

The pst-pdf package*

Rolf Niepraschk[†] Hubert Gäßlein

2017/06/22

1 Introduction

The package `pst-pdf` simplifies the use of graphics from PSTricks and other PostScript code in PDF documents. As in building a bibliography with `BIBTEX` additional external programmes are being invoked. In this case they are used to create a PDF file (`\PDFcontainer`) that will contain all this graphics material. In the final document this contents will be inserted instead of the original PostScript code.

2 Usage

2.1 Package options

active Activates the extraction mode (DVI output). An explicit declaration usually is not necessary (default in `LATEX` mode).

inactive No special actions; only the packages `pstricks` and `graphicx` are loaded (default in `VTEX`). Can be used to just convert the document with `LATEX` into a DVI file while avoiding the automatic extraction mode.

pstricks The package `pstricks` is loaded (default).

nopstricks The package `pstricks` does not get loaded. Once it is detected that `pstricks` was loaded however in some other way, the `pspicture` environment is treated as if the option “`pstricks`” was given.

draft From the `\PDFcontainer` file included graphics is displayed as frame in `pdfLATEX` mode.

final From the `\PDFcontainer` file included graphics is correctly displayed in `pdfLATEX` mode (default).

tightpage The graphics’ dimensions in the `\PDFcontainer` file match exactly those of the corresponding `TEX` boxes (default).

notightpage The dimensions of the `TEX` box corresponding to its graphics is not always correct, since a PostScript statement can draw outside its box. The option “`notightpage`” makes the graphics in the `\PDFcontainer` file to be at

*This document corresponds to `pst-pdf` v1.2d, dated 2017/06/22. Thanks to Peter Dyballa for the translation.

[†]`Rolf.Niepraschk@gmx.de`

least the size of the whole page. To be able to make use of the graphics' in a later pdfL^AT_EX run, the `\PDFcontainer` file needs to be finished in a way that each graphics gets reduced in size to its visible part. For this an external programme like `pdfcrop`¹ can be useful. Its use can save declaring the option “trim” (see also section 2.4).

displaymath In PDF mode the mathematical environments `displaymath`, `eqnarray`, and `$$` get also extracted and included as graphics. This way additional PSTricks extensions can easily be added to the contents of these environments. (Question: how do AMSL^AT_EX environments behave?)

(other) All other options are passed to `psctricks` package.

2.2 Program calls

The following table shows the course necessary to create a PDF document containing PostScript graphics². As comparison the analogous course for a bibliography is shown.

PostScript graphics	bibliography
<code>pdflatex document.tex</code>	<code>pdflatex document.tex</code>
<i>auxiliary calls</i>	
<code>latex document.tex</code>	
<code>dvips -o document-pics.ps document.dvi</code>	
<code>ps2pdf document-pics.ps</code>	<code>bibtex document.aux</code>
<code>pdflatex document.tex</code>	<code>pdflatex document.tex</code>

While creating the output only code from inside a `pspicture` or `postscript` environment is considered. PostScript graphics files, which are passed as parameter of an `\includegraphics` statement, too are included into the `\PDFcontainer` file. This file's name is by default `\jobname-pics.pdf`. It can be changed by re-defining the macro `\PDFcontainer`.

2.3 User commands

`pspicture` `\begin{pspicture}[\langle keys \rangle] (\langle x0,x1 \rangle) (\langle y0,y1 \rangle) ... \end{pspicture}`
 The `pspicture` environment is not available when the option “nopstricks” was given. It is to be used the same way as if in PSTricks. In pdfL^AT_EX mode this environment's contents is only displayed when the `\PDFcontainer` file was created before.

`postscript` `\begin{postscript}[\langle keys \rangle] ... \end{postscript}`
 The `postscript` environment can contain any code except floats. In pdfL^AT_EX mode its contents is take too off the `\PDFcontainer` file. Other as in the `pspicture` environment the necessary space is not always preserved when the `\PDFcontainer` file does not exist yet.

`\includegraphics` `\includegraphics[\langle keys \rangle]{\langle filename \rangle}`

¹CTAN: support/pdfcrop/

²The T_EX distribution “teT_EX” contains a UNIX shell script `ps4pdf` which executes all the necessary steps. See: CTAN: macros/latex/contrib/ps4pdf/

To be used as in `graphics/graphicx` defined. In pdfL^AT_EX mode it is now additionally feasible to pass the name of an EPS file. Its visible contents too is taken from the `\PDFcontainer` file.

<code>\includegraphics</code>	<code>\includegraphics[<i><keys></i>](<i><pfxadd></i>)<<i><ovpfgd></i>>[<i><ovpbgd></i>] {<i><filename></i>}</code> Wie im Paket <code>psfragx</code> definiert zu verwenden.
<code>\savepicture</code>	<code>\savepicture{<i><name></i>}</code> The last output graphics (result of the <code>pspicture</code> or <code>postscript</code> environments or the <code>\includegraphics</code> statement with an PostScript file as argument) is being saved in a file under the name as given by the parameter.
<code>\usepicture</code>	<code>\usepicture[<i><keys></i>]{<i><name></i>}</code> Die zuvor mit <code>\savepicture</code> gespeicherte Grafik wird ausgegeben. Der optionale Parameter entspricht dem bei der Anweisung <code>\includegraphics</code> möglichen.
<code>pst-pdf-defs</code>	<code>\begin{pst-pdf-defs} ... \end{pst-pdf-defs}</code> Sollen eigene Makros oder Umgebungen definiert werden, die das Zeichen <code>&</code> (andere?) im Ersetzungstext enthalten, so müssen diese Definitionen von der Umgebung <code>pst-pdf-defs</code> umschlossen werden.

