

Das Paket `pst-pdf`*

Rolf Niepraschk[†] Hubert Gäßlein

2008/05/02

1 Einleitung

Das Paket `pst-pdf` vereinfacht die Verwendung von PSTricks-Grafiken und anderem PostScript-Code in PDF-Dokumenten. Ähnlich wie beim Erstellen des Literaturverzeichnisses mit `bibTeX` werden zusätzlich externe Programme aufgerufen. Sie dienen in diesem Fall dazu, eine PDF-Datei, die sämtliche Grafiken enthält, zu erzeugen. Ihr Inhalt wird im endgültigen Dokument statt des ursprünglichen PostScript-Codes eingefügt.

2 Anwendung

2.1 Paketoptionen

active Aktiviert den Extraktionsmodus (DVI-Ausgabe). Die explizite Angabe ist normalerweise unnötig (Standard im `LATEX`-Modus).

inactive Keine besonderen Aktionen; es werden nur die Pakete `pstricks` und `graphicx` geladen (Standard bei Verwendung von `VTEX`). Kann dazu benutzt werden, um das Dokument mit `LATEX` in eine DVI-Datei zu wandeln und dabei die automatische Verwendung des Extraktionsmodus¹ zu vermeiden.

pstricks Das Paket `pstricks` wird geladen (Standard).

nopstricks Das Paket `pstricks` wird nicht geladen. Wird später festgestellt, dass `pstricks` doch noch anderweitig geladen wurde, wird die Umgebung `pspicture` nachträglich in der Weise behandelt, als wäre die Option “`pstricks`” doch angegeben worden.

draft Im `pdfLATEX`-Modus werden aus der Containerdatei eingefügte Grafiken nur als Rahmen dargestellt.

final Im `pdfLATEX`-Modus werden aus der Containerdatei eingefügte Grafiken vollständig dargestellt (Standard).

tightpage Die Abmessung Grafiken in der Containerdatei entsprechen denen der zugehörigen `TEX`-Boxen (Standard).

*Dieses Dokument bezieht sich auf `pst-pdf` v1.1r vom 2008/05/02.

[†]`Rolf.Niepraschk@ptb.de`

notightpage Die Abmessung der zur Grafik gehörenden T_EX-Box ist manchmal nicht korrekt, da PostScript-Anweisungen auch außerhalb der Box zeichnen können. Die Option “notightpage” führt dazu, dass die Grafiken in der Containerdatei mindestens die Größe des gesamten Blattes einnehmen. Um die Grafiken im späteren pdfL^AT_EX-Lauf verwenden zu können, muss die Containerdatei nachbearbeitet werden, so dass die Größe der Grafiken auf die der sichtbaren Bestandteile reduziert ist. Dazu kann z. B. das Programm `pdfcrop`¹ dienen. Die Anwendung dieses Verfahrens kann die Angabe der Option “trim” erübrigen (siehe Abschnitt 2.4).

displaymath Es werden zusätzlich die mathematischen Umgebungen `displaymath`, `eqnarray` und `$$` extrahiert und im pdf-Modus als Grafik eingefügt. So können zusätzliche PSTricks-Ergänzungen leicht dem Inhalt dieser Umgebungen zugefügt werden. (Frage: Wie verhalten sich die AMSL^AT_EX-Umgebungen?)

⟨*other*⟩ Alle anderen Optionen werden an das Paket `pstricks` weitergereicht.

2.2 Programmaufrufe

Die folgende Tabelle zeigt den Ablauf, der nötig ist, um ein PDF-Dokument mit PostScript-Grafiken zu erzeugen². Im Vergleich dazu ist der analoge Ablauf für Literaturverzeichnisse angegeben.

PostScript-Grafiken	Literaturverzeichnis
<code>pdflatex document.tex</code>	<code>pdflatex document.tex</code>
<i>Hilfsaufrufe</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>

Bei der Erzeugung wird nur Code berücksichtigt, der sich innerhalb der Umgebungen `pspicture` oder `postscript` befindet. Ebenfalls werden PostScript-Grafiken, die als Parameter von `\includegraphics` angegeben wurden, der Containerdatei hinzugefügt. Der Name dieser Datei ist standardmäßig `\jobname-pics.pdf`. Er kann durch Undefinieren des Makros `\PDFcontainer` geändert werden.

