The contour package

Harald Harders (h.harders@tu-bs.de)
Morten Høgholm (moho01ab@student.cbs.dk)

File version v2.14, date 2004/11/18, printed September 27, 2012

Abstract

This package generates a colored contour around a given text in order to enable printing text over a background without the need of a color box around the text. By default this is done by placing copies of the text around the text itself and can be switched to use real outlines if supported by the backend driver.

Contents

Copyright

1 Introduction

Sometimes it is necessary to print text over a background that is not pure white, e.g., in gnuplot plots. It is not always wished to plot the text in a rectangular box since this can cover important information or look poor. This package provides a solution to cover less space with the text and ensure readable text in the same time.

The used technique is quite simple. By default, in a circle around the original text position the same text is printed evenly distributed 16, 32, or a a given number times. The default radius for the circle is 0.03 em. If requested and supported by the used driver (dvips, pdfTeX, VTeX, DVIPSONE) a real outline can be chosen instead of text copies.

2 User interface

To use this package place

```
\usepackage[<options>]{contour}
```

in the preamble of your document.

If loaded without package option or with the option `copies` the contour is printed by placing copies of the text around the original text, as described below. This can be changed by the option `outline`. If specified this option the contour is printed by a real outline of the text instead of copies. This increases speed as well as quality[1] and reduces the file size. But some prerequisites have to be fulfilled:

- The backend driver has to be supported. Currently, dvips, pdfTeX, VTeX, and DVIPSONE are supported. Normally, the file `contour.cfg` tries to find out which driver has to be loaded.[2] But you can also specify the driver by giving one of the package options `dvips`, `xdvi`, `pdftex`, `vtex`, or `dvipsone`. `xdvi` refers to `dvips` as driver. In xdvi and DVIWindo, the contours are silently ignored.
- Outline (vector) fonts have to be used. With dvips, this means that Type 1 fonts have to be used. With pdfTeX, Type 1 or TrueType fonts are supported. If you are using METAFONT fonts you have to use copies (package option `copies`).

`\contour` The package provides the commands

```
\contour{<color>}{<text>}
\contour[<number>]{<color>}{<text>}
\contour*{<color>}{<text>}
```

which produce the text ⟨*text*⟩ with a ⟨*color*⟩ed contour around. The text itself is typeset in the normal active color. The normal `\contour` command produces 16 copies of the text while the `*` variant makes 32 copies. If ⟨*number*⟩ is given, the given number of copies is generated around the text. If ⟨*number*⟩ is `auto` instead of a number, the number of copies is calculated automatically depending on the font size.

`\contourlength` The radius of the circle (= thickness of the contour) can be changed using the

[1] Especially in Acrobat Reader, the quality is better.
[2] DVIPSONE cannot be determined automatically.

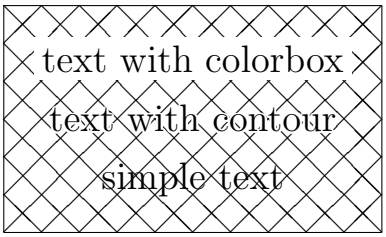


Figure 1: Comparison of the commands `\colorbox` and `\contour`

command `\contourlength{`⟨*length*⟩`}`, where ⟨*length*⟩ is a length understood by LaTeX. The length is not expanded when defining the contour length but when using it. For example when using `0.05em`, 0.05 em of the font used in the command `\contour` are used.

`\contournumber` By default, `\contour` generates 16 copies of the text. You can change this by using `\contournumber`, e.g.,

```
\contournumber{27}
```

If you specify `auto` instead of a number, `\contour` will use the automatically calculated number of copies. You may also use the package option `auto` to reach this behaviour:

```
\usepackage[auto]{contour}
```

If using outlines the number of copies is ignored.

3 Examples

The command

```
\colorbox{black}{This text is not visible
  \contour{white}{but this is.}}
```

produces:

but this is.