2.4 Command options

The behaviour of the `\includegraphics` and `\usepicture` statements and the `postscript` environment can be modified with any of the following parameters (key value syntax):

- frame**=*<true|false>* As with the `\fbox` statement a frame is drawn around the graphics. Any change of size due to rotation is taken into account. Drawing happens in pdfL^AT_EX mode; before, in creating the `\PDFcontainer` file, it is ignored. Default: `false`.
- innerframe**=*<true|false>* As in “`frame`”, but the frame is drawn around the graphics, not its box.
- ignore**=*<true|false>* If “`true`” no graphics is output. With `\savepicture{<name>}` the graphics can be used later in a different place via `\usepicture`. Default: `false`.
- showname**=*<true|false>* A caption of minimal font size records the used file’s name. Default: `false`.
- namefont**=** Controls the font used when “`showname=true`” is set. Default: `\ttfamily\tiny`

All parameters can be set globally as in `\setkeys{Gin}{<key=value>}`.

3 Implementation

1 *<*package>*

3.1 Package options

2 `\newcommand*\ppf@TeX@mode{-1}`

```

3 \newcommand*\ppf@draft{false}
4 \newif\if@ppf@PST@used\@ppf@PST@usedtrue
5 \newif\if@ppf@tightpage \@ppf@tightpagetrue
6 \DeclareOption{active}{\OptionNotUsed}
7 \DeclareOption{inactive}{\def\ppf@TeX@mode{9}}
8 \DeclareOption{ignore}{\def\ppf@TeX@mode{999}}
9 \DeclareOption{pstricks}{\@ppf@PST@usedtrue}
10 \DeclareOption{nopstricks}{\@ppf@PST@usedfalse}
11 \DeclareOption{displaymath}{%
12   \PassOptionsToPackage\CurrentOption{preview}}
13 \DeclareOption{draft}{\def\ppf@draft{true}}
14 \DeclareOption{final}{\def\ppf@draft{false}}%
15   \PassOptionsToPackage\CurrentOption{graphicx}}

16 \DeclareOption{notightpage}{\@ppf@tightpagefalse}%
17 \DeclareOption{tightpage}{\@ppf@tightpagetrue}%
18 \DeclareOption*{%
19   \PassOptionsToPackage\CurrentOption{pstricks}}
20 \ProcessOptions\relax
21 \ifnum\ppf@TeX@mode=999\relax\expandafter\endinput\fi

```

3.2 Compiler tests

It is tested which \TeX compiler in which mode of operation is actually used (see ‘graphics.cfg’ in $\text{te}\TeX/\text{TeX}$ Live). Accordingly the environments `pspicture` and `postscript` gain each a different range of functions. This test is only executed when the options `active` or `inactive` were not given.

```

22 \RequirePackage{ifpdf,ifxetex,ifvtex}
23 \ifnum\ppf@TeX@mode=-1\relax
24   \ifpdf
25     ⇒ pdf $\TeX$  or Lua $\TeX$  are running in PDF mode
26     \def\ppf@TeX@mode{1}%
27     \RequirePackage{luatex85}%
28   \else
29     \ifvtex
30     ⇒ V $\TeX$ 
31     \def\ppf@TeX@mode{9}%
32   \else
33     \ifxetex
34     ⇒ Xe $\TeX$ 
35   \def\ppf@TeX@mode{9}%
36 \else
37   ⇒ DVI mode
38   \def\ppf@TeX@mode{0}%
39   \fi
40 \fi
41 \fi
42 \fi

39 \newcommand*\PDFcontainer{}
40 \edef\PDFcontainer{\jobname-pics.pdf}
41 \newcounter{pspicture}
42 \newcommand*\ppf@other@extensions[1]{}

```

```

43 \newcommand*\usepicture[2] [] {}
44 \newcommand*\savepicture[1] {}

```

pst-pdf-defs

```

45 \newenvironment*{pst-pdf-defs}{%
46   \endgroup
47   %   ??? \@currencline
48 }{%
49   \begingroup
50   \def\@currenvir{pst-pdf-defs}%
51 }

52 \RequirePackage{graphicx}[2017/06/01]%
53 \let\ppf@Gininclude@graphics\Gininclude@graphics
54 \let\ppf@Gin@extensions\Gin@extensions
55 \let\ppf@Gin@ii\Gin@ii

```

```

56 \newif\ifppf@pdfTeX@graphic
57 \newif\ifGin@frame\Gin@framefalse
58 \newif\ifGin@innerframe\Gin@innerframefalse
59 \newif\ifGin@showname\Gin@shownamefalse
60 \newif\ifGin@ignore\Gin@ignorefalse

```

\ifpr@outer in fact is defined in package preview. We have to do it here too since otherwise T_EX could “stumble and fall” while parsing the \ifcase structure.

```

61 \newif\ifpr@outer

```

\ppf@is@pdfTeX@graphic

Parameter #1 is the name of a graphics file with or without extension, parameter #2 contains the valid extensions in PDF mode, parameter #3 contains the valid extensions in DVI mode. If it works to process the graphics in PDF mode, then the statements in #4 are executed, otherwise those in #5.

```

62 \newcommand*\ppf@is@pdfTeX@graphic[5] {%
63   \@ppf@pdfTeX@graphicfalse%
64   \begingroup
65   \edef\pdfTeXext{#2}%

```

Instead of loading the found graphics, only a test on file name extension.

```

66   \def\Gin@setfile##1##2##3{%
67     \edef\@tempb{##2}%
68     \@for\@tempa:=\pdfTeXext\do{%
69       \ifx\@tempa\@tempb\global\@ppf@pdfTeX@graphictrue\fi}}%