2.3 Nutzeranweisungen

pspicture `\begin{pspicture}[\langle keys \rangle](\langle x0,x1 \rangle)(\langle y0,y1 \rangle)...\end{pspicture}`
 Die `pspicture`-Umgebung steht zur Verfügung, wenn nicht die Option “nopstricks” angegeben wurde. Sie wird so wie in PSTricks üblich verwendet. Im pdfL^AT_EX-Modus wird ihr Inhalt nur dann dargestellt, wenn vorher die Containerdatei erzeugt wurde.

postscript `\begin{postscript}[\langle keys \rangle]...\end{postscript}`

¹CTAN: `support/pdfcrop/`

²Die Shell-Skripte `ps4pdf` bzw. `ps4pdf.bat` führen alle angegebenen Programmaufrufe automatisch aus.

Die `postscript`-Umgebung kann beliebigen Code mit Ausnahme von Gleitumgebungen aufnehmen. Im pdfL^AT_EX-Modus wird ihr Inhalt ebenfalls der Containerdatei entnommen. Ist diese Datei nicht vorhanden, wird – anders als bei der `pspicture`-Umgebung – der später benötigte Platz möglicherweise nicht korrekt frei gehalten.

<code>\includegraphics</code>	<code>\includegraphics[⟨keys⟩]{⟨filename⟩}</code> Wie in <code>graphics/graphicx</code> definiert zu verwenden. Zusätzlich ist es nun möglich, auch im pdfL ^A T _E X-Modus EPS-Dateien als Argument anzugeben und ihren Inhalt darzustellen. Er wird dazu ebenfalls der Containerdatei entnommen.
<code>\includegraphicsx</code>	<code>\includegraphicsx[⟨keys⟩](⟨pfxadd⟩)<⟨ovpfgd⟩>[⟨ovpbgd⟩]{⟨filename⟩}</code> Wie im Paket <code>psfragx</code> definiert zu verwenden.
<code>\savepicture</code>	<code>\savepicture{⟨name⟩}</code> Die zuletzt ausgegebene Grafik (Ergebnisse der Umgebungen <code>pspicture</code> , <code>postscript</code> und der <code>\includegraphics</code> -Anweisungen mit PostScript-Dateien) wird unter dem als Parameter übergebenen Namen gespeichert.
<code>\usepicture</code>	<code>\usepicture[⟨keys⟩]{⟨name⟩}</code> Die zuvor mit <code>\savepicture</code> gespeicherte Grafik wird ausgegeben. Der optionale Parameter entpricht 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

Das Verhalten der Anweisungen `\includegraphics`, `\usepicture` und der Umgebung `postscript` kann mit den folgenden optionalen Parametern beeinflusst werden (key-value-Syntax):

frame=⟨true|false⟩ Es wird – ähnlich wie bei der Anweisung `\fbox` – ein Rahmen um die Grafik gezeichnet. Die durch Rotation geänderte Gesamtgröße wird dabei berücksichtigt. Das Zeichnen geschieht im pdfL^AT_EX-Modus; vorher beim Erzeugen der Containerdatei wird dieser Parameter ignoriert. Standard: false.

innerframe=⟨true|false⟩ Wie “**frame**” jedoch wird der Rahmen nur um die Grafik selbst, nicht aber um die resultierende Box gezeichnet.

ignore=⟨true|false⟩ Bei “**true**” wird die Grafik nicht ausgegeben. Bei Angabe von `\savepicture{⟨name⟩}` kann sie später jedoch an anderer Stelle mit `\usepicture` verwendet werden. Standard: false.

showname=⟨true|false⟩ Gibt in kleiner Schrift den tatsächlich verwendeten Dateinamen unter der Grafik aus. Standard: false.

namefont=⟨font commands⟩ Beeinflusst die Schriftart, die bei “**showname=true**” benutzt wird. Standard: `\ttfamily\tiny`

Alle Parameter können auch global per `\setkeys{Gin}{⟨key=value⟩}` gesetzt werden.

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 Compilertests

Es wird getestet, welcher T_EX compiler in welchem Modus läuft (siehe ‘graphics.cfg’ von t_EX/L_{AT}E_XLive). Entsprechend dem Ergebnis bekommen die Umgebungen `pspicture` und `postscript` unterschiedliche Funktionalität. Der Test wird nur ausgeführt, wenn nicht die Paketoptionen `active` oder `inactive` angegeben wurden.

```
22 \ifnum\ppf@TeX@mode=-1\relax
23   \begingroup
Default (TEX with a dvi-to-ps converter)
24   \chardef\x=0 %
Check pdfTEX
25   \@ifundefined{pdfoutput}{}{%
26     \ifcase\pdfoutput\else
27       \chardef\x=1 %
28     \fi
29   }%
Check VTEX
30   \@ifundefined{OpMode}{}{\chardef\x=2}%
31   \expandafter\endgroup
32   \ifcase\x
⇒ DVI mode
33   \def\ppf@TeX@mode{0}%
34   \or
```

