LaTeX and graphics: Basics and packages

Aleksandra Hankus and Zofia Walczak

1 Introduction

There are the number of distinct ways of producing graphics, each with advantages and disadvantages in terms of flexibility, device independence and ability to include arbitrary TeX text. In this paper we will discuss two ways of placing graphics inside a LaTeX document. The first is about graphics imported to the TeX file from an external graphic program, and the second about creating graphics inside a TeX document. We will discuss documents with graphics which are intended to be printed, and presentations made with the `beamer` class.

2 Importing graphics into LaTeX

When we want to include graphics in a document we have to take into account that LaTeX cannot manage pictures directly. LaTeX just creates a box with the desired size for the image we want to include and embeds a reference to the picture, without any other processing. This means we have to take care of the format and size of the images to be included. This is not such a hard task because LaTeX supports the most common picture formats around.

2.1 The graphicx package

Since LaTeX can't manage pictures directly, we load the `graphicx` package for help by placing the following in the preamble of our document:

```
\usepackage{graphicx}
```

The image formats we can use depend on the driver that `graphicx` is using, and since the driver is automatically chosen according to the compiler (TeX variant) being used, in practice the allowed image formats depend on the compiler.

The only format you can include while compiling with `latex` is Encapsulated PostScript (EPS). An EPS file declares the size of the image, which makes it easy for LaTeX to arrange the text and the graphics in the best way. EPS is (typically) a vector format, meaning that it can have very high quality if it is created properly, namely with programs that are able to manage vector graphics.

If we are compiling with `pdflatex` to produce a PDF, we have a wider choice. We can insert graphics in JPG, PNG, PDF. EPS format can also be used if it is converted to PDF; in current distributions, that happens automatically with the help of `epstopdf`.

The same LaTeX source can be compiled in both `latex` and `pdflatex` without any change, as long as we avoid using particular packages. We can use both compilers for documents with pictures as well, if we remember to provide the pictures in proper format (both EPS and one of JPG, PNG or PDF).

2.2 Including graphics

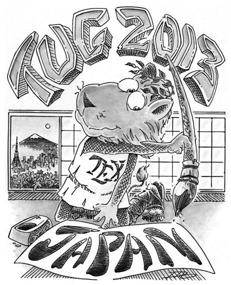
After we have loaded the `graphicx` package in the preamble, we can include images using the command `\includegraphics`, whose syntax is the following:

```
\includegraphics[arg1,arg2,...,argn]{imgname}
```

The arguments in square brackets are optional, whereas arguments in curly braces are compulsory.

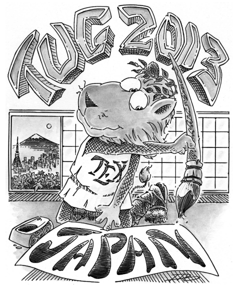
2.3 Examples

For scaling images we can use the optional argument `scale=`⟨*number*⟩:

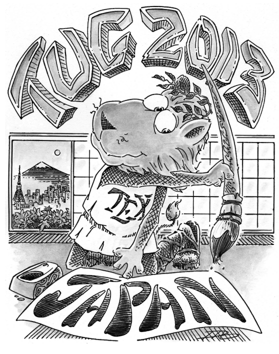


`\includegraphics[scale=.16]{name}`

We can give image dimensions with either or both of the optional arguments `width=`⟨*number*⟩ and `height=`⟨*number*⟩. When we specify only one or the other, the second will be chosen proportionally. When we specify both, the image will be resized without preserving proportions.



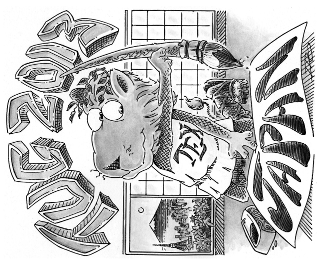
`\includegraphics[width=3cm]{name}`



`\includegraphics[height=4.5cm]{name}`

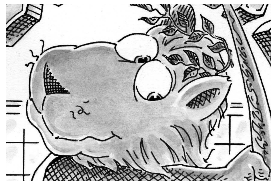
```
\includegraphics[width=5cm,height=3cm]{name}
```

To rotate the image, the option `angle=`⟨*number*⟩ is used.



```
\includegraphics[scale=0.18,angle=90]{name}
```

And finally, here is an example of how to crop an image to focus on one particular area of interest. For this purpose we use the `trim` argument; in order, its parameters are ⟨*left*⟩ ⟨*bottom*⟩ ⟨*right*⟩ ⟨*top*⟩.



```
\includegraphics[trim=.5cm 1.1cm .3cm .5cm,clip,
  width=3.5cm]{name}
```

We can specify which image file is to be preferred by `pdflatex` through this preamble command:

```
\DeclareGraphicsExtensions{.pdf,.png,.jpg}
```

This specifies the files to include in the document (in order of preference), if there exist files with the same name but with different extensions.

2.4 The `figure` environment

There are many situations where we want to add to the image a caption and possibly a cross-reference. We can do that with the `figure` environment, but we have to take into account that this is a so-called "floating" environment. The minimum required code is the following:

```
\begin{figure}[pos]
\includegraphics{image name.png}
\end{figure}
```

where the optional argument `[pos]` stands for the allowed positions of the figure on the page. Such a float placement specifier can consist of the following characters in any order: `htb!p`. For example, specifying `[bt]` means that our picture can be placed on the bottom or top area of the page.

The above code is relatively trivial, and doesn't offer much functionality. The next sample shows an extended use of the figure environment which is almost universally useful, offering a caption, label and centering the image, placed at either the current position ("here") or the top of the page.

```
\begin{figure}[ht]
\centering
\includegraphics{name}
\caption{Caption}
\label{fig:1}
\end{figure}
```

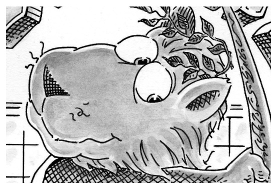


Figure 1: Just the lion

3 Supporting the creation of graphics directly in LaTeX

There are many packages to do pictures in (LA)TeX itself rather than importing graphics created externally, starting with simple use of the LaTeX `picture` environment.

3.1 The `picture` environment

The `picture` environment is used to draw pictures composed of text, straight lines, arrows and circles. The objects in the picture are positioning by specifying their coordinates. The first picture environment was created by Leslie Lamport.

The basic syntax for the environment is:

```
\begin{picture}(width,height)(xoffset,yoffset)
  picture commands
\end{picture}
```

Thus, the `picture` environment has one mandatory argument, which specifies the size of the picture. The environment produces a rectangular box with width and height determined by the values of these two arguments. Coordinates are specified in the usual way with respect to an origin, which is normally at the lower-left corner of the picture. The optional positioning argument following the size argument can change the origin. If we decide to modify our picture by shifting everything, we can just add the appropriate optional argument.

Everything that appears in a picture is drawn by the `\put` command.

Using the picture environment we can "decorate" our previous image, for example to add to our image "glasses".

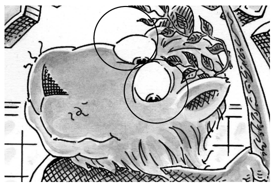


Figure 2: Adding glasses

```
\setlength{\unitlength}{1cm}
\begin{picture}(4,2.5)
\includegraphics[width=3.5cm]{image name.jpg}
\put(-1.9,1.9){\circle{.8}}
\put(-1.5,1.2){\circle{.8}}
\end{picture}
```

3.2 XY-pic package

XY-pic is a special package for drawing diagrams. It works smoothly with most formats, including LaTeX, AMS-LaTeX, AMS-TeX, and plain TeX. To use it, add the following line to the preamble of your document:

```
\usepackage[all]{xy}
```

where "all" means you want to load a large standard set of functions from XY-pic, suitable for developing complex diagrams. Below we show an example.

$$\begin{tikzcd} G/T \arrow[r, "="] \arrow[d] & G/T \arrow[d] \\ E \arrow[r, "\hat f"] \arrow[d, "\pi"] & BT \arrow[d, "p"] \\ S^{2k} \arrow[r, "f"] & BG \end{tikzcd}$$

```
{
G/T \ar[r]^=\ar[d]
& G/T \ar[d]\\
E\ar[r]^{\hat f}
\ar[d]^{\pi}
& BT \ar[d]^{p}\\
S^{2k}\ar[r]^f& BG
}
```

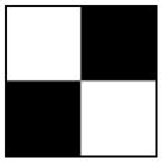
3.3 PSTricks — the pstricks package

Here, the basic package to use is pstricks, to be loaded with the usual command \usepackage in the document preamble. PSTricks commands are usually placed in a pspicture environment, whose mandatory argument gives the coordinates of the upper-right corner (the lower-left is the origin by default).

```
\begin{pspicture}(x1,y1)
  pstricks commands
\end{pspicture}
```

Here is a basic example. The \psframe command draws an unfilled rectangle, and its starred version makes it filled.

```
\begin{pspicture}(6,4)
\psframe(1,1)(3,3)
\psframe*(1,1)(2,2)
\psframe*(2,2)(3,3)
\end{pspicture}
```



3.4 PSTricks — the psfrag package

The psfrag package allows LaTeX users to replace text strings in EPS files created by external programs with LaTeX text or equations. To use psfrag, create an EPS file and then perform the following steps

- In the document, use the \psfrag command to specify the PostScript text in the EPS to be replaced, and the replacement LaTeX string. This makes the specified substitution occur in any subsequent \includegraphics command issued in the same environment.
- Use the \includegraphics command as usual.

The \psfrag command has the following syntax:

```
\psfrag{PStext}[posn][PSposn][scale][rot]
    {text}
```

Remark: PSfrag cannot be used with pdfTeX. If such substitution is needed, one option is to use the LaTeX-to-DVI-to-PostScript-to-PDF route that was used before pdfTeX. PSfrag also doesn't work with beamer. An alternative there is to use the Ti*k*Z package (with the EPS figure converted to PDF).

3.5 The amscd package

The amscd package provides a CD environment that emulates the commutative diagram capabilities of AMSTeX version 2.x. This means that only simple rectangular diagrams are supported, with no diagonal arrows or more exotic features.

$$\begin{tikzcd} A_{PL}(Y) \arrow[r] & A_{PL}(X) \arrow[r] & A_{PL}(F) \\ (\Lambda V_Y, d) \arrow[r] \arrow[u, "m_Y"] & (\Lambda V_Y \otimes \Lambda V, d) \arrow[r] \arrow[u, "m"] & (\Lambda V, \bar d) \arrow[u, "\bar m"] \end{tikzcd}$$

```
$$ \CD
A_{PL}(Y) @>>> A_{PL}(X)
               @>>> A_{PL}(F)\\
@A{m_Y}AA @A{m}AA
               @A{\bar m}AA\\
(\Lambda V_Y,d)  @>>>
(\Lambda V_Y\otimes\Lambda V,d)
@>>> (\Lambda V,\bar d)
\endCD $$
```

Remark: The amscd package does not work with the beamer class.

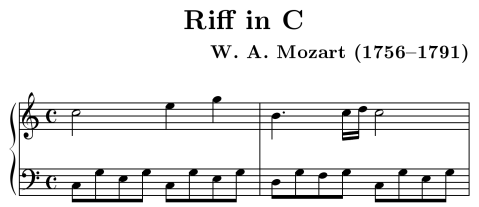
3.6 MusiXTEX

MusiXTEX is a set of macros and fonts which enables music typesetting within the TEX system.

It contains symbols for staves, notes, chords, beams, slurs and ornaments, ready to be arranged to form a sheet of music.

But it must be told how to position those symbols on the page. This can be done manually, if you elect to proceed by entering MusiXTEX commands manually into an input file.

However most users will find it far less taxing to let such decisions be made largely by the preprocessor PMX, which also uses a much simpler input language than MusiXTEX. Here is an example of the output.



Remark: MusiXTEX needs LATEX, which is automatically invoked when needed; but in general, LATEX and MusiXTEX cannot be combined. For typesetting a large musical score it is better to use another alternative.

3.7 Graphics with PGF/TikZ

One possible solution for drawing graphics directly with TEX commands is PGF/TikZ. TikZ can produce portable graphics in both PDF and PostScript formats using either plain (pdf)TEX, (pdf)LATEX or ConTEXt. It comes with very good documentation, and there is an extensive collection of examples at `http://www.texample.net/tikz`.

Using TikZ in a LATEX document requires loading the tikz package

```
\usepackage{tikz}
```

somewhere in the preamble, as usual. This automatically loads the pgf package. To load further libraries use

```
\usetikzlibrary{list of libraries}
```

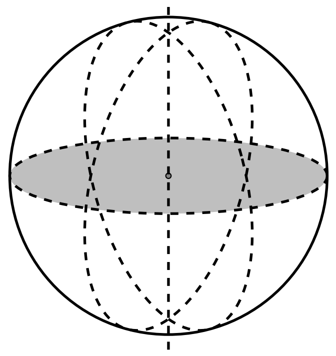
Some useful existing libraries are: arrows, automata, backgrounds, calendar, chains, matrix, mindmap, patterns, petri, shadows, shapes.geometric, and there are plenty more.

Drawing commands are usually enclosed in a tikzpicture environment:

```
\begin{tikzpicture}[options]
tikz commands
\end{tikzpicture}
```

or alternatively we can use the `\tikz` command:

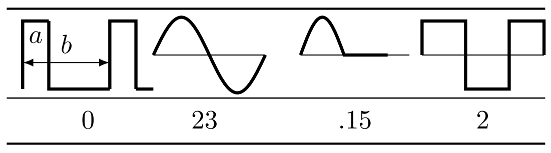
```
\tikz[options]{tikz commands}
```



If we specify the bounding box (it's an optional argument to the environment) with the baseline option as we show here:

```
\begin{tikzpicture}
  [x=0.0714\textwidth,y=0.5cm,
   baseline=(current bounding box.east)]
\path[use as bounding box](0,-1)rectangle(2,1);
\draw (0,0)--(2,0);
```

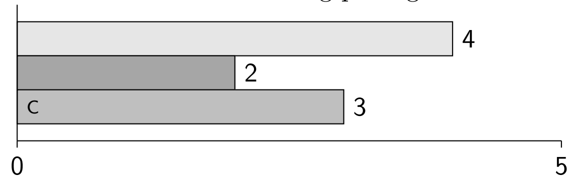
we can draw a table with different pictures in every cell in the row, all aligned together.



4 Some packages based on PGF/TikZ

4.1 The bchart package

bchart is a LATEX package for drawing simple bar charts with horizontal bars on a numerical x-axis. It is based on the TikZ drawing package.

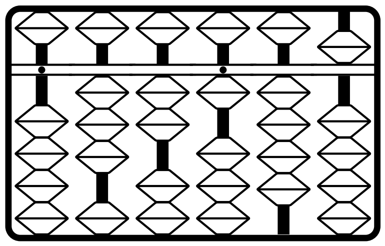


```
\begin{bchart}[max=5,scale=.9]
\bcbar[color=gray!20]{4}
\bcbar[color=gray!70]{2}
\bcbar[text=\scriptsize{C},color=gray!50]{3}
\end{bchart}
```

Remark: The bchar package can be used with both latex and pdflatex, and it also works with the beamer class.

4.2 The `pgf-soroban` package

The soroban is an abacus developed in Japan; the `pgf-soroban` package lets us typeset representations of soroban values. We load the package in the usual way, with `\usepackage{pgf-soroban}` in the preamble. There is no need to load any corresponding graphics package, as all required packages are loaded by the soroban package. The package sets a base unit as 1 mm, as well as other lengths. If we want to change the size, the units can be changed with, e.g., `\ladj{0.25}`. The soroban picture below represents the number 321.45.



```
\ladj{0.5}
\begin{tikzpicture}
\tige{1}{0}{1} \tige{2}{3}{0}
\tige{3}{2}{0} \tige{4}{1}{1}
\tige{5}{4}{0} \tige{6}{5}{0}
\cadre{6}
\end{tikzpicture}
```

There is also a soroban package for PSTricks, named `pst-soroban`.

References

[1] D.P. Carlisle, Packages in the 'graphics' bundle, 2005, `ctan.org/pkg/grfguide`.

[2] A. Delmotte, `pgf-soroban` — Create images of the soroban using Ti*k*Z/PGF, `ctan.org/pkg/pgf-soroban`.

[3] M. Goossens, F. Mittelbach, S. Rahtz, D. Roegel, H. Voß, LaTeX Graphics Companion, second edition, Addison-Wesley, 2007.

[4] H. Kopka and P.W. Daly, *A Guide to LaTeX 2$_\varepsilon$*, fourth edition, Addison-Wesley, 2003.

[5] T. Kuhn, bchart: Simple bar charts in LaTeX, version 0.1.2, `ctan.org/pkg/bchart`.

[6] L. Lamport, *LaTeX: A Document Preparation System*, Addison-Wesley, second edition, 1994.

[7] L. Lamport, *LaTeX: A Document Preparation System*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2004 (in Polish).

[8] E. Rafajłowicz and W. Myszka, LaTeX, Zaawansowane Narzędzia, Akad. Ofic. Wydawn. PLJ, Warszawa, 1996 (in Polish).

[9] D. Taupin, MusiXTeX. Using TeX to write polyphonic or instrumental music, Version 1.15, `ctan.org/pkg/musixtex`.

[10] Z. Walczak, *LaTeX for the Impatient*, Wydawn. Uniwersytetu Łódzkiego, 2012 (in Polish).

◇ Aleksandra Hankus
Institute of Mathematics,
University of Silesia, Poland
`aleksandra.hankus (at) us dot edu dot pl`

◇ Zofia Walczak
Faculty of Mathematics and Computer Science,
University of Lodz, Poland
`zofia.walczak (at) math dot uni dot lodz dot pl`