deltaxn sub tipy tipfat 2 div add
tailx tailfat 2 div add taily deltayw sub
tailx tailfat 2 div add taily curveto

tailfat pl

tailx tailfat 2 div sub taily deltaye sub
tipx tipdepth sub deltaxs sub tipy tipfat 2 div sub
tipx tipdepth sub tipy tipfat 2 div sub  curveto

tipspread tipfat sub 2 div pd closepath fill
grestore } store
```

The proc first grabs all your arrow position data. It then defines **enthu** or the **enthuasiasm** of the spline control points. **0.28** of the cardinal spline length is often a good compromise between "too straight" and "too loopy".

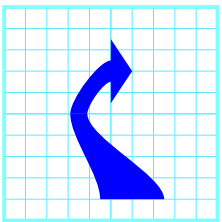
The actual control point positions are then calculated by the **deltax** and **deltay** routines. Only one delta is needed if the fattail arrows ends up in the same direction as it starts. Four are required if the arrow goes "around a bend". The "other" control point values simply follow an appropriate cardinal direction.

Actual drawing then starts at the point tip and then does one of the splines. The tail is then sent off in the appropriate direction. This is followed by the second spline and completing the arrowhead.

Single-spline-per-edge solutions seem limited to needing an "all positive" playpen for the splines to work in.

Getting Fancy

A sneaky trick can greatly expand your fat arrow possibilities. One that lets you do arrows like this one…



Note that we can make any arrowhead "disappear" by setting both **tipdepth** and **tipspread** to zero. So, what we have done above is place a "tipless" fattail arrow immediately below a "regular" fattail arrow. Giving us two splines on each side. And greatly extending the possibilities of just where the tip is allowed to be with respect to the tail.

The coding might look something like this…

```
[ 35 15 32.5 13 1 0.8 1 3 ] fattailse
[ 32.5 13 35 9 0.8 3 0 0 ] fattailsn
```

A reminder that Distiller versions above 8.1 will default to preventing most disk access. The workaround is to run **acrodist-F** from the xp command line.

For More Help

Additional examples and tutorials appear in our **GuruGram** and **PostScript** libraries. Sourcecode for this GuruGram can be found **here**.

Seminars, training, **consulting**, and direct PostScript programming projects are available. You can **email me** or call (928) 428-4073 for more details.