⇒ pdfT_EX is running in PDF mode

```

35 \def\ppf@TeX@mode{1}%
36 \else
    ⇒ VTEX is running
37 \def\ppf@TeX@mode{9}%
38 \fi
39 \fi

40 \newcommand*\PDFcontainer{}
41 \edef\PDFcontainer{\jobname-pics.pdf}
42 \newcounter{pspicture}
43 \newcommand*\ppf@other@extensions[1]{}
44 \newcommand*\usepicture[2][{}]{%
45 \newcommand*\savepicture[1]{}

```

pst-pdf-defs

```

46 \newenvironment*{pst-pdf-defs}%
47 {%
48 \endgroup
49 % ??? \@currentvline
50 }{%
51 \begingroup
52 \def\@currentvir{pst-pdf-defs}%
53 }

54 \RequirePackage{graphicx}%
55 \let\ppf@Gininclude@graphics\Gininclude@graphics
56 \let\ppf@Gin@extensions\Gin@extensions
57 \let\ppf@Gin@ii\Gin@ii

58 \newif\ifppf@pdftex@graphic
59 \newif\ifGin@frame\Gin@framefalse
60 \newif\ifGin@innerframe\Gin@innerframefalse
61 \newif\ifGin@showname\Gin@shownamefalse
62 \newif\ifGin@ignore\Gin@ignorefalse

```

\ifpr@outer wird eigentlich im Paket preview definiert. Wir müssen es aber bereits hier zusätzlich tun, da sonst T_EX u. U. beim Parsen der \ifcase-Struktur “außer Tritt” kommt.

```

63 \newif\ifpr@outer

```

\ppf@is@pdfTeX@graphic Parameter #1 ist der Name einer Grafikdatei mit oder ohne Endung, Parameter #2 enthält die gültigen Dateieendungen im pdf-Modus, Parameter #3 enthält die gültigen Dateieendungen im dvi-Modus. Ist es möglich, die Grafik im pdf-Modus zu verarbeiten, werden die Anweisungen in #4 ausgeführt, sonst die in #5.

```

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

```

Statt Einladen einer identifizierten Grafik nur Test der Grafikendung.

```

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

```

Es müssen Dateitypen beider Moden gefunden werden, um die Fehlermeldung “File ‘#1’ not found” zu vermeiden.

```
72 \edef\Gin@extensions{#2,#3}%
Testaufruf. Dabei Ausgabe vollständig verhindern.
73 \pr@outerfalse\ppf@Gininclude@graphics{#1}%
74 \endgroup
75 \if@ppf@pdftex@graphic#4\else#5\fi
76 }
```

```
77 \ifcase\ppf@TeX@mode\relax
```

3.3 Extraction mode (dvi output)

Die Umgebung `pspicture` behält die Definition aus `pstricks.tex`. Ausschließlich der Code der Umgebungen `pspicture` und `postscript` sowie `\includegraphics` mit PS-Dateien bewirken Einträge in die DVI-Datei. Der restliche Code des Dokuments wird bei der Ausgabe der DVI-Datei ignoriert. Nach Wandlung der DVI-Datei über PostScript (“dvips”) nach PDF (Datei `\PDFcontainer`) nimmt jede Grafik genau eine Seite der pdf-Datei ein. Der `TeX`-Compiler mit DVI-Ausgabe sowie die Paketoption “active” erzwingen diesen Modus.

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

```

Es werden auch die im pdf_{La}TeX-Modus erlaubten Endungen von Grafikdateien benötigt.

```
88 \begingroup
89 \let\AtBeginDocument\@gobble \let\PackageWarningNoLine\@gobbletwo
90 \def\pdftexversion{121}\input{pdftex.def}%
91 \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}
92 }%
93 \x

```

Für die im PDF-Modus möglichen Grafikformate dürfen keine speziellen Regeln definiert sein (z. B. wegen ‘dvips’-Erweiterungen). Für sie wird die universelle EPS-Regel verwendet, damit sie zumindest gefunden werden.

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

```

In diesem Modus sollten undefinierte keys keinen Fehler bewirken.

```
101 \@for\@tempa:=\ppf@other@extensions\do{%

```

```

102     \expandafter\let\csname Gin@rule@\@tempa\endcsname\relax}%
103     \DeclareGraphicsRule{*}{eps}{*}{*}%

In diesem Modus keine Funktion.

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

109     \ifppf@tightpage\else
110         \def\PreviewBbAdjust{%
111             -600pt -600pt 600pt 600pt}%
112         \AtEndDocument{%
113             \PackageWarningNoLine{pst-pdf}{Picture container needs cropping.}}%
114     \fi

