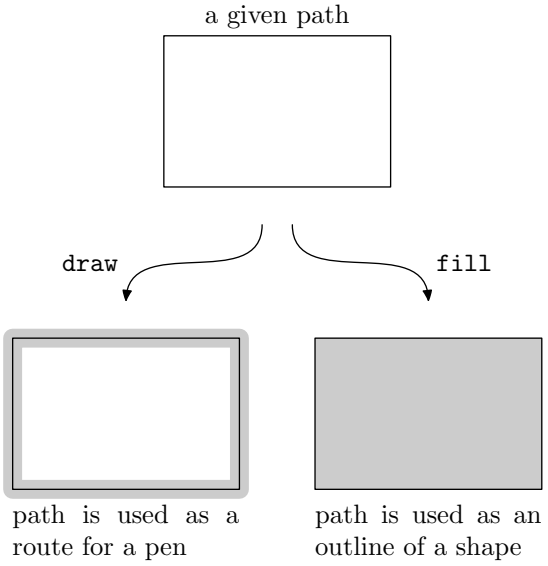
From drawn to filled paths

Linus Romer

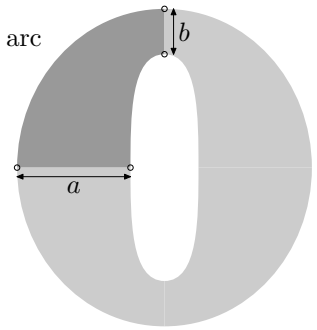
Abstract

While drawing given paths by pens is one ability of METAFONT, it is filled paths, the so-called *outlines*, which play the key role in workaday typefaces such as *Computer Modern*. The following article shows in short the relationship between drawn and filled paths.

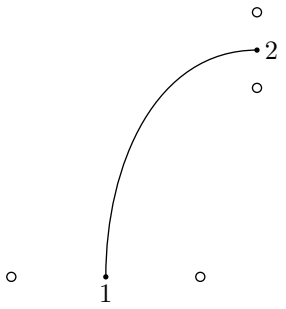


1 Construction of the letter "o"

For the sake of simplicity, this article concentrates on one exemplary shape: The letter "o". It is quite natural to divide this construction problem into four analogous parts, one of them highlighted below. Let us call such a part an *arc*. We will only look at the highlighted arc that connects the left side with the top of the "o".



We assume that we already know the beginning and the end of the arc as well as the widths a and b. The shape of the arc is then essentially determined by the shape of the path in the middle of the arc:



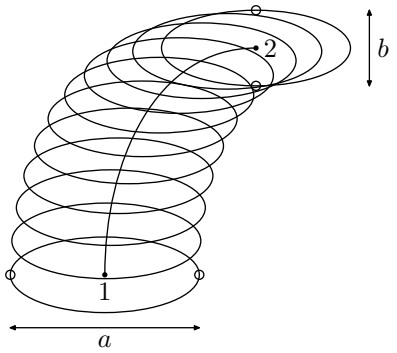
The shape of this path could be verbalized as follows: "Leave point 1 in upwards direction and make a nice curve to point 2, such that it would leave there traveling rightwards". The translation to METAFONT is much less redundant:

```
z1{up} ... z2{right};
```

With this midpath one can imagine already roughly the final shape of the arc. There are now several approaches that lead to the desired arc. The next sections present you six techniques that are available in the current METAFONT84 or were part of the old METAFONT78. Obsolete commands and methods of METAFONT78 are marked with a dagger (†).

2 Drawing with fixed elliptical pens

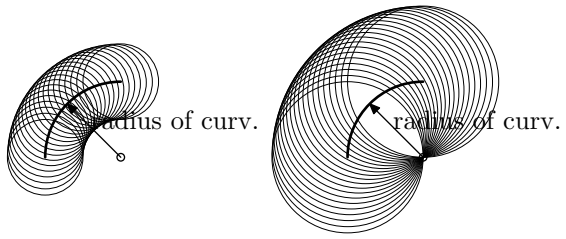
This primitive approach comes from the idea that one lets a pen glide over the given midpath. For every point on the path, the same shape is drawn on the surface (therefore "fixed"). In most cases one uses elliptical shaped pens.



For the shown arc, the width and the height of the ellipse must correspond to a and b respectively.

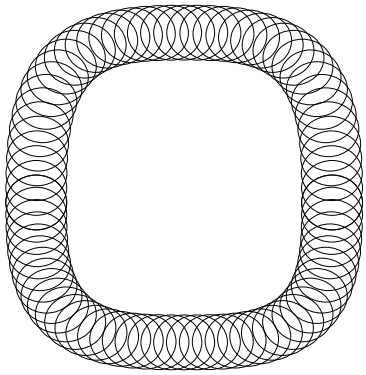
```
pickup pencircle xscaled a yscaled b;
draw z1{up} ... z2{right};
```

The figure reveals already a big problem that can occur in this method: The final height of the arc at point 2 can become larger than b. However, if the diameter of the ellipse is smaller than the radius of curvature, one can use this method confidently. Put more simply, this means that only "big" pens lead to critical cases.



uncritical case critical case

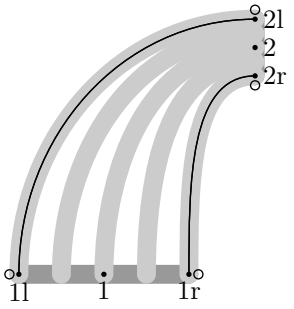
Donald E. Knuth used a fixed elliptic pen for all letters of the METAFONT logo:



As you can see, the used pen is nearly circular (the ellipses have flattening factor of only 0.1).

