Fabrice Popineau
SUPELEC –
Campus de Metz
2, rue E. Belin
F–57070 Metz
Fabrice.Popineau@supelec.fr

metapost
Practical METAPOST[1]

abstract
In this article, I will explain how to pratically use METAPOST. This program is very different from usual drawing programs, but it fits very well in a TEX based typesetting system.

résumé
Ces quelques pages ont pour objectif de montrer comment utiliser pratiquement METAPOST. Ce programme est très différent des logiciels classique de dessin, mais s'intègre extrêmement bien dans la chaîne de composition basée sur TEX.

samenvatting
De volgende pagina's laten zien hoe METAPOST praktisch kan worden gebruikt. Dit programma verschilt sterk van de klassieke tekenprogramma's, maar laat zich voortreffelijk combineren met op TEX gebaseerde tekstverwerkingssystemen.

Where can METAPOST be found?

Today, METAPOST can be found in all "complete" TEX distributions:

- TETEX for Unix,
- TEXLIVE CD-ROM for Unix and Windows,
- MIKTEX for Windows,
- OZTEX, CMACTEX and if you want to use CONTEXT under OS/X then you need TETEX.

In the following treatment, I base myself upon TEXLIVE 6 that runs under both Unix and Windows. The example graphics are taken from the METAFUN Manual by Hans Hagen (see further down).
The main program (`mpost` or `mpost.exe`) is accompanied by some auxiliary programs:

- `makempx`: a control program that extracts the text parts of your `.mp` files and converts the lower level commands to a file with `.mpx` extension. This extraction is only done when the `.mpx` file is older than the `.mp` file. Once extraction is accomplished the following two programs take care of the conversion.
- `mpto`: this program extracts the `btex ... etex` parts and the `verbatimtex ... etex` parts from your `.mp` file and places them in a TEX file.
- `dvitomp`: converts `.dvi` files to `.mpx` files.

Further programs in a basic distribution include a set of so–called "macro" files to construct diagrams and graphics based on circles and boxes (for an example see figure 1). Within the TEX Directory Structure – TDS – look for these files under `/texmf/metapost`, for documentation under `/texmf/doc/metapost`, and for many examples under `/texmf/doc/guides/metapost-examples`.

The complete set of sources, documentation, examples and further contributions can be downloaded from the Comprehensive TEX Archive Network – CTAN – to be found on the Web under `ftp://ftp.dante.de/pub/tex/graphics/metapost/`. A recent

1. First published in Cahiers GUTenberg n° 41 — Novembre 2001. Authorized translation from the French by Karel and Hanny Wesseling.

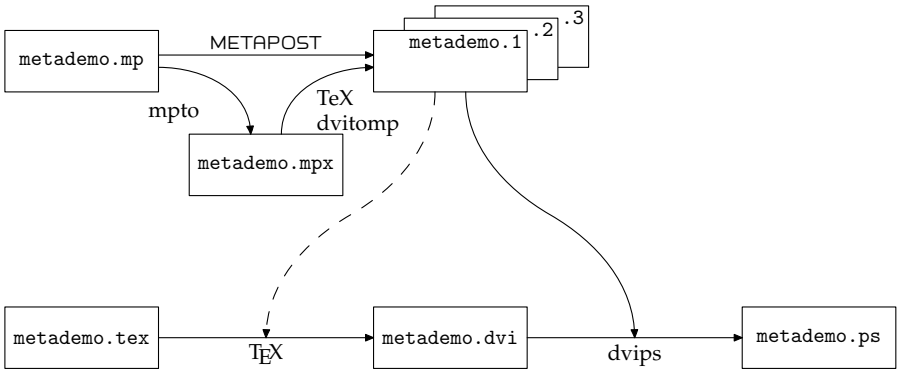


Figure 1 Compilation process of a METAPOST file

CD-ROM copy of this archive, published by Lehmanns Fachbuchhandlung is regularly distributed to members of TEX users groups without charge, but can be ordered at `bestellung@lehmanns.de`

The basics of using METAPOST with LATEX

Let us start with the two needed source documents:

`metademo.tex` containing the LATEX document, and
`metademo.mp` with the METAPOST graphics composition.