```

postscript Die Umgebung postscript wertet die trim-Option in derselben Weise wie \includegraphics aus (Angaben ohne Maßeinheit werden als bp interpretiert).

```

115     \newenvironment{postscript}[1][]{%
116     {%
117         \global\let\ppf@PreviewBbAdjust\PreviewBbAdjust
118         \ifppf@tightpage
119             \begingroup
120                 \setkeys{Gin}{#1}%
121                 \xdef\PreviewBbAdjust{%
122                     -\Gin@vllx bp -\Gin@vllx bp \Gin@vurx bp \Gin@vury bp}%
123             \endgroup
124         \fi
125         \ignorespaces
126     }%
127     {\aftergroup\ppf@RestoreBbAdjust}%

128     \PreviewEnvironment{postscript}%
129     \AtBeginDocument{%
130         \@ifundefined{PSTricksLoaded}{}%
131     }%

```

pspicture Originaldefinition preview bekannt machen.

```

132     \PreviewEnvironment{pspicture}%

```

psmatrix Originaldefinition preview bekannt machen.

```

133     \@ifundefined{psmatrix}{}%
134     {%
135         \PreviewEnvironment{psmatrix}%
136         \newcommand*\ppf@set@mode{}%
137         \newcommand*\ppf@test@mmode{%
138             \ifmmode
139                 \ifinner
140                     \let\ppf@set@mode=$%
141                 \else
142                     \def\ppf@set@mode{$$}%
143                 \fi
144             \else
145                 \let\ppf@set@mode=\@empty

```

```

146      \fi
147    }%
148    \let\ppf@psmatrix=\psmatrix
149    \expandafter\let\expandafter\ppf@pr@psmatrix%
150      \expandafter=\csname pr@\string\psmatrix\endcsname
151    \let\ppf@endpsmatrix=\endpsmatrix
152    \def\psmatrix{\ppf@test@mode\ppf@psmatrix}
153    \expandafter\def\csname pr@\string\psmatrix\endcsname{%
154      \ppf@set@mode\ppf@pr@psmatrix}%
155    \def\endpsmatrix{\ppf@endpsmatrix\ppf@set@mode}%
156  }%

```

Internes Makro `\pst@object` bekanntmachen, um manchen PSTricks-Code außerhalb von `pspicture`-Umgebungen ebenfalls verwenden zu können. Derzeit sind Aufrufe der folgenden Art möglich:

```

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

```

Mehr als drei optionale Argumente am Ende des Aufrufs, wie beispielsweise bei `\psline` denkbar, sind noch nicht möglich.

```

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

```

Mehrfaches testweises Setzen von Tabelleninhalten durch “`tabularx`” verhindern.

```

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

```

Unterstützung von `\includegraphicx` aus dem Paket `psfrag`.

```

169    \@ifundefined{pfx@includegraphicx}{}{%
170      \PreviewMacro[{}{}]{\pfx@includegraphicx}%
171    }%

```

`\Gscale@@box` Skalieren verhindern.

```

172    \def\Gscale@@box#1#2#3{%
173      \toks@{\mbox}%
174    }

```

`\Ginclude@graphics` Alle Grafiken mit bekanntem Format (z.B. EPS-Dateien) werden normal verarbeitet, was in diesem Modus bedeutet, dass sie der Preview-Funktionalität unterliegen. Andere Grafiken (z.B. PDF-Dateien) werden ignoriert.

```

175    \def\Ginclude@graphics#1{%
176      \ifpr@outer

```

Im allgemeinen Fall sollen pdf_TE_X-Grafiken bevorzugt werden (Einfügen erst im pdf_TE_X-Modus). Ist nur eine DVIPS-Graphik vorhanden, dann wirkt wieder die Originaldefinition und Registrierung beim `preview`-Paket muss erfolgen.

```

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

```


Dummy-Box, um Division durch Null bei Skalierung/Rotation zu vermeiden. Wird ansonsten ignoriert.

```
178     {\rule{10pt}{10pt}}%
179     {\ppf@Ginclude@graphics{#1}}%
180     \else
```

Innerhalb von PS-Umgebungen (pspicture usw.) muss sich \includegraphics wie die Originaldefinition verhalten (nur die DVIPS-Graphik-Typen sind gültig).

```
181     \ppf@Ginclude@graphics{#1}%
182     \fi
183 }%

184 \PreviewMacro[{}]{\ppf@Ginclude@graphics}%
185 \let\pdfliteral\@gobble%
186 \or
```

