Xdvipsk: Dvips ready for OpenType fonts and more image formats

Sigitas Tolušis, Arūnas Povilaitis and Valentinas Kriaučiukas

Abstract

We present two extensions to `dvips`. One allows flexible inclusion of bitmap images and was implemented on top of the FreeImage library. The second extension solves quite a long-standing task: adding OpenType font support to `dvips`. Our extended `dvips`, `xdvipsk`, goes the "LuaTEX way" in OpenType font management: it works on DVI files compiled by LuaTEX and expects to find the necessary Unicode map files, obtained as by-products of the compilation. The providing of these map files is ensured by a special LATEX package.

1 Motivation and history

The Dvips(k) page [5] says that "it would be great to add OpenType and perhaps TrueType support to dvips".

We had our own motivation too. We saw that complete elimination of the PostScript stage from our publishing workflow LATEX→PDF would be either very costly or almost impossible, for different reasons. Among them are requirements to produce web-optimized PDF, use of Adobe's Acrobat Distiller and of other local utilities built into the workflow going through PostScript.

Some `dvips` shortcomings, such as restricted support of graphic formats, can be quite easily compensated for by graphics preprocessing tools. An attempt to use OpenType fonts meets bigger problems. It is possible to transform one OT font to many new Type1 fonts, but then one needs to introduce the new fonts in TEX styles and ensure their correct use in LATEX texts. As a side effect of this, all advantages of OpenType fonts are lost. Plus, these steps are painful, so the wish to avoid them by extending `dvips` is natural.

The locally used versions of `dvips` were modified long ago (before 2000), but behavior modifications were not deep, like ignoring unknown and recognizing private specials, and writing a log file. As the `dvips` program was not actively developed at that time, the local patching to new versions of `dvips` was an easy task.

About five years ago, more advanced handling of graphic files was implemented, based on the FreeImage [6] library. Standard `dvips` mainly works with EPS files only, so all non-EPS graphics had to be converted into EPS format. It allows restricted use of bitmap images (BMP, PCX, PICT formats, no scaling or rotation), but this is not exposed in the main documentation [8]. The extended `dvips` now accepts BMP, PCX, TIFF, JPEG, PNG formats and performs the same actions as with EPS: scaling, rotating, trim, viewport (but the `graphics` package does not yet implement the operations of clipping, trimming and viewport).

The work on providing OpenType font support started about a year ago, when possible components were tested; later they were connected into a working chain. The current stage of `xdvipsk` development can probably be characterized as beta.

The program name `xdvipsk` starts with 'x' denoting the Unicode (OT fonts) extension and ends with 'k' denoting use of the Kpathsea library (as with `dvipsk`). The standard Kpathsea library (from TEX Live) does not work with our main development environment on MS Windows (Visual Studio), so it was separately compiled for `xdvipsk`.

2 New options

New features of the extended `dvips` can be switched on or off using new command-line options. The summary of the options, presented in Fig. 1, is output when `xdvipsk` is called with no arguments or with the standard `--help` option. For more convenient review, in this presentation the new options are printed in frame-boxes. A more detailed description of the new options, as included in the documentation (again differing only in formatting details) is as follows:

`-g` Mode of logging into the file named ⟨*dvi file name*⟩`.xdvips.log`; default on. For a successful run, the log file contains only the message `!!!Success!!!`.

`-H` 32-bit turbo mode for inclusion of PostScript graphics (writes EPS files directly to PS file) using 10 MB dynamic buffer; default off.

`-I`⟨*pixel-form filters*⟩ Resizing mode for bitmap images included with `em: graph` specials; default off. ⟨*pixel-form filters*⟩ is a comma-separated tuple of up to four pairs ⟨*pixel-form*⟩:⟨*filter*⟩, where ⟨*pixel-form*⟩ can be one of

`BW:` black/white 1-bit pixels,
`GR:` gray 8-bit pixels,
`RGB:` colored 24-bit pixels,
`CMYK:` colored 32-bit pixels,

and ⟨*filter*⟩ can be one of the following:

`b:` box filter,
`t:` bilinear filter,
`B:` B-spline filter,
`m:` Mitchell–Netravali bicubic filter,

```
Usage: dvips [OPTION]... FILENAME[.dvi]
Convert DVI input files to PostScript.
Options:
-a*  Conserve memory, not time       -A   Print only odd (TeX) pages
-b # Page copies, for posters e.g.   -B   Print only even (TeX) pages
-c # Uncollated copies               -C # Collated copies
-d # Debugging                       -D # Resolution
-e # Maxdrift value                  -E*  Try to create EPSF
-f*  Run as filter                   -F*  Send control-D at end
-g*  write log file                  -G*  Shift low chars to higher pos.
-h f Add header file                 -H*  Turbo mode for PS graphics
-i*  Separate file per section       -I*  Resize mode for emTeX graphics
-j*  Download T1 fonts partially     -J*  Download OpenType fonts partially
-k*  Print crop marks                -K*  Pull comments from inclusions
-l # Last page                       -L*  Last special papersize wins
-m*  Manual feed                     -M*  Don't make fonts
-mode s Metafont device name
-n # Maximum number of pages        -N*  No structured comments
-noomega  Disable Omega extensions
-noptex   Disable pTeX extensions
-noluatex   Disable LuaTeX extensions
-noToUnicode   Disable ToUnicode CMap file generation for OpenType fonts
-o f Output file                     -O c Set/change paper offset
-p # First page                      -P s Load config.$s
-pp l Print only pages listed
-q*  Run quietly                     -Q*  Skip VTeX private specials
-r*  Reverse order of pages          -R*  Run securely
-s*  Enclose output in save/restore  -S # Max section size in pages
-t s Paper format                    -T c Specify desired page size
-u s PS mapfile                      -U*  Disable string param trick
-v   Print version number and quit   -V*  Send downloadable PS fonts as PK
                                     -W*  Extended search for emTeX graphics
-x # Override dvi magnification      -X # Horizontal resolution
-y # Multiply by dvi magnification   -Y # Vertical resolution
-z*  Hyper PS                        -Z*  Compress bitmap fonts
    # = number   f = file   s = string  * = suffix, '0' to turn off
    c = comma-separated dimension pair (e.g., 3.2in,-32.1cm)
    l = comma-separated list of page ranges (e.g., 1-4,7-9)
```

Figure 1: Xdvipsk option summary with new options indicated.

`l`: Lanczos-windowed sinc filter,
`c`: Catmull–Rom and Overhauser splines,
`r`: resample image (remove rows and columns in the bitmap),
`w`*i*: MS Windows GDI filter, where $i = 1, 2, 3, 4$ means modes BLACKONWHITE, WHITEONBLACK, COLORONCOLOR and HALFTONE, respectively.

Not all ⟨*pixel form*⟩:⟨*filter*⟩ combinations are possible:

- filters `w`*i* can be used on MS Windows systems only and just for BW, GR, and RGB pixel forms; for CMYK, any `w`*i* filter is replaced by the `r` filter;
- on Linux and other systems, filters `w`*i* are also changed to `r` filter;
- for monochrome graphics, only filters `r` and `w`*i* are applicable.

`-I` (without filters) Resizing mode with the following filter tuples:
`BW:w1,GR:w3,RGB:w3,CMYK:r` on Windows;
`BW:r,GR:r,RGB:r,CMYK:r` on other systems.

`-j` Type 1 fonts partial download; default off (contrary to dvips).

`-J` Download only needed characters from OT fonts; default on.

`-noluatex` Disable LuaTEX extensions and support of OpenType fonts.

`-noToUnicode` Omit generation of map (to Unicode) files for OT fonts, which can be used by Acrobat Distiller to enable text search; default on.

`-Q` Mode of skipping VTeX specials: any content of `\special` commands prefixed with `mt:`, `vtex:`, `MC:`, `BMC:` or `EMC:` is silently ignored; default off.

`-W` Extended search mode for image files indicated by `em: graph` specials: when no file with the specified name is found, the file names with other extensions (`.pcx`, `.bmp`, `.tif`, `.jpg`, `.png`) are tried; default off.