Another example is shown in Figure 1.

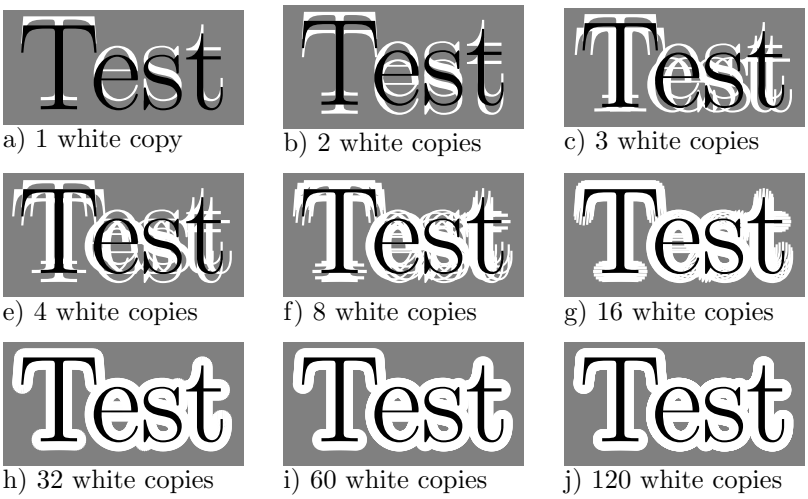
In Figure 2, you can see how the output depends on the number of copies.

4 Advanced internals

Normally, it is not planned to switch between outline and copy mode. If you really have to do it you can use the boolean `\ifcon@outline`. `\con@outlinetrue` switches on outlines, while replacing `true` by `false` switches them off. Don't forgett to surround it by `\makeatletter` and `\makeatother` to allow `@` in command names.

See the difference between copy and outline mode:

Copies Outline



a) 1 white copy b) 2 white copies c) 3 white copies
e) 4 white copies f) 8 white copies g) 16 white copies
h) 32 white copies i) 60 white copies j) 120 white copies

Figure 2: Contours with different numbers of copies with a large distance between text and copies

5 Acknowledgements

Thanks to Richard Pfeiffer who had the idea for this package and wrote some code that did it.

Thanks to Dietrich Grau who asked me for support of DVIPSONE and who did the testing of all my code ideas for this driver since I do not have Y&Y TEX.

6 The implementation

Heading of the package:

```
1 ⟨package⟩\NeedsTeXFormat{LaTeX2e}
2 ⟨dvips⟩\ProvidesFile{dvips.cnt}
3 ⟨pdftex⟩\ProvidesFile{pdftex.cnt}
4 ⟨vtex⟩\ProvidesFile{vtex.cnt}
5 ⟨dvipsone⟩\ProvidesFile{dvipsone.cnt}
6 ⟨cfgfile⟩\ProvidesFile{contour.cfg}
7 ⟨package⟩\ProvidesPackage{contour}
8 ⟨package | drv | cfgfile⟩  [2004/11/18  v2.14  Print contoured text (HH, MH)]
9 ⟨*package⟩
```

6.1 The package

This package requires the color package from the graphics bundle:

```
10 \RequirePackage{color}
```

This package requires the trig package:

```
11 \RequirePackage{trig}
```

A command that makes `\color` inactive prevent the background text color from being changed:

```
12 \def\con@coloroff{%
```

```
  \def\@undeclaredcolor[##1]##2{}%
  \def\@declaredcolor##1{}%
}
```

Set text to a specified relative position without using space:

```
\newcommand*\con@put[3]{%
  \rlap{\hskip#1\raisebox{#2}[0pt]{#3}}%
}
```

`\contourlength` Define the commands for changing the base length:

```
\newcommand*\contourlength[1]{\def\con@base@length{#1}}
\contourlength{0.03em}
```

`\contournumber` Define the commands for changing the number of text copies:

```
\newcommand*\contournumber[1]{\def\con@default@copies{#1}}
\contournumber{16}
```