3.4 pdfL^AT_EX mode (pdf output)

Ist die Datei \PDFcontainer (default: <jobname>-pics.pdf) vorhanden, so wird der Inhalt der Umgebungen pspicture und postscript ignoriert. Stattdessen wird die zugehörige Grafik aus der Datei \PDFcontainer eingebunden.

```
187 \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (pdfTeX mode)}%
188 \@temptokena{%
189   \let\Gin@PS@file@header\@gobble\let\Gin@PS@literal@header\@gobble
190   \let\Gin@PS@raw\@gobble\let\Gin@PS@restored\@gobble
191   \@ifundefined{PSTricksLoaded}{}{}}%
```

Für PSTricks < 2.0 nötig.

```
192   \PSTricksOff
193   \@ifundefined{c@lor@to@ps}{\def\c@lor@to@ps#1 #2@@{}{}}{}}%
```

Prevent pdfT_EX's message Non-PDF special ignored!.

```
194 \if@ppf@PST@used
195   \let\ppf@temp\AtBeginDvi\let\AtBeginDvi\@gobble
196   \RequirePackage{pstricks}\let\AtBeginDvi\ppf@temp
197 \fi
```

PostScript-Ausgabe jetzt verhindern und später noch einmal.

```
198 \the\@temptokena %%% ???
199 \expandafter\AtBeginDocument\expandafter
200   {\the\@temptokena\@temptokena{}}%
201   \@ifundefined{PSTricksLoaded}{}{}}
202 {%
```

Zum Parsen der Argumente von PSTricks' \pst@object laden wir preview im active-Modus, restaurieren aber die standardmäßigen Definitionen von \output und \shipout. \pr@startbox und \pr@endbox dienen hier nur dazu, um \pst@object wirkungslos zu machen und stattdessen die zugehörige Grafik aus der Containerdatei einzuladen. Derzeit werden nur maximal 3 optionale Parameter in runden Klammern am Ende von \pst@object unterstützt, was für viele, aber nicht für alle Fälle ausreichend ist.

```
203   \newtoks\ppf@temptoken
204   \ppf@temptoken\expandafter{\the\output}%
205   \let\output\@gobble
206   \let\ppf@nofiles\nofiles \let\nofiles\relax
```

```

207 \RequirePackage[active]{preview}[2005/01/29]%
208 \let\shipout=\pr@shipout \let\nofiles\ppf@nofiles
209 \output\expandafter{\the\ppf@temptoken}%
210 \ppf@temptoken{}%

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

```

(Siehe auch identische Definition im DVI-Modus.)

```

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

```

Es werden auch die im DVI-Modus erlaubten Endungen von Grafikdateien benötigt.

```

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

```

Dummy-Definition für die im DVI-Modus gültigen Dateitypen.

```

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

```

(Nur intern zu benutzen!)

```

251 \edef\@tempa{\toks@{\noexpand\frame{\the\toks@}}}%
252 \ifcase#1\relax
253 \ifGin@innerframe\else\let\@tempa\relax\fi
254 \or
255 \ifGin@frame\else\let\@tempa\relax\fi
256 \fi
257 \@tempa
258 }%
259 \define@key{Gin}{showname}[true]{%
260 \lowercase{\Gin@boolkey{#1}}{showname}}%
261 \define@key{Gin}{namefont}{%
262 \begingroup
263 \temptokena\expandafter{\ppf@namefont#1}%
264 \edef\x{\endgroup\def\noexpand\ppf@namefont{\the\@temptokena}}%
265 \x
266 }%
267 \newcommand*\ppf@filename{}%
268 \newcommand*\ppf@namefont{\tiny\ttfamily}%
269 \newcommand*\ppf@Gin@keys{}%
270 \let\ppf@Gin@setfile\Gin@setfile

```

`\Gin@setfile` Realen Dateinamen und ggf. Seitenzahl zur späteren Verwendung merken.

```

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

```

`\Gin@ii` Auswertung der Optionen “frame”, “ignore” usw. sowie weiterer Spezialfälle.

```

274 \def\Gin@ii[#1]#2{%
275 \begingroup

```

Der Wert `\ifGin@innerframe` muss bereits vor Zeichnen des inneren Rahmens bekannt sein. Die Werte für `\ifGin@showname` und `\ppf@namefont` müssen auch nach Darstellung der Grafik verfügbar sein. Daher durch eine Gruppe geschützt vorher Auswertung der Optionen.

```

276 \setkeys{Gin}{#1}%
277 \@temptokena{#1}\def\@tempb{#2}%