```

File types for both modes need to be determined to prevent a wrong error message “File ‘#1’ not found”.

```

70   \edef\Gin@extensions{#2,#3}%

```

Trial invocation. Output is completely inhibited.

```

71   \pr@outerfalse\ppf@Gininclude@graphics{#1}%
72   \endgroup
73   \if@ppf@pdfTeX@graphic#4\else#5\fi
74 }

```

```

75 \ifcase\ppf@TeX@mode\relax

```

3.3 Extraction mode (DVI output)

The `pspicture` environment retains any definition from `pstricks.tex`. Only the code from the environments `pspicture` and `postscript` as well as `\includegraphics` with PostScript files leads to records into the DVI file. The remainder of the document's code is ignored for output. After conversion of the DVI file via PostScript (“`dvips`”) into PDF (`\PDFcontainer` file) each graphics takes exactly one page in the `\PDFcontainer` file. The \TeX compiler with DVI output and the package option “`active`” both force this mode.

```

76 \PackageInfo{pst-pdf}{%
77   MODE: \ppf@TeX@mode\space (dvi -- extraction mode)}
78 \nofiles
79 \let\makeindex\@empty \let\makeglossary\@empty
80 \AtBeginDocument{\overfullrule=\z@}%
81 \if@ppf@PST@used\RequirePackage{pstricks}\fi
82 \RequirePackage[active,dvips,tightpage]{preview}[2005/01/29]%
83 \newcommand*\ppf@PreviewBbAdjust{}
84 \newcommand*\ppf@RestoreBbAdjust{}
85 \let\PreviewBbAdjust\ppf@PreviewBbAdjust}%

```

The pdf \LaTeX mode compliant graphics file formats are needed too.

```

86 \begingroup
87 \let\AtBeginDocument\@gobble \let\PackageWarningNoLine\@gobbletwo
88 \chardef\pdftexversion=121 %
89 \newcount\pdfoutput
90 \pdfoutput=1 %
91 \input{pdftex.def}%
92 \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}
93 }%
94 \x

```

In PDF mode no rules must be defined for its compliant (PNG, JPEG, PDF) graphics file formats (because of for example ‘`dvips`’ extensions). The universal EPS rule is used to at least find these files.

```

95 \AtBeginDocument{%
96   \ifpackageloaded{keyval}{%
97     \def\KV@errx#1{\PackageInfo{keyval}{#1}}%
98     }{}%
99   \ifpackageloaded{xkeyval}{%
100     \def\XKV@err#1{\PackageInfo{xkeyval}{#1}}%
101     }{}%

```

In this mode undefined keys should not be an error.

```

102 \@for\@tempa:=\ppf@other@extensions\do{%
103   \expandafter\let\csname Gin@rule@\@tempa\endcsname\relax}%
104 \DeclareGraphicsRule*{eps}{*}{*}{}%

```

No function in this mode.

```

105 \define@key{Gin}{innerframe}[true]{}%
106 \define@key{Gin}{frame}[true]{}%
107 \define@key{Gin}{ignore}[true]{}%
108 \define@key{Gin}{showname}[true]{}%
109 \define@key{Gin}{namefont}{}%
110 \@ifundefined{Gin@page}{\define@key{Gin}{page}{}{}}{}

```

```

111 \if@ppf@tightpage\else
112 \def\PreviewBbAdjust{%
113     -600pt -600pt 600pt 600pt}%
114 \AtEndDocument{%
115     \PackageWarningNoLine{pst-pdf}{Picture container needs cropping.}}%
116 \fi

```

postscript The postscript environment utilises the trim option in the same manner as does `\includegraphics` (any specification without dimension is interpreted as if given in bp).

```

117 \newenvironment{postscript}[1] []%
118 {%
119     \global\let\ppf@PreviewBbAdjust\PreviewBbAdjust
120     \if@ppf@tightpage
121         \begingroup
122             \setkeys{Gin}{#1}%
123             \xdef\PreviewBbAdjust{%
124                 -\Gin@vllx bp -\Gin@vlly bp \Gin@vurx bp \Gin@vury bp}%
125         \endgroup
126     \fi
127     \ignorespaces
128 }%
129 {\aftergroup\ppf@RestoreBbAdjust}%

130 \PreviewEnvironment{postscript}%
131 \AtBeginDocument{%
132     \@ifundefined{PSTricksLoaded}{}%
133     {%

```

pspicture Announce preview original definition.

```

134     \PreviewEnvironment{pspicture}%

```

psmatrix Announce preview original definition.

```

135     \@ifundefined{psmatrix}{}%
136     {%
137         \PreviewEnvironment{psmatrix}%
138         \newcommand*\ppf@set@mode{}%
139         \newcommand*\ppf@test@mmode{%
140             \ifmmode
141                 \ifinner
142                     \let\ppf@set@mode=$%
143                 \else
144                     \def\ppf@set@mode{$$}%
145                 \fi
146             \else
147                 \let\ppf@set@mode=\@empty
148             \fi
149         }%
150         \let\ppf@psmatrix=\psmatrix
151         \expandafter\let\expandafter\ppf@pr@psmatrix%
152         \expandafter=\csname pr@\string\psmatrix\endcsname
153         \let\ppf@endpsmatrix=\endpsmatrix
154         \def\psmatrix{\ppf@test@mmode\ppf@psmatrix}
155         \expandafter\def\csname pr@\string\psmatrix\endcsname{%

