Viewing DVI files with Acrobat Reader — DVIPDF gives birth to AcroDVI —

Sergey Lesenko
Institute for High Energy Physics (IHEP)
Protvino (Moscow Region)
142284 Russia
lesenko@mx.ihep.su

Laurent Siebenmann
Mathématique, Bât. 425
Université de Paris-Sud
91405-Orsay, France
Laurent.Siebenmann@math.u-psud.fr

Abstract

The free DVIPDF program of the first author converts from TEX's DVI output files (called DVI since DeVice Independent) to Adobe's PDF (= Portable Document Format). Although DVIPDF has existed as a prototype for about three years, the uses to which it will be put by the TEX community are only gradually emerging. This article presents one concrete application. DVIPDF has been adapted under the Windows 9X/NT operating systems to allow immediate viewing of DVI files with Adobe's free Acrobat Reader. We call the resulting viewer AcroDVI: it involves DVIPDF and Acrobat Reader operating in concert.

AcroDVI is intended for viewing legacy DVI files and thus aims to support the the commonest `\special` additives. Since the mid 1980s, it has been traditional to present, along with a DVI file, all auxiliary graphics as files in format EPS (= Encapsulated PostScript). To integrate and view such auxiliary EPS files, AcroDVI needs help from a PostScript interpreter such as the freely distributed Ghostscript. This makes viewer performance sluggish. To fix this, we are recommending an evolutive change of electronic publishing practice: the EPS format should be replaced by various 'optimal' formats: JPEG or PNG for bitmaps, and PDF for vector graphics. When AcroDVI is exploited via the convenient drag-and-drop interface of MS Windows, each of these is instantly integrated, or separately viewable, or editable, or printable, or convertible to EPS. As an alternative to the monolithic PDF format, we are promoting DVI with such additives. This evolutive format's polyvalence and compactness make it preferable to PDF where CD-ROM distribution or modem transfer are concerned.

Introductory viewing experience

> *. . . Egli e' scritto in lingua matematica, e i caratteri son triangoli, cerchi, ed altre figure geometriche . . .*
>
> [*Galileo writing on physical science*]

[1]

To view a DVI using AcroDVI you can simply push the DVI file's icon onto that of AcroDVI or onto a shortcut icon to AcroDVI conveniently located on the desktop. The DVI file is quickly converted to PDF (read on for rates); then a window automatically pops up for viewing by Acrobat Reader (of which a version ≥ 4 of 1999 must be installed). If you are not already familiar with Acrobat Reader, the bigest thrill will surely be the the top quality graphics and typography, both superior in various respects to what web browsers offer. Worth notice from TEX users are the hypertext features familiar from web browsers. All this is remarkable, but not new.

[1] * [This article has not yet been edited. – Ed.]

For those familiar with Acrobat Reader, the main novelty in the AcroDVI viewing experience is currently enhanced visibility of the graphics objects. They appear not only in the PDF page view but also autonomously in 'native' formats suitable for reuse and also for display at an optimal scale. The three formats that AcroDVI deals with directly are PNG for bitmapped high contrast graphics, JPEG for color photos, and PDF for vectorial graphics (more on these later!). The corresponding file extensions are ".png", ".jpg" and ".pdf" respectively. One of these norms should be optimal for just about any (still) graphics object.

If you push the icon of a PNG or JPEG file onto that of AcroDVI, then it will be immediately viewed in an Acrobat Reader window. If you do the same with a PDF graphics object, something similar will happen. Likewise for EPS files, provided Acrobat Distiller or Ghostscript is accessible and enough fonts are available. Latent in Acrobat Reader, which does not itself process PNG or JPEG files, are broad graphics viewing capabilities, and DVIPDF has boldly tapped into them; Adobe clearly could have done as much for Acrobat Reader, but chose not to do so.

There are many specialized tools for both viewing *and* editing PNG and JPEG files, notably the free XNview under Windows & Linux [Gou] and the shareware Graphics Converter on the Macintosh [Lem]. If you take care to view at scale 100%, then you will see the bitmapped graphics at best possible quality.

The most widely used tools for viewing PNG and JPEG graphics are certainly the HTML browsers. This is an open invitation to make double use of the graphics in an article: firstly, in an illustrated HTML introduction, and, secondly, in the DVI file for the article's body.

Thus, AcroDVI provides striking polyvalence for graphics. At the same time, it provides a basic polyvalence for text: namely, the possibility to view the same DVI file with Acrobat Reader and with classical DVI viewers.

Such polyvalence (and efficiency) was the immediate motivation for devellopping AcroDVI. Indeed, the CD-ROM project called MathCD [Mcd], badly needed a compact and polyvalent format for mathematics journal content, basically because it has an order of magnitude less space available for many journals than a single journal can afford on internet.

Where space is at a premium, as on some CD-ROMs and in personal electronic libraries, DVI format plus auxiliary 'native' graphics can now reasonably replace PDF format. Where space is pretty well unlimited, as on many internet sites, expect to see more formats then ever!

There are relatively few hyperreferences in the electronic journal articles on MathCD. Currently, DVIPDF does support hyperreferences using a `\special` syntax parallel to Acrobat Distiller's 'pdfmark' syntax. But, by an accident of its youth, it unfortunately does not yet support the commonest `\special` syntax of today's DVI files, namely the one introduced by xhdvi and parallelling HTML.