```

Leerer Dateiname beim Aufruf von `\usepicture` aus.

```

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

```

Grafiken aus Containerdatei sind bereits skaliert usw. Nicht noch einmal, daher optionalen Parameter ignorieren.

```

280 {%
281 \ifx\@tempb\PDFcontainer
282 \@temptokena{page=\GPT@page}%
283 \fi
284 }%
285 {%
286 \refstepcounter{pspicture}%
287 \@temptokena{page=\the\c@pspicture}\def\@tempb{\PDFcontainer}%
288 }%
289 \fi
290 \ifGin@ignore\else

```

“frame@@=0” = innerer Rahmen, “frame@@=1” = äußerer Rahmen.

```

291     \edef\@tempa{\noexpand\ppf@Gin@ii[frame@@=0,\the\@temptokena,
292         frame@@=1]{\@tempb}}%
293     \@tempa
294     \ifGin@showname
295         \ppf@namefont
296         \raisebox{-\ht\strutbox}[Opt][Opt]{\llap{\ppf@filename}}%
297         \gdef\ppf@filename{}%
298     \fi
299 \fi
300 \endgroup
301 }%

302 \IfFileExists{\PDFcontainer}%
303 {%

```

\ppf@container@max Die Anzahl der in der Containerdatei enthaltenen Seiten.

```

304     \pdfximage{\PDFcontainer}%
305     \edef\ppf@container@max{\the\pdfastximagepages}%

306     \AtEndDocument{%
307         \ifnum\c@pspicture>\z@

```

Warnung ist nur sinnvoll, wenn überhaupt Grafiken benötigt wurden.

```

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

```

`\ppf@isnum` Ist Parameter #1 numerisch, werden Anweisungen in #2 sonst die in #3 ausgeführt (siehe `bibtopic.sty`).

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

`postscript` Beide Umgebungen ignorieren ihren Inhalt und laden stattdessen die zugehörige Grafik aus der Containerdatei. Auf den Wert des dabei benutzten Zählers (`pspicture`) kann per `\label/\ref` zugegriffen werden.

`psmatrix`

```
340 \newcommand*\ppf@set@mode{%
341 \newcommand*\ppf@test@mmode{%
342 \ifmmode
343   \ifinner
344     \let\ppf@set@mode=$%
345   \else
346     \def\ppf@set@mode{${}%
347   \fi
348 \else
349   \let\ppf@set@mode=\@empty
350 \fi
351 }
352 \newenvironment{postscript}[1] []
353 {%
354   \ppf@test@mmode
355   \gdef\ppf@Gin@keys{%
356     \def\@tempa{postscript}\ifx\@tempa\@currenenv\gdef\ppf@Gin@keys{#1}\fi
357   \expandafter\let\expandafter\pst@object
358     \csname pr@\string\pst@object\endcsname
359   \pr@outerfalse
```

Innerhalb der Umgebung ist das Parsen der Argumente von `\pst@object` unnötig, daher wieder Originaldefinition verwenden.

Nötig für `\psmatrix`.

```
360 \@makeother\&%
361 \def\Gin@ii[##1]##2{\setbox\@tempboxa=\vbox\bgroup
362 \ppf@set@mode
363 }%
364 {\ppf@set@mode\egroup\aftergroup\ppf@@getpicture}%
365 \AtBeginDocument{%
366   \@ifundefined{PSTricksLoaded}{}%
367   {%
368     \iffalse
369       \PreviewEnvironment{pspicture}% Why doesn't it work?
370       \g@addto@macro\pspicture{%
371         %\pr@outerfalse% necessary, or already there anyway?
372         \@makeother\&% necessary?
373       \def\Gin@ii[##1]##2{%
374         }%
375       \g@addto@macro\endpspicture{\ppf@@getpicture}%
376     \else
377       \def\pst@@@picture[##1](##2,##3)(##4,##5){\postscript}%
```

```

378     \def\endpspicture{\endpostscript\endgroup}%
379     \fi
380     \@ifundefined{psmatrix}{}%
381     {\let\psmatrix=\postscript\let\endpsmatrix=\endpostscript}%
382     }%
383     \@ifundefined{pfx@includegraphics}{}{%

```

Die im pdfTeX-Modus unnütze Umdefinition von `\includegraphics` (Paket `psfrag`) führt zu zweifachem Einfügen des Ergebnisses, weshalb die Originaldefinition wiederhergestellt wird.

```

384     \let\includegraphics=pfx@includegraphics
385     \def\pfx@includegraphics#1#2{\ppf@@getpicture}%
386     }%
387     }%

