

Graphics in L^AT_EX

Claudio Beccari

Email claudio.beccari@polito.it
Address Politecnico di Torino, Turin, Italy

Abstract This tutorial describes some facilities offered by L^AT_EX and its extension packages for producing line art graphics directly in the source document. Some of these facilities are standalone, in the sense that they do not require functionalities of external programs; some, on the other hand, rely on external programs.

1 Introduction

When a book is produced it is very unlikely that it does not contain some graphic material, be it pictures, diagrams, line art, and the like.

In this paper I will focus on ways to produce graphics within the T_EX or L^AT_EX systems. I will not discuss how to insert graphics that are already available in files produced by external equipment (for example photo cameras) or external programs, such as any of the commercial or freeware drawing programs available outside the T_EX-system distributions, or even METAPOST[†], that comes with the distribution.

The standard package `graphics` and its “extension” `graphicx`, already available with the system, offer simple macros with which such inclusion may be performed with a number of optional actions available, such as rotation, scaling, trimming, clipping, et cetera.

Maybe these extension packages assume that the user is well aware that if L^AT_EX is being used the only graphic formats that can be included are those belonging to the PostScript family (including the METAPOST[†]output), while if pdfL^AT_EX is being used, the only formats admissible are PDF, JPEG, PNG, and METAPOST[†]output, provided in the latter that its file extension is changed to `mps`. It is true that some other formats may be included if a separate file containing

the bounding box information is available, but as far as I know this possibility is seldom used because it requires some hacking ability.

At the same time, any or most T_EX-system distributions contain such little programs as `ps2pdf` in order to transform (encapsulated) PostScript file into PDF ones, or `jpegtops` to transform JPEG files to PostScript format. A number of such programs are available so that a transformation from one format to another allows the use of any image with both “plain” L^AT_EX or pdfL^AT_EX, or ConT_EXt, or...

Here I focus my attention on native and imported graphic functionality that is available with (pdf)L^AT_EX and ConT_EXt. With that I mean the standard graphics macros and the extended macros available with extension packages that conform with the L^AT_EX “language”; METAPOST will be left out of this tutorial, even if it belongs to the T_EX-system bundle, because its macros require the user to learn another language. The interested reader may look up the METAPOST documentation, [1]; be aware, though, that ConT_EXt MetaFun package, [2], allows exploiting modern T_EX-system implementations in order to simultaneously run the T_EX and METAPOST engines and to produce line art graphics directly from one source file.

2 The standard L^AT_EX picture environment

L^AT_EX was born in 1984 with a native picture environment that allowed one to draw lines, vectors, circles and ‘ovals’ with some limitations; special fonts were used to draw all graphic objects. This was a severe limitation, because at L^AT_EX’s birth all fonts usable with L^AT_EX could contain only 128 glyphs; this was true with text fonts as well as with L^AT_EX special drawing fonts.

This implied that there were only two thicknesses available for all graphic objects, and, worst of all, the line and vector slopes and the circle diameters were available in only a limited number. Line slope parameters could only be integer relatively prime numbers in the range $-6 \text{ --- } +6$, while for vectors they could range only within $-4 \text{ --- } +4$. This means that the only lines and vectors that could be drawn are represented (with just positive slopes) in figure 1. Filled circles (disks) were limited to diameters from 1pt to 15pt, while unfilled circles were limited to diameters from 1pt to 40pt; circles of diameter larger than 15pt were drawn as four quarter circles, and each of these were available also for the

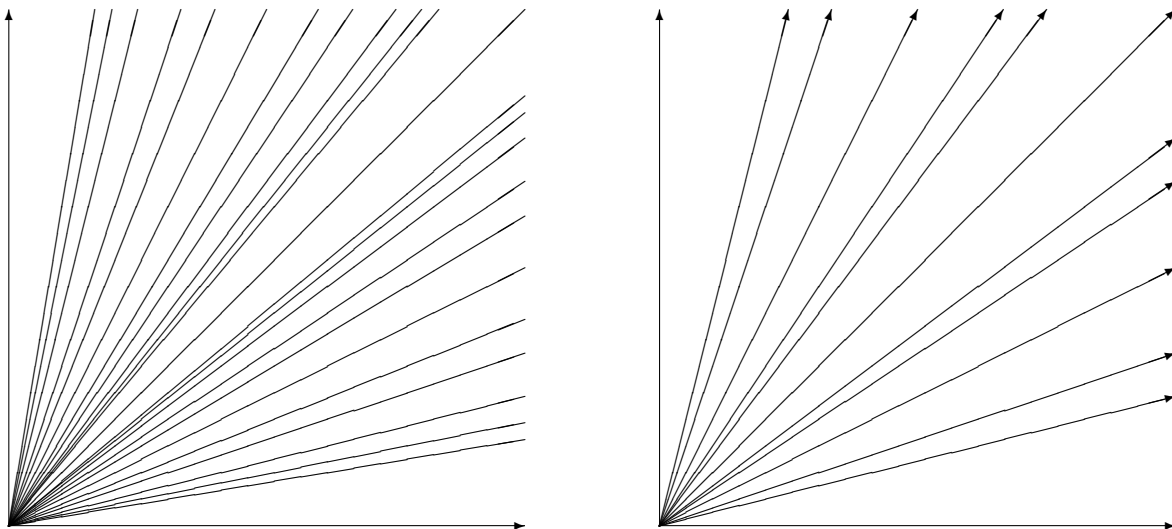


Figure 1: Possible positive slopes for lines and vectors in the standard `picture` \LaTeX environment

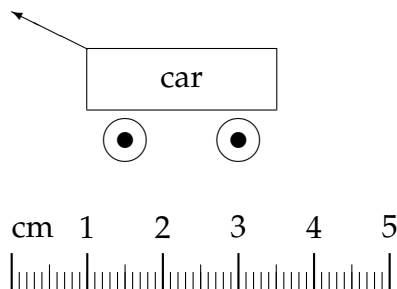


Figure 2: A \LaTeX `picture` environment simple drawing

rounded corners of ‘ovals’.