Define new offset length:

```
\newlength{\con@length}
```

Define the angles for using '⟨*#1*⟩' copies unless they've previously been defined. We use 'trig' for this.

```
\newcommand*\con@define@copyangles[1]{%
  \edef\con@tempa{auto}%
  \edef\con@tempb{#1}%
```

Set the number of copies automatically.

```
  \ifx\con@tempa\con@tempb
    \expandafter\ifdim\f@size pt<15pt\relax
      \edef\con@tempb{16}%
    \else
      \expandafter\ifdim\f@size pt<25pt\relax
        \edef\con@tempb{32}%
      \else
        \expandafter\ifdim\f@size pt<50pt\relax
          \edef\con@tempb{60}%
        \else
          \edef\con@tempb{120}%
        \fi
      \fi
    \fi
  \fi
  \@tempcnta=\con@tempb\relax
  \@ifundefined{con@\number\@tempcnta @copies}{%
    \global\@namedef{con@\number\@tempcnta @copies}{}%
    \@tempdima=360\p@
    \divide\@tempdima by \@tempcnta
    \@tempdimb=\z@
    \@tempcntb=\z@
    \loop
      \ifnum\@tempcntb<\@tempcnta
        \edef\con@temp@fdim{\strip@pt\@tempdimb}%
        \CalculateSin{\con@temp@fdim}%
        \CalculateCos{\con@temp@fdim}%
        \advance\@tempcntb \@ne
```

```
      \advance\@tempdimb \@tempdima\relax
    \repeat
  }{}%
  \let\con@tempa\@undefined
  \let\con@tempb\@undefined
}
```

\contour The starred version of \contour prints 32 copies.

```
\newcommand*\contour{\@ifstar{\@contour[32]}{\@contour}}
\newcommand*\@contour[3][\con@default@copies]{%
  \ifvmode
    \leavevmode
  \fi
  \setlength\con@length{\con@base@length}%
```

Print outline or copies?

```
  \ifcon@outline
    \typeout{contour: Using real outline for `#3'\on@line.}%
    \@contour@outline{#2}{#3}%
  \else
    \begingroup
```

Calculate the copy angles.

```
      \con@define@copyangles{#1}%
      \typeout{contour: Using \the\@tempcnta\space copies for `#3'\on@line.}%
      \@tempdima=360\p@
      \divide\@tempdima by \@tempcnta\relax
      \@tempdimb=\z@
      \@tempcntb=\z@
```

Set color ans switch off color command inside argument.

```
      \color{#2}%
      \con@coloroff
```

Print the copies.

```
      \loop
        \ifnum\@tempcntb<\@tempcnta\relax
          \edef\con@temp@fdim{\strip@pt\@tempdimb}%
          \con@put
            {\UseSin{\con@temp@fdim}\con@length}%
            {\UseCos{\con@temp@fdim}\con@length}%
            {#3}%
          \advance\@tempcntb \@ne\relax
          \advance\@tempdimb\@tempdima\relax
      \repeat
    \endgroup
```

Print the main text.

```
    \mbox{#3}%
  \fi
}
```

By default, no driver is active.

```
\providecommand*\con@driver{\@empty}
```

Boolean for using outline or copies.

```
\newif\ifcon@outline
```

Options:

```
\DeclareOption{auto}{\contournumber{auto}}
\DeclareOption{dvips}{\def\con@driver{dvips.cnt}}
\DeclareOption{xdvi}{\ExecuteOptions{dvips}}
\DeclareOption{pdftex}{\def\con@driver{pdftex.cnt}}
\DeclareOption{vtex}{\def\con@driver{vtex.cnt}}
\DeclareOption{dvipsone}{\def\con@driver{dvipsone.cnt}}
\DeclareOption{outline}{\con@outlinetrue}
\DeclareOption{copies}{\con@outlinefalse}
```