```

`\savepicture` Speichert die Nummer der aktuellen Grafik in einem Makro mit Namen `\ppf@@#1`.

```

388     \def\savepicture#1{%
389         \expandafter\xdef\csname ppf@@#1\endcsname{\the\pdfastximage}}%

```

`\usepicture` Fügt Grafik mit symbolischem Namen #2 ein. Der Name muss vorher mit `\savepicture{<Name>}` vereinbart worden sein. Statt des Namens kann auch eine Zahl angegeben werden, die dann direkt eine Grafik aus der Containerdatei adressiert. Der optionale Parameter #1 entspricht dem bei `\includegraphics`.

```

390     \renewcommand*\usepicture[2] []{%
391         \@ifundefined{ppf@@#2}%
392         {%
393             \ppf@isnum{#2}%
394             {\ppf@getpicture{#1}{#2}}%
395             {\latexerror{picture ‘#2’ undefined}\@ehc}%
396         }%
397         {%
398             \begingroup
399             \def\Gininclude@graphics##1{%
400                 \xdef\ppf@filename{#2}%
401                 \setbox\z@\hbox{\pdfrefximage\@nameuse{ppf@@#2}}%
402                 \Gin@nat@height\ht\z@ \Gin@nat@width\wd\z@
403                 \def\Gin@llx{0} \let\Gin@lly\Gin@llx
404                 \Gin@defaultbp\Gin@urx{\Gin@nat@width}%
405                 \Gin@defaultbp\Gin@ury{\Gin@nat@height}%
406                 \Gin@bboxtrue\Gin@viewport@code
407                 \Gin@nat@height\Gin@ury bp%
408                 \advance\Gin@nat@height-\Gin@lly bp%
409                 \Gin@nat@width\Gin@urx bp%
410                 \advance\Gin@nat@width-\Gin@llx bp%
411                 \Gin@req@sizes
412                 \ht\z@\Gin@req@height \wd\z@\Gin@req@width
413                 \leavevmode\box\z@}%
414             \define@key{Gin}{type}{}%
415             \includegraphics[scale=1,#1]{}%
416             \endgroup
417         }}%

```

`\ppf@getpicture` Fügt die Seite (Grafik) mit Nummer #2 aus der Containerdatei ein. Parameter #1: Optionen wie bei `\includegraphics`.

```

418 \newcommand*\ppf@getpicture[2]{%
419   \@tempcnta=#2\relax%
420   \ifnum\@tempcnta>\ppf@container@max
421     \PackageWarningNoLine{pst-pdf}{%
422       pspicture No. \the\@tempcnta\space undefined}%
423   \else
424     \includegraphics[draft=\ppf@draft,#1,page=\the\@tempcnta]%
425       {\PDFcontainer}%
426   \fi
427   \gdef\ppf@Gin@keys{}}%
```

`\ppf@@getpicture` Fügt die nächste Seite (Grafik) aus der Containerdatei ein.

```

428 \newcommand*\ppf@@getpicture{%
429   \ifpr@outer
430     \refstepcounter{pspicture}%
431     \expandafter\ppf@getpicture\expandafter{\ppf@Gin@keys}%
432     {\the\c@pspicture}%
433   \fi}%

```

`pst-pdf-defs` Umgebung, die keine eigene Gruppe aufmacht. Innerhalb der Umgebung bekommt das Zeichen & den Kategoriecode „other“. Gedacht für eigene Makros, die z. B. eine `psmatrix` enthalten. (Einen “Hook” verwenden, falls andere Zeichen auch noch benötigt werden!?)

```

434 \renewenvironment*{pst-pdf-defs}%
435 {%
436   \endgroup
437 %   ??? \@currentvline
438   \chardef\ppf@temp=\catcode'\&%
439   \@makeother\&%
440 }{%
441   \catcode'\&=\ppf@temp
442   \begingroup
443   \def\@currentvir{pst-pdf-defs}%
444 }
445 \else

```

3.5 Inactiver Modus

Es werden nur die Pakete `pstricks` und `graphicx` geladen – keine weitere Einflussnahme. Die Paketoption „inactive“ sowie der \TeX -Compiler erzwingen diesen Modus.

```

446 \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (inactive mode)}%
447 \newenvironment{postscript}[1][\ignorespaces]{%
448 \let\ppf@is@pdfTeX@graphic\relax
449 \fi

450 \InputIfFileExists{pst-pdf.cfg}{%
451 \PackageInfo{pst-pdf}{Local config file pst-pdf.cfg used}}{}
452 \</package>