This environment allowed typesetting of text by means of extensions of the `\makebox` macros, and framed text by means of `\framebox`; fine tuning of the position of the text allowed placement of any text in the precise required position and, most important of all, the fonts used were the same ones used in the text of the document as a whole. Typographically, this feature was and remains the most valuable of this built-in environment. Figure 2 displays the same drawing Lamport used in his handbook to describe his `picture` environment.

3 PiCT_{EX}

The first graphic extension available on the market was PiCT_{EX}[3]. It was designed to work with plain T_{EX}, in an initial stage of T_{EX}'s "history"; after L^AT_{EX} became available, some macros were set up in order to let L^AT_{EX} import those PiCT_{EX} macros and use them at almost full power. Those were the times when T_{EX} and L^AT_{EX} were run on mainframes and PCs were in their infancy; the T_{EX}-system was already available for those small PCs, but the memory limitations were so strong that on such machines it was quite normal to get the dreadful message that there had been a memory overflow and processing aborted.

Nowadays the PC memory limitations have substantially vanished, and a medium price PC has enough main memory, virtual memory and disk space to do more than the mainframes available in the eighties. As a matter of fact, a new version of PiCT_{EX}, named m-PiCT_{EX}, [4], is part of the ConT_{EX}t system as a normal accessory.

Although the functionalities of PiCT_{EX}, or most of them, were transferred to more modern extensions, I often dig into its code in order to learn the exceptionally clever tricks implemented by Michael Wichura in order to have the T_{EX}-engine perform fractional decimal number computations, from logarithms to trigonometric functions, and the like.

The problem with PiCT_{EX} (and also with more recent macro sets) is that arbitrary lines are drawn by setting a myriad of dots partially superimposed on one another so as to simulate continuous line drawing; this multitude of dots implies such a heavy memory usage that remains locked until the picture is shipped out into the dvi file; a single picture may saturate the available memory during its processing, not to speak of what happens when several figures containing PiCT_{EX} graphics are queued by L^AT_{EX} before shipping them out.

In spite of these drawbacks, PiCT_{EX} is very good at drawing line art of arbitrary complexity, including that form of graphical mathematical objects known as commutative diagrams. It is true that 2-dimensional commutative diagrams may be typeset with the `cmd.sty` package made available by the American Mathematical Society; but PiCT_{EX} can draw 3-dimensional ones; a very attractive commutative diagram decorates the first page of the PiCT_{EX} manual, which is not free, so I can't reproduce it; a partial imitation is shown in figure 3, where I used the standard macros available within the L^AT_{EX} picture environment.

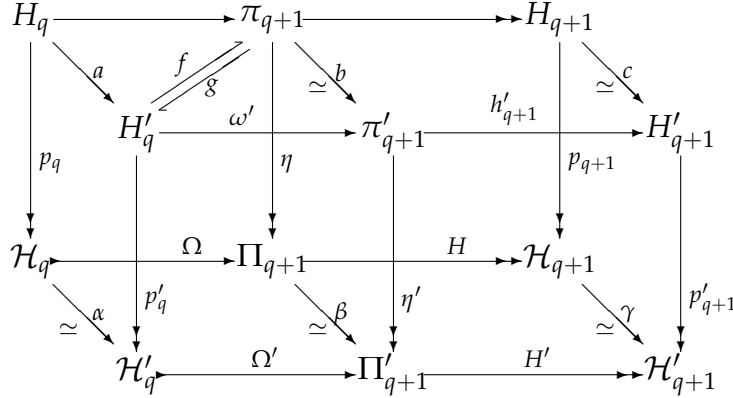


Figure 3: A three dimensional commutative diagram

4 The epic and eepic picture extensions

In 1986 Sunil Podar published his extension package `epic` [5], to the standard \LaTeX picture environment; in 1988 Conrad Kwok published the further enhancement `eepic` [6].

Both were conceived so as to relieve the strong limitations of the standard \LaTeX picture environment, in particular the limitation on the line and vector slopes and the limited range of circle radii.

Both programs were conceived for use under the ‘old’ \LaTeX , today known as $\text{\LaTeX} 209$, in contrast with the new $\text{\LaTeX} 2_{\epsilon}$ (which is not ‘new’ anymore, since it is about twelve years old). In particular `eepic` was conceived in order to translate pictures drawn with the `xfig` program into a set of macros that `epic` could use for composing complex drawings. `xfig` is a drawing program, originally from UNIX platforms, that can output its contents into a variety of formats, including \LaTeX picture commands. Nevertheless, such commands bear the limitations described above, so that the `fig` format is preferred, but such a format must be translated into commands \LaTeX can actually execute by itself or with the intermediary of a macro package. This is exactly what `eepic` is for.

It must be said that most of the limitations of the old \LaTeX picture environment are overcome by the set of macros available with the `pict2e` extension package, which I will speak about in section 8. Therefore these extensions `epic` and `eepic`, although not obsolete, are essentially outdated.