The general structure of an example METAPOST file looks like:

```
prologues:=2 ;
color c[];
c1:=red; c2:=green+red; c3:=green; c4:=blue;

def star (expr size, n, pos, color) =
  for a=0 step 360/n until 360 :
    draw (origin -- (size/2,0))
         rotatedaround (origin,a)
         shifted pos withcolor color ;
  endfor ;
enddef ;

for n = 1 upto 4:
  beginfig(n) ;
    pickup pencircle scaled 2mm ; star(2cm,n+n+3,origin,c[n]) ;
  endfig ;
endfor ;

end
```

Compilation of this file with `mpost` produces a series of four POSTSCRIPT output files, each containing one of the four declared figures (1) through (4) as shown below:

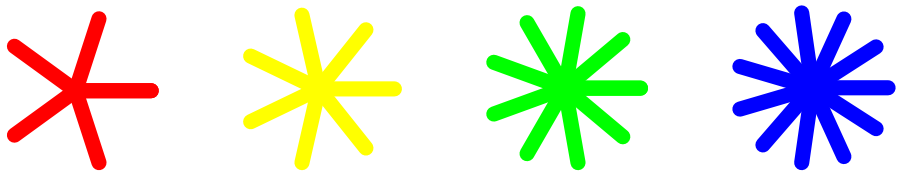
```
c:\>mpost metademo.mp
This is MetaPost, Version 0.641 (Web2c 7.3.3.1)
(metademo.mp [1] [2] [3] [4] )
4 output files written: metademo.1 .. metademo.4
Transcript written on metademo.log.
```

It is now time to include the figures just generated in your `.tex` document. This is simple when you are using a recent LATEX version in which the `graphicx` package treats the files output by METAPOST as if they were POSTSCRIPT.

For older versions you might consider renaming the files such that the file extension becomes `.eps`. For example, rename `metademo.1` to `metadem1.eps`. These files can then be handled without problems. Perhaps a better solution is to add a line `\DeclareGraphicsRule` with parameters as shown in the LaTeX document below. This line specifies to LaTeX that files included with the `\includegraphics` command that have an extension that is not recognized should be treated as PostScript files.

```
\documentclass[a4paper,11pt]{article}
\usepackage{graphicx}

\begin{document}

Some ``stars'' taken from the MetaFun manual:\\[4mm]

\includegraphics[width=1.5cm]{metademo.1} \hfill
\includegraphics[width=1.5cm]{metademo.2} \hfill
\includegraphics[width=1.5cm]{metademo.3} \hfill
\includegraphics[width=1.5cm]{metademo.4}

\end{document}
```

Now compile this file and pass it to `dvips` to obtain a PostScript file which looks like the starry result below.



In its default configuration, MetaPost does not create encapsulated, i.e. self contained, PostScript but as long as you compile to DVI and use `dvips` exclusively to obtain a PostScript output file, there is no problem. But if you want to use the PostScript generated by MetaPost with other applications such as Adobe Illustrator for example, you must include in your `.mp` file a `prologue` variable which is given the value 2, as we saw in the MetaPost program above, as follows:

```
prologues := 2 ;
```

This variable is 0 by default. A positive value forces MetaPost to generate encapsulated PostScript. Use a value of 1 for `troff`, 2 for `dvips`. With a prologue value of 2, MetaPost will look for the file `psfonts.map`, the font map file that is usually read by `dvips`. These mappings, however, are not included in the MetaPost output file. Thus, they have to be supplied together with the output, unless they already are resident in the target application. The immediate drawback of this approach is that you face problems as soon as non–ASCII character codes are used. The code vectors assumed by `dvips` will simply be ignored by other applications.

MetaPost and PDF

MetaPost maintains a particular relationship to PDF, even though it generates code that could be a priori incompatible with PDF. Fortunately, the range of instructions that MetaPost uses is a sufficiently small subset of PostScript that it is more or less “accidentally” compatible with the PDF operators.

The CONTEXT system[2] contains a module `supp-pdf` that permits the direct conversion by TEX of EPS files produced by METAPOST into PDF instructions. The advantage of this module is that it is independent of CONTEXT and can be used in other environments, for example LATEX. When used in conjunction with PDFTEX, the `graphicx` package recognizes `.mps` files containing METAPOST instructions and charges `supp-pdf` with the conversion of such graphics.
However, METAPOST produces graphic output files with extensions `.1`, `.2` and so on, so we have to tell the `graphicx` package that such files are actually `.mps`files. Therefore we always add an IF statement as follows:

```
\usepackage{ifpdf}
...
\ifpdf
  \DeclareGraphicsRule{*}{mps}{*}{*}
\else
%  Recent LaTeX versions don't require the next line
%  \DeclareGraphicsRule{*}{eps}{*}{*}
\fi
```

Thus we see that METAPOST vector graphics files *are compatible with* POSTSCRIPT *as well as* PDF *output*. The same source files serve two output formats, and there is no need to keep and maintain two file versions, one for each output type.

CONTEXT has an equally easy utility to achieve this called `mptopdf`. It is a Perl script accompanied by a reduced format file `mptopdf.efmt` that automatically includes the necessary commands when compiling METAPOST files through `texexec`, the CONTEXT work bench of many uses.

CONTEXT also offers a new format for METAPOST called MetaFun which loads its proprietary macros on top of those of METAPOST. They include flashy effects such as the one demonstrated below (figure 2): a graded background to a line of text, going from red to yellow and back to red again.

