Now for each character in the message, Metafont determines the position of this letter on the path, and records this information for subsequent retrieval by TEX.

First, suppose the point z_0 marks our current position on the curve. Then we use Metafont's `solve` macro to find the point z_1 such that the length of the chord $z_1 - z_0$ is the same as the width of the current character. (Plain Metafont sets the value of `solve`'s tolerance to `0.1`; we need to decrease to `tolerance=0.001` or even smaller). It's easy for Metafont to determine the angle of the chord, and thus the angle of rotation for the letter. (As you see, we are approximating the path by a series of chords which inscribe the path such that each face of the polygonal approximate path will be the exact width of each character in the message.)

TEX will eventually need two pieces of information about each letter in order to typeset it properly—the x- and y-offsets of that letter from the previous letter. We pass this information to TEX using kerning pairs (as we did at the outset of this discussion to pass coordinate pairs to TEX in order to label a Metafont diagram).

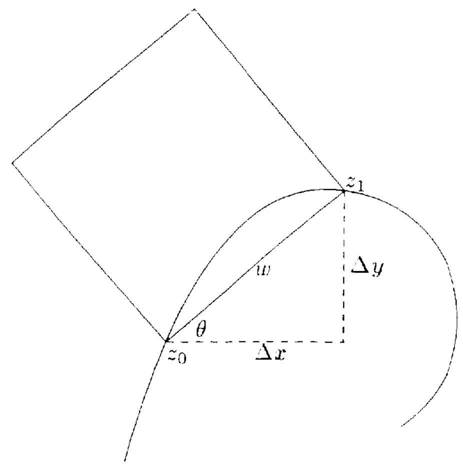

Figure 21: *Needed information.*

The Metafont process uses so many more files than TEX does that it's worth examining the alterations to the normal Metafont production cycle in a slight more detail. Normally, to produce a font of `cmr12` (for example), Metafont first reads a parameter file (`cmr12.mf`), which calls the driver file (`roman.mf`), which finally calls the program files (`romanu.mf, romanl.mf, romand.mf,` etc.).

$$\boxed{\text{MF}} \Longrightarrow \boxed{\texttt{cmr12}} \longrightarrow \boxed{\texttt{roman.mf}} \longrightarrow \boxed{\texttt{romanu.mf, romand.mf, ...}}$$

I mirror this process, except I rename the files by adding the prefix 'sp' (for special purpose)

$$\boxed{\text{MF}} \Longrightarrow \boxed{\texttt{spcmr12}} \longrightarrow \boxed{\texttt{sproman.mf}} \longrightarrow \boxed{\texttt{spromanu.mf, spromand.mf}}$$

and add two new files to the process. The first is `make.mf`, which contains the actual macros for analyzing and storing the curved baseline. The driver file `sproman` calls this file. A second file contains the actual definition of the path together with any special alterations to the macros to achieve whatever special effects you are after. You call this file whatever you want, but don't forget to `input` it in the `make` file. (In the absence of such a file, `make` defines defaults for all such special purpose things.)

$$\begin{tikzcd} & & \boxed{\texttt{make}} \arrow[r] & \boxed{\texttt{??}} \\ \boxed{\text{MF}} \arrow[r, Rightarrow] & \boxed{\texttt{spcmr12}} \arrow[r] & \boxed{\texttt{sproman}} \arrow[u] \arrow[r] & \boxed{\texttt{spromanu, spromand}} \end{tikzcd}$$

The '`??`' in this figure represents the file you need to prepare and which will contain the special definitions pertaining to your path.

As a practical matter, all fonts are named `spcmr12` in this scheme. As with any meta-font, you have to convert the generic pixel files into packed pixel `pk` files, and place it and the corresponding font metric file in the places in your system where TEX expects to find them. I have gotten into the habit of transferring these files into these distant directories with new names indicative of whatever they illustrate.

```
\font\rofont=spiral
  ⋮
\curvetype{ABCDEFG ...} % text for curved message
```

Figure 22: *Mounting—placing—the curved type on the page.*

5.4 Step Three: Mounting the Message

Finally, it's TEX's turn again. The file `mount.tex` contains all the positioning macros, as well as the machinery for extracting the position information from the kerns. All you have to do is tell TEX the name of the special purpose font containing your message, for example,

```
\font\rofont=spiral
```

5.5 Examples

In figure 23 you see some text typeset along a wiggly, self-intersecting curve. Unfortunately, it doesn't seem to be possible to have TEX automatically make space so it doesn't overprint at the point of intersection.