5 The curves package

The CTAN archives contain the package prepared by Ian MacLaine-Cross initially in 1991, then revised for L^AT_EX_{2 ϵ} . The latest release in the year 2000 is the current one, [7].

The package `curves` accepts a number of options designed to simplify the output file. These options allow many graphic objects to be drawn by the programs that process T_EX's output (device drivers). Without these specifications, the curves drawn by the package are done by the superposition of a multitude of black disks; this creates a lot of overhead and when possible it is better to use the specific driver option. While there is an option for `dvips` there is unfortunately not one for `pdfTEX`. If a PDF output format is desired, it is necessary to obtain it through the lengthy path L^AT_EX \longrightarrow `dvips` \longrightarrow `pstopdf`.

In any case, the main package `curves` contains many new features compared with the standard situation; in fact all of the graphic objects can be drawn with 'arbitrarily' thick lines and lines with arbitrary slopes, and curves may be drawn by means of second-degree Bèzier splines (parabolas) where it is necessary to specify only the interpolation nodes, since control points are determined by the macros themselves. If the user is inclined to specify few nodes, the macros try to do their best, but more often than not curves end up having spurious loops, especially if inflection points are implied. But if the user does not spare the interpolation nodes, the curves turn out to be very nice and smooth, even if inflection points are involved. Let's remember that second-order Bèzier splines are those used to describe the contours of the TrueType fonts, while the third order splines are used to describe fonts in the METAFONT output format and in PostScript Type 1 format.

This package allows one to draw objects, save them and redraw them with 'arbitrary' transformations (rotation, x -scaling and y -scaling) so that fine composition of such objects is possible. Figure 4 represents a perspective view of a square washer drawn with curves.

6 PSTricks

The previous package `curves`, although capable of working as a standalone L^AT_EX macro package, is the first example of a drawing package where some of the

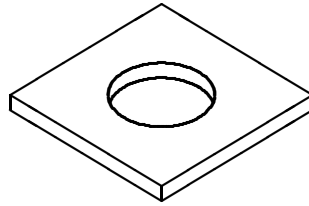


Figure 4: A square washer drawn with curves

functionality is provided by an external program, in this case `dvips` and a few other similar PostScript-based drivers.

The set of packages that are collectively known as `PSTricks` do the same, in the sense that they exploit the PostScript language to a great degree by writing to the output file, a `dvi` one in our case, all the `\special` commands that introduce the raw PostScript code necessary to draw all of the required objects.

The PostScript language is a very powerful programming language specifically designed to describe a typeset page and all the objects it contains; of course images and line art are at home in a typeset page, so that graphics handling is an integral part of the PostScript language.

Basically this language can handle fonts and their glyphs and can draw lines and curves and fill the paths they enclose with colors and/or patterns; but the most important feature is that PostScript can define macros more or less in the same way that the \TeX -engine does. These macros contain a full set of control statements, so that PostScript is capable of doing actions that one might reserve only for higher programming languages. The difficult part of the PostScript language is that it uses Reverse Polish notation, which seems unnatural to humans. The code appears cryptic to anybody who is not a PostScript programmer (at least it is cryptic to me). Fortunately, Timothy van Zandt, who wrote the interface, spares the user from Reverse Polish notation by providing a large set of \TeX macros that allow the user to write in the usual (\LaTeX) style. These macros translate the (\LaTeX) information into PostScript code. In addition, the macros do the necessary calculation for transferring all the internal quantities in typesetter's points (72.27pt to 1in) into PostScript points (72bp to 1in); the difference might appear small and may be negligible, but after many calculations the accumulated error affects the precise positioning of graphic objects. In any case the user does not have to learn the PostScript language.

It would be too lengthy to describe the available commands; suffice to say that the bundle of PSTricks packages covers almost any possible graphics situation; from structured graphs, to electric circuits, from optics setups to mechanical springs, from box diagrams to 2- and 3-dimensional graphics.

The point is that L^AT_EX can process files where one or more PSTricks packages have been used to draw something; the raw PostScript code written in the output dvi file by means of the `\special` commands may be processed only by further programs that understand and execute PostScript code; typically this program is `dvips` and the result of this process is a PostScript file. This is good for many applications, but more often than not a PDF file is required, so that the PS file must be ‘distilled’ by any of the various programs that convert a PS file into a PDF one. The normal process, therefore, is L^AT_EX \longrightarrow `dvips` \longrightarrow `pstopdf`; many T_EX-system distributed editors already contain a ‘button’ to click that execute the complete triple process, so that the user need not be bothered.

But all this implies also that the user cannot employ `pdfLATEX` as the engine to process an input L^AT_EX file, because the `pdfLATEX` engine is not capable of understanding and executing the raw PostScript code, in spite of the fact that the uncompressed PDF page description language is a subset of PostScript.

This incompatibility shows up in other areas, for example the inclusion of external graphic material; since L^AT_EX is the only program that can process PSTricks macros and their `\special` commands, this implies that all external graphic material must be in encapsulated PostScript format, besides the `METAPOST` output; if the user wants to include pictures coming from a digital photo camera, s/he must first transform the picture format, probably JPEG, into an encapsulated PostScript.

7 X_Y-pic

Another very powerful graphics package that relies on using an external software for actually rendering graphic objects is X_Y-pic, [9].

This package is designed to work with L^AT_EX, plain T_EX, and AMS-T_EX. It can draw diagrams of any ‘mathematical’ kind, from commutative diagrams to those used in category theory, automata theory, algebra, neural networks, and database theory.