Load configuration file if existing.

```
\InputIfFileExists{contour.cfg}{%
  \typeout{Loading configuration file ‘contour.cfg’.}%
}{%
  \typeout{No configuration file ‘contour.cfg’ found.}%
}
```

Process the options.

```
\ProcessOptions\relax
```

Load the driver file.

```
\expandafter\ifx\con@driver\@empty
  \ifcon@outline
    \PackageError{contour}{Chosen package option ‘outline’ but no
      driver defined}{Leave out the ‘outline’ option or define a driver}%
  \else
    \PackageWarning{contour}{No driver defined (which does not matter
      when using copies)}%
  \fi
\else
  \InputIfFileExists{\con@driver}{%
    \typeout{contour: Using driver file ‘\con@driver’.}%
  }{%
    \PackageError{contour}{Driver file ‘\con@driver’ does not exist}{}%
  }%
\fi
```

⟨/package⟩

6.2 Driver Files

6.2.1 Dvips

`\@contour@outline` Prints the text and contour using real outlines. `\@contour@outline{`⟨*color*⟩`}{`⟨*text*⟩`}`

```
⟨*dvips⟩
\newcommand*\@contour@outline[2]{%
  \begingroup
```

Double the width of the contour since the inner half is overprinted by the normal text; convert `pt` to `bp`.

```
    \setlength\con@length{2\con@length}%
    \setlength\con@length{0.99626400996\con@length}%
```

Set the contour color and disable color command.

```
    \color{#1}%
    \con@coloroff
```

PostScript preamble to print an outline for the text.

```
    \special{ps:
```

First, save all graphics settings to avoid side effects.

```
      gsave
```

Start a new path and choose a round pen.

```
        newpath
        1 setlinejoin
        1 setlinecap
```

Set the line width and scale it according to the PostScript scale.

```
        Resolution 72 div DVImag mul
        \strip@pt\con@length\space mul setlinewidth
```

Save `show` to be able to restore it later.

```
        /cntorigshow /show load def
```

Redefine the `show` command that prints a text to do the outline instead of the text.

```
        /show { false charpath } def
    }%
```

Typeset the outline text.

```
    \rlap{#2}%
```

PostScript postamble.

```
    \special{ps:
```

Finally, do the outline.

```
        stroke
```

Restore the original settings.

```
        /show /cntorigshow load def
      grestore
    }%
  \endgroup
```

Print the main text.

```
  \mbox{#2}%
}
```

⟨/dvips⟩

6.2.2 pdfTEX

`\@contour@outline` Prints the text and contour using real outlines. `\@contour@outline{`⟨*color*⟩`}{`⟨*text*⟩`}`

⟨*pdftex⟩

```
\newcommand*\@contour@outline[2]{%
  \begingroup
```

Double the width of the contour since the inner half is overprinted by the normal text; convert `pt` to `bp`.

```
    \setlength\con@length{2\con@length}%
    \setlength\con@length{0.99626400996\con@length}%
```

Set the contour color and disable color command.

```
    \color{#1}%
    \con@coloroff
```

PDF preamble.

```
    \pdfliteral{%
```

Save the graphics settings.

```
      q
```

Choose a round pen.

```
      1 j
      1 J
```

Switch text to print an outline instead of fill.

```
      1 Tr
```

Set the line width.

```
      \strip@pt\con@length\space w
    }%
```

Typeset the outline text.

```
    \rlap{#2}%
```

PDF postamble.

```
    \pdfliteral{%
```

Restore original settings.

```
      Q
    }%
  \endgroup
```

Print the main text.

```
  \mbox{#2}%
}
```

⟨/pdftex⟩

6.2.3 VTEX

`\@contour@outline` Prints the text and contour using real outlines. `\@contour@outline{`⟨*color*⟩`}{`⟨*text*⟩`}`

⟨*vtex⟩

```
\newcommand*\@contour@outline[2]{%
  \begingroup
```

Double the width of the contour since the inner half is overprinted by the normal text; convert `pt` to `bp`.

```
    \setlength\con@length{2\con@length}%
    \setlength\con@length{0.99626400996\con@length}%
```

Set the contour color and disable color command.

```
    \color{#1}%
    \con@coloroff
```

PostScript preamble to print an outline for the text.

