Textpos: absolute positioning of text on the page

Norman Gray
(http://nxg.me.uk)

Version 1.7g, 2010/09/30

This package facilitates placing boxes at absolute positions on the LaTeX page. There are several reasons why this might be useful, but the reason which originally motivated this package is to help produce a large-format conference poster. However the facility is also useful for filling in forms or other special purpose layout.

This package provides a single environment, which contains the text (or graphics, or table, or whatever) which is to be placed on the page, and which specifies where it is to be placed.

The package tries not to get in the way. That is, you should be able to use most of the apparatus of LaTeX in your poster, such as section headings, citations, graphics inclusion, and so on. Please let me know if you experience problems in this respect.

This package requires the services of Martin Schröder's package everyshi. If this is not already part of your TeX installation, you will need to download this package from CTAN. See <http://www.tex.ac.uk/tex-archive/macros/latex/contrib/supported/ms/> or one of the other CTAN hosts.

An article describing Textpos appeared in TUGboat in 2004 (Norman Gray, 'Absolute Positioning with Textpos', TUGboat **23** (3/4), pp341–4, 2002, available at <http://www.tug.org/TUGboat/Contents/contents23-3-4.html>). Textpos has a home page at <http://purl.org/nxg/dist/textpos>.

1 Description

Load the package as usual, with

```
\usepackage⟨package-options⟩{textpos}
```

The ⟨*package-options*⟩ are as described in section 1.1.

`textblock` The environment is used as follows

```
\begin{textblock}{⟨hsize⟩}(⟨hpos⟩,⟨vpos⟩)
text...
\end{textblock}
```

The ⟨*hsize*⟩ and ⟨*hpos*⟩ arguments are given in units of a module `\TPHorizModule`, and ⟨*vpos*⟩ is given in units of a module `\TPVertModule`. You set these using the command `\setlength{\TPHorizModule}{`⟨*dimen*⟩`}`, and similarly for `\TPVertModule`. The arguments may be any dimension, and you may use the modules as units elsewhere in your document if you wish to, for example in `\makebox[2\TPHorizModule]{gnus}`. The text in the environment will be set in a box ⟨*hsize*⟩ modules wide, and placed on the page with its upper left corner at the position (*hpos,vpos*). As is natural in TEX, the ⟨*vpos*⟩ parameter indicates distance *down* from the 'anchor point' (see below).

The {`textblock`} parameters may be whole numbers or fractional. If you want or need to give explicit sizes here, see the {`textblock*`} environment below.

Notice that the positioning arguments for the {`textblock`} command – the coordinates `...(`⟨*hpos*⟩`,`⟨*vpos*⟩`)` – are in *round* brackets, not curly ones. This is in imitation of the `picture` environment, and whether or not this is sensible, it's not going to change now.

relative & absolute mode

This package works in two modes, relative and absolute. In the first one, the default, the block-positioning parameters in the {`textblock`} envirionment are taken to be relative to an 'anchor point' which is the current position on the page, that is, the place where (the bottom left of) a character would appear if it were typed at this point. This will be appropriate if you are laying out text within a `figure` environment or the like. In this mode, you will typically give several {`textblock`} environments one after the other, so that they are all relative to the same point.

If, however, your entire document is to be laid out piece by piece (which is the case in the canonical use of the package, to lay out a poster), then you might want to be more sure of where the origin is. In this case, you make the package work in its absolute mode, by invoking it with the [absolute] option: `\usepackage[absolute]{textpos}`. In this mode, all the block-positioning parameters are given relative to a single origin on the page. By default, this 'anchor point' is the top-left corner of the page, but you may change it with the command `\textblockorigin{`⟨*hpos*⟩`}{`⟨*vpos*⟩`}`. Here ⟨*hpos*⟩ and ⟨*vpos*⟩ are dimensions such as '10mm', relative to the top-left corner of the paper. You may use this command only if the package was invoked with the [absolute] option. See also section 2.2.

The textblocks are placed on the page in the order in which they appear in the file. This means that later textblocks will be placed on top of earlier ones, which may matter if one or other contains, for example, a block of opaque colour (I believe this to be true in practice for all output mechanisms, though I doubt it's guaranteed in principle). This order was unspecified before textpos 1.7e; it was changed, and specified, in that version.