```

```

156         \ppf@set@mode\ppf@pr@psmatrix}%
157         \def\endpsmatrix{\ppf@endpsmatrix\ppf@set@mode}%
158     }%

```

Announce internal macro `\pst@object` to enable the use of some PSTricks code outside of `pspicture` environments. At the moment invocations of the following kind are feasible:

```

\pst@object {<m>}<*>[<o>]{<o>}{<o>}<(o)><(o)><(o)>
(m = necessary, * = optional, o = optional)

```

More than three optional arguments at the call's end, as in `\psline` possible, do not work yet.

```

159     \PreviewMacro[{}*[]%
160     ?\bgroup{#{#1}{#1}}{}%
161     ?\bgroup{#{#1}{#1}}{}%
162     ?(#{#1}){({#1})}{}%
163     ?(#{#1}){({#1})}{}%
164     ?(#{#1}){({#1})}{}%
165     ]{\pst@object}

```

Prevent multiple test-wise setting of table contents by “`tabularx`”.

```

166     \@ifundefined{tabularx}{}{%
167         \newcolumntype{X}{c}%
168         \expandafter\let\expandafter\tabularx\csname tabular*\endcsname
169         \expandafter\let\expandafter\endtabularx\csname endtabular*\endcsname
170     }%

```

Support of `\includegraphicx` from the package `psfragx`.

```

171     \@ifundefined{pfx@includegraphicx}{}{%
172         \PreviewMacro[{}{}]{\pfx@includegraphicx}%
173     }%

```

`\Gscale@@box` Disable scaling.

```

174     \def\Gscale@@box#1#2#3{%
175         \toks@{\mbox}%
176     }

```

`\Ginclude@graphics` All graphics content of well known format (for instance EPS files) is treated in a regular way, which in this mode denotes that it is subject to preview functions. Other graphics content (for instance PDF files) is ignored.

```

177     \def\Ginclude@graphics#1{%
178         \ifpr@outer

```

Generally pdf_{TEX} supported graphics formats are intended to be preferred (inclusion in final pdf_{TEX} run). If it's a PostScript type graphics, then the original definition is in function again and registration for the `preview` package is necessary in order to convert this PostScript type graphics into PDF.

```

179         \ppf@is@pdfTeX@graphic{#1}{\ppf@other@extensions}{\Gin@extensions}%

```

Dummy box to prevent a division by zero while scaling or rotating. Otherwise ignored.

```

180         {\rule{10pt}{10pt}}%
181         {\ppf@Ginclude@graphics{#1}}%
182     \else

```


Inside a PostScript environment (pspicture etc.) `\includegraphics` has to behave as in its original definition (only DVIPS supported graphics formats are allowed).

```

183     \ppf@Ginclude@graphics{#1}%
184     \fi
185   }%

186   \PreviewMacro[{}]{\ppf@Ginclude@graphics}%
187   \let\pdfliteral@gobble%
188 \or

```

3.4 pdf \LaTeX mode (PDF output)

When the `\PDFcontainer` file (default: `\jobname`-pics.pdf) exists, the contents of the environments `pspicture` and `postscript` is ignored. Instead the corresponding graphics from the `\PDFcontainer` file is used.

```

189   \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (pdfTeX mode)}%
    Prevent pdf $\TeX$ 's message Non-PDF special ignored!.

190   \ifppf@PST@used
191     \let\ppf@temp\AtBeginDvi\let\AtBeginDvi@gobble
192     \def\c@lor@to@ps#1 #2@@@{}
193     \RequirePackage{pstricks}\let\AtBeginDvi\ppf@temp
194   \fi

195   \@temptokena{%
196     \let\Gin@PS@file@header@gobble\let\Gin@PS@literal@header@gobble
197     \let\Gin@PS@raw@gobble\let\Gin@PS@restored@gobble
198     \@ifundefined{PSTricksLoaded}{}{}%

```

Necessary if `PSTricks < 2.0`.

```

199     \PSTricksOff
200     \@ifundefined{c@lor@to@ps}{\def\c@lor@to@ps#1 #2@@@{}{}}{}%

```

PostScript output is now inhibited and later once again.

```

201   \the\@temptokena
202   \expandafter\AtBeginDocument\expandafter
203   {\the\@temptokena\@temptokena{}}%
204   \@ifundefined{PSTricksLoaded}{}{}%

```

To parse the arguments of `PSTricks'` `\pst@object` we load `preview` in active mode, but restore the default definitions of `\output` and `\shipout`. `\pr@startbox` and `\pr@endbox` serve here only to disable `\pst@object` and to load the corresponding graphics from the `\PDFcontainer` file. At present a maximum of three optional parameters in round braces (parenthesis) at the end of `\pst@object` is supported, which is sufficient, but not always enough.

```

205   \newtoks\ppf@output
206   \ppf@output\expandafter{\the\output}%
207   \let\ppf@nofiles=\nofiles \let\nofiles=\relax
208   \let\ppf@shipout=\shipout
209   \RequirePackage[active]{preview}[2005/01/29]%
210   \let\shipout=\ppf@shipout \let\ppf@shipout=\relax
211   \let\nofiles=\ppf@nofiles \let\ppf@nofiles=\relax
212   \output\expandafter{\the\ppf@output} \ppf@output{}%

```

`\pr@startbox`, `\pr@endbox`: simpler over original definitions.

```

213 \long\def\pr@startbox#1#2{%
214   \ifpr@outer
215     \toks@{#2}%
216     \edef\pr@cleanup{\the\toks@}%
217     \setbox\@tempboxa\vbox\bgroup
218     \everydisplay{}%
219     \pr@outerfalse%
220     \expandafter\@firstofone
221   \else
222     \expandafter\@gobble
223   \fi{#1}}%
224 \def\pr@endbox{%
225   \egroup
226   \setbox\@tempboxa\box\voidb@x
227   \ppf@getpicture
228   \pr@cleanup}%