3 Extension for graphics

The extension for bitmap images does not require changes to the user-level syntax; the LaTeX command `\includegraphics` should work as described in the documentation of `graphics` and `graphicx` [2]; that is, after inclusion in the preamble of either
`\usepackage[`⟨*driver*⟩`]{graphics}`
or
`\usepackage[`⟨*driver*⟩`]{graphicx}`
where the file ⟨*driver*⟩`.def` contains all the necessary declarations and is registered in `graphics.sty` (examples can be found in the presentation [10]). As `xdvipsk` accepts images in formats BMP, JPEG, PCX, PNG, and TIFF, they should all be declared in the form of graphic inclusion rules in the driver file, most likely `dvips.def`:

```
\@namedef{Gin@rule@.tif}#1{{bmp}{.tif.bb}{#1}}
\@namedef{Gin@rule@.tiff}#1{{bmp}{.tiff.bb}{#1}}
\@namedef{Gin@rule@.jpeg}#1{{bmp}{.jpeg.bb}{#1}}
\@namedef{Gin@rule@.jpg}#1{{bmp}{.jpg.bb}{#1}}
\@namedef{Gin@rule@.png}#1{{bmp}{.png.bb}{#1}}
```

Several things for authors of TeX packages and papers to know:

- Bitmap image file names are included in DVI files inside arguments of `\special` commands with prefix `em:graph` (the name has roots in the time of the EmTeX distribution).
- Bitmaps can be of different color models: BW, gray, RGB, CMYK, indexed RGB.
- The program ignores the content of `\special` commands with unknown prefixes.
- For more precise image positioning, Xdvipsk inserts the PostScript `HiResBoundingBox`.

4 How Xdvipsk works with OpenType fonts

Our solution comes from the decision to use OpenType font information directly, as with LuaTeX and the `luaotfload` package [9]. The current version of `luaotfload` operates with only one writable cache, which incorporates file paths specific to an OS, which for us is inconvenient. Our production environment contains different operating systems: Linux servers, Linux and Windows workstations, and a shared TeX-tree resource with multiple TeX Live versions on a Linux server, accessible by Windows clients through the local network. We wanted to have things as flexible as possible in presence of different OSes.

Some additional tools are necessary for `xdvipsk` to work:

1. PostScript header file `texcid.pro` is used for inclusion of OpenType fonts in PostScript files. It is an analogue of `texps.pro` that is used in case of Type 1 fonts.
2. A LaTeX package `luafonts`, which is just an interface to Lua code generating two additional maps. It is loaded like any other LaTeX package:

   `\usepackage{luafonts}`

One map generated by the package at compilation time is analogous to `psfonts.map` and contains information about OpenType fonts used in a particular article. The map format is as follows:

⟨*tfm name*⟩␣⟨*ps name*⟩␣⟨*texfont name*⟩␣><⟨*file name*⟩

where ⟨*tfm name*⟩ is the same as what is written in the DVI file by LuaTeX, and ⟨*ps name*⟩ and ⟨*file name*⟩ come from `luaotfload` Lua tables. This ⟨*ps name*⟩ is a PostScript font name and ⟨*texfont name*⟩ is an internal font name seen by `luaotfload` as `fullname`. ⟨*file name*⟩ is modified so that the directory prefix, corresponding to the actual TeX tree used, is replaced by variable `$SELFAUTOPARENT`. Examples of ⟨*tfm name*⟩, ⟨*file name*⟩ and ⟨*ps name*⟩ are given, respectively, in Figs. 2, 3 and 4.

Another map generated by `luafonts` stores information about characters. It consists of triples ⟨*internal tex character code*⟩, ⟨*opentype font glyph index*⟩, ⟨*unicode equivalent*⟩; examples are in Fig. 5.

These maps are used by Xdvipsk (a) to find CIDs (character identifiers [1]) for insertion in PS files and (b) to prepare TOUNICODE cmaps [3], like the one shown in Fig. 6. They are needed for searching in PDF files. A utility `make2unc` was created to incorporate TOUNICODE cmaps for PDF.