What really is AcroDVI?

DVIPDF and AcroDVI currently run under all versions of MS Windows, excepting the early versions 3.x. They will be freely available on internet.

The technologically aware user will tend to see AcroDVI as the sum of its parts: DVIPDF plus Acrobat Reader plus some MS Windows programming using the Dynamic Data Exchange (DDE) protocol as framework for collaboration between DVIPDF and Acrobat Reader. DVIPDF will remain the name for the system-independent core program; the advent of AcroDVI is more like childbirth than graduation.

However, to the passing user, the whole will be more important than the parts. Indeed, AcroDVI acts as a viewer that accepts most DVI files (modulo font availability — read on), and all files of norm PNG, JPEG, and PDF. Significantly, it is the first viewer to accept all of these. We have decided to dignify the whole with the the acronym AcroDVI pronounced "acro-d-v-eye" having accents on the first and last syllables. The eye of the viewer is what you should see in the logo:

AcroDVI

The Windows icon is itself a colored iris (an eye-con!); and onto it, the files to view will be dragged and dropped. (See black/white renditions of the current icon on later pages.)

In its present infancy, AcroDVI involves a single binary executable that currently bears the name DVIPDF while the distribution directory is called AcroDVI. That makes AcroDVI rather like a marsupial 'mother-with-baby-in-pouch'. To facilitate portability, there is just a decent segregation of the C++ source code into modules devoted exclusively to the AcroDVI viewer functions and modules that can hopefully be compiled as a 'black box' processor to implement DVIPS as stand-alone DVI-to-PDF converter. Incidentally, most of the `\special` features developed recently for viewing legacy DVI files (more on this below!) have become permanant

additions to the 'black box' part of the DVIPDF program.

What shape will maturity bring to AcroDVI? Two options currently hold our attention.

It was first proposed in [Les2] 1998 to build a DVIPDF 'plugin' for Acrobat Reader. With that approach, to view a given DVI file in Acrobat Reader, one would push its icon onto the Acrobat Reader icon rather than onto the DVIPDF icon. The plugin architecture promises to promote portability of AcroDVI.

A second reasonable option would make AcroDVI an autonomous Windows application distinct from DVIPDF. This architecture promises to facilitate orchestration by AcroDVI of *multilateral* collaborations among *DVIPDF, Netscape, Zip, Acrobat Reader, Ghostscript, ...*.

Incidentally, as soon as Ghostscript/Ghostview under Windows provide support for the key functions "Open Doc" and "Close Doc" of DDE, there will promptly come into being an alternative viewer GhostDVI, a sibling of AcroDVI. It will probably be less luxurious, but it will, in addition, accept PS and EPS files. Also, it will serve several platforms not supported by Acrobat Reader.

Fonts

DVI files do not contain fonts. That is one reason why they are so compact. Where, then, are fonts for AcroDVI to come from? The best one can hope, is that, in practice, enough Type 1 fonts will be in AcroDVI's expansible repertoire.

Adobe's Type 1 is currently the only font norm supported by DVIPDF. Thus TT (TrueType) fonts are not accepted by DVIPDF. Nor are Adobe Type 3 fonts allowed, bitmapped or not; Acrobat Reader would in any case handle them poorly. Fortunately, DVIPDF can rely on the extensive BaKoMa Type 1 font collection [Ma1] of Basil Malyshev for essentially all the fonts commonly used used in freely distributed electronic science publications.

The renowned Adobe Type Manager (currently $100 approx.) that first made screen viewing with scalable (vectorized) fonts a significant reality is *not* needed by AcroDVI since the relevant functions have been absorbed into Acrobat Reader.

On MathCD, there are just a few DVI files that call for commercial Adobe Type 1 fonts. DVIPDF will not currently handle these unless you have them installed in Type 1 format. Since many of these have acceptable TrueType (= TT) versions preinstalled by MS Windows, more support for TrueType would be desirable.

Until then, we recommend Malyshev's own DView for this purpose. It offers essentially universal font support — although different graphics support. DVIew is also recommended for articles on MathCD that call upon fonts available only in Metafont and ".pk" formats.

Installing AcroDVI

As it will appear on MathCD, AcroDVI is a directory of approximately 1.5 megabytes, not including the BaKoMa font collection, which is another directory of a few megabytes that should stand next to the AcroDVI directory during installation. As for many Windows programs, an installer program is used.

AcroDVI is largely autonomous in that it requires only the presence of the free Acrobat Reader (preinstalled in version ≥ 4), and non-invasive in that it alters the behavior of nothing outside its own folder. To deinstall it, one just trashes its directory.

Hopefully this means that AcroDVI it will be as simple to use as Acrobat Reader itself. Sophisticated users do have an extensive configuration file to play with.

Considerations of speed and space-efficiency

The following performance figures are for a 1997 PC with a Pentium I processor operating at clock speed 200 Mhz under MS Windows 95. For other MS Windows environments, a simple correction for clock speed should give a good first approximation to performance.

For a typical mathematics article, the conversion to PDF format goes at about 15 pages per second. This speed is 5 times greater than for Acrobat Distiller. Comparison is relevant since it would be possible to publish compressed PostScript files without included fonts, while giving Acrobat Distiller access to the same BaKoMa font collection.