```

(See also the identical definition in DVI mode.)

```

229 \AtBeginDocument{%
230   \ifundefined{pst@object}{}%
231   {%
232     \PreviewMacro[{}*[]%
233     ?\bgroup{#{#1}#{#1}}{}%
234     ?\bgroup{#{#1}#{#1}}{}%
235     ?({#{#1}){({#1})}}{}%
236     ?({#{#1}){({#1})}}{}%
237     ?({#{#1}){({#1})}}{}%
238     }]{\pst@object}}%
239   }%
240 }%

```

Too the supported file name extensions from DVI mode are needed.

```

241 \begingroup
242   \input{dvips.def}%
243   \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}}%
244   \x

```

Dummy definition for in DVI mode supported file formats.

```

245 \DeclareGraphicsRule*{eps}{*}{*}{%
246   \define@key{Gin}{innerframe}[true]{%
247     \lowercase{\Gin@boolkey{#1}{innerframe}}%
248   \define@key{Gin}{frame}[true]{%
249     \lowercase{\Gin@boolkey{#1}{frame}}%
250   \define@key{Gin}{ignore}[true]{%
251     \lowercase{\Gin@boolkey{#1}{ignore}}%
252   \define@key{Gin}{frame@}@{}%

```

(For internal use only!)

```

253   \edef\@tempa{\toks@\noexpand\frame{\the\toks@}}%
254   \ifcase#1\relax
255     \ifGin@innerframe\else\let\@tempa\relax\fi
256   \or
257     \ifGin@frame\else\let\@tempa\relax\fi
258   \fi

```

```

259   \@tempa
260   }%
261   \define@key{Gin}{showname}[true]{%
262     \lowercase{\Gin@boolkey{#1}}{showname}}%
263   \define@key{Gin}{namefont}{%
264     \begingroup
265     \@temptokena\expandafter{\ppf@namefont#1}%
266     \edef\x{\endgroup\def\noexpand\ppf@namefont{\the\@temptokena}}%
267     \x
268   }%
269   \newcommand*\ppf@filename{%
270     \newcommand*\ppf@namefont{\tiny\ttfamily}%
271     \newcommand*\ppf@Gin@keys{}%
272     \let\ppf@Gin@setfile\Gin@setfile

```

`\Gin@setfile` Save real file name and, if applicable, page number for later use.

```

273   \def\Gin@setfile#1#2#3{\ppf@Gin@setfile{#1}{#2}{#3}}%
274   \xdef\ppf@filename{%
275     #3\ifx\Gin@page\empty\else(\Gin@page)\fi}}%

```

`\Gin@ii` Examine the options “frame”, “ignore”, etc. as soon as other special cases.

```

276   \def\Gin@ii[#1]#2{%
277     \begingroup

```

The value of `\ifGin@innerframe` has to be known before the inner frame is drawn. The values for `\ifGin@showname` and `\ppf@namefont` need to be available after rendering the graphics too. Thus beforehand and protected inside a group examine the options.

```

278     \@temptokena{#1}\def\ppf@tempb{#2}%

```

Finds empty file name when calling `\usepicture`.

```

279     \ifx\ppf@tempb\empty\else
280       \ppf@is@pdfTeX@graphic{#2}{\Gin@extensions}{\ppf@other@extensions}%

```

Graphics out of `\PDFcontainer` are complete – scaled, rotated, etc. Don’t apply these things again and therefore ignore the optional parameters.

```

281       {%
282         \setkeys{Gin}{#1}%
283         \ifx\ppf@tempb\PDFcontainer
284           \@temptokena{page=\Gin@page}%
285         \fi
286       }%
287       {%
288         \refstepcounter{pspicture}%
289         \@temptokena{page=\the\c@pspicture}\def\ppf@tempb{\PDFcontainer}%
290       }%
291       \fi
292       \ifGin@ignore\else

```

“frame@@=0” = inner frame, “frame@@=1” = outer frame.

```

293         \edef\@tempa{\noexpand\ppf@Gin@ii[frame@@=0,\the\@temptokena,
294           frame@@=1]{\ppf@tempb}}%
295         \@tempa
296         \ifGin@showname
297           \ppf@namefont

```

```

298         \raisebox{-\ht\strutbox}[0pt][0pt]{\llap{\ppf@filename}}%
299         \gdef\ppf@filename{}%
300     \fi
301 \fi
302 \endgroup
303 }%

304 \IfFileExists{\PDFcontainer}%
305 {%

```

`\ppf@container@max` The number of pages as contained in `\PDFcontainer` file.

```

306     \pdfximage{\PDFcontainer}%
307     \edef\ppf@container@max{\the\pdflastximagepages}%

308     \AtEndDocument{%
309         \ifnum\c@pspicture>\z@

A warning only makes sense when a graphics is needed at all.

310         \ifnum\c@pspicture=\ppf@container@max\else
311             \PackageWarningNoLine{pst-pdf}{%
312                 ‘\PDFcontainer’ contains \ppf@container@max\space pages
313                 \MessageBreak but \the\c@pspicture\space pages are requested:
314                 \MessageBreak File ‘\PDFcontainer’ is no more valid!
315                 \MessageBreak Recreate it
316             }%
317         \fi
318     \fi
319 }%
320 }%
321 {%
322     \def\ppf@container@max{0}%
323     \AtEndDocument{%
324         \ifnum\c@pspicture>\z@
325             \filename@parse{\PDFcontainer}%
326             \PackageWarningNoLine{pst-pdf}{%
327                 File ‘\PDFcontainer’ not found.\MessageBreak
328                 Use the following commands to create it:\MessageBreak
329                 -----
330                 \MessageBreak
331                 latex \jobname.tex\MessageBreak
332                 dvips -o \filename@base.ps \jobname.dvi\MessageBreak
333                 ps2pdf \filename@base.ps\MessageBreak
334                 -----
335             }%
336         \fi
337     }%
338 }%