`\TPGrid`

You will often wish to set up a grid on your page. Rather than calculate and specify the two modules explicitly, you can set up the grid with a com-

mand \TPGrid{⟨*nhoriz*⟩}{⟨*nvert*⟩}, which sets \TPHorizModule to be ⟨*paper width*⟩/⟨*nhoriz*⟩, and \TPVertModule to be ⟨*paper height*⟩/⟨*nvert*⟩. This takes an optional pair of dimension arguments, which specify a coordinate, as follows.

```
\TPGrid[⟨x⟩,⟨y⟩]{⟨nhoriz⟩}{⟨nvert⟩}
```

If these are present, then the modules are set up to leave a border of the given size around the grid. That is, \TPHorizModule is set to be (⟨*paper width*⟩ − 2⟨*x*⟩)/⟨*nhoriz*⟩, and similarly for \TPVertModule. Further, if the package was given the [absolute] option, then the text origin is set to be (⟨*x*⟩,⟨*y*⟩) through a call to \textblockorigin (see below). For example, the declaration

```
\TPGrid[40mm,20mm]{10}{5}
```

would choose \TPHorizModule and \TPVertModule so as to give a grid of 10 intervals across and 5 intervals down, after leaving 40mm of a border on the right and left sides, and a 20mm border top and bottom.

You may give a optional argument to the {textblock} environment, specifying which point in the box is to be placed at the specified point:

```
\begin{textblock}{⟨hsize⟩}[⟨ho⟩,⟨vo⟩](⟨hpos⟩,⟨vpos⟩)
text...
\end{textblock}
```

The coordinates ⟨*ho*⟩ and ⟨*vo*⟩ are fractions of the width and height of the text box, respectively, and state that the box is to be placed so that the reference point (⟨*ho*⟩,⟨*vo*⟩) within the box is to be placed at the point (⟨*hpos*⟩,⟨*vpos*⟩) on the page. The default specification is [0,0], the top left of the box: [0,1] would be the bottom left, and [0.5,0.5] the middle.

\TPMargin By default, the box that is positioned by the {textblock} environment is a tight fit to the block of text (or other material) inside it. This looks rather odd when you also use the \textblockcolour macro described below, or specify a larger than default \TPboxrulesize, in order to get a noticeable border round a piece of text. In those cases, you will want to request a non-zero margin around your text. If you give the command

```
\TPMargin{⟨size⟩}
```

then the block of text inside the textblock will be decreased in width, enough to give the specified margin on each side. That is, the ⟨*hsize*⟩ of the textblock, and thus the coloured block (or the edge of the displayed border), remains the same, but the text width inside it decreases. The parameter ⟨*size*⟩ may be any non-negative dimension, and may as usual be in units of \TPHorizModule or \TPVertModule. The default behaviour is recovered by giving a ⟨*size*⟩ of 0pt.

There is a starred variant of this command, \TPMargin*{⟨*size*⟩}, where the argument must again be non-negative. In this case, the text block inside the box is set with the ⟨*hsize*⟩ specified in the {textblock} environment, but the coloured block is increased in size, such that there is again a margin of the specified size around the text block.

The {textblock} environment will most often be used in vertical mode. If it is called in horizontal (ie, paragraph) mode, however, it will silently create a paragraph break by inserting a \par command before the environment; it remains in vertical mode after the environment is finished. It should have no further effects on spacing, and if you find that it does, that's a bug. If you try to use the environment when in maths mode, the package objects (as it should!).

textblock*

There is an alternative, starred, form of the {textblock} environment. In the argument to the {textblock*} environment, the block width, and the block position (but *not* the specification of the block reference point) are given as absolute dimensions, rather than as numbers in units of the horizontal and vertical modules. Thus

```
\begin{textblock*}{⟨hsize⟩}[⟨ho⟩,⟨vo⟩](⟨hpos⟩,⟨vpos⟩)
text...
\end{textblock*}
```

produces a textblock of the given size, where this time ⟨*hsize*⟩, ⟨*hpos*⟩ and ⟨*vpos*⟩ are absolute dimensions, but ⟨*ho*⟩ and ⟨*vo*⟩ are pure-number offsets (that is, fractions of the width and height of the textblock), as above.

Each {textblock} environment takes up zero space on the page (which means, by the way, that it cannot detect that it's overprinting or being overprinted), so you can (and typically will) use several of the environments in a row to scatter text all over the page.

