dvi and EPS: The Ideal Author-to-Publisher Interface?

Berthold K.P. Horn
MIT AI Laboratory, 545 Technology Square, Cambridge, MA 01741 USA
Internet: bkph@ai.mit.edu

Abstract

dvi files specifying text and document format, along with EPS (Encapsulated PostScript) files for included figures, are rapidly becoming the *de facto* standard for interchange of machine-readable manuscripts in technical publishing. While dvi file format and EPS file format are standardized, the glue holding them together, namely the \special macro, is not. This is presently the weak link in the chain. Unfortunately, in the absence of an officially sanctioned standard, every publishing organization is developing its own standard, often inelegant and inextensible. Electronic publishing has arrived. Publishers in specialized technical areas are using machine-readable material now. Their needs for standardization have become critical.

Introduction

Device-independent (dvi) files specifying text and document format, along with Encapsulated PostScript (EPS) files for included figures, are rapidly becoming the *de facto* standard for interchange of machine-readable manuscripts in technical publishing.

The advantage of dvi files over raw TEX files is that there is no need for the publisher or service bureau to bring up the special version of TEX used by the author, nor does the publisher have to deal with the author's macro packages. dvi files are supremely standardized, portable and compact.

The advantage of dvi files over PostScript (PS) files produced by present-day dvi-to-PS converters is that dvi files are resolution-independent, while PostScript files containing bitmapped fonts are not. As long as dvi-to-PS converters continue to use bitmapped fonts, they will have to be run over the dvi file again and again, each time an output device with different resolution is to be used.

While dvi file format and EPS file format are standardized, the glue holding them together, namely the \special macro, is not. This is presently the weak link in the chain. Unfortunately, in the absence of an officially sanctioned standard, every publishing organization is developing its own standard, often inelegant and inextensible.

The only viable alternative to the combination of dvi and EPS files is the resolution-independent PS file. Resolution-independent PS files containing both text and illustrations are possible now that high-quality outline font programs are available for Computer Modern.

One advantage of resolution-independent PS files over dvi files is that they contain only ASCII characters and so can be more conveniently stored and transmitted. Perhaps more significant is the fact that resolution-independent PS files can be sent to a service bureau that is not knowledgeable about TEX and dvi and does not have access to high-resolution bitmapped fonts. This lowers costs considerably and gives the editor or author complete control over the final appearance the work.

The Best Medium of Interchange?

We probably would all agree that when writing on a technical subject, particularly one requiring the use of mathematical formulae, an author these days finds few viable alternatives to the use of TEX for preparing papers and books. In the past, the author's manuscript, after review and revision, was typeset, with the author required to proofread the result, which quite often was less pleasing than the original TEX output submitted!

This whole process is expensive, slow, frustrating, and error-prone. It is, of course, being displaced by the obvious alternative. But so far this transition has been slow and painful. There are a number of critical areas that need urgent attention if the change is to progress more smoothly.

First of all, what is the best medium of interchange between author and publisher? Should

it be (a) TEX source, (b) dvi files, or (c) PostScript code produced from dvi files? I will argue that at the present state of development the dvi file is the best of the three alternatives. The reason is that dvi files are standardized, portable, and compact. (And unlike some other 'standards' the format of dvi files *really* has not changed for many years.)

TEX source and macro files. If the author supplies TEX source, the publisher or typesetting service bureau needs to be able to run the dialect of TEX used by the author, and also have access to the macro packages used. This may involve moving a complex web of interrelated files. More seriously, it requires considerable investment in computer hardware, software, and a level of sophistication that is *not* required if only dvi files are being manipulated.

Another problem is that the publisher may have stale versions of some of the macro files. One way to make the use of TEX source slightly more bearable, and to circumvent the stale macro file problem, is to create a program, called TeXExpand perhaps, that creates a single (large) file by (recursively) inserting files called for in the original TEX source file.

PostScript code. PostScript code generated from dvi files in the past was not resolution-independent, since dvi-to-PS converters used bitmapped fonts. This meant that the publisher had to tell the author in advance what device the text would be typeset on, and the author had to build the (large) bitmapped font files required for that device.