```

`\ppf@isnum` If parameter #1 is numeric, the instructions in #2, otherwise those in #3 are executed (see `bibtopic.sty`).

```

339     \newcommand\ppf@isnum[1]{%
340         \if!\ifnum9<1#1!\else_\fi\expandafter\@firstoftwo
341         \else\expandafter\@secondoftwo\fi}%

```

`psmatrix` Both environments ignore their contents and load instead the corresponding graphics out of the `\PDFcontainer` file. The value of the herein used `pspicture` counter's value can be used in `\label/\ref`.

`postscript`

```

342 \newcommand*\ppf@set@mode{%
343 \newcommand*\ppf@test@mmode{%
344 \ifmode
345 \ifinner
346 \let\ppf@set@mode=$%
347 \else
348 \def\ppf@set@mode{$$}%
349 \fi
350 \else
351 \let\ppf@set@mode=\@empty
352 \fi
353 }

354 \RequirePackage{environ}%
355 \newenvironment{postscript}[1][{}]{%
356 \def\@tempa{postscript}%
357 \ifx\@tempa\@currenvr
358 \def\ppf@Gin@keys{#1}%
359 \else
360 \def\ppf@Gin@keys{}%
361 \fi
362 \ppf@@getpicture
363 \Collect@Body@gobble}{}%
364 \AtBeginDocument{%
365 \@ifundefined{PSTricksLoaded}{-}{%
366 \def\pst@@@picture[#1](#2,#3)(#4,#5){\postscript}%
367 \def\endpspicture{\endpostscript\endgroup}%
368 \@ifundefined{psmatrix}{-}{%
369 \let\psmatrix=\postscript
370 \let\endpsmatrix=\endpostscript}%
371 }%
372 \@ifundefined{pfx@includegraphics}{-}{%

```

The useless redefinition of `\includegraphics` in pdf \TeX mode (package `psfrag`) is leading to double insertion of the result. We go back to the original meaning.

```

373 \let\includegraphics=pfx@includegraphics
374 \def\pfx@includegraphicx#1#2{\ppf@@getpicture}%
375 }%
376 }%

```

`\savepicture` Saves the recent graphics' number in a macro named `\ppf@@@#1`.

```

377 \def\savepicture#1{%
378 \expandafter\xdef\csname pfx@@@#1\endcsname{\the\pdfastximage}}%

```

`\usepicture` Inserts graphics with symbolic name #2. This name has to be declared beforehand in `\savepicture{<name>}`. Instead of a name a number can be used too, which directly addresses a graphics in the `\PDFcontainer` file. The optional parameter #1 corresponds to the one in `\includegraphics`.

```

379 \renewcommand*\usepicture[2][{}]{%

```

```

380 \@ifundefined{ppf@@#2}%
381 {%
382   \ppf@isnum{#2}%
383   {\ppf@getpicture{#1}{#2}}%
384   {\@latex@error{picture ‘#2’ undefined}\@ehc}%
385   }%
386   {%
387     \begingroup
388     \def\Gin@include@graphics##1{%
389       \xdef\ppf@filename{#2}%
390       \setbox\z@\hbox{\pdfrefximage\@nameuse{ppf@@#2}}%
391       \Gin@nat@height\ht\z@ \Gin@nat@width\wd\z@
392       \def\Gin@llx{0} \let\Gin@lly\Gin@llx
393       \Gin@defaultbp\Gin@urx{\Gin@nat@width}%
394       \Gin@defaultbp\Gin@ury{\Gin@nat@height}%
395       \Gin@bboxtrue\Gin@viewport@code
396       \Gin@nat@height\Gin@ury bp%
397       \advance\Gin@nat@height-\Gin@lly bp%
398       \Gin@nat@width\Gin@urx bp%
399       \advance\Gin@nat@width-\Gin@llx bp%
400       \Gin@req@sizes
401       \ht\z@\Gin@req@height \wd\z@\Gin@req@width
402       \leavevmode\box\z@}%
403       \define@key{Gin}{type}{}%
404       \includegraphics[scale=1,#1]{}%
405     \endgroup
406   }}%

```

`\ppf@getpicture` Inserts the page (graphics) with number #2 from the `\PDFcontainer` file. Parameter #1: any option as in `\includegraphics`.

```

407 \newcommand*\ppf@getpicture[2]{%
408   \@tempcnta=#2\relax%
409   \ifnum\@tempcnta>\ppf@container@max
410     \PackageWarningNoLine{pst-pdf}{%
411       pspicture No. \the\@tempcnta\space undefined}%
412   \else
413     \includegraphics[draft=\ppf@draft,#1,page=\the\@tempcnta]%
414     {\PDFcontainer}%
415   \fi
416   \gdef\ppf@Gin@keys{}}%

```

`\ppf@@getpicture` Inserts next page (graphics) from the `\PDFcontainer` file.

```

417 \newcommand*\ppf@@getpicture{%
418   \ifpr@outer
419     \refstepcounter{pspicture}%
420     \expandafter\ppf@getpicture\expandafter{\ppf@Gin@keys}%
421     {\the\c@pspicture}%
422   \fi}%