3 Double drawing with circular pens†

In METAFONT78 there was no direct way to fill contours. Instead, Knuth defined in [1] `ddraw`†, a predecessor of the `filldraw` command (see section 5). With `ddraw`† (for *double drawing*) one could simulate the filling–in of certain contours by stroking inside the contour several times.



To get our arc filled with `ddraw`†, we do not look at the midpath but rather apply its description on the points to the left and right and get the two outlining side paths.

```
z1l{up} ... z2l{right};
z1r{up} ... z2r{right};
```

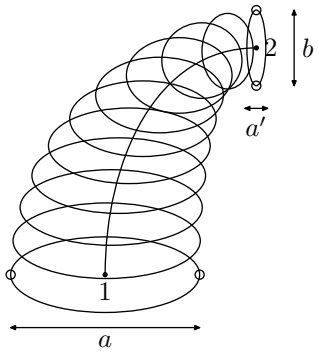
These side paths are now drawn with a normal circular pen (called `cpen`† in [1]). Of course, one has to shift the starting and ending points of these paths by the amount of the pen radius to get the correct borders.

The exact filling–in then works with a sufficiently large number of interpolated paths. The arc shown above clearly does *not* contain sufficiently many interpolated paths. METAFONT78 made sure that this could not happen in most cases. Nevertheless, if it happened for complicated paths one could increase the so-called *safety factor*†, which allowed more overlapping curves but also consumed more computational power.

4 Drawing with variable width pens

For more convenient coding, Knuth defined in [1] a high–level command based on `ddraw`†. With this mechanism, METAFONT78 could vary the pen width while drawing along a path.

```
† hpen;
† draw |a|1{0,1} ... |a'|2{1,0};
```



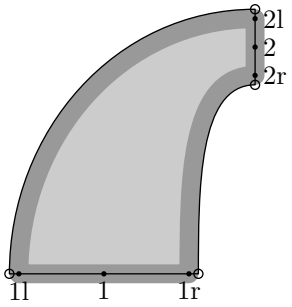
This kind of generalized drawing was only allowed for special pens. For our arc I have used an elliptical pen with fixed height b but variable horizontal width (called `hpen`†). Every drawing command with `hpen`† or its vertical counterpart `vpen`† became automatically translated to a `ddraw`† command, as detailed in [1].

The old *Computer Modern* base file at [3] indicates that Knuth gave up this approach around 1982: The obsolete `darc`† macro for drawing two connected arcs is still based on drawing with variable width pens, whereas the replacement `arc` macro directly uses `ddraw`†.

5 Filling and drawing the outlines

The whole `ddraw`† construction was somewhat cumbersome and inefficient, but there was no proper filling command available at that time in METAFONT78. The page description language *PostScript*, which supports the filling of contours, entered the community not before 1984. Knuth implemented a method to fill contours for the upcoming METAFONT84 and described the method in [4] and its implementation in [5]. Since then `ddraw`† has been

replaced by `filldraw` (filling and drawing at the same time).



As with `ddraw`†, the lines which outline our arc are shifted a bit, filled as well as drawn.

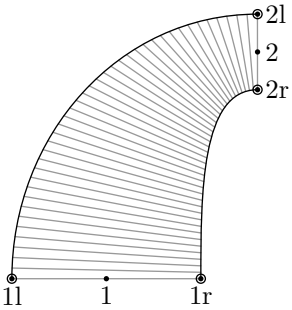
```
filldraw z1l{up} ... z2l{right}
 -- z2r{left} ... z1r{down} -- cycle;
```

The string `--` stands for straight paths and `cycle` for returning to the starting point.

6 Penstroking

For me, `penstroke` is *the* connecting link between the pen world and the outline world.

Double drawing or filldrawing with circular pens does not much depend on the size of the circular pens as long as they are much smaller than the radius of curvature. One can also make these circular pens infinitely small which alters the outlining contours to infinitely thin curves. The idea behind `penstroke` is the following: Corresponding points on the left and right part of the path are connected like a helix by straight and infinitely thin lines to fill the area in between. These lines may also be understood as pens with finite width and zero height, so-called *razor pens*.



The advantage of such penstroking over `ddraw`† and `filldraw` is that you do not have to worry about problematic pen offsets. In addition, vertices are now sharp and no longer rounded by circular pens.

