The bmeps program and library, version 1.2.4

Dipl.-Ing. D. Krause

March 22, 2005

Contents

1 Overview

The “bmeps” package contains a library and a command line tool to convert PNG and other images to EPS. It is intended to be used together with LaTeX and dvips. Using the bmeps package dvips can convert images to EPS “on the fly”, it is not longer necessary to convert the files explicitly.

Different EPS features can be used depending on the selected PS level. PS level 2 enables color output and ASCII85 output encoding instead of ASCII-Hex-encoding. PS level 3 enables flate compression. PS level and EPS features can be specified when dvips or bmeps is run.

2 License

The software in this package – except the files and directories mentioned in the paragraph below – is licensed to you under a BSD style license as follows:

- Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  - Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  - Redistributions in binary form must reproduce the above opyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  - Neither the name of the Dirk Krause nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

- This software is provided by the copyright holders and contributors “as is” and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed.
  In no event shall the copyright owner or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

Exceptions

- dvips-mods subdirectory
  The files in the dvips-mods subdirectory are derived from the original dvips sources and remain under the same license conditions and copyright statements as the original sources. The current version of dvips – the version you should use to build a modified dvips – can be found in the teTeX distribution[1], the file is tetex-src.tar.gz. After unpacking the archive you will find a directory tetex-src-2.0.2/texk/dvipsk (the version number may vary), see the sources and documentation in this directory for copyright and license details.

- contrib/kant_krishna directory and DOCU/kant_krishna/excel_to_eps.txt file
  The file excel_to_eps.vb was provided by Krishna Kant. No copyright and/or license terms were specified for this file. You should leave the Author:...-line intact when distributing this file.

[1] ftp://ftp.dante.de/tex-archive/systems/unix/teTeX/current/distrib

3 Installation

3.1 Prerequisitions

The following software packages should be installed before you start to install bmeps:

- **zlib** (required)
  A general purpose compression library[2].
- **libpng** (required)
  A library for dealing with PNG images[3].
- **jpeglib** (optional)
  The Independent JPEG Group's Free Software to handle JPEG files[4]. After "make install" copy all *.h-files to /.../include and the *.a-files to /.../lib.
- **NetPBM Tools** (optional)
  Tools and libraries for converting bitmap images[5].

Install the packages in the order given here and make sure to set CFLAGS and LDFLAGS.

3.2 Installation procedure

Unpack the distribution and change into the bmeps directory. Run

```
./configure
make
make install
```

to build and install the software.
A second makefile "Makefile-shared-linux" is created. This can be used on Linux systems to build a shared library and a binary using this shared library.

```
./configure
make -f Makefile-shared-linux
make -f Makefile-shared-linux install
```

Note: Dealing with shared libraries is not recommended to the wide majority of users, only experienced users, developers and people creating binary installation

[2] http://www.gzip.org/zlib

[3] http://www.libpng.org/pub/png/libpng.html

[4] ftp://ftp.uu.net/graphics/jpeg/

[5] ftp://ftp.metalab.unc.edu/pub/Linux/apps/graphics/convert/

packages for Linux distributions and LaTeX distributions should do this. The directory containing the libbmeps.so library must be added to the LD_LIBRARY_PATH environment variable or listed in the /etc/ld.so.conf file. For details about placing shared libraries and configuring search paths consult you systems manuals.

3.3 Dvips modification

3.3.1 Overview

There are two ways to convert PNG and JPEG images to EPS when dvips is running:

- Run the bmeps program. This option is available immediately after the bmeps package is installed and requires no further work.
- Invoke conversion routines directly within dvips. This requires to build a modified dvips. The steps to do so are shown below.

3.3.2 Getting the sources

To build dvips you need the sources for two further packages:

- The kpathsea library. This library is used by dvips to search for files.
- The dvips program itself.

These packages are available at CTAN[6], download “tetex-src.tar.gz”.

This archive contains the sources for the entire teTeX distribution, we need to deal only with kpathsea and dvipsk in tetex/texk.

3.3.3 Build and install kpathsea

In the kpathsea directory type

```
./configure
make
make install
```

If you have downloaded the “tetex-src.tar.gz” archive, run

```
./configure
```

in the tetex directory and

```
make
make install
```

in the tetex/texk/kpathsea directory.

[6] ftp://ftp.dante.de/tex-archive/systems/unix/teTeX/current/distrib/

3.3.4 Build the unmodified dvips

First we need to make sure that we are able to build an unmodified dvips. To do so run

```
./configure
make
```

in the dvipsk directory. If you have downloaded the “tetex-src.tar.gz” archive it is not necessary to run configure again, running make in the tetex/texk/dvipsk directory is sufficient.

3.3.5 Create a backup of dvips

Type

```
which dvips
```

to see where your dvips resides. Create a backup copy of the existing dvips and name it “dvips.original”. The copy should get the same permissions as the original.