The package is compatible with the *calc* package, so that you may use calc-style expressions when specifying lengths. Thus

```
\usepackage{calc}
\textblockorigin{56.9055pt-10mm}{0pt+1cm}
\begin{textblock*}{10mm+14cm}(0.3cm*5,10\TPVertModule+5mm)
text...
\end{textblock*}
```

Note that you can only use calc-style expressions where you would specify a length with units, such as the width and location arguments of {textblock*} or the arguments to \textblockorigin – you can't use them when specifying a length in units of the horizontal and vertical modules, such as in the width and location arguments to the (unstarred) {textblock} environment.

Figure and table environments

Textpos changes the behaviour of any {figure} and {table} environments *within* instances of the {textblock} environment, in such a way that the figure or table contents do not float away from the {textblock} environment. For the same reason, \marginpar is forbidden within a textblock. It makes no change, however, outside the environment, where figures float as normal, and you are still able to use *textblock* within figures, as described above. Within a {textblock}, these environments do nothing beyond accepting the usual \caption command, which behaves correctly with respect to caption numbering and \label commands (it also respects \@makecaption, so you can tinker with that if you like that sort of thing). There's no real need to use either the {figure} or {table} environments within a textblock – you don't require them to allow \includegraphics

or {tabular} to work, for example – but many people automatically use them to surround graphics or tables, and also expect to use these environments to number figures and tables within {textblock} environments; they are therefore here on a principle of least surprise.

The support here is admittedly simple, and it is known to fail in the case where there are {figure} (or {table}) environments both inside and outside {textblock}s on the same page (the LoF is ordered incorrectly in this case, due to the different times that the various environments write to the .lof file). I don't have immediate plans to fix this: the situation is surely sufficiently rare as not to justify the (potentially fragile) complication of the fix – if you disagree, let me know.

Since both LaTeX's floats mechanism (that is, {figure} and {table}) and the [absolute] mode are designed to move content around, we can't expect them to play together nicely. With [absolute] mode on, the contents of a {textblock} inside a floating {figure} probably isn't going to end up where you expect it to. The only case I can think of where this would inconvenience you, is if you wanted some absolutely-positioned material to appear on the 'next' page. You might at first try to use a \begin{figure}[p] – that won't work, but the following will, if you first load the *afterpage* package:

```
\afterpage{%
  \newpage%
  \begin{textblock*}{297mm}(0mm,0mm)%
    \includegraphics{picture.png}% a full-page picture?
  \end{textblock*}%
  \null%
  \newpage}
```

This inserts a complete page, with some graphic on it, immediately after the end of the current page (thanks to Matthias Gloede for this technique).

\textblockcolour

The text blocks can be coloured in. If you load the *color* package, then the commands of that package, \textcolor, \pagecolor and the like, should work as usual. The *textpos* package adds a new command, \textblockcolour. If you give the command

\textblockcolour{⟨*colour*⟩}

all text blocks following will have their background filled with the specified colour, which must be one of the standard colours or have been declared in a \definecolor declaration in the document preamble. This colour may be overridden for individual text blocks by giving this command within the {textblock} environment. If you wish a block not to have any background colour, you can suppress it, again for one block at a time, with the command \textblockcolour{} inside the {textblock} environment.

\textblockrulecolour

You can similarly change the colour of the borders around the text block. If you give the command

\textblockrulecolour{⟨*colour*⟩}

then following text blocks will have their border in the given colour, which must again be either one of the standard ones of declared in the document preamble.

\textblockcolor For the benefit of those who observe Webster's spelling reforms, \textblockcolor is defined as a synonym for \textblockcolour, but those who would condemn such anaemic half measures can use \tekstblokkulur instead. There are also the corresponding spelling-reform variants of \textblockrulecolour.

\tekstblokkulur
\textblockrulecolor
\tekstblokroolkulur

1.1 Package options

There are several package options:

[showboxes] When you are laying things out, it can be useful to have the boxes drawn in for you. This option draws a box fitting closely round the set text. You can turn this on and off within the file using the \TPshowboxestrue and \TPshowboxesfalse commands.

[noshowtext] This suppresses the display of the text in each block (so it's not really usable without the [showboxes] option). The resulting box will be the correct size, but empty, unless the \textblocklabel command has been given. This can be useful when you are previewing a document.

[absolute] If this is present, then the positions on the {textblock} environment are taken to be absolute positions on the page. See above for more detail, and see section 2.2 for the interaction with the \newpage command.

[overlay] When using the absolute-position mode, the textblocks are placed under any other text on the page. This is normally what you want, but if you have page contents, and they have something which *obscures* the textblocks (for example, a block of opaque colour), then the positioned textboxes disappear. In this case, specify the option [overlay], to request that the positioned blocks of text overlay any other page contents, rather than being overlaid.

[verbose], [quiet] The package writes a few messages to the output, describing its calculations. These are potentially irritating, so you can turn them off with the [quiet] option or on with the [verbose] option. The default is currently [verbose], but this might change in future.

1.2 Package parameters

\TPHorizModule **\TPHorizModule** The length unit which is used for the horizontal positioning and size parameters of the {textblock} environment. Set it using the command \setlength{\TPHorizModule}{⟨*dimen*⟩} (or indeed \addtolength). The default is one sixteenth of the paper width.

\TPVertModule **\TPVertModule** The length unit which is used for the vertical positioning and size parameters of the {textblock} environment. Set it using the command \setlength{\TPVertModule}{⟨*dimen*⟩} (or \addtolength). The default is one sixteenth of the paper height.

\TPshowboxestrue and \TPshowboxesfalse You can control whether text blocks have the rule around them by using the `\TPshowboxestrue` and `\TPshowboxesfalse` commands. The [showboxes] option simply sets the initial value of this switch.

`\TPboxrulesize` **\TPboxrulesize** When you use the [showboxes] option, the lines drawn are of this width. If this too small to show up when you are previewing your document, or if you simply like bold frames and wish to make them a feature of your poster's design, you may adjust the size using `\setlength` or `\addtolength`. The default is 0.4pt. See also the `\textblockrulecolour` command.

`\textblocklabel` **\textblocklabel** This may be used within any {`textblock`} environment. It is ignored, unless the [noshowtext] option has been specified, when it will be used to label the textblock it is inside. Use: `\textblocklabel{Identifying text}`.

`\showtextsize` **\showtextsize** When `\textblocklabel` is being shown, the text appears in size `\showtextsize`, which is defined by default to be `\normalsize`. If this is too small, you may adjust it using `\newcommand{\showtextsize}{\large}`, or whatever size you prefer.

`\textblockorigin` **\textblockorigin** Sets the position of the top-left of the printable area. See above.

2 Notes

2.1 Suggestions: Producing large-format posters

If you are producing a large-format poster, such as A0 size, you might want to use Gerlinde Kettl and Matthias Weiser's `a0poster` class, which painlessly deals with the miscellaneous hassles of printing to a large-format postscript printer.

I have a collection of suggestions for producing such posters at `<http://purl.org/nxg/note/posters>`.

The text on a large poster will typically use a very large font. It can be a hassle to create (or have dvips create) these fonts, and they take up a good deal of space on your disk. You might want to investigate the BlueSky/AMS fonts (available at CTAN), which are postscript versions of the Computer Modern fonts, and which can therefore be scaled arbitrarily.

2.2 Absolute mode and \newpage

You can sometimes get rather puzzling behaviour when you use `\newpage` in absolute mode.

When using the [absolute] option, you most typically have all of the text on your page inside {`textblock`} environments. In this case, TEX does not believe that you have anything on the page at all, and so if you give a `\newpage` command

to start a second sheet (perhaps you have a particularly generous poster space allocation at your conference, or you are filling out a form), TEX thinks it is redundant, and ignores it, so that all your {textblock} environments end up on just one page. To work round this, use \null\newpage instead: this is enough to persuade TEX to respect the page-break.

This happens for the same reason that, also quite surprisingly, two \newline commands in a row do not produce a blank page.

2.3 Interactions

textpos & prosper

Textpos does not appear to get on terribly well with Prosper (a problem reported by Erik Van Eynde). This is because Textpos in absolute mode places its text onto the page at the last moment before the page is shipped out, which will be *after* Prosper has rotated the page from portrait to landscape format, so that the {textblock} contents end up at the correct position on the portrait page, but the wrong position on the landscape page. This unfortunately means that Textpos in absolute mode is incompatible with Prosper. But all is not lost: since Prosper slides start at a consistent point, you can fairly happily use Textpos in relative mode as long as the {textblock} environment is the first thing on the slide. The same appears to be true with the beamer package. Indeed *anything* doing things at \shipout time (which includes Textpos in absolute mode) is going to be in a precarious position with respect ot anything else which plays games here.

... & beamer

textpos & color

There's also an unfortunate interaction with the color package. Textpos in absolute mode, and the pdftex graphics driver, both squabble over who gets to redefine the \shipout command: using the \pagecolor command with pdflatex causes textpos absolute mode to behave strangely[1]. A fix may appear here in time, but for the present, there are three workarounds. The first is to place the \pagecolor{...} command before \begin{document}, which causes the redefinitions of \shipout to happen in a working order. If this is impossible for some reason, then (as with the Prosper workaround above) you can use textpos not in absolute mode: if the textblocks are the first printable material on the page, then they're anchored at a fixed position, and you should be able to use textblocks and \pagecolor with pdflatex much as normal. The final possibility is to use latex, dvips and ps2pdf to produce PDF, since the dvips driver happens not to have this problem (thanks to Joris Vankerschaver for the initial report, and to Heiko Oberdiek for one of the workarounds).

page grid

Gabriel Zachmann suggested having Textpos put a grid on the page, so that it is easier to work out {textblock} coordinates. I may yet do this, but it may not be necessary, since Rolf Niepraschk's eso-pic package can help you create this grid yourself. There is a vivid example of using Textpos along with Rolf's eso-pic package and the calc package on the Textpos web pages, at <http://purl.org/nxg/dist/textpos>.

textpos & texdraw

Finally, Robert Wenner reported a problem when using Textpos along with the texdraw package, with \move(0,0) apparently making a difference when it

[1] See the comp.text.tex thread 'Colour in a0 poster' starting 2007 April 3

should be a no-op. I haven't worked out what's going on here.

3 Credits

Olaf Maibaum, Olaf.Maibaum@informatik.uni-oldenburg.de, made an elegant improvement to an earlier version of this package, by producing the code which I've incorporated here as the 'absolute mode' (I'd had something like this before, but it was *very* kludgy).

Bjoern Pedersen, bjoern@poseidon.org.chemie.tu-muenchen.de, made the excellent suggestion that the horizontal and vertical modules should be independent, and provided code to implement this.

Rolf Niepraschk, niepraschk@ptb.de, provided the code changes which made textpos compatible with the *calc* package.

Wybo Dekker, wybo@servalys.nl, reported a problem when box 255 was (erroneously) not a vbox, and passed on a fix from Hans Hagen.

Jenny Maresh and Matthias Jerg independently suggested that it would be useful to specify a margin around the block of text in a {textblock} environment. That resulted in the \TPMargin command (after an unconscionably long gestation period). Rusen Lu suggested that one should be able to specify the colours of the box borders, and that it would be useful to turn the bordering feature on and off within the file.

Axel Sommerfeldt provided elegant code to fix incorrect behaviour of \caption within the {figure} environment.

Thanks also for bugreports and other suggestions to Jozef Bednarcik, Daniel Richard G, Wolfgang Fleischer, Giovanni Radilla and Robert Whittaker.

If you've reported a bug or made a suggestion and I haven't credited you here, please do accept my apologies, and please let me know.

4 Example

Here is a short example file.

```
⟨*example⟩
\documentclass{article}

\usepackage[absolute]{textpos}

\setlength{\TPHorizModule}{30mm}
\setlength{\TPVertModule}{\TPHorizModule}
\textblockorigin{10mm}{10mm} % start everything near the top-left corner
\setlength{\parindent}{0pt}

\begin{document}

\begin{textblock}{3}(0,0)
This block is 3 modules wide, and is placed with its top left corner
```

```
at the 'origin' on the page.  Note that the length of the block is not
specified in the arguments -- the box will be as long as necessary to
accomodate the text inside it.  You need to examine the output of the
text to adjust the positioning of the blocks on the page.
\end{textblock}

\begin{textblock}{2}(2,1)
\textblocklabel{block two}
Here is another, slightly narrower, block, at position (2,1) on the page.
\end{textblock}

\begin{textblock}{3}[0.5,0.5](2,3)
This block is at position (2,3), but because the optional argument
[0.5,0.5] has been given, it is the centre of the block which is
located at that point, rather than the top-left corner.
\end{textblock}

\end{document}
⟨/example⟩
```