The feature that makes it so versatile is that mathematicians were the first ones

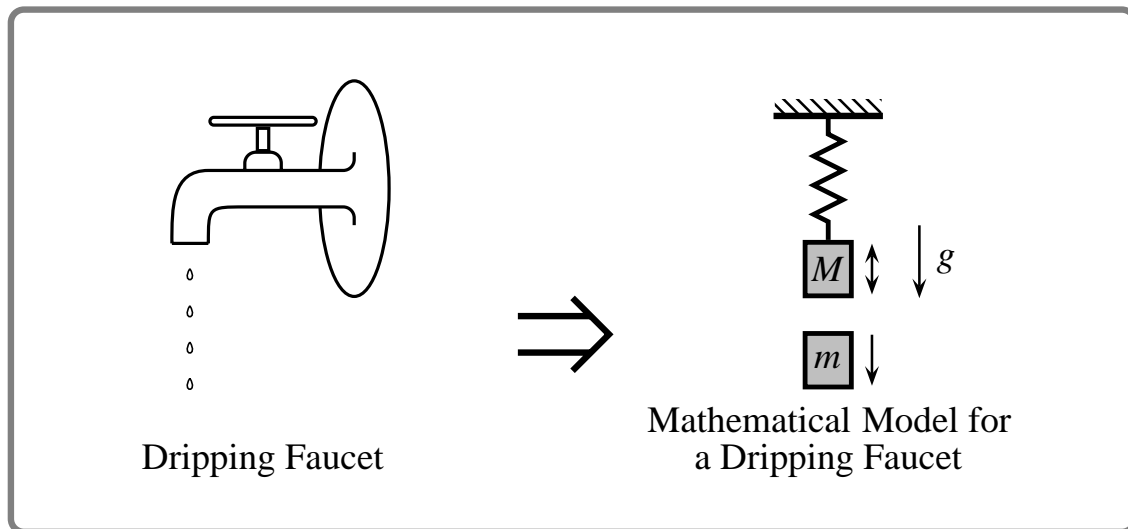


Figure 5: A PostScript line art figure drawn with PSTricks; this appears on the cover page of the PSTricks manual

to use plain $\text{T}_{\text{E}}\text{X}$, and, apparently keep preferring this incarnation of the $\text{T}_{\text{E}}\text{X}$ -system rather than using more user-friendly packages; plain $\text{T}_{\text{E}}\text{X}$ and $\text{AMST}_{\text{E}}\text{X}$ are similar and refrain from using those bells and whistles that make $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ so widespread. Nevertheless, the developers adapted it also to $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$, but the users must cope with what the authors consider a ‘logical composition of visual components’. The idea is brilliant, but in practice I find the syntax of the various object descriptions a little too cryptic, or maybe I am too old to learn new sophisticated graphic description languages.

I can read the manual, see the examples and agree that the power of $\text{X}_{\text{Y}}\text{-pic}$ is highly underestimated; I am guilty on my own for not daring to sit down and try hard to get the best out of it. In any case I show a simple figure taken from the manual, figure 6; the commutative diagram is simple, but it contains objects such as the curved and dotted arrows that are not easily drawn with other packages.

What is amazing with $\text{X}_{\text{Y}}\text{-pic}$ is the fact that its syntax is extremely compact; the whole commutative diagram of figure 6 is described in just six lines of code in the two-column formatted $\text{X}_{\text{Y}}\text{-pic}$ Guide.

One little point of warning: $\text{X}_{\text{Y}}\text{-pic}$ uses several characters as special purpose ones; among these characters there is the double quote `"`, which acts also as an ac-

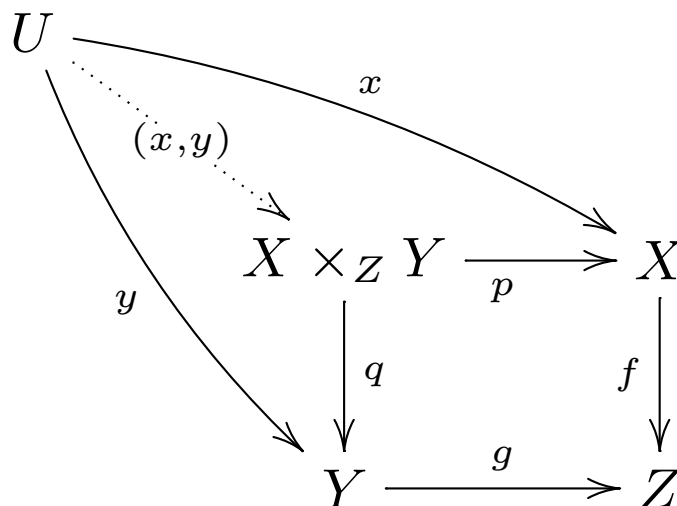


Figure 6: A simple commutative diagram drawn with $\text{\texttt{Xy-pic}}$

tive character in most babel language description files, perhaps all languages but English. In order to use $\text{\texttt{Xy-pic}}$, it is necessary to turn off the shortcuts associated with the double quote by means of the babel command `\shorthandoff{"}`; this can be done just upon entering the `xypic` environment, and without this action $\text{\texttt{Xy-pic}}$ is virtually unusable when the current language option is different from `english`!