For its PDF output, DVIPDF does both font subsetting and stream compression. (The new compressed Type 1 font format has yet to be exploited by DVIPDF.) The efficiency of its default PDF output is thus respectable but not yet optimal. For example, it is comparable to that of the PDF files currently published by the Amer. Math. Soc. for its one free electronic research journal ERA (= Electronic Research Announcements). However, by playing with the settings of Distiller, we were able to do better with Distiller by a ratio 3:2. Do not rush to conclude that this ratio in favor of Distiller applies to all math journals. Indeed, the advantage swung in favor of DVIPDF for the next test below by nearly a 2:3 ratio. It seems that both these PDF

compilers could still reduce PDF bulk somewhat, in spite of many years of effort in this direction.

The DVI files used by AcroDVI are far less bulky than the PDF files used by Acrobat Reader. Here are a few examples from a the first 1999 volume of the Electronic Journal of Probability, which, in 1998, added to its web offering PDF format compiled by Distiller:

Article	*Pages*	*.pdf*	*.pdf.gz*	*.dvi*	*.dvi.gz*	*Adv*
1	11	402	284	48	22	12.9
2	19	437	320	78	27	11.8
3	19	459	343	85	35	9.8
4	81	1162	960	412	154	6.2
5(?)	12	251	112	55	33	3.4

All sizes here are in kilo-octets (Ko). Notice that DVI files regularly compress to about 40% of their original size, while PDF files compress far less (as big internal chunks are precompressed). The last column of the table labelled "Adv" meaning 'DVI advantage' gives the size ratio of compressed PDF files to compressed DVI files. This is an accurate measure of modem transfer speed ratios — whether the files are compressed or not — because during modem transfer, all material is compressed. The same ratio will be roughly the file size advantage of DVI files on a CD-ROM like MathCD that attempts to make the best use of available space. Indeed, a thoroughly precompressed form of PDF would be chosen for such a CD-ROM, while the DVI files would probably be zip-compressed along with any auxiliary graphics files.

The last (fifth) article was anomalous in a number of respects. It had `\special` commands; there were two ".eps" figures, and these were complemented by their ".pdf" versions from Distiller, and the total of these additives was less than 12 Ko of insertions (compressed). The explanation for PDF being only 3.4 times less efficient than DVI turned out, on investigation, to be mostly due to a common error in the production of PDF; namely, it was made with bitmapped TEX fonts, which perform disastrously in Acrobat Reader. When this is corrected, one can expect a PDF size similar to that of the first article.

Here are a couple of further examples, the shortest and longest available from the 1999 volume of ERA (Amer. Math. Soc.):

Article	*Pages*	*.pdf*	*.pdf.gz*	*.dvi*	*.dvi.gz*	*Adv*
1	3	105	84	12	5.5	15.3
2	12	248	215	66	27	8.0

These examples were reworked by us using well tuned settings for Distiller (Windows version), and the results (see below) are more flattering for the PDF format, while leaving substantial advantage to DVI. Note that the *difference* between efficient and inefficient PDF is often many times greater than the total size of a DVI version!

Article	*Pages*	*.pdf*	*.pdf.gz*	*.dvi*	*.dvi.gz*	*Adv*
1	3	62	50	12	5.5	9.1
2	12	144	118	66	27	4.4

In the same vein, we note that the electronic journal "Geometry and Topology" (see www.emis.de) posts no TEX format whatever, just the Adobe formats PS and PDF. For their 1st 1000 pages the average PDF bulk per page is 10 Ko (fonts included) or about 8 Ko compressed. Thus, the "DVI advantage" would probably be somewhat less than 4. Expert use of PDF does make a big difference.

The modem bottleneck Modem transfer speed is an important time factor. With a good telephone modem and a good line one can hope to get a transfer rate of about 5 Ko per second of compressed material. Now, a mathematics article in DVI format is about 2 Ko per compressed page, and hence the transfer rate is about 2.5 pages per second. This is about the speed at which one can scroll through the article with the 200 MHz PC used for these tests. Note that DVIPDF converts to PDF format at 6 times this speed. The time taken is perhaps time lost, but it is negligible.

With bad telephone lines or modems, or again congested web conditions, transfers that last more than a minute or two are likely to be broken; clearly the big PDF files are the ones at greatest risk and with the present web protocols, the partial transfer is completely wasted.

Article transport costs To get a very rough cost estimate, consider a mathematics article of 100 pages posted on internet and ultimately downloaded by 1000 readers (the typical number of subscriptions to a paper journal). Let is assume, to get an easily calculated figure, that everyone uses a contempory 56K baud modem with telephone charges of $2 per hour and in compensation let us neglect all other charges. Then reckoning with the above figures, the telephone cost for delivering the article is about $22 with DVI format and between $70 and $250 for PDF format. Such figures suggest that use of DVI does reduce data transport costs significantly.

Improving the AcroDVI environment Here is another recent economy feature that is made possible by the speed of AcroDVI. First, the problem to be solved. In browsing the literature, it is not

uncommon to look quickly at dozens of articles. This can quickly eat up many megaoctets of disk space if PDF format is involved. Now, one of Murphy's computing laws asserts that hard disks are nearly full, no matter what their capacity, since "data expands to fill any void". Hence DVIPDF constantly risks running out of space.