3.3.6 Build the modified dvips

Change into the dvipsk directory. Copy the files from . . . /bmeps/dvips-mods into the dvipsk directory.
Run

```
./configure
make
cp dvips `which dvips`
```

to build and install the modified dvips. If you specified options for ./configure when building kpathsea you should specify the same options to ./configure here.
Type

```
dvips --version
```

to verify proper installation. The output should contain a line like

```
dvips(k) 5.86 modified for bitmap graphics support
```

showing that a modified dvips is in use.

4 Usage

4.1 Bounding box creation

LaTeX needs bounding box information for all graphics to include. Run a shell script like the following to create bounding box information files for all PNGs in a directory:

```
#! /bin/csh
foreach i (*.png)
 set j = $i:r
 set j = "${j}.bb"
 bmeps -b $i $j
end
```

To create bounding box information for a single PNG file use bmeps as follows:

```
bmeps -b x.png x.bb
```

4.2 Register file type

In your LaTeX document's preamble write

```
\DeclareGraphicsRule{.png}{eps}{.bb}{`bmeps #1}
```

This tells LaTeX that files having suffix .png can be converted into EPS file format. Bounding box information is expected in a file having the same filename but the suffix .bb. When the .dvi file is processed by dvips the program bmeps is invoked to convert the file to EPS.

If you have build a modified dvips – I strongly recommend to do so – you should specify

```
\DeclareGraphicsRule{.png}{eps}{.bb}{}
```

instead. The dvips program now invokes the conversion routines from the bmeps library directly without running external programs.

If you want to use your LaTeX source with both LaTeX/dvips and pdflatex write

```
\usepackage{ifpdf}
\ifpdf
...
\else
...
\DeclareGraphicsRule{.png}{eps}{.bb}{`bmeps #1}
\fi
```

respectively

```
\usepackage{ifpdf}
\ifpdf
...
\else
...
\DeclareGraphicsRule{.png}{eps}{.bb}{}
\fi
```

instead.

4.3 Setting up PS output

4.3.1 The EPSOUTPUT environment variable

PS level and PS features can be configured via the EPSOUTPUT environment variable. The following settings can be made here:

- **Verbose mode**
  Use “V” to turn verbose mode on. By default it is turned off. If bmeps encounters an error when reading input files it writes an error message as PostScript comment to the output.
  In verbose mode the error message is also written to the standard error output.

- **PS level**
  Use “1”, “2” or “3” to specify the PS level.

- **Colored/grayscaled output**
  Use “c” to obtain colored output, “g” for grayscaled output.

- **showpage operator**
  By default bmeps does not print a showpage operator at the end of the output file (version 1.0.9 and above).
  The showpage operator can be printed using the “h” character.

- **PostScript memory managment**
  Bmeps can install a separated dictionary using the “u” character.
  The “r” character can be used to add a “1 vmreclaim” instruction.

- **DSC comments**
  By default bmeps does not issue DSC comments (except “%!PS-Adobe...”). Earlier versions (before 1.0.2) of bmeps had a bug, DSC comments were not written in correct order. Recent versions of GhostScript/GhostView complained about this. Comment order is corrected now, additionally DSC comment output is disabled now by default. Use “x” to enable DSC comments.

- **Encoding and compression**
  If you have selected at least PS level 2 you can specify “8” to use ASCII85-encoding instead of ASCII-Hex-encoding. Furthermore you can specify “r” for run-length-compression. If PS level 3 is used you can specify “f” to use flate compression. Multiple encoding algorithms can be combined.

- **Exact resolution**
  Specify “e” to use the resolution information found in the PNG file’s pHYs

chunk. This option should not be used to produce images for inclusion in LaTeX documents or DTP applications. It is intended for creating EPS files for standalone viewing.

- **Draft mode**
  If you specify “d” for draft mode only placeholders are printed instead of converted pictures.

- **Alpha channel usage**
  When PS level 3 is in use you can convert an alpha channel into a clipping mask. To do so specify “a”. If the alpha channel expresses opacity (default) specify “o”, if it expresses transparency specify “t”. Normally only pixels with full transparency are masked out. To change this so that all pixels with transparency not 0 are masked out specify “l”.
  If you want to mix foreground and background color depending on the alpha channel value specify “m”. A triple of numbers (i.e. “128,255,128” for light green) can be used to specify a default background color. The background color specification must appear after the “a” for alpha channel settings. If you want your background color specification to superseed the background chunk contained in the file, specify “s”.

Examples:

- In a C-Shell (or a derivate) use

  ```
  setenv EPSOUTPUT 2gr8
  ```

  to set up EPS export for PS level 2, grayscaled output, run-length-compression and ASCII85-encoding. Most PS level 2 printers should be able to print this.

- In a Bourne-Shell (or a derivate) use

  ```
  EPSOUTPUT=3crf8
  export EPSOUTPUT
  ```

  to get PS level 3, colored output, run-length and flate compression and ASCII85-encoding. This will produce output suitable as input for ps2pdf.

- On an DOS-prompt on Windows type

  ```
  set EPSOUTPUT=3crf8ams128,255,128
  ```

  to convert a transparancy alpha channel to a masking bitmap. Depending on the alpha channels value the foreground picture of each pixel is mixed against a lightgreen background. The specified background color is used, a background chunk in the image file is ignored.

- In a C-Shell you can use

  ```
  setenv EPSOUTPUT d
  ```

  to produce drafts only.

4.3.2 Command line options

The EPSOUTPUT environment variable is a convenient way to set up default options.
These defaults can be overwritten by command line options:

- -b
  Generate a bounding box only. No other options should be specified together with this option.
- -d
  Draft mode, generate a symbol only. No other options should be specified together with this option.
- -V
  Enables verbose output.
- -p *level*
  Set the PS level, "1", "2" or "3".
- -c
  Colored printing, requires PS level 2.
- -g
  Grayscaled printing. You can specify either "-c" or "-g".
- -e *encoding*
  Specify compression and encoding mechanisms, the string consists of the following key characters:
  - "8"
    Use ASCII-85-encoding. Requires PS level 2.
  - "r"
    Use run-length compression. Requires PS level 2.
  - "f"
    Use flate compression. Requires PS level 23.

- -a *alpha-options*
  Specify how to handle the alpha channel. The string consists of the following entries:
  - o
    Treat alpha channel data as opacity (default).
  - t
    Treat alpha channel data as transparency. You can use either “o” or “t”, not both.
  - l
    Use alternative trigger level for image mask creation. By default only pixels with no opacity at all are masked out. This option masks out all pixels except those with full opacity.
  - m
    Mix image against white background or a specified color (see below).
  - *rgb-triple*
    Specify background color as RGB-triple, i.e. “128,255,128” for light green.
  - s
    Give the user-specified background color (see entry above) a higher priority than the color specified in the PNG file’s background chunk.
- -t *filetype*
  Specify the source file type if the program is running as a pipe or unusual file names are in use. Use either “png”, “jpg” or “pnm”.
- -s
  Print DSC comments.
- -o
  Print a showpage operator at the end of the output.
- -u
  Undefine “/pstr” and “/inputf” to allow the garbage collection to release the memory.
- -r
  Force garbage collection at the end of file via “1 vmreclaim”.
- -q
  Use resolution information from the PNG file’s pHYs chunk, if available.

4.4 Specifying dvips command line options

If you have build a modified dvips you can overwrite the information from EPSOUTPUT by command line options. Use “**-I** *option*” like

```
dvips -I 2gr8 ...
```

to specifiy EPS output options.

4.5 Recommended options for typical usage scenarios

4.5.1 PS level 2 printers

If you plan to use the EPS images in a document to be printed on a printer capable of PS level 2 you should use the bmeps options

```
-p2 -g -e8r -u
-p2 -c -e8r -u
```

for grayscaled/colored printing. For the EPSOUTPUT variable use:

```
2g8ru
2c8ru
```

For dvips use the option:

```
-I 2g8ru
-I 2c8ru
```

4.5.2 PS level 3 for distilling

If you plan to use the EPS images in a document to be distilled or passed to Ghostscript to produce a PDF you should use the bmeps options

```
-p3 -c -e8rf -u -r
```

For the EPSOUTPUT variable use:

```
3c8rfu
```

For dvips use the option:

```
-I 3c8rfu
```

4.6 Examples

The EXAMPLES subdirectory contains some PS files derived from the same example.tex file using different dvips options. The picture “stefan_255_rgba.png” was obtained from the
MISCELLANEOUS TRANSPARENT PNGS USING IMAGE TAGS page on the libpng homepage [7]. It was provided courtesy of Stefan Schneider.

Some files (names start with “bg…”) have a page background color.

I recommend to use “gv” [8] on UNIX rather than “ghostview” to view the examples. It needs the “Xaw3d” library [9].

- The file “1g.ps” was produced by:

  ```
   latex example
   dvips -I 1g example
  mv example.ps 1g.ps
  ```

  It is a PS level 1 file containing a grayscaled picture.

- File “2g8r.ps” was produced using options “2g8r” and contains the same grayscaled picture run-length-compressed and ASCII85-encoded.

- File “2c8r.ps” was produced using options “2c8r” and contains a colored version of the picture, run-length-compressed and ASCII85-encoded.

- File “3c8rf.ps” was produced using options “3c8rf”. Flate compressions was used additionaly.

- File “bg3c8rf.ps” was produced using options “3c8rf”. It has a cyan page background.

- File “bg3c8rfa.ps” was produced using options “3c8rfa”. The alpha channel in the picture was converted into an image mask. All pixels having at least some opacity are drawn in the color specified by the file, pixels with 0 opacity are skipped.

- File “bg3c8rfam.ps” was produced using options “3c8rfam”. For each pixel the color is calculated from the foreground color specified in the file and the background color from the background chunk in the file depending on the

[7] ftp://ftp.freesoftware.com/pub/png/index.html
[8] http://wwwthep.physik.uni-mainz.de/~plass/gv/
[9] ftp://ftp.x.org/contrib/widgets/Xaw3d/

alpha channel.
If there was no background chunk in the PNG file white background would be used.

- “bg3c8rfams0_255_255.ps” was produced using “3c8rfams0,255,255”. The pixel colors are calculated from the foreground color specified in the file and the background color cyan (specified as RGB 0 255 255) depending on the alpha channel.
  You can get “free-standing” objects by defining the same background color as used as page background color.

5 Programmer’s manual

5.1 How to use libbmeps in applications

5.1.1 Module configuration

There are two ways to configure libbmeps’ output:

- Provide a configuration string using

  ```
  void bmeps_cfg(char *cs);
  ```

  The configuration string *cs* corresponds to the EPSOUTPUT environment variable.

- Manual configuration.
  To do so we need some variables:

  ```
  int pslevel, colored, enc_a85, enc_rl, enc_fl,
  is_draft, alpha, trans, altrig, mix, spec,
  bg_red, bg_green, bg_blue,
  shp, dic, vmr, usr;
  ```

  We retrieve the current settings:

  ```
  pslevel  = bmeps_get_pslevel();
  /* PS level */
  colored  = bmeps_get_colored();
  /* 1=colored, 0=grayscaled */
  enc_a85  = bmeps_get_enc_a85();
  /* 1=use ASCII85 encoding */
  enc_rl   = bmeps_get_enc_rl();
  /* 1=use run-length-encoding */
  enc_fl   = bmeps_get_enc_fl;
  /* 1=use flate compression */
  is_draft = bmeps_get_draft();
  /* 1=draft only */
  alpha    = bmeps_get_alpha();
  /* 1=special alpha handling wanted */
  trans    = bmeps_get_trans();
  /* 1=alpha channel is transparency */
  altrig   = bmeps_get_altrig();
  /* 1=alternative masking trigger level*/
  mix      = bmeps_get_mix();
  /* 1=mix background and foreground */
  ```

```
spec     = bmeps_get_spec();
/* 1=specified background */
/* background color */
bg_red   = bmeps_get_bg_red();
bg_green = bmeps_get_bg_green();
bg_blue  = bmeps_get_bg_blue();
shp      = bmeps_get_showpage();
dic      = bmeps_get_dictionary();
vmr      = bmeps_get_vmreclaim();
/* keep resolution */
usr      = bmeps_get_usr();
```

- After processing configuration files and command line arguments we need to write the changed variables back into libbmeps:

```
bmeps_setup(
  pslevel, colored, enc_a85, enc_rl, enc_fl,
  is_draft, alpha, trans, altrig, mix, spec,
  bg_red, bg_green, bg_blue,
  shp, dic, vmr, usr
);
```

5.1.2 Checking support for a given filename

The function

```
int bmeps_can_handle(char *name);
```

can be used to check, whether the file with the given name can be converted. The check is performed based on the file name extension, the file contentes is not inspected. The function returns a value not 0 if the file can be converted, 0 if not.

5.2 Error messages

To change verbose mode use

```
void bmeps_set_verb(int flag);
```

verbose mode is turned off if *flag = 0* and turned on for all other values.

Use

```
void bmeps_set_app(int flag);
```

to indicate whether your application produces “normal” logging (*flag = 0*) or “special” logging (like dvips, where we should write a newline before we start the error message).

The functions

```
int bmeps_get_verb(void);
int bmeps_get_app(void);
```

can be used to check these settings.

5.2.1 Writing an image

Here is a simple example how to write an image.
For a detailed example covering transparency… look into “pngeps.ctr”.

```
FILE *out;
char *name;
unsigned long width, height;
...
bmeps_header(out, name, width, height);
if(bmeps_get_draft()) {
  bmeps_draft(out, width, height);
} else {
  bmeps_begin_image(out, width, height);
    for(y = 0; y < height; y++) {
      for(x = 0; x < width; x++) {
        bmeps_add_rgb(r(x,y), g(x,y), b(x,y));
      }
    }
  bmeps_end_image(out);
}
bmeps_footer(out);
```

5.3 MT-Level

This library is not thread-safe. Configuration data and other data are stored in module-wide static variables.
You can convert only one image at a time in a given process.