8 The `pict2e` and `curve2e` extensions to the standard $\text{\texttt{L\^A T\^E X}}$ picture environment

Leslie Lamport, in his second edition of the $\text{\texttt{L\^A T\^E X}}$ handbook [10], fixed the syntax of a new extended picture environment, where most if not all limitations of the standard implementation could be overcome: unlimited slopes of lines and vectors, any circle radius, arbitrary line thickness also for curved lines, real third order Bèzier curves, et cetera.

Eventually in 2003 H.Gäßlein and R.Niepraschk published an implementation of Lamport's extensions with their package `pict2e` [11], which implements these extensions by means of the driver capabilities.

Figure 7 shows some of the new possibilities: vectors of any slope and thick-

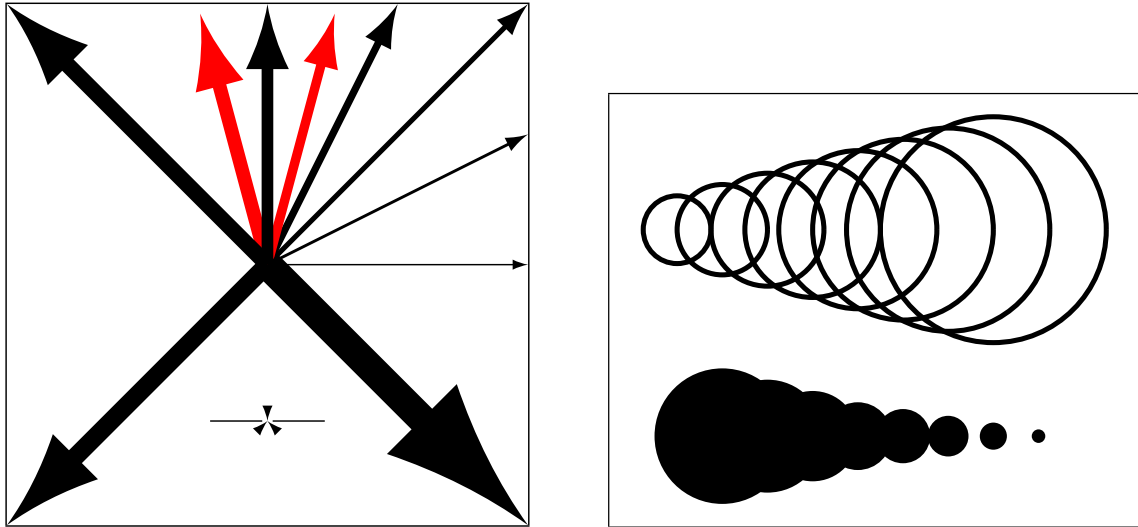


Figure 7: Some examples of `pict2e` capabilities

ness and circles and disks of any diameter. Of course that is a simple example, but it gives a good idea of the progress made with the `pict2e` extension.

I did some personal extensions to the already available extensions offered by `pict2e` [12]. My goal was to prove the point that `pict2e` could be freed from the remaining constraints of integer slope parameters imposed by Lamport to the new package syntax; it is true that the new slope parameters are constrained to be any signed three-digit integer, but this does not improve the performance of tracing lines and vectors.

As any \TeX -system user knows, the \TeX -engine is not capable of making computations with decimal fractional numbers; the only ones it deals with are the scale factors for dimensions. It is not difficult to create macros for simulating addition, subtraction and multiplication, but division is another thing; let's not speak of square roots or, even worse, trigonometric and other transcendental functions. Many of the described graphic packages worked around these limitations with very clever algorithms, but apparently Lamport did not want to extend his new picture environment that far.

But with a suitable division routine even the line and vector tracing algorithms can exploit any fractional slope parameter; and this opens the door for 'turtle graphics'. My package `curve2e` does exactly that, allowing also for circu-

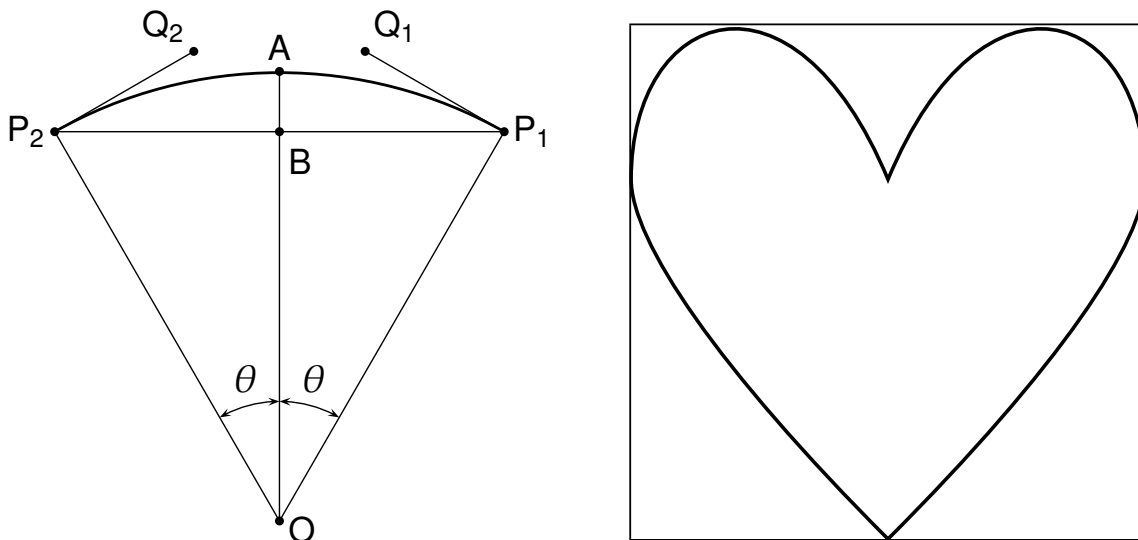


Figure 8: Some examples with package curve2e

lar arcs, circular vectors with the arrow tip at one or both ends, to generic curves obtained by specifying just the interpolation nodes and the tangent directions in such nodes. The simple examples in figure 8 show some of the functionality of this package.