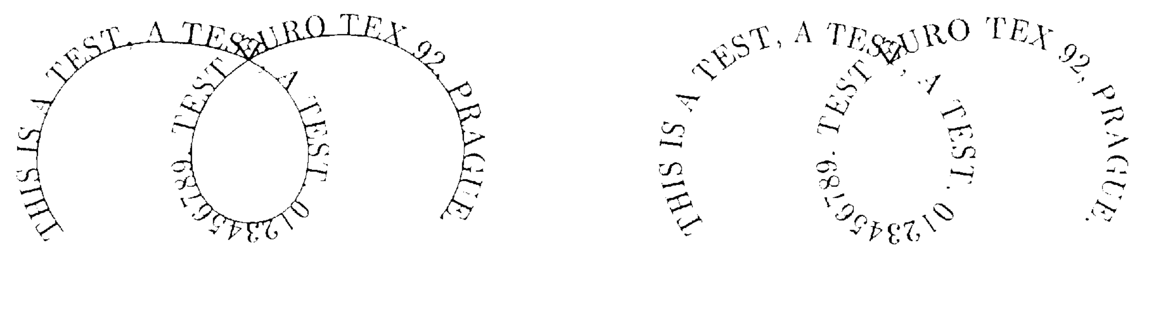

Figure 23: *Type along a wiggly curve.*

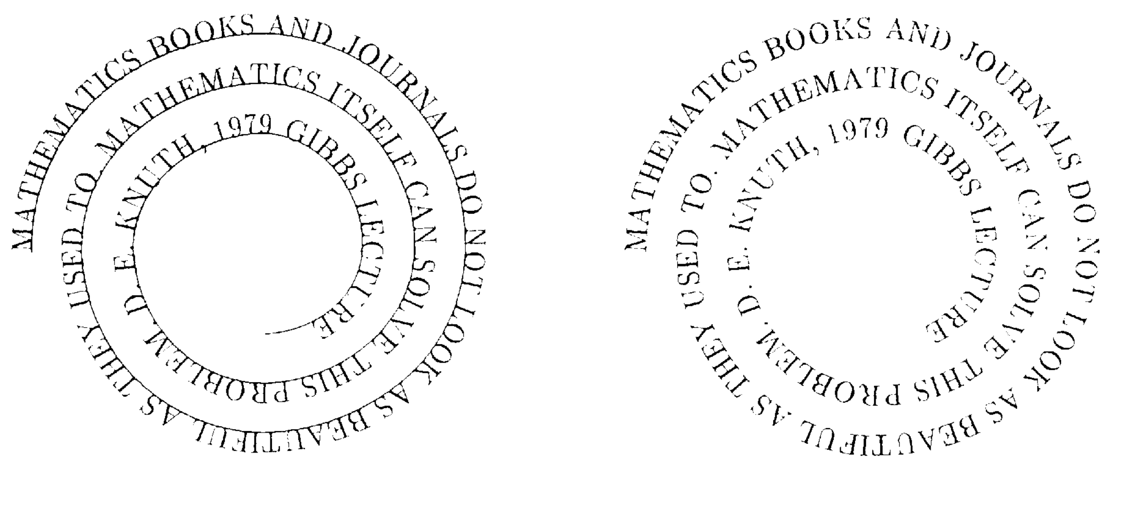

Figure 24: *Type along a spiral.*

In figure 24, TEX typesets along a spiral. It is harder than you think to draw a spiral in Metafont, so it is nice to be able to print the curved type twice—with and without the underlying path, so we can convince ourselves that TEX and Metafont have remained true to the curve. For some reason, certain device drivers choked when trying to print this bit of text.

Figure 25 was inspired by a promotional piece developed by Blue Sky Research, purveyors of TEXtures for the Macintosh. The radii of the ellipse are in golden ratio $\frac{1+\sqrt{5}}{2}$, a particularly pleasing ratio. It proved to be straightforward to achieve the font change.