To largely eliminate this overflow risk, there will be a setting for AcroDVI that makes the PDF file ephemeral and invisible. As soon as the next DVI file is processed, the previous invisible PDF will be erased. (That is no loss, since it can quickly be regenerated.) With this scheme, it suffices to verify at the beginning of a browsing session that your hard disk volume has enough space for the largest single PDF file you expect to read, plus enough space for the relatively tiny DVI files.

Review of Strengths and Weaknesses of the PDF format versus DVI

Adobe's Acrobat Reader has as its native file format the Portable Document Format (=PDF). This format is very autonomous – for example:

- graphics objects are always embedded within the PDF file.
- fonts most often are contained. (The alternative to use system fonts has proved somewhat unreliable.)

These positive features bring some disadvantages:

- bitmapped graphics are unlikely to be displayed on screen at optimum quality, since that means no scaling. Vectorial graphics may not be seen in their full glory since that often requires the full screen.
- It is difficult to export graphics objects from the PDF file in the most useful formats.
- PDF files tend to be many times larger than DVI files. This is partly because of the font burden.[2] But, in addition, the complexity of the PDF file structure brings substantial hidden costs.

Besides its space economy, the DVI format has virtues worth mentioning. Like all of TEX, the DVI format is very stable, in spite of its `\special` appendages. This is important for archiving. Secondly, DVI is simple: – just a few pages of "TEX the Program" [Knu] define it adequately. Finally one can get to all other formats from it, except TEX source (i.e. the ".tex" file).

[2] The journal "Geometry and Topology" posts PDF format both with and without included fonts; omission of fonts economizes 25% over the 1st 1000 pages of articles

The strongest argument for PDF format has been the wide availability and high performance of the free Acrobat Reader. Particularly outstanding are the user interface, the graphics quality, and the graphics speed. Adding to this:– search, hypertext, copy & paste to text files, annotations, printing facilities and PostScript (or EPS) export, it is clear that Acrobat Reader is a major contender for the affections of the reading public.

This does not prove that Acrobat Reader has no rivals among DVI readers; for example the recent BaKoMa DView can do better in speed and quality and scope of typography. emTEX provides better search and text export. Interesting new DVI viewers continue to appear. — for example, check out A. Lingnau's TKDVI at

http://www.tm.informatik.uni-frankfurt.de/~lingnau/tkdvi

and K. Peeters' nDVI at

http://norma.nikhef.nl/~t16/ndvi_doc.html

It would be destructive not to serve such DVI readers. And ultimately destructive of TEX itself since TEX systems are typically built around them.

Extending to PDF, PNG and JPEG the traditional EPS g

Be aware that the schemes to be described are just two of many that have been elaborated for integration of graphics into the PDF output of DVIPDF, see [Les2][Le3].

Let us begin by considering the graphics integration issue that arose for electronic journal articles to appear on MathCD. The DVI format is usually one of two or three presented, and it entails, for each article, one DVI file accompanied perhaps by some EPS graphics files. For MathCD it was important that the TEX version of the articles *not* be necessary for the integration of the reformatted graphics files.

Since DVIPDF cannot, on its own, convert EPS graphics to PDF, it was initially decided to provide, via Distiller, PDF versions of the graphics objects. One reason for this decision was that the PDF versions of vectorial graphics files are of optimal quality and often quite efficient, provided that font subsetting is used in creating them. Inasmuch as these PDF files can be immediately viewed by Acrobat Reader on most platforms, this conversion is of immediate benefit to essentially all users.

Gradually, it appeared that conversion to PNG and JPEG formats by various methods sometimes offers greater advantages. Fortunately, the solution to be described for PDF extends to PNG and JPEG graphics files.

The `\special` syntax in the DVI files used for EPS integration was most often (and by far) the one

used by Tomas Rokicki's "epsf.tex" [Rok]. Aiming to exploit pre-existing DVI files and maintain perfect coherence with Rokicki's DVIPS, we decided to have DVIPDF interpret the existing Rokicki syntax as the following example indicates:

```
\special{PSfile=test.eps llx=11 lly=22
      urx=33 ury=44 rwi=550 rhi=660}
```

This is probably the world's most common `\special` syntax![3]

— the unit for the first four "bounding box" entries is 1 big point. llx is the x-coordinate of the lower left corner of the bounding box, etc. Most often (but not always), this bounding box has simply been copied by TEX from the bounding box indicated in the EPS file header.

— the unit for the last two entries is 0.1 big point. One of the two entries tagged by "rwi" (for real width), and by "rhi" (for real height) may be absent, in which case uniform scaling is used. These last two entries specify the width and height of the integrated bounding box on the output page. The integrated box traditionally has its lower left corner placed at the DVI insertion point. It will have the the width and height prescribed.

As is well known, the (expanded) argument of this `\special` command is passed intact into the DVI file. On the other hand it is usually generated inside of TEX; the author sees only some high level commands, say those of "epsf.tex".

For both the EPS file "test.eps" (say) and the derived PDF file "test.pdf" the figure is located on a coordinate plane with unit of length = 1 big point; also, the scale and orientation are the same for both planes.

In case "test.pdf" was created by Ghostscript the two coordinate systems will be exactly the same. Then the rules of integration coming from Rokicki's DVIPS are applied without modification and the results are identical.

In case "test.pdf" was created by Distiller the two coordinate systems are related by a translation and some care is required required to make it predictable. We omit the details.