9 The pgf and xcolor packages

The acronym PGF stands for ‘Portable Graphics Format’; the package pgf [13], implements or re-implements all commands of the standard \LaTeX picture environment and those of the graphics package so as to create a coherent set of commands capable of doing most of the graphic operations in \LaTeX and in pdf\LaTeX . Many commands are added to that fundamental set so as to extend the \LaTeX graphic capabilities almost to the level of PSTricks; its capabilities are further extended by means of the xcolor package that on its own extends the possibilities of color handling offered by the standard color package.

This package automatically examines the default configuration files and inserts in the output files the $\backslash\text{special}$ commands suited for the output driver; if \LaTeX is being used (and therefore dvips or some of its kin is used) the suit-

able PostScript code is inserted in the output file. If pdfL^AT_EX is being used, the `\specials` contains PDF code that is therefore immediately active within the output PDF file.

Since PDF code at the moment is not as powerful as the PostScript code, when pdfL^AT_EX acts as the typesetting engine it is not possible to exploit the full power of the PSTricks package, but Till Tantau is working hard to extend this excellent graphic package so that the full functionality may be achieved.

A personal experience: in September 2006 I bought myself a new laptop; I installed all the packages I am used to, among which, of course, the whole T_EX distribution, the latest version available at that time. I had to prepare my presentation for the Marrakech TUG2006 Conference, so I polished up what I had already prepared using my previous laptop. I did not realize that the PGF package had changed so much; I used the facilities I was used to, but I was not aware that a major revision had added a lot of new features to that package. If I had known, I would have withdrawn my contribution to that conference. Fortunately enough, in my presentation I was praising the PGF bundle, but I was not describing it, so that my presentation was still “presentable”...

What I discovered while preparing this paper was that a completely new interface had been created so that PGF graphics could be drawn in a much simpler way.

The new PGF bundle, version 1.10, contained a new package and its new graphic environment that is called TikZ; this acronym stands for “TikZ ist kein Zeichenprogramm” (TikZ is not a drawing program), but this is an understatement set forth by Till Tantau in order to avoid bragging.

Yes, dear readers, PGF and its new package tikz is a very nice drawing interface can produce almost anything. It does not have the full power of PSTricks, but goes a long way in that direction. The manual explains in good detail what can’t be done with the PostScript interface, and, what’s best, it explains that the PGF format and program were specifically designed to be used with pdfL^AT_EX, where it performs best.

I had little time to get familiar with and to master TikZ, but after my first experiments and using it for actual drawings I can definitely state that I will stick to this software for the duration. Of course, the first thing I have to do is transform the circuit drawing environment I spoke about in my earlier paper [12] in order to exploit the best features of TikZ. But for other drawings I’ll use TikZ

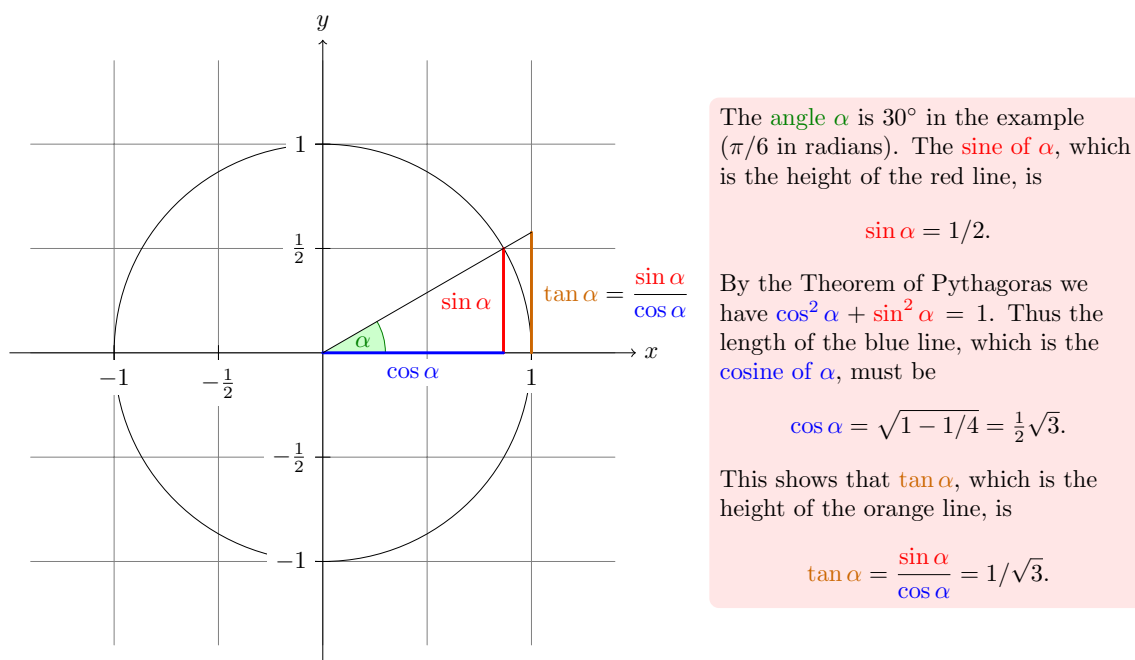


Figure 9: An example taken from Tantau's manual [13]

directly. I have already been using the PGF bundle for handling pictures when using the presentation software beamer (another achievement of Till Tantau's), and now almost everything can be done with it.