At this point, VTEX does an ugly job since it interprets this code immediately and thus puts the path itself into the output instead of just let the printer do the work. I do not know how to change this.

```
    \special{pS:
```

First, save all graphics settings to avoid side effects.

```
      save
```

Start a new path and choose a round pen.

```
        newpath
        1 setlinejoin
        1 setlinecap
```

Set the line width.

```
        \strip@pt\con@length\space setlinewidth
```

Redefine the command that prints a text to do the outline instead of the text.

```
        /show { false charpath } def
    }%
```

Typeset the outline text.

```
    \rlap{#2}%
```

PostScript postamble.

```
    \special{pS:
```

Finally, do the outline.

```
        stroke
```

Restore the original settings.

```
      restore
    }%
  \endgroup
```

Print the main text.

```
  \mbox{#2}%
}
```

⟨/vtex⟩

6.2.4 DVIPSONE

`\@contour@outline` Prints the text and contour using real outlines. `\@contour@outline{`⟨*color*⟩`}{`⟨*text*⟩`}`

⟨*dvipsone⟩

```
\newcommand*\@contour@outline[2]{%
```

First, print the text. This is a hack and avoids problems when `\contour` is used in some contexts, e.g., at the begin of the text.

```
  \rlap{#2}%
  \begingroup
```

Double the width of the contour since the inner half is overprinted by the normal text; convert `pt` to `bp`.

```
    \setlength\con@length{2\con@length}%
    \setlength\con@length{0.99626400996\con@length}%
```

Set the contour color and disable color command.

```
    \color{#1}%
    \con@coloroff
```

PostScript preamble to print an outline for the text.

```
    \special{ps:
```

First, save all graphics settings to avoid side effects.

```
      gsave
```

Save the current position to be used for the new path.

```
        currentpoint
```

Start a new path and go back to the old position.

```
        newpath
        moveto
```

Choose a round pen.

```
        1 setlinejoin
        1 setlinecap
```

Set the line width and scale it according to the PostScript scale. The factor 65693.4 is chosen by trial and error and may be inexact.

```
        65693.4 \strip@pt\con@length\space mul setlinewidth
```

Save `show` to be able to restore it later.

```
        /cntorigshow /show load def
```

Redefine the `show` command that prints a text to do the outline instead of the text.

```
        /show { false charpath } def
    }%
```

Typeset the outline text.

```
    \rlap{#2}%
```

PostScript postamble.

```
    \special{ps:
```

Finally, do the outline and save the position.

```
        currentpoint
        stroke
```

Restore the original settings and position.

```
        /show /cntorigshow load def
      grestore
      moveto
    }%
  \endgroup
```

Print the main text.

```
  \mbox{#2}%
}
```

⟨/dvipsone⟩

6.3 Configuration Files

This configuration file is just a copy of a part of `graphics.cfg` from TEXLive.

⟨∗cfgfile⟩

```
 % Select an appropriate default driver
\begingroup
  \chardef\x=0 %
  % check pdfTeX
  \@ifundefined{pdfoutput}{}{%
    \ifcase\pdfoutput
    \else
      \chardef\x=1 %
    \fi
  }%
  % check VTeX
```

```
  \@ifundefined{OpMode}{}{%
    \chardef\x=2 %
  }%
\expandafter\endgroup
\ifcase\x
  % default case
  \ExecuteOptions{dvips}%
\or
  % pdfTeX is running in pdf mode
  \ExecuteOptions{pdftex}%
\else
  % VTeX is running
  \ExecuteOptions{vtex}%
\fi
```
⟨/cfgfile⟩

Change History

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.