```

Change History

v1.0a	General: Initial version. 1	v1.0l	General: Options “framesep”, “framerule”, “linewidth” removed, “fname” and “innerframe” added. (RN) 1
v1.0b	General: Some code and documentation cleaning. (RN) 1	v1.0m	General: New package option “notightpage” added. (RN) . . . 1
v1.0c	General: New options “pstricks”, “nopstricks”, “draft” and “final”. (RN) 4	v1.0n	General: Changed macro names (<code>\savepicture</code> and <code>\usepicture</code>). (RN) 1 Some code cleaning. (RN) 1
v1.0d	General: Redefinition of <code>\includegraphics</code> in modes 0 und 1. Now using of eps graphics directly in pdf \LaTeX is possible. (RN) 1	v1.0o	General: New code for “notightpage”. (RN) 7 Option “fname” renamed to “showname”. (RN) 1
v1.0e	<code>postscript</code> : “trim” option added. (RN) 7	v1.0p	General: Some code and documentation cleaning. (RN) 1
v1.0f	General: Config file loading added. (RN) 15 <code>\savepicture</code> : New macro <code>\savepspicture</code> . (RN) 14 <code>\usepicture</code> : New macro <code>\usepspicture</code> . Useful for putting a PSTricks graphic in a box or something else. (RN) 14	v1.0q	<code>\usepicture</code> : Now <code>\usepspicture</code> works for all kind of graphics. (RN) 14
v1.0g	General: Definition of <code>\PDFcontainer</code> now with <code>\edef</code> . (RN) 5 <code>\usepicture</code> : Now <code>\usepspicture</code> does accept a numerical parameter. (RN) 14	v1.0r	<code>\ppf@is@pdfTeX@graphic</code> : Changed <code>\ppf@is@known@graphic</code> to <code>\ppf@is@pdfTeX@graphic</code> . Now pdf \TeX graphics are preferred. (RN) 5
v1.0h	<code>postscript</code> : Based no more on the comment environment from the verbatim package. (RN) 13	v1.0s	General: Scaling e.g. of PostScript pictures now only in extraction mode. Some code cleaning. (RN) 1 <code>\Gin@ii</code> : Rewritten. (RN) 11
v1.0i	<code>\ppf@is@pdfTeX@graphic</code> : No more errors for given files without extensions. (RN) 5	v1.1a	General: Support for the internal PSTricks macro <code>\pst@object</code> . (HjG/RN) 8
v1.0j	General: Check <code>AtBeginDocument</code> for package ‘pstricks’ even if “nopstricks” is given. (RN) . . . 1	v1.1b	General: Ignore the call of <code>\nofiles</code> inside of <code>preview</code> . (RN) 9 Some code and documentation cleaning. (RN) 1
v1.0k	<code>\Gin@setfile</code> : Show also the pagenummer if exists. (RN) 11 <code>\Gin@include@graphics</code> : Prevent division by zero. (RN) 8	v1.1c	General: New package option “tightpage” added. (RN) 1 Special support for “tabularx”. (RN) 8

Supress handling of pdfL ^A T _E X graphic formats in DVI mode. (RN)	6	defs: Support for PSTricks envi- ronment “psmatrix” inside user definitions. (RN,HjG)	1
v1.1d		v1.1l	
psmatrix: Support for PSTricks en- vironment “psmatrix”. (RN) . .	13	General: Support for the package “psfragx”. (RN)	8
v1.1e		v1.1m	
General: New option “display- math” (see preview package). (HjG/RN)	4	General: Merge english and german version of the documentation. (RN)	1
v1.1f		v1.1n	
General: Package option “ignore” reimplemented. Now the com- pilation of the dtx file in L ^A T _E X mode is possible. (RN)	4	General: \nofiles added (sugges- tion of Torsten Bronger).	6
v1.1g		v1.1o	
psmatrix: “psmatrix” environ- ment (preserve math mode). (RN/HjG)	13	\Gscale@@box: Disable scaling. (RN)	8
pspicture: pspicture environment must still parse its arguments. (RN/HjG)	13	v1.1p	
v1.1h		General: \nofiles makes \makeindex and \makeglossary to \relax. \@empty is better be- cause of later \renewcommand’s. . .	6
\Ginclude@graphics: Check if in- side of a PS-related environ- ment (correct graphic inclu- sion). (RN)	8	v1.1pl	
v1.1i		General: \let\output\@gobble be- fore loading of “preview” added. (RN)	9
General: \ifpr@outer must be pre- defined. (HjG/RN)	5	v1.1q	
Package option “final” also for “graphicx”. (RN)	4	General: Problem with “tabularx” and “threeparttable” solved. (RN)	8
\Ginclude@graphics: Correction of the inside