```

`pst-pdf-defs` Environment without grouping. The character & has the catcode “other”. Useful for user-defined macro definitions with e.g. `psmatrix` inside.

```

423 \renewenvironment*{pst-pdf-defs}%
424 {%
425   \endgroup

```

```

426 %   ??? \@currentvline
427   \chardef\ppf@temp=\catcode'\&%
428   \@makeother\&%
429   }{\%
430   \catcode'\&=\ppf@temp
431   \begingroup
432   \def\@currentvir{pst-pdf-defs}%
433   }

434 \else

```

3.5 Inactive Mode

Only the packages `pstricks` and `graphicx` are loaded – no further exertion of influence. The package option “inactive” as soon as the \TeX compiler force this mode.

```

435 \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (inactive mode)}%
436 \newenvironment{postscript}[1][\ignorespaces]{}
437 \let\ppf@is@pdfTeX@graphic\relax
438 \fi

439 \InputIfFileExists{pst-pdf.cfg}{%
440 \PackageInfo{pst-pdf}{Local config file pst-pdf.cfg used}}{}
441 \</package>

```

Change History

v1.0a	box or something else. (RN) . 13
General: Initial version. 1	General: Config file loading added. (RN) 15
v1.0b	
General: Some code and documentation cleaning. (RN) . 1	v1.0g
v1.0c	<code>\usepicture</code> : Now <code>\usepicture</code> does accept a numerical parameter. (RN) 13
General: New options “pstricks”, “nopstricks”, “draft” and “final”. (RN) 3	General: Definition of <code>\PDFcontainer</code> now with <code>\edef</code> . (RN) 4
v1.0d	v1.0h
General: Redefinition of <code>\includegraphics</code> in modes 0 and 1. Now using of eps graphics directly in pdf \LaTeX is possible. (RN) 1	<code>psmatrix</code> : Based no more on the comment environment from the verbatim package. (RN) 13
v1.0e	v1.0i
<code>postscript</code> : “trim” option added. (RN) 7	<code>\ppf@is@pdfTeX@graphic</code> : No more errors for given files without extensions. (RN) 5
v1.0f	v1.0j
<code>\savepicture</code> : New macro <code>\savepicture</code> . (RN) 13	General: Check <code>AtBeginDocument</code> for package ‘pstricks’ even if “nopstricks” is given. (RN) . . . 1
<code>\usepicture</code> : New macro <code>\usepicture</code> . Useful for putting a PSTricks graphic in a	v1.0k
	<code>\Gin@setfile</code> : Show also the

	pagenumber if exists. (RN) . . .	11		Special support for “tabularx”. (RN)	8
	<code>\Gininclude@graphics</code> : Prevent division by zero. (RN)	8		Supress handling of pdfL ^A T _E X graphic formats in DVI mode. (RN)	6
v1.0l	General: Options “framesep”, “framerule”, “linewidth” removed, “fname” and “innerframe” added. (RN)	1	v1.1d	<code>postscript</code> : Support for PSTricks environment “psmatrix”. (RN)	13
v1.0m	General: New package option “notightpage” added. (RN) . . .	1	v1.1e	General: New option “displaymath” (see preview package). (HjG/RN)	3
v1.0n	General: Changed macro names (<code>\savepicture</code> and <code>\usepicture</code>). (RN)	1	v1.1f	General: Package option “ignore” reimplemented. Now the compilation of the dtx file in L ^A T _E X mode is possible. (RN) .	3
	Some code cleaning. (RN)	1	v1.1g	<code>postscript</code> : “psmatrix” environment (preserve math mode). (RN/HjG)	13
v1.0o	General: New code for “notightpage”. (RN)	7		<code>pspicture</code> : pspicture environment must still parse its arguments. (RN/HjG)	13
	Option “fname” renamed to “showname”. (RN)	1	v1.1h	<code>\Gininclude@graphics</code> : Check if inside of a PS-related environment (correct graphic inclusion). (RN)	8
v1.0p	General: Some code and documentation cleaning. (RN) .	1	v1.1i	<code>\Gininclude@graphics</code> : Correction of the inside check. (RN/HjG) .	8
v1.0q	<code>\usepicture</code> : Now <code>\usepspicture</code> works for all kind of graphics. (RN)	13		General: <code>\ifpr@outer</code> must be predefined. (HjG/RN)	5
v1.0r	<code>\ppf@is@pdfTeX@graphic</code> : Changed <code>\ppf@is@known@graphic</code> to <code>\ppf@is@pdfTeX@graphic</code> . Now pdfT _E X graphics are prefered. (RN)	5		Package option “final” also for “graphicx”. (RN)	4
v1.0s	<code>\Gin@ii</code> : Rewritten. (RN)	11	v1.1k	General: New environment pst-pdf-defs: Support for PSTricks environment “psmatrix” inside user definitions. (RN,HjG)	1
	General: Scaling e.g. of PostScript pictures now only in extraction mode. Some code cleaning. (RN)	1	v1.1l	General: Support for the package “psfragx”. (RN)	8
v1.1a	General: Support for the internal PSTricks macro <code>\pst@object</code> . (HjG/RN)	8	v1.1m	General: Merge english and german version of the documentation. (RN)	1
v1.1b	General: Ignore the call of <code>\nofiles</code> inside of <code>preview</code> . (RN)	9	v1.1n	General: <code>\nofiles</code> added (suggestion of Torsten Bronger).	6
	Some code and documentation cleaning. (RN)	1			
v1.1c	General: New package option “tightpage” added. (RN)	1			