Typesetting could not proceed from the same file used by the author to produce draft output for review. The publisher did not have the ability to later alter the choice of output device, since the resolution was frozen in the files.

dvi files. The above clearly suggest that TEX source and PostScript output are less satisfactory than dvi files. To many people the idea that the dvi file is the best medium of interchange is so alien that, even after being told several times to send dvi files, they continue to submit PostScript files; and when reminded not to do this, they will send TEX source files along with a web of macro files!

The only minor drawback of dvi files is that they are binary, requiring care in transmission.

What about Illustrations?

Next we come to the question of illustrations. Presently the most satisfactory method here appears to be the use of encapsulated PostScript files. Properly constructed — that is, conforming — EPS files can be resolution-independent and print well on any PostScript image-setting device. Some of the alternatives are less satisfactory, although they have their uses in specialized situations, e.g.:

a. LATEX line and circle fonts permit construction of certain kinds of simple figures;
b. PICTEX can generate graphs and figures of limited complexity; and
c. METAFONT can generate bitmap images, although these are not resolution-independent, and will look 'pixelated' when printed on a high-resolution device.

In most cases then, the combination of dvi and EPS files appears to be the best combination for transfer of material from the author to the publisher.

Indeed, dvi files specifying text and document format, along with EPS files for included figures, are rapidly becoming the *de facto* standard for interchange of machine-readable manuscripts in technical publishing.

dvi Files are *not* Device Independent

The only problem with this rosy picture is that dvi files are not truly device independent! Yes, unfortunately there are two areas in which the extensions provided for by TEX lead to difficulties. And these extensions are the very ones that we can no longer imagine living without. They are:

a. Inserting illustrations using \special; and
b. Using fonts other than those in the Computer Modern family.

What's so Special about \special?

The problem with use of \special for figure insertion is the more complex of the two problems, but also the one more urgently in need of a solution. In the absence of an officially sanctioned standard, every publishing organization using TEX or dvi files is developing its own *de facto* standard, sometimes inelegant or inextensible.

A major stumbling block to completion of the transition to electronic publishing is that every dvi processing program supports a different convention for usage of \special. This means that every job is a custom job. Instead of a smooth operation involving only the transfer of the author's dvi and EPS files, a serious programming effort is often required to deal with yet another way of using \special.

While \special is the open door to extensions of TEX usage, we need concern ourselves here

only with the use of \special for figure insertion. Arguments over uses of \special for other purposes should not drown out discussion of the urgent need for a simple standard for figure insertion.

No one can anticipate all the possible uses for this powerful extension of the TEX language, nor is it likely that the community can soon agree on the details of how such extensions are to be implemented. But this should not stand in the way of satisfying what has now become an urgent need: a standard way of using \special to include figures in text.

When the discussion of standards for dvi drivers first started, there was little urgency, since the routine need for these capabilities had not yet arisen outside a small number of research laboratories. Progress in this field has been rapid, however, outpacing deliberations of the standard committees, with journals rapidly switching to machine-readable material. Similarly, books are now routinely produced from TEX output.

This represents one of two major obstacles to seamless electronic publishing. Therefore,

- The time for publication of a simple standard for figure insertion in papers and books is *now*.

Simple requirements. What is required is in fact really quite straightforward. All that is usually needed is a means for inserting a figure, possibly scaled, at the desired position in TEX. Sometimes it is also useful to be able to shift and perhaps rotate and clip the figure. Informal statistics show that 80% of the time simple figure insertion is enough, while scaling is also called for in perhaps 20% of the cases. Shifting, rotating, and clipping are almost never used, but should perhaps be provided — just for generality's sake.

More important, no use seems to be made of the ability to insert verbatim PostScript commands, to call on PostScript functions native to a particular dvi processing program, or to produce overlays. While these transformations represent interesting and powerful extensions, they apparently are not vital to the production of even the most sophisticated texts.

There may be several reasons for the limited use authors presently make of the more complex figure manipulations:

a. Apparently even the most sophisticated textbooks can be produced using little more than simple figure insertion.
b. It is relatively easy to modify a file that obeys the EPS structuring convention to achieve the desired graphical transformation.
c. Authors know that exploiting esoteric features of particular dvi processors will reduce the portability of their document and consequently restrict themselves to the simplest operations that will accomplish their objective.

Lack of standardization of usage of \special for figure insertion is the main obstacle to seamless electronic publishing using TEX. A simple standard is urgently required.