Tantau says that TikZ is intended to offer the user a simplified and uniform interface to allow composing drawings with the least possible fuss; in his manual he wrote an introductory small teaching demo so as to produce figure 9. I will not copy the code here, because anybody can find it in the manual [13], which in any case has to be used intensively (it's a about 370 pages long and it is very well done and well organized) because of the multitude of drawing commands available for any possible situation.

As you can see in figure 9 the program allows for both black and white graphics and color graphics, and the text uses the same fonts as the default ones in the PDF document; this is one of the features that is missing from most external drawing packages and forces the user to circumvent this vacancy with a number of tricks. TikZ frees the user from using any trick for this purpose because it is fully integrated with (pdf)LaTeX.

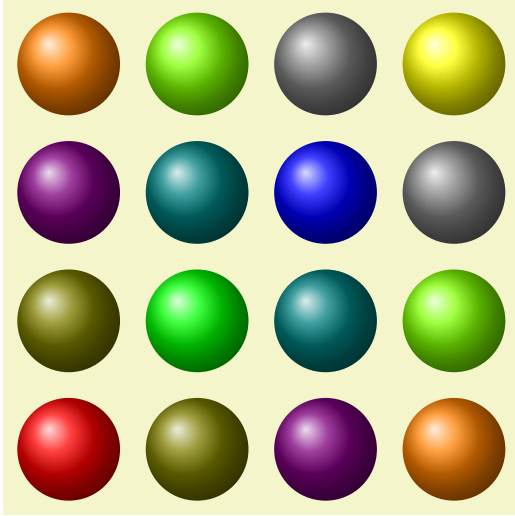


Figure 10: Color gradients and mixtures from Tantau’s manual [13]

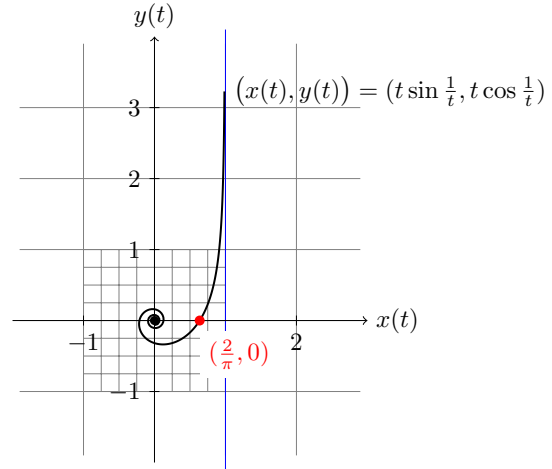


Figure 11: A spiral plotted by TikZ the coordinates of which are computed by GnuPlot; from Tantau’s manual [13]

Color is dealt with using advanced commands; in particular it is possible to do gradients and to mix colors; figure 10 displays an array of balls with circular gradients and mixes the four colors that appear in the matrix secondary diagonal.

The PGF bundle contains a lot of libraries of additional commands; the standard drawing objects are available also without resorting to these libraries, but the latter add a lot of options and functionality. It is possible to color the inside of closed paths and to make node drawings and automata diagrams. There is an enormous variety of “arrow” tips. There is a large array of background choices, including the page background over which the regular pages are typeset. It offers entity-relationship diagrams, mindmap diagrams (very attractive), a choice of background patterns for technical drawings, Petri-net drawings, and so on.

The bundle handles the drawing of plots which may be defined as a series of two- or three-coordinate pairs or triplets for doing 3D drawings. It can interact with GnuPlot, a freeware external program that can compute and store coordinate pairs or triplets for subsequent use by pdfL^AT_EX¹; actually it is not necessary to

1. This feature requires activation of the `\write18` (pdf)T_EX facility, which is disabled by default, but may be enabled for particular tasks to reduce the risks of using it. Note that if it was enabled by default it could allow the user of the ever present malware to launch destructive programs,

launch GnuPlot from within pdfL^AT_EX; it can be run in deferred mode by the user himself, and in a second run of pdfL^AT_EX it can retrieve the files produced by GnuPlot, and then the plotting facility of TikZ executes the plots; see figure 11 for another example taken from Tantau’s manual [13]. This plotting facility is completed by the possibility of marking the plotting nodes (the experimental points) with different symbols so that plots presenting different lines can be better distinguished even if colors are not used.


The “snake” library allows changing the shape of most drawing objects, specifically straight or curved lines. A couple of special shapes I appreciate are the curved arrows, that allow connecting different nodes with non-intersecting and non-blurring arrows or lines, and the helical shape that is used so frequently in technical drawings.

The “shape” library contains the definitions of a lot of typical shapes, that can be colored inside and that contain text or symbols used in a lot of schematic block diagrams. A useful one is the “forbidden sign” that can be used in a variety of situations. The example on Tantau’s manual contains the text “Smoking” so that the strong meaning “It’s forbidden to smoke” emerges with due emphasis.

The “to path” library allows one to draw curved lines with third-order Bèzier splines without the need of specifying the node tangent directions. Such Bèzier third-order splines may be used freely even without the use of this library provided the node tangents are specified. This library renders the curve drawing task much easier.