Internally, `penstroke` is translated to a filling command. You can tell METAFONT to

```
penstroke z1e{up} ... z2e{right};
```

and according to [4] METAFONT automatically does

```
fill z1l{up} ... z2l{right}
 -- z2r{left} ... z1r{down} -- cycle;
```

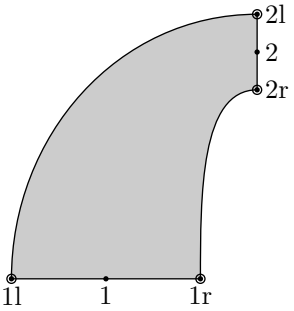
So you can *think* in pen terms and *do* outline filling at the same time, which is a big relief for programming.

7 Filling the outlines

Of course, you can directly use the more complex `fill` command:

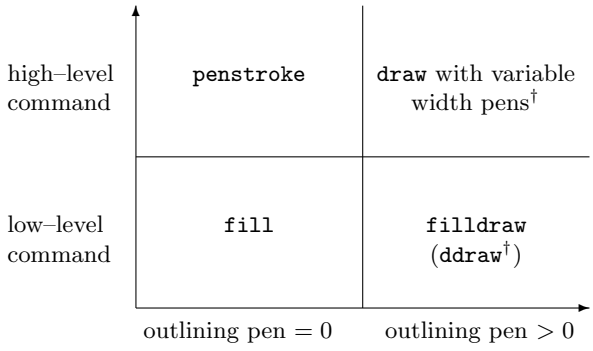
```
fill z1l{up} ... z2l{right}
 -- z2r{left} ... z1r{down} -- cycle;
```

This makes sense when you want to design outlines that can hardly be understood as penstrokes (e.g. serifs).



8 Close relatives

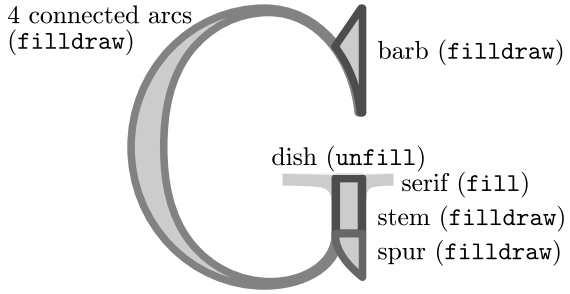
The methods that we have just seen in sections 3 through 7 are very similar to each other. I will try to point out their relationships in the following table:



9 *Computer Modern* and pens

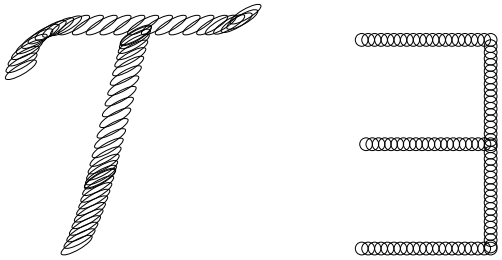
Computer Modern is the family of typefaces used by default by TEX and was first written using METAFONT78. The source was published in [2]. Knuth thereafter completely rewrote *Computer Modern* for METAFONT84. So the statement "*Computer Modern* is based only on pens" is surely true for the old sources, as METAFONT78 did not have any true filling of contours. For the current sources I claim: "*Computer Modern Roman* is a typeface based mainly on outlines. Pens are additionally used to soften vertices."

Let us look at the letter "G" of *Computer Modern Roman* to understand this claim better:



I have marked the several parts with their names and the way they are "painted" in the output. You can see that there is no standalone `draw` command used here. Note that `unfill` is just a special case of `fill` with white instead of black colour. The singular purpose of the `filldraw` command here is to make vertices rounded. In theory, this could also be solved by a more complex filling command.

Most parts of the huge *Computer Modern* family are based mainly on outlines. There are, however, also some parts which use pens exclusively:



The calligraphic uppercase letters (like the "$\mathcal{T}$" here) are for instance created from paths that are traced by a rotated elliptic pen. Mathematical symbols (like the "$\exists$" here) often make use of circular pens.

10 Conclusion

No matter whether you prefer drawn or filled paths, METAFONT can handle both. In any case the significance of METAFONT lies not in the power of pens but in its ability to parametrize fonts in large generality.

References

[1] Donald E. Knuth. *TEX and METAFONT: New directions in typesetting*. American Mathematical Society, 1979.

[2] Donald E. Knuth. *The Computer Modern family of typefaces*. Stanford University, 1980.

[3] Donald E. Knuth. cmbase.mf. `http://www.saildart.org/CMBASE.MF[MF,SYS]1`, 1982.

[4] Donald E. Knuth. *The METAFONTbook*. Addison–Wesley, 1986.

[5] Donald E. Knuth. *METAFONT: The Program*. Addison–Wesley, 1986.

◇ Linus Romer
Hirzlistrasse 3
Uznach, 8730
Switzerland
`linus.romer (at) gmx dot ch`