Existing schemes. For inspiration one might consider some of the existing schemes:

a. The use of \special in UNIX's DVI2PS is simple, and provides most of the listed features. An example:

```
\special{psfile=figure.eps
        hscale=0.66 vscale=0.66}
```

b. The use of \special in Blue Sky Research's *Textures* is also satisfactory, although it does not provide all of the features indicated (but in turn provides some others). An example:

```
\special{illustration figure.eps
        scaled 667}
```

c. The proposed use of \special in Nelson Beebe's next release of DVIALW has many desirable features (although it is perhaps more complex than needed). For example:

```
\special{language=PS include=figure.eps}
```

It should be possible to use \special for figure insertion without reference to internal procedures of a particular dvi-to-PS converter or inclusion of verbatim PostScript code.

It should be clear in any case that a standard syntax for figure insertion using \special should be established as soon as possible.

Font-Naming Woes

The other device-dependent aspect of dvi files is the naming of non-Computer Modern fonts. This is the easier of the two problems to analyze — and to fix.[1]

For Computer Modern there exists a standardized way of relating the font names used in TEX to the files containing font metric information and the files containing the actual outlines or bitmaps of that font.

[1] The reason the discussion of the font naming problem covers more paper here than discussion of the more serious problem of standardization of \special for figure insertion is precisely that it is the simpler of the two issues.

We do not usually waste much time worrying about this, but there needs to be a mapping between three entities: (a) the name used to refer to a font in the TEX source document, (b) the name of the TEX font metric (`tfm`) file for that font (which TEX needs to do its job), and (c) the name of a font program file (or a font program in the printer) that actually draws the characters (which the `dvi` processing program needs to know about).

Unfortunately, there is no general agreement yet on how to build such a mapping for fonts other than Computer Modern. The problem would be slightly simpler if it were not for the fact that the name used to refer to a font in TEX used to be constrained to be no more than 6 characters long — and is in any case constrained to no more than 8 characters by some operating systems such as MS-DOS. What is done now — as a stop gap measure — is for `dvi` processing programs to accept an auxiliary file that contains the mapping. This file must be supplied by the author or constructed by the publisher after obtaining the required information from the author.

One reason the font-naming problem is becoming more of an issue is that many publishers are urging authors to be more ecumenical about font selection. So far, such pressures have encountered strong resistance because of the sparsity of satisfactory non-CM fonts for typesetting mathematical formulae. But there is now at least one alternative: Bigelow and Holmes' LucidaMath fonts published by Adobe.

Lack of portability. This lack of standardization has proved to be a source of frustration when `dvi` files are ported from one computer system to another, as is common when publishing journal articles and books from author-supplied material. As it stands now, each project requires customization, compelling the typesetting service bureau to set up yet another new font-name translation table. Perhaps more seriously, without a uniform naming convention, it may happen that the `dvi` processing program and TEX have conflicting notions about what fonts are being referred to — with disastrous consequences.

The above represents the other major obstacle to seamless electronic publishing. Therefore,

- A standard naming convention for fonts other than those in the Computer Modern family should be established as soon as possible.

This is particularly important for the existing collection of fonts in Adobe Type 1 format. This collection is both popular and very large. Thirty vendors supply over 7000 fonts in this format (at the time of writing), with 1300 in the Adobe Font Library alone. Here an unaesthetic standard is better than no standard at all — or a standard that is not extensible enough to deal with the continuing flood of typefaces being converted into this format.

One solution would be to establish some permanent organization to invent abbreviations or at least act as a clearing house for proposed abbreviations of font names. This does not seem practical, since it is unlikely that such an organization could deal in a timely fashion with the rapid growth in the collection of fonts in this format. Consequently,

- One should use established font-naming schemes whenever possible.

This will reduce confusion and avoid the need for a central registry of abbreviations. Adobe, for example, has already found it necessary to invent 6-character abbreviations for its fonts — it seems inefficient not to use these.[2] This in fact will take care of a significant part of the font-naming problem, since presently the most commonly called for non-CM fonts are Adobe Type 1 fonts.

Remapping of character code assignments. Unfortunately, the above isn't the full story. Each font has its own mapping between the numeric character codes (typically 0–255) and character glyphs. There are nine different standard mappings used by Computer Modern fonts: roman (e.g., `cmr10`), text italic (e.g., `cmti10`), typewriter (e.g., `cmtt10`), typewriter italic (e.g., `cmtti10`), small caps (e.g., `cmcsc10`), TEX ASCII (e.g., `cmtex10`), math italic (e.g., `cmmi10`), math symbol (e.g., `cmsy10`), and math extended (e.g., `cmex10`).

A non-CM font can be used with the encoding it came with, or can be remapped to one of the above standard encodings. It must be possible to distinguish between `tfm` files for the original font and the remapped font. The easiest way to do this is to use different, but related, file names for the two versions. One simple scheme is the following:

- The file name of the `tfm` file for a remapped font has an '`x`' appended.

This doesn't completely solve the problem, since it doesn't specify *which* remapping was chosen. Unfortunately, the `tfm` file format does not provide for an encoding vector mapping numeric character

[2] Some of Adobe's font downloaders happen to limit the font-name part of the file name to six characters, which conforms exactly to the old restriction in TEX.

codes to character names, only an *optional* field that may contain the name of a remapping (and only the nine names mentioned above are in any way considered standard).

Fortunately, the need for remapping fonts is greatly reduced by the advent of TEX 3.0, which can deal with 8-bit character codes.

Resolution-Independent PostScript

The only viable alternative to the combination of dvi and EPS files is the resolution-independent PS file. Resolution-independent PS files containing both text and illustrations are possible now that high-quality outline fonts are available for Computer Modern.

An aside. Some readers may have low expectations for the quality of rendering using outline fonts, perhaps having seen the results of some early experiments. Properly hinted Type 1 fonts, however, are compact, support across-job font-caching, and most important, produce beautiful characters. Type 3 outline fonts, used in some early experiments, suffered from the 'dot-growth' phenomenon inherent in use of the PostScript `fill` operator. Furthermore, *unhinted* Type 1 fonts do not render well on low resolution devices such as computer display monitors.[3]

To return to the topic at hand, note that resolution-independent PS files derived from dvi and EPS files:

a. should not make any assumptions about the output device resolution;
b. should not rescale or round coordinates given in dvi files; and
c. should not refer to bitmapped fonts.

One advantage of resolution-independent PS files over dvi files is that they contain only ASCII characters and so can be more conveniently stored and transmitted. (Extra work is required to safely transport binary files across networks or even serial lines connecting disparate computer systems.) There is no need to redo the conversion from dvi to PS form when a printer or image-setter of different resolution is used.

Perhaps more significant is the fact that resolution-independent PS files can be sent to a service bureau that is not knowledgeable about TEX or dvi files, and does not have access to high-resolution bitmap fonts. This lowers costs considerably and gives the editor or author complete control over the final appearance of the work.

Summary

dvi and EPS files are the preferred medium of interchange of material between author and publisher.

Electronic publishing has arrived — although it is not quite seamless yet. Publishers in specialized technical areas are using machine-readable material now. Their needs for standardization have become critical.

One of the areas in need of attention is that of usage of `\special` for inclusion of illustrations:

- A standard syntax for figure insertion using `\special` should be established as soon as possible.

This should not close the door on future, as yet unanticipated, uses of `\special`. All that is needed now is a simple syntax for insertion of illustrations. There is serious danger that in the absence of any guidance *ad hoc* standards will come into widespread use that are neither elegant nor extensible. The window of opportunity for influencing developments in this area is open now, but will not remain open indefinitely.

The other problem area is that of naming conventions for fonts other than Computer Modern:

- A standard naming convention for fonts other than those in the Computer Modern family should be established as soon as possible.

Finally, note that there is an alternative to the use of dvi and EPS files, namely the resolution-independent PS file. As a parting thought, consider the following table of estimated costs:

> $30–40 per page for traditional typesetting;
> $9–10 per page for service bureau work from TEX source; and
> $2–3 per page to print resolution-independent PostScript.

There is a clearly an incentive to consider seamless electronic publishing. And there is clearly an even greater incentive to consider resolution-independent PostScript.

[3] Also, a particular character's shape may be described in many different ways by using lines and Bézier curves. Some such description may contain many more elements than really neccessary, and may not obey the strict rules specified in the Type 1 standard. Rendering using such an outline is likely not to be as fast or as clean as that of a properly constructed outline.