The “tree” library allows the drawing of trees of nodes with parents and children so that most of the logical visual drawings can be performed with no difficulty.

The TikZ syntax is rather simple; its statements start with a command, continue with options and coordinates and finish with a semicolon; they remind of the METAFONT or METAPOST commands and syntax, but TikZ is not a replacement of METAPOST or METAFONT. The options and the object names are similar to some PSTricks commands, but TikZ is not a replacement for PSTricks.

A true novelty is the possibility of making simple inline drawings such as this:  without opening an environment. The intuitive code for drawing the above shaded button is the following:

without the user being aware of what’s happening.

[...] such as this: `\tikz \shade[ball color=red] (2ex,1ex) circle(1ex);`
without opening [...]

A final comment: *TikZ* may be used also with plain \TeX , pdf\TeX , and Con\TeXt ; it is thought of as a modern package usable in different situations and with different typesetting engines. It recognizes by itself which engine is being used and changes its performance, syntax and interface accordingly, and eventually issues the necessary `\special` commands, which are transparent to the user, and which match the specific driver needed for rendering the document in human readable form. Those who use `dvips` will need to run the necessary filters, but pdf\LaTeX users do not have to do anything in order to display their work, because pdf\LaTeX outputs directly the readable document.

10 Conclusion

This tutorial scans a variety of packages for drawing simple or complicated drawings, with a variety of programming interfaces, and with different levels of sophistication in the complexity of the final output.

Certainly the basic standard \LaTeX picture environments pale in comparison to the performance of *PSTricks* or those of the PGF bundle, even if the standard environment is upgraded with `pict2e` and `curve2e`. However, this simple drawing tool may be used by learning only a few commands. *Xy-pic*, *PSTricks*, and *TikZ* definitely have a larger feature set but also have a steeper learning curve.

With *TikZ* I am personally experiencing the satisfaction of drawing technical line art that I wasn't able to do with other \LaTeX drawing interfaces; the results are worth the little strain of learning a new interface language.

This tutorial shows only small examples of how to use the drawing tools discussed. For more detail the interested reader can view the required handbooks and manuals. These are available from the international CTAN archives, and are also provided with the various bundles, most of which are already included in most modern distributions of the \TeX system, be it for a Windows, a UNIX/Linux or a Macintosh platform.

If you explore the CTAN archives you can certainly find other graphic drawing interfaces; either they are very specific or they are overridden by the packages I describe in this tutorial. Nevertheless if you need to draw something very special

(as I have to, for example, with my electronics circuits) it's a good idea to explore CTAN, where most likely the `graphics/` folder contains the specific package for your needs.²

I set forth some little warnings derived from my experience. To tell the truth, I do not have a lot of experience with some of these packages yet. Nevertheless I collected information from colleagues who use them. The warning about disabling the double quote active character when using `Xy-pic` derives from a small article written on the web site of the Italian TUG, written by a user who was very disappointed when the Italian language description file was upgraded to include some “double quote” driven markups. When I introduced them into the language description file, I had the impression that all other language description files, except for English, contained such markups, so why not in Italian, where the keyboard deficiency is remarkable³ There are other circumstances where some markups are necessary and the double quote stands out as the only character that Knuth did not use for something else (well, the hexadecimal numbers, but one can use the octal or the decimal ones instead). I am surprised that no such complaints were raised from users writing in other languages.

I encourage (pdf)L^AT_EX users to experiment extensively with the PGF package; after a while they will become addicted...

References

- [1] John Hobby, `METAPOST`, [CTAN:graphics/metapost/](#)
- [2] Hans Hagen, *MetaFun*, [CTAN:manuals/metafun-p.pdf](#)
- [3] Michael Wichura, *PiCT_EX*, [CTAN:graphics/pictex/](#)
- [4] Michael Wichura *m-PiCT_EX*, [CTAN:help/Catalogue/entries/m-pictex.html](#)
- [5] Sunil Podar, *The epic package*, [CTAN:macros/latex/contrib/epic/](#)

2. Why, then, am I going to rewrite my electronics circuit drawing package, if there is one already there in CTAN? Because even if that package contains a larger variety of components, they are drawn according the North American standards which are not standard in Europe.

3. The Italian standard keyboard does not contain the ever present T_EX characters such as both curly braces (!), the tilde and the grave accent; the latter is the only accent to be used on any Italian vowel, except for ‘e’ which takes the acute accent instead.

- [6] Conrad Kwok, *The eepic package*, [CTAN:macros/latex/contrib/eepic/](#)
- [7] Ian Maclaine-Cross, *The curves and curveslm packages*, [CTAN:macros/latex/contrib/curves/](#)
- [8] Timothy van Zandt, *The PSTricks bundle*, [CTAN:graphics/pstricks/](#); the documentation is in [CTAN:applications/PSTricks/](#)
- [9] Kristofer Høgsbro Rose and Ross Moore, *The Xy-pic bundle*, [CTAN:applications/Xy-pic/Xy-pic.html](#)
- [10] Leslie Lamport, *A document preparation system — L^AT_EX — User's guide and reference manual*, Addison Wesley Publ. Co., Reading, mass., 2nd ed., 1994.
- [11] Hubert Gäßlein and Rolf Niepraschk, *The pict2e package*, [CTAN:macros/latex/contrib/pict2e/](#)
- [12] Claudio Beccari, *The curve2e package*, [CTAN:macros/latex/contrib/curve2e/](#)
- [13] Till Tantau, *TikZ and PGF*, [CTAN:graphics/pgf/](#)