4.1 Process in steps

Step 1. Run `dvilualatex` ⟨*article*⟩`.tex` where file ⟨*article*⟩`.tex` uses `luafonts`:

Input: ⟨*article*⟩`.tex`
`tex/luatex/luafonts/luafonts.sty`
`tex/luatex/luafonts/luafonts.lua`
`...`

Output: ⟨*article*⟩`.dvi`
`.xdvipsk/`⟨*ps name*⟩`.encodings.map`
`...`
`.xdvipsk/`⟨*article*⟩`.opentype.map`

Step 2. Run `xdvipsk` ⟨*article*⟩`.dvi`:

Input: ⟨*article*⟩`.dvi`
`.xdvipsk/`⟨*ps name*⟩`.encodings.map`
`...`
`.xdvipsk/`⟨*article*⟩`.opentype.map`
`texmf-dist/dvips/base/texcid.pro`
`...`

```
FandolFang-Regular
FandolFang-Regular:mode=node;script=latn;language=DFLT;+tlig;
TeXGyreAdventor
TeXGyreAdventor/B
TeXGyreAdventor/BI
TeXGyreAdventor/I
TeXGyreAdventor:mode=node;script=latn;language=DFLT;+pnum;+onum;
[lmroman10-bold]:+tlig;
[lmroman10-italic]:+tlig;
[lmroman10-regular]:+tlig;
```

Figure 2: Examples of ⟨*tfm name*⟩.

```
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/fandol/FandolFang-Regular.otf
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/tex-gyre/texgyreadventor-regular.otf
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/tex-gyre/texgyreadventor-bold.otf
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/tex-gyre/texgyreadventor-bolditalic.otf
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/tex-gyre/texgyreadventor-italic.otf
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/tex-gyre/texgyreadventor-regular.otf
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/lm/lmroman10-bold.otf
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/lm/lmroman10-italic.otf
>$SELFAUTOPARENT/texmf-dist/fonts/opentype/public/lm/lmroman10-regular.otf
```

Figure 3: Examples of ⟨*file name*⟩.

```
FandolFang-Regular
TeXGyreAdventor-Regular
TeXGyreAdventor-Bold
TeXGyreAdventor-BoldItalic
TeXGyreAdventor-Italic
TeXGyreAdventor-Regular
LMRoman10-Bold
LMRoman10-Italic
LMRoman10-Regular
```

Figure 4: Examples of ⟨*ps name*⟩.

```
59964,707,00AF
59965,708,00AF
59966,709,00200331
59967,710,0304
59968,711,02DA
59969,712,0020030A0301
59970,713,0020030A0301
59971,714,030A
59972,715,02DC
```

Figure 5: An excerpt from a map, specifying TEX characters' OpenType glyph indices and Unicode equivalence codes.

Output: ⟨*article*⟩`.ps`
`...`
`.xdvipsk/`⟨*article*⟩`-cid`⟨*num*⟩`.tounicode`
`...`

where ⟨*num*⟩ is a font index in the DVI file and is used here to have distinct file names.

Step 3. Convert the PostScript file to PDF (using Ghostscript, Acrobat or any other tool):

Input: ⟨*article*⟩`.ps`
Output: ⟨*article*⟩`.pdf`

Step 4. Add TOUNICODE cmaps to the PDF file using `make2unc` utility:

Input: ⟨*article*⟩`.pdf`
`...`
`.xdvipsk/`⟨*article*⟩`-cid`⟨*num*⟩`.tounicode`
`...`
Output: ⟨*article*⟩`.pdf` (searchable)

5 Development environment

At present, we use a rather split and mixed environment, compared with the TEX Live build ecosystem. As mentioned above, the main development and building of executables is done on a MS Windows workstation using the Visual Studio 2013 IDE.