Figure 2 MetaFun's shady background.

Integration of METAPOST graphics in your source document

With `\write18`, which is part of the WEB2C system, it has become possible, finally, to include METAPOST graphics transparently in your TEX document, i.e. that graphics compilation is automatically launched by TEX at the time of the compilation of the main document. But note: for security reasons the `\write18` command is disabled by default. To make it operational you must call TEX with the option `-shell_escape`, or you could modify your `texmf.cnf` file. Look for:

```
% Enable system commands via \write18{...}?
shell_escape = t
```

CONTEXT is the first provider of an environment to fully exploit the possibilities of METAPOST graphics integration in the body of a source document. See how it is done in practice by reading the code below which produces the effect shown in figure 2 above:

2. CONTEXT, whose principal author is Hans Hagen, is a complete text composition system more modern and more ambitious than LATEX. For more information on CONTEXT look at the NTG Web site `www.ntg.nl/context/`

```
\definecolor[a][yellow]
\definecolor[c][darkred]

\startuniqueMPgraphic{CircularShade}
  path p ;
  p := unitsquare xscaled \overlaywidth
                  yscaled \overlayheight ;
  circular_shade(p,0,\MPcolor{a},\MPcolor{c}) ;
 \stopuniqueMPgraphic

\defineoverlay[circular shade]
              [\uniqueMPgraphic{CircularShade}]
\framed[background=circular shade,frame=off]
       {\bf Circular shade background}
```

A powerful aspect of CONTEXT is that it offers extensive possibilities of information exchange between TEX and METAPOST since TEX may pass parameters to METAPOST because its macros are expanded in time before the METAPOST file is generated. And using its driver modules, CONTEXT can create graphics perfectly encapsulated in typeset text composed by TEX or vice versa, as shown above with the shading example. This requires, of course, several compilation passes: the first generates the graphic definition file, the second composes the ensemble of text and graphics; between the runs metapost is called. When `write18` is enabled, only one pass is needed.

This pleasant property of CONTEXT of the fusion of figures and text in one single document has also been ported lately to LATEX thanks to the `emp` package which stands for *Embedded MetaPost*. This package can be used to produce effects similar to the example above, as demonstrated with the listing below, and the result shown in figure 3:

```
\documentclass[a4paper,11pt]{article}
\usepackage{palatino}
\usepackage{mflogo,graphicx,emp,ifpdf}

\ifpdf
  \DeclareGraphicsRule{*}{mps}{*}{}
\fi

\begin{document}

% Commands included in the Metapost file:
\empaddtoTeX{%
  \usepackage{palatino}
}

% Start of metapost figures
\begin{empfile}
  % General definitions:
  \begin{empcmds}
    color yellow  ; yellow := red + green ;
    \input mp-spec ; % or \input metafun ;
  \end{empcmds}
  % First Metapost figure:
  \begin{empdef}[fig1](5cm,5cm)
    draw fullsquare            withcolor .625red ;
    draw fullsquare rotated 45 withcolor .625red ;
    picture cp ; cp := currentpicture ;
    def copy = addto currentpicture also cp enddef ;
    copy scaled .9 withcolor .625white ;
```

```
      copy scaled .7 withcolor .625yellow ;
      copy scaled .6 withcolor .625white ;
      copy scaled .4 withcolor .625red ;
      copy scaled .3 withcolor .625white ;
      fill fullcircle scaled .2 withcolor .625yellow ;
      currentpicture := currentpicture scaled 50 ;
    \end{empdef}
    % (more figure can follow)
\end{empfile}
% On-the-fly metapost compilation:
\immediate\write18{mpost -tex=latex \jobname}

Example of a \MP\ figure included in the body of a \LaTeX\ document.
\begin{figure}[ht]
  \begin{center}
    \empuse{fig1}
    \caption{A \MP{} graphic embedded in a \LaTeX\ document}
  \end{center}
\end{figure}

\end{document}
```

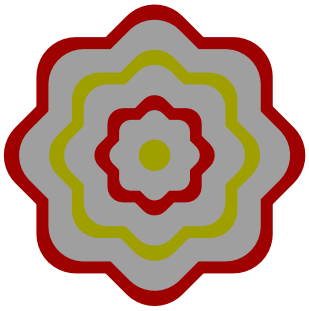


Figure 3 A METAPOST graphic embedded in a LATEX document.

Unfortunately, and unlike in CONTEXT, the `emp` package does not allow the exchange of parameters between TEX and METAPOST.

Conclusion

I hope that with this publication I have eliminated most of the practical difficulties that one may meet initially when attempting to use METAPOST. I also hope to have induced you to undergo the joys of geometry graphics. Bonne chance. Good luck.