MetaPost with groff

John Hobby's MetaPost program had an *AT&T troff* interface which have been changed to use *GNU troff* instead.

Usage

Since MetaPost is a picture-drawing language that outputs PostScript, it is necessary to use the `-mpspic` macro package, which is automatically included when *groff*(1) is invoked with the `-Tps` option to prepare output for PostScript printers or previewers.

Suppose you have written some figures in MetaPost and placed the input in a file `figures.mp`. Running

```
mp -T figures
```

to invoke the MetaPost interpreter produces output files `figures.1`, `figures.2`, `...`

Standalone EPS pictures can be produced from such files by `fixgrofffonts`, that fixes comments according to Adobe Document Structuring Conventions (DSC), embeds font reencoding commands and non-standard fonts.

```
fixgrofffonts < figures.1 > figures1.eps
fixgrofffonts < figures.2 > figures2.eps
...
```

Before these figures can be included into *groff*, they have to be processed with the `fixmp` script: It adds an encoding vector for all text fonts back to the file (which MetaPost can't add itself) and changes the Adobe Document Structuring Conventions (DSC) comments to something *groff* can handle.

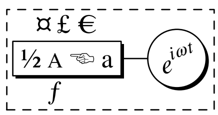
```
fixmp < figures.1 > figures.1ps
fixmp < figures.2 > figures.2ps
...
```

Now the converted files can be included in a *groff* document via macro calls such as

```
.PSPIC figures.1ps [width [height]]
```

as explained in the *grops*(1) documentation. Note that the picture gets rescaled if the height and width in the `.PSPIC` command don't match *mp*'s idea of the picture dimensions.

An example



This figure was derived from a file `grdemo.mp` and included at this point by invoking the `.PSPIC` macro (omitting the height and width parameters to avoid rescaling).

The file `grdemo.mp` looks like this:

```
verbatimtex
.EQ
delim $$
.EN
\# etex

input boxes

beginfig(1);
  pair shadowshift;
  shadowshift = (1, -1) * bp;

  def drawshadowed(text t) =
    forsuffixes $=t:
      fill bpath$ shifted shadowshift;
      unfill bpath$;
      drawboxed($);
    endfor
  enddef;

  boxit.a(btex \[12] \s8A\s+2 \[lh] a etex);
  circleit.b(btex $e sup {i omega t}$ etex rotated 20);

  b.w - a.e = (10bp, 0);

  drawshadowed(a, b);
  draw a.e .. b.w;

  label.top(btex \[Cs] \[Po] \[Eu] etex, a.n);
  label.llft(btex \[Fn] etex, a.s);

  draw bbox currentpicture dashed evenly;
endfig;

end
```

Changing GNU troff pipeline

Note that, by default, the typesetting commands in the `btex` ... `etex` blocks in the above example are processed by

```
eqn -Tps -d\$\$ | troff -Tps
```

If a different *troff* pipeline must be used it can be specified via the `TROFF` environment variable. For *groff*, the following is recommended:

```
TROFF='groff -teZ'
```

It adds *tbl* to the pipeline in addition to *eqn*, producing output for *groff*'s default 'ps' device. Note that you then have to specify the left and right delimiters of *eqn*'s in-line equations within the document (which is good practice anyway).

Macro definitions and such can be added via the standard `verbatimtex` ... `etex` mechanism that adds the given material to the *troff* input. Such material should not generate any output since this would get mixed up with the next `btex` ... `etex` block. Note the comment escape right before the first `etex` command in the above example to avoid an empty line.