The DVIPDF Program

Sergey Lesenko

Abstract

This article describes the DVIPDF program, a DVI driver producing as its output Portable Document Format. It reviews the current state of development of the program and makes some suggestions for \special syntax.

1 Introduction

Among the common problems in the world of TEX is the question of how to produce documents with hypertext capability and high-quality printing at the same time; the Portable Document Format (PDF)[1] by Adobe permits us to resolve this particular issue. There are currently several ways to generate PDF output:

- TEX —→ DVIPS [4] —→ Adobe Distiller
- TEX —→ DVIPS —→ GhostScript [2]
- TeX2PDF [5]
- TEX —→ DVIPDF

The most frequently used solution (the first case above) permits us to generate the most functionally complete PDF files, thanks to the *hyperref* package [3]. This route can be compared with the DVIPDF program. Since it is based on Distiller (the commercial product from Adobe), the former may stay ahead in terms of features; since the PDF specification has recently been upgraded, there is need for further development after shipment of the new version of Distiller.

But the DVIPDF way has its effective side too. If we consider the process of getting output from a `.dvi` file, then we have only the following step:

Step One: Translation from DVI to PDF

But the process using Distiller has two steps:

Step One: Translation from DVI to PDF

Step Two: Translation from PDF to PDF

In the latter case there is a loss of precision for characters, rules and other objects because intermediate values are used in the `.ps` file, not those actually present in the `.dvi` file. DVIPDF makes it possible to generate output with high precision, although currently with a limited set of features.

[1] Described in [1], and also available from `http://www.adobe.com/supportservice/devrelations/devtechnotes.html` and `ftp://ftp.adobe.com/pub/adobe/Applications/Acrobat/SDK/TECHDOC/PDFSPEC.PDF`.

2 Current features and those in development

DVIPDF is based on DVIPS by Tomas Rokicki, and in future may be integrated with it, I would like to hope. What can the DVIPDF program do now and what will it be able to do in future? The current version supports the following features:

- Rotated and scaled text;
- Rotated and scaled graphics (BMP and JPEG formats);
- Colors for text and background;
- Annotations and bookmarks;
- HTML links and links to other PDF files;
- Partial font loading;
- Reencoding.

At present only two graphic formats are supported. The BMP format allows the insertion of illustrations in the PDF file with black & white (1-bit), gray (8-bit) and color (24-bit) models. The JPEG format allows gray and color models. The capabilities for text and background colors correspond to those in DVIPS. Geometric transformation of text as a graphic object may be nested up to sixteen times in any way desired. As far as the hypertext capabilities are concerned, annotations may be nested; for bookmarks this is limited to six levels.

Embedded fonts are PostScript Type 1, using partial font downloading. Re-encoding can be performed on internal (embedded) fonts as well as external (referenced) fonts. Use of external fonts decreases the size of the output, but the potential user has to have these fonts available.

The most important problems for future development are:

- Support for Encapsulated PostScript illustrations;
- Support for new features of the PDF 1.2 specification.

There is currently a way to insert EPS (by producing BMP using GhostScript, and then inserting the figure in BMP format), but the result is not scalable.

There are no plans to support all features of PDF; for example, bitmapped fonts will not be addressed, since they render very badly with the Acrobat Reader; on the other hand, features such as thumbnails may be added, but only much later."

3 Suggestions for \special syntax

Since PDF has some unique features, I had some problems choosing the optimal variants of \special

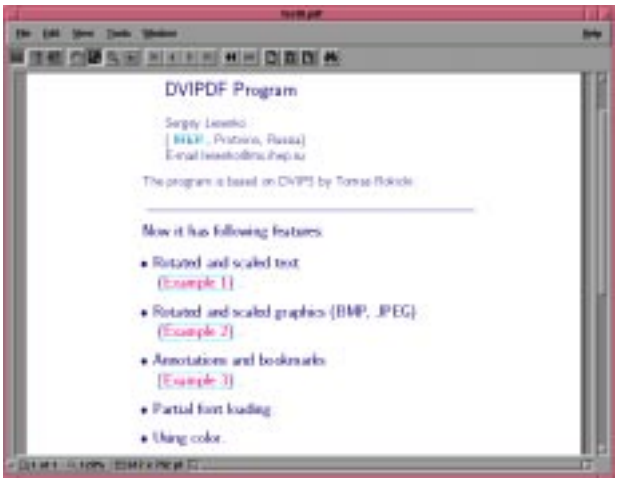

Figure 1:

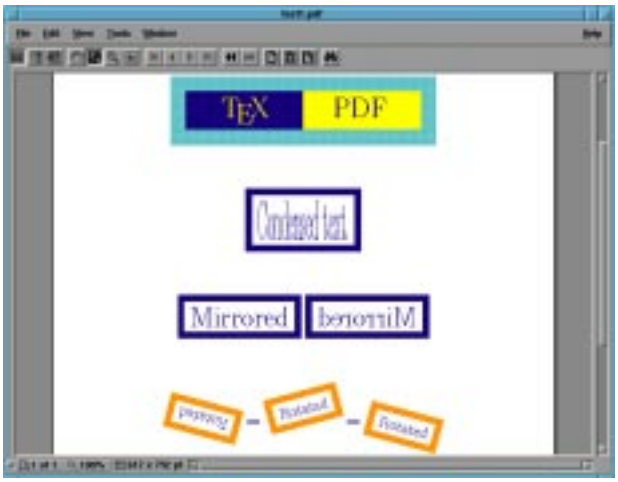

Figure 2:

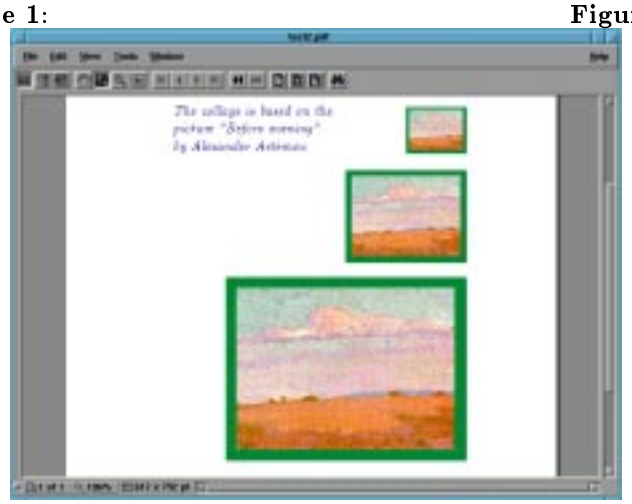

Figure 3:

commands. Let us consider two variants and call them *universal* and *pdf*:

- *universal* to support existing `\special` conventions;
- *pdf* to be oriented only to PDF output.

The *universal* `\special` permits us to generate either a `.ps` file or a `.pdf` file from the same `.dvi` file. This is very useful, if we distribute only the `.dvi` file without TEX sources. However, the *universal* `\special` may need some information only for PostScript output, and some destined solely for Distiller. This leads to redundancy and increased size of the `.dvi` file.

The *pdf* `\special` is more compact and simpler for parsing; since PDF is developing as a standard format, it seems that this second way is preferable, and this was what I have implemented in the current version of DVIPDF.

I would like to introduce some suggestions for syntax:

- `pdf:` – first token to identify a *pdf* `\special`;
- `/ABC` – token consisting only of uppercase characters for definition of type;
- `/Abc` or `/abc` – token for subtype;
- `Abc` or `abc` – alphabetic parameters;
- `123` or `123.000` – numeric parameters;
- `<<` and `>>` – tokens to mark `push` and `pop`.

The above syntax description makes for simple parsing of the `.dvi` file. Some examples are offered in the following table:

Begin rotation	pdf: /ROT 30 <<
End rotation	pdf: /ROT >>
Begin scaling	pdf: /SC 4.0 2.0 <<
End Scaling	pdf: /SC >>
Begin color	pdf: /C Blue <<
End color	pdf: /C >>
Begin annotat.*	pdf: /ANN /LNK /Dest test <<
End annotation	pdf: /ANN >>
Graphics	pdf: /GRAPH filename 123 123

* here some secondary parameters are omitted

I would like to suggest that we pass size parameters for graphics in scaled points (*sp*), not big points (*bp*), since DVIPDF deals with *sp* when illustrations are inserted into some object and then scaled or rotated; recall that *bp*, as a unit, is used only for producing output. The program simply calculates new coordinates and does not need to worry about converting *bp* into *sp*.

To then estimate the effectiveness of the *pdf* `\special` set, I produced DVI output for some TEX files with different `\special` commands. These results are presented in the following table. To generate this data, I used the same sources as for producing the PDF slide files discussed in the next section.

File	*universal*	*pdf*
TEST0.DVI	3200	2276
TEST1.DVI	4244	2400
TEST2.DVI	2676	1720
TEST3.DVI	7284	3808

We can see that `.dvi` files with the *pdf* `\special` format are more compact.

4 Some results

To demonstrate the results I would like to present some figures from slides which were prepared for TUG'96.[2] The collection consists of four `.pdf` files (they are DVIPDF output): main file (`TEST0.PDF`) as a menu, and three auxiliary files (`TEST1.PDF`, `TEST2.PDF` and `TEST3.PDF`).

The main file has one HTML link (IHEP on Fig. 1) and three links (Example 1, Example 2 Example 3 on Fig. 1) to auxiliary files.

When we click on the HTML link, the Acrobat Reader passes a request to our browser (Netscape, for example) and it asks for the IHEP home page (if, of course, our computer is connected to the Internet).

If we click on any link to auxiliary files, the chosen file will be loaded and the Reader will view it. Each file has a link (Return on Fig. 2, Fig. 3) to the main file, so that we can return to the main file and examine the other auxiliary files.

If all goes well, I hope to release DVIPDF for testing towards the end of 1996.

[2] 17th Annual Meeting of the TEX Users Group in Dubna, Russia, July 28th – August 2th, 1996

Acknowledgements

I would like to thank Michel Goossens and Sebastian Rahtz for supporting this project, and Mimi Burbank and Sebastian Rahtz for editing this paper.

References

[1] Tim Bienz and Richard Cohn. *Portable Document Format Reference Manual.* Addison Wesley, Reading, MA, 1993.

[2] L. Peter Deutsch. Aladdin ghostscript, 1996. version 4.01, available at `ftp://ftp.cs.wisc.edu/pub/ghost/aladdin`.

[3] Yannis Haralambous and Sebastian Rahtz. LATEX, hypertext and PDF, *or* the entry of TEX into the world of hypertext. *TUGboat*, 16(2):162–173, June 1995.

[4] Tomas Rokicki. Dvips: A TEX driver, 1994. version 5.58, available at `ftp://labrea.stanford.edu/pub/tex`.

[5] Petr Sojka and Han The Thanh. The Joy of TEX2PDF — Acrobatics with an alternative to DVI format. *TUGboat*, 17(3), 1996.

◇ Sergey Lesenko
P.O. Box 35
Institute for High Energy Physics
Scientific Information Department
Protvino, Moscow Region, Russia
`lesenko@mx.ihep.su`