In parallel, we build the code on two more architectures: Linux and Mac OS X. For these, we are quite close to the TEX Live build environment except for prebuilt architecture-dependent versions of `tiff`, `lzma` and `jbig` libraries for `xdvipsk` and MuPDF [7] library for `make2unc`. There is no doubt that this is easier than incorporating the mentioned libraries into the TEX Live build ecosystem in the proper way. It allowed us to provide, with minimum effort, our time-limited solution for incorporating OpenType fonts into a Dvips-based workflow.

```
/CIDInit /ProcSet findresource begin
12 dict begin
begincmap
/CMapName /t6-cid002 def
/CMapType 2 def
/CIDSystemInfo <<
  /Registry (TeX)
  /Ordering (BHCDARZO+002)
  /Supplement 0
>> def
1 begincodespacerange
<0000> <FFFF>
endcodespacerange
24 beginbfchar
<001D> <0061>
<0024> <0062>
<002E> <002C>
<0030> <0064>
<0033> <0065>
<0042> <0049>
<0043> <0069>
<0049> <006C>
<004C> <006D>
<004E> <006E>
...
<040B> <0037>
<040E> <0036>
<0415> <0033>
<0419> <0032>
endbfchar
endcmap
CMapName currentdict /CMap defineresource pop
end
end
```

Figure 6: An example of a TOUNICODE map.

Other needed libraries are taken from TEX Live distributions. The current `xdvipsk` version is based on `dvips` 5.996, `web2c+kpathsea` 6.22, TEX Live 2016.06.07, `jpeglib` 9b, `libpng` 1.62, `libtiff` 4.06, `zlib` 1.2.8.

Comparing with the `dvips` source, the changes in the code structure are the following:

- New modules:

  `charcode.c` `emspecialex.c`
  `luamap.c` `sfntload.c`
  `utarray.h` `uthash.h`
  `writecid.c`

  and `texcid.lpro` — a PostScript procset with comments.
- Removed modules:

  `emspecial.c`
- Changes made in 22 modules. All changes are tagged with markers:

  ```
  //AP--begin
  //AP--end
  ```
- New directories:

  `graflib:` simplified and adapted code from the FreeImage [6] library;
  `otflib:` adapted code from `dvipdfmx` [4].

6 Availability

The source code is available from `https://github.com/vtex-soft/texlive.xdvipsk`.

References

[1] Adobe Systems Inc. *Adobe CMap and CIDFont Files Specification*, June 1993. Version 1.0. `https://www.adobe.com/content/dam/Adobe/en/devnet/font/pdfs/5014.CIDFont_Spec.pdf`.

[2] David P. Carlisle. *Packages in the 'graphics' bundle*. The LATEX3 Project, December 2016. `http://mirrors.ctan.org/macros/latex/required/graphics/grfguide.pdf`.

[3] Cmap — character to glyph index mapping table. `https://www.microsoft.com/typography/otspec/cmap.htm`. Visited 2017-04-10.

[4] Dvipdfmx — an extended version of dvipdfm. `https://ctan.org/pkg/dvipdfmx`, March 2010.

[5] Dvips(k). `https://tug.org/dvips/`, April 2017.

[6] The FreeImage project. `http://freeimage.sourceforge.net/`, April 2017.

[7] MuPDF. `http://mupdf.com/`.

[8] Tomas Rokicki. *Dvips: A DVI-to-PostScript Translator*, April 2016. `https://tug.org/texlive/Contents/live/texmf-dist/doc/dvips/dvips.pdf`.

[9] Elie Roux, Khaled Hosny, and Philipp Gesang. *The luaotfload package*, January 2017. `https://ctan.org/pkg/luaotfload`.

[10] Sigitas Tolušis, Arūnas Povilaitis, and Valentinas Kriaučiukas. Xdvipsk: dvips ready for opentype fonts and more image formats. `http://www.gust.org.pl/bachotex/2017-pl/presentations/stolusis-etal-1-2017.pdf`, May 2017. Slides of presentation at TUG@BachoTEX 2017.

◇ Sigitas Tolušis
VTeX, Mokslininkų 2a, Vilnius
LT-08412, Lithuania
`sigitas (at) vtex dot lt`

◇ Arūnas Povilaitis
VTeX, Mokslininkų 2a, Vilnius
LT-08412, Lithuania
`arunasp (at) vtex dot lt`

◇ Valentinas Kriaučiukas
VTeX, Mokslininkų 2a, Vilnius
LT-08412, Lithuania
`valius (at) vtex dot lt`