v1.1o		<code>postscript</code> : Using <code>environ</code> the environment <code>postscript</code> is now simple and more robust. (RN)	13
	<code>\Gscale@@box</code> : Disable scaling. (RN)		8
v1.1p	General: <code>\nofiles</code> makes <code>\makeindex</code> and <code>\makeglossary</code> to <code>\relax</code> . <code>\@empty</code> is better because of later <code>\renewcommand</code> 's.	v1.1u	General: <code>\pdfoutput</code> must be set when loading "pdftex.def" in DVI mode. (RN)
			6
v1.1pl	General: <code>\let\output@gobble</code> before loading of "preview" added. (RN)	v1.1v	<code>\Gin@ii</code> : Key settings only for pdf graphics. (RN)
			11
v1.1q	General: Problem with "tabularx" and "threeparttable" solved. (RN)	General: Local redefinition of <code>\pdfoutput</code> to be a counter. (RN)	6
			8
v1.1r	General: Fixed values for <code>\PreviewBbAdjust</code> because <code>\paperwidth</code> is not always defined (suggested by Will Robertson).	v1.2a	General: Engine tests changed (RN)
			4
v1.1s	General: Dummy definition of the page key in DVI mode.	v1.2b	General: Loading packages 'if*' at wrong place (RN)
			4
v1.1t	General: Remove the line " <code>\let\output@gobble</code> " because of bad side effects. (RN)	v1.2c	General: "postscript" environment no more allowed before <code>\begin{document}</code> (changed example file).
			1
		v1.2d	General: <code>\color@to@ps</code> must not be undefined before loading 'PSTricks'
			9
			Version parameter for 'graphicx' and rename <code>\GPT@page</code> to <code>\Gin@page</code> at several places
			5

P	
<code>\PassOptionsToPackage</code>	12, 15, 19
<code>\PDFcontainer</code>	39, 40, 283, 289, 304, 306, 312, 314, 325, 327, 414
<code>\pdflastximage</code>	378
<code>\pdflastximagepages</code>	307
<code>\pdfliteral</code>	187
<code>\pdfoutput</code>	89, 90
<code>\pdfrefximage</code>	390
<code>\pdfTeXtext</code>	65, 68
<code>\pdftexversion</code>	88
<code>\pdfximage</code>	306
<code>\pfx@includgraphics</code>	373
<code>\pfx@includgraphicx</code>	172, 374
<code>\postscript</code>	366, 369
<code>postscript</code> (environment)	2, 117, 342
<code>\ppf@getpicture</code>	227, 362, 374, 417
<code>\ppf@container@max</code>	306, 310, 312, 322, 409
<code>\ppf@draft</code>	3, 13, 14, 413
<code>\ppf@endpsmatrix</code>	153, 157
<code>\ppf@filename</code>	269, 274, 298, 299, 389
<code>\ppf@getpicture</code>	383, 407, 420
<code>\ppf@Gin@extensions</code>	54
<code>\ppf@Gin@ii</code>	55, 293
<code>\ppf@Gin@keys</code>	271, 358, 360, 416, 420
<code>\ppf@Gin@setfile</code>	272, 273
<code>\ppf@Gin@graphics</code>	53, 71, 181, 183, 186
<code>\ppf@is@pdfTeX@graphic</code>	62, 179, 280, 437
<code>\ppf@isnum</code>	339, 382
<code>\ppf@namefont</code>	265, 266, 270, 297
<code>\ppf@nofiles</code>	207, 211
<code>\ppf@other@extensions</code>	42, 92, 102, 179, 243, 280
<code>\ppf@output</code>	205, 206, 212
<code>\ppf@pr@psmatrix</code>	151, 156
<code>\ppf@PreviewBbAdjust</code>	83, 85, 119
<code>\ppf@psmatrix</code>	150, 154
<code>\ppf@RestoreBbAdjust</code>	84, 129
<code>\ppf@set@mode</code>	138, 142, 144, 147, 156, 157, 342, 346, 348, 351
<code>\ppf@shipout</code>	208, 210
<code>\ppf@temp</code>	191, 193, 427, 430
<code>\ppf@tempb</code>	278, 279, 283, 289, 294
<code>\ppf@test@mmode</code>	139, 154, 343
<code>\ppf@TeX@mode</code>	2, 7, 8, 21, 23, 25, 29, 32, 34, 75, 77, 189, 435
<code>\pr@cleanup</code>	216, 228
<code>\pr@endbox</code>	224
<code>\pr@outerfalse</code>	71, 219
<code>\pr@startbox</code>	213
<code>\PreviewBbAdjust</code>	85, 112, 119, 123
<code>\PreviewEnvironment</code>	130, 134, 137
<code>\PreviewMacro</code>	159, 172, 186, 232
<code>\psmatrix</code>	150, 152, 154, 155, 369
<code>psmatrix</code> (environment)	135, 342
<code>pspicture</code> (environment)	2, 134, 342
<code>pst-pdf-defs</code> (environment)	3, 45, 423
<code>\pst@opicture</code>	366
<code>\pst@object</code>	165, 238
<code>\PSTricksOff</code>	199
R	
<code>\raisebox</code>	298
<code>\refstepcounter</code>	288, 419
<code>\rule</code>	180
S	
<code>\savepicture</code>	3, 44, 377
<code>\setkeys</code>	122, 282
<code>\shipout</code>	208, 210
<code>\string</code>	152, 155
<code>\strutbox</code>	298
T	
<code>\tabularx</code>	168
U	
<code>\usepicture</code>	3, 43, 379
V	
<code>\voidbox</code>	226
X	
<code>\XKV@err</code>	100