Such are the complications that arose from demanding perfect compatibility with legacy DVI files made for DVIPS. In fact MathCD has mostly used Distiller, encountering only occasional problems.

Fortunately, the reader of an article will be completely oblivious to such complications; only www site editors, CD-ROM editors, and conscientious authors are concerned.

Generalizing to PNG and JPEG bitmapped graphics There is an important variant of the above mechanism that is optimal for bitmaps. EPS and PDF are very general norms that can accommodate vectorial or bitmapped images; however, for bitmaps, EPS tends to be bulky and slow, and both both seem to obstruct the the recovery of embedded bitmaps. On the other hand, everyone has access to quite capable freeware/shareware bitmap manipulation tools such as XNview under Windows and linux/unix [Gou] or Graphics Converter for Macintosh [Lem], and these can produce EPS format at any time. Thus, to the extent that you wish to grant full control of bitmapped graphics to the reader of your article, you may wish to use a 'native' bitmapped norm.

The converse applies too: one can lock a PDF file or restrict its use in various ways. And it must be conceded that PDF manages to inherit the space efficiency of both leading public bitmapped norms PNG and JPEG.

Recall that PNG (= Portable Network Graphics) is the most efficient contemporary norm for faithful bitmap compression and is suitable for scientific figures and for most of the myriad uses which the commercial norm GIF enjoys on the web. PNG is typically 15% more compact than GIF. (Alas, the leading browsers Netscape and Internet Explorer, have been tardy and halfhearted in their support for PNG. It may be necessary for AcroDVI to support GIF.)

JPEG is the dominant 'lossy' norm for compression of low contrast color images such as photos. JPEG (like GIF) is well supported by current versions of the Web browsers.

Hopefully the above considerations will encourage more TEX users to exploit PNG and JPEG bitmaps. Those who are still restricted in TEX to vector graphics should be reminded, every time they see a web browser, that the full gamut of (still) bitmapped images, color included, as seen on the web, are begging to be used in TEX.

What we have said about PNG and JPEG being 'native' or 'editable' graphics formats is to some extent true for PDF. Indeed, the Windows graphics program Mayura Draw [Kar] uses it as its storage format; however, it does not read arbitrary PDF files.

It should be plausible from the above, that the preparation 'ab initio' of a manuscript in DVI format with PNG bitmapped graphics additives can

[3] It is not the simplest for the job; indeed one could get by without the "bounding box" entries, cf. [Si1].

use the conventional EPS integration mechanism. This applies not just to PNG but also to JPEG and PDF. Here are some details:– Convert the PNG graphics to EPS; integrate the eps file using the 'consensus' special command; and finally, replace the EPS file by the original PNG file. (The EPS file is usually bulky and is perhaps best discarded since it can be regenerated if the need arises.) The enduser then just pushes the DVI file onto the AcroDVI icon and viewing in Acrobat Reader will begin — using the original PNG graphics.

But there is a shortcut! It is unnecessary to generate an EPS file in XNview or similar program. Provided that in a certain configuration file one has specified

```
DoBBoxFile  =YES
```

one can instead first preview the PNG file by pushing its icon onto that of AcroDVI. The previewing creates, as a sideproduct, an auxiliary file with extension ".bb", which contains the 'BoundingBox:' comment as in an EPS file header. With a suitable macro package such as "boxedeps.tex" [Si1] (versions $\geq$ 1999) or the LaTeX graphics packages [CaR], the ".bb" file can be used *as is* in lieu of an EPS file.

Key roles for Ghostscript The first is on-the-fly integration by AcroDVI of EPS files into DVI format; optional settings of AcroDVI can enable this when Ghostscript is present. Useful for viewing legacy 'DVI plus EPS' postings!

When an author or publisher is preparing an article for publication in DVI format with graphics additives, the strategy should be to vary graphics format in order to maximize image quality, while minimizing bulk. Effort spent on this often holds surprises and pays handsome dividends. For some guidance, see the documentation of boxedeps [Si1] (versions $\geq$ 1999). Here we just mention that Ghostscript is a valuable converter to bitmap formats from PS, EPS, and some PDF; it has command line options for parameters like resolution.

Unfortunately, Ghostscript has its quirks as rasterizer. In particular, it is not quite the same as Acrobat Reader. In view of such difficulties it is advisable to urge authors to present originals of all graphics objects; this often makes life easier. In preparing MathCD, the lack of such originals has been more of a vexation than the lack of TeX source files!

We have mentioned that the Adobe PS and PDF based systems seem loth to surrender internally stored bitmaps. They can be likened to a bank so eager for deposits that it has forgotten to provide for withdrawals. When need for withdrawals comes, Ghostscript may be your best friend.

The medium molds the message One has to bear in mind that the various graphics norms and the various viewing mechanisms may influence what ultimately reaches the human eye. The pages of *TUGboat*, for example, are printed in black and white by photo-offset methods and will never faithfully represent the colored iris that is the icon for AcroDVI.

The quest for image clarity and beauty is an empirical art; with no testing, the results of photo-offset printing may be disappointing. We apologize in advance.

On the need for DVI's efficiency

There is currently at best tepid support for the use of efficient methods. What can be done efficiently by astute programming is by preference done by the liberal expenditure of RAM or disk space or processor power. This admittedly has the beneficial effect of driving the industries that create these fundamental resources.

Thus, for AcroDVI to be taken seriously, cogent evidence is wanted that the resources economized through maintaining and devellopping the efficient DVI norm can be used decisively.

This is perhaps most evident with CD-ROMs. A CD-ROM contains about 650 Mo of data. If exploited to archive mathematics in compressed DVI form (and no other) a CD-ROM could contain about 300,000 pages of mathematics. That is enough space to record all the mathematics currently on the xxx.lanl.gov 'e-print' archive (said to be about 200,000 pages). Alternatively, it is enough to distribute all the mathematics research articles published in one year (on paper *or* electronically). Again, it is enough space to reprint the whole of Annals of Math (the most prestigeous math journal) plus the whole of Crelle (the oldest math journal).

Going beyond mathematics, it might be possible to present complete encyclopedias on a single CD (or two) using compressed DVI (and graphics) files for storage, and AcroDVI for viewing. Currently, the most used storage format for encyclopedias is Microsoft's RTF (Rich Text Format). RTF enjoys efficiency comparable to that of DVI; viewing with Acrobat Reader is possible via Acrobat Distiller (or a subset). On the other hand, AcroDVI offers better typography, as well as freer viewer availability, and greater speed.

Although such projects may not be realized in the immediate future, MathCD is intended to hint at them all.

There should also be evidence that CD-ROM capacity will not grow so fast that it outstrips the increasing demand for such permanent storage. If it does, then it is plausible that there is room for waste. The spectacular 1000-fold growth of the capacity of inexpensive (say $500) hard disks in the last dozen years has fed wild expectations of storage technologies. But the reality for CD-ROMs is sobering. It is now known that the next (second) generation of CD-ROMs, called DVD (Digital Versatile Disc) coming about 15 years after the first, will be based on a simple evolution of the current CD-ROMs: a rough doubling of density is involved plus use of both sides of the disk. The capacity gain to 4.5 Gig will be somewhat less than 10-fold (not 1000-fold).[4] This is a factor frighteningly similar to the wastage factor that would be imposed by universal adoption of the bulky PDF norm. Furthermore, it could be a decade before the new CD-ROM norm is sold at the affordable prices of today's CD-ROMs, since that is the time it took for today's CD's to reach mass consumer prices. This is one of the strongest arguments for retaining the efficient DVI norm. Happily, DVD readers will accept today's CD-ROMs.

The current pause in progress of telephone modem speeds gives additional arguments. The 56 kilobaud telephone modems of today are considered to be the last gasp of a tired technology up against what is called the 'Shannon limit'. In this case, a dramatic switch to ADSL (Asymetrical Digital Subscriber Lines) is being promoted with great speed gains: nominally 1.5 megabits/sec download and .5 megabits for upload (but, in practice, perhaps only 1/3 or 1/4 of that). However, there remains the question whether and when this technology will be as widely available and as affordable as the present modem technology.

Although hard disk capacity has been growing prodigiously, electronic libraries such as ELibMath EMS (http://www.emis.de) could come to need DVI's polyvalence and efficiency. Thus far, an inexpensive ($500) hard disk of a few gigabytes is ample to lodge the entire library of a few dozen journals. At such an affordable price for storage, dozens of mirror copies of the library have been established worldwide. As time passes, journals are not only multiplying and individually growing but are offering more and more formats for downloading, notably the bulky PDF. If and when this causes overflow of the current generation of hard disk, either more economical formats will have to be used or ELibMath's costs will abruptly jump.

The xxx.lanl.gov 'e-print' server has shown the way on economy, by deriving essentially all formats from ".tex" format, on demand. This server successfully deploys immense expertise and resources under unix systems and manages to compile any document from ".tex" format and derive on-the-fly any other format the user requests for viewing/printing. To do much the same on a CD-ROM, but using ".dvi" format, seems just within the realm of possibility — relying heavily on the greater simplicity and wide acceptance of DVI format. The first edition of MathCD will nevertheless be far more liberal (heteroclyte) than the xxx.lanl.gov server.

We conclude that the economy and polyvalence of TEX's original DVI norm may indeed be the magical stuff from which dreams can be woven.

DVI viewing is a leading role for DVIPDF

As a front end to Acrobat Reader for DVI viewing, how well does AcroDVI face competition?

There are several interesting *indirect* competitors — that we merely mention in historical order: Ghostscript/Ghostview, then Distiller [Acro] teamed with Acrobat Reader, and most recently pdfTEX [Ha1] [Ha2], (cf. [Le3]), teamed likewise. Perhaps the most dangerous competitor would be Acrobat Reader itself using a radically improved version of PDF format — but there no sign of that.

One direct competitor of AcroDVI is Malyshev's BaKoMa DView [Ma2] which not only has the broadest typographic capabilities in the TEX world, but also the ability to output PDF files. We leave the user to judge the relative virtues of these two. Both will be provided on MathCD.

A second direct competitor is dvipdfm by Mark A. Wicks [Wic], which surfaced in 1998. It is an autonomous converter quite similar in concept to DVIPDF. Executable binaries have already been published on CTAN for 2 platforms, W9X/NT and i386 linux. The *TUGboat* referee informed us of many compiled dvipdfm binaries on the "TEX-Live 4" CD-ROM about to be distributed by TEX Users Group. The platform / OS combinations served are: DEC alpha / OSF4, HP / HPUX10, i386 / Linux, SGI / IRIX6.2, RS6000 / AIX4.1.4, Sparc / Solaris 2.5-2.6, and Windows (32 bit).

Thus far, neither of these competitors has provided close integration with Acrobat Reader. It is probably fair to say that both are presently aiming

[4] Double that for two-layer versions — whose durability is unfortunately in doubt.

at PDF publication, *not* DVI viewing. They are not yet competing frontally but they soon could.

As for support of the most used `\special` commands, BaKoMa DView is well advanced, thanks to adherence to dvips syntax (Rokicki's). AcroDVI is less advanced here, and has some catching up to do, because it originally fashioned its own `\special` syntax; basic functionality for color and hyperreferences are, however, present. Least adapted for viewing legacy DVI files is dvipdfm — because of its reliance on 'pdfmark' syntax; however it has good basic `\special` functionality.

AcroDVI is also alone in supporting the important PNG norm, not to mention the integration of PDF, PNG, and JPEG with a *single* `\special` syntax, nor the direct viewing of bitmaps.

On the other hand, the recent wide porting of dvipdfm and its universal CD-ROM distribution with *TUGboat* could well eclipse DVIPDF, and with it AcroDVI. If that is our fate, we hope that both DVIPDF and AcroDVI will nevertheless be remembered as seminal 'proofs of feasibility'.

The idea of exploiting DVIPDF and Acrobat Reader together as a feature-rich DVI reader has been a subject of discussions between us two authors ever since the 1996 TUG meeting in Dubna, Russia. For a long time, this project remained on a back burner while basic features of DVIPDF were perfected by the first author. As the MathCD project took shape, it offered many stimulating design challenges such as convenient graphics handling and the use of legacy DVI files with legacy syntax. The last year has brought substantial progress that seems to justify our early optimism.

To obtain the current AcroDVI news and distributions, follow pointers on the MathCD web pages [Mcd]. Hopefully, ftp postings will be maintained on ftp://sirius.ihep.su and the the CTAN archive.

References

[1] Acrobat, a series of products by Adobe Inc., including Acrobat Reader, and Acrobat Distiller; the former is free while the latter is sold; but low prices for Distiller have been available to academic users in many countries. Supported platforms include: Windows 3.x, Windows 9x, Windows NT, Macintosh, OS/2-Warp, Linux, IBM-AIX, SunOS, Solaris, SGI-IRIX, HP-UX, and Digital Unix.
There is an active news list comp.text.pdf that can provide user support. See also [5], in particular Nelson Beebe's comments of 23 April 1999.
http://www.adobe.com/prodindex/acrobat/readstep.html
ftp://ftp.adobe.com/pub/adobe/acrobatreader

[2] Bienz Tim, Cohn Richard and Meehan James, Portable Document Format Reference Manual, Adobe Systems Incorporated, 1993; Addison-Wesley Publishing Company, ISBN 0-201-62628-4,
http://www.adobe.com/supportservice/devrelations/PDFS/TN/PDFSPEC.PDF

[3] Carlisle David and Rahtz Sebastian, *The graphics and graphicx packages*,
CTAN `ftp://ftp.tex.ac.uk` etc.

[4] Deutsch L. Peter, Aladdin Ghostscript, electronic distribution,
ftp://ftp.cs.wisc.edu://pub/ghost/aladdin.

[5] Electronic Math Journals Discussion List, archive:
http://math.albany.edu:8800/hm/emj
glimpse index:
http://nyjm.albany.edu:8000/SF/emjsearch.html
1125

[6] Gougelet Pierre-E, XNview, freeware bitmap editor/converter for MS Windows, Linux, etc. http://latour.univ-paris8.fr/~pierre.

[7] Han The Thanh, Petr Sojka, and JiříZlatuška, "The joy of TEX2PDF — Acrobatics with an alternative to DVI format", *TUGboat* 17, 3, September 1996, pages 244–251.

[8] Han The Thanh, "The pdfTEX Program", Proceedings of Euro-TEXConference, St. Malo, March, 1998; Cahiers GUTenberg, n° 28–29, mars 1998, pages 231–241.

[9] Karunakaran Rajeev, Mayura Draw, graphics program whose native format is a dialect of PDF, electronic distribution: http://www.wix.com/mdraw210.zip

[10] Knuth Donald, "TEX The Program", Addison Wesley, Reading, Mass., 1986.

[11] Lang Russell, GSview, electronic distribution:
ftp://ftp.cs.wisc.edu/pub/ghost/rjl/gsview*.zip

[12] Lemke Thorsden, Graphics Converter, shareware ($30 approx.) bitmap editor and converter for Macintosh, http://www.lemkesoft.de

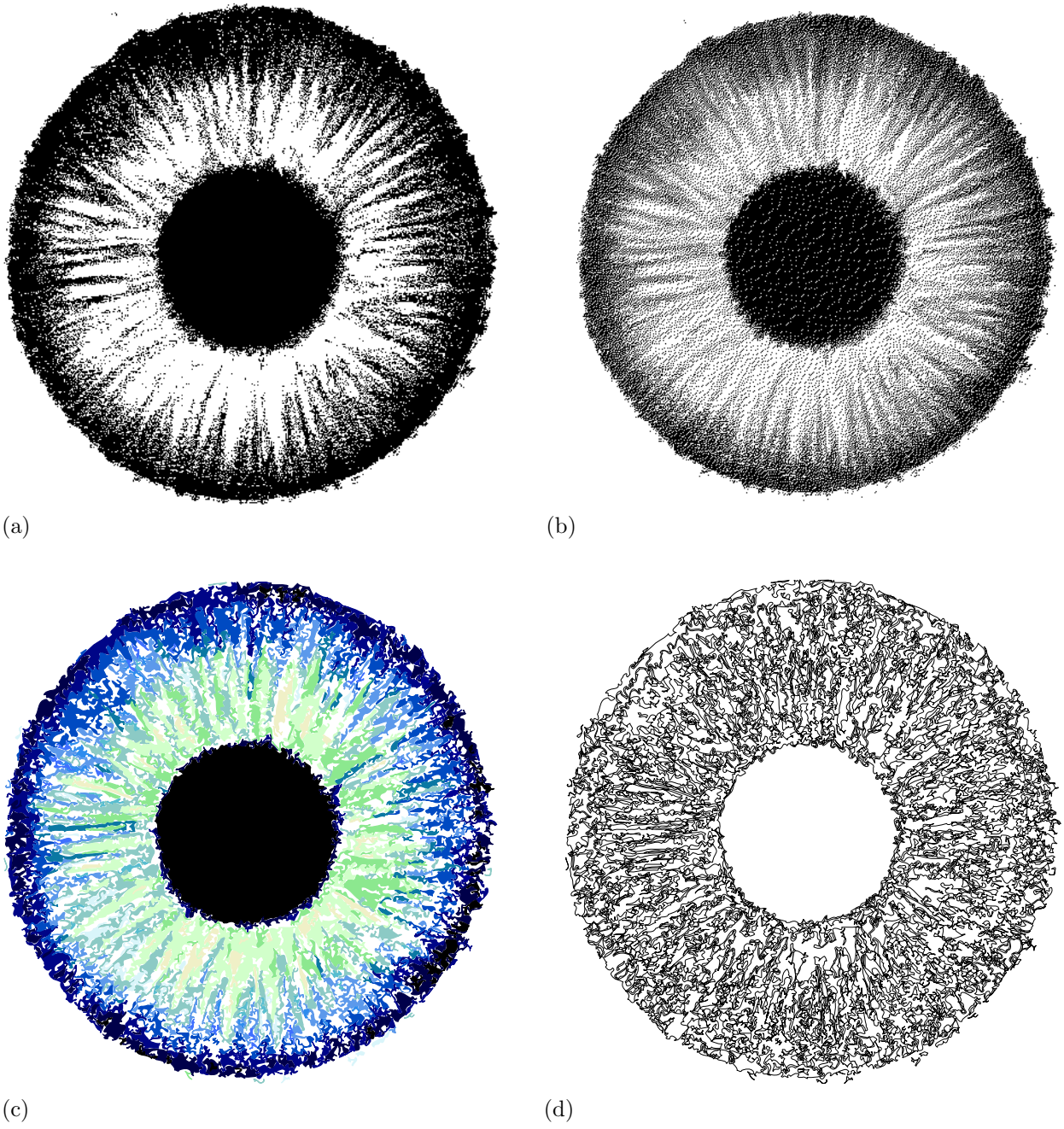
[13] Lesenko Sergey, "The DVIPDF Program", *TUGboat* 17, 3, September 1996, pages 252-254.

[14] Lesenko Sergey, "DVIPDF and Graphics", "TEX and Scientific Publishing on the Internet", Proceedings TUG'97, San Francisco, 28 July – 1 August 1997; *TUGboat* 18, 3, September 1997, pages 166–169.

[15] Lesenko Sergey, "DVIPDF and Embedded PDF", Proceedings of Euro-TEXConference, St. Malo, March, 1998; Cahiers GUTenberg, n° 28–29, mars 1998, pages 231–241.

[16] "MathCD", a CD-ROM in preparation, devoted chiefly to electronic mathematics journals and mathematical software:
http://www.math.washington.edu/~burdzy/MathCD.html
http://rsp.math.brandeis.edu/MathCD.html
http://topo.math.u-psud.fr/~lcs/MathCD.html

[17] Rokicki Tomas G., DVIPS, available on CTAN in dviware directory.



(a) (b)

(c) (d)

For the reader's amusement, here are several rather different black and white renditions of the iris, all of which derive from one multicolored pastel original contributed by Tina and Keira Miyata. This was scanned in 32 bit color to a 450 by 450 pixel bitmap, and stored as a 57 Ko file in JPEG format.

(a) arises from a suitable projection to black-white [Lem]; size 30 Ko as ".eps.gz", 20 Ko as ".pdf.gz", 14 Ko as PNG.

(b) is derived by Floyd-Steinberg filtering [Lem]; size 36 Ko as ".eps.gz", 26 Ko as ".pdf", and 18 Ko as ".png",

(c) arises from (color) vectorisation by 16 regions of flat color according to Adobe Streamline (Version 3); size 144Ko as ".eps.gz", 146 Ko as ".pdf.gz", and 41 Ko as ".png".

(d) is the 1000 or so curves that are the boundaries of the 16 colors of (c); size 203 Ko as ".eps.gz", 186 Ko as ".pdf.gz", and 53 Ko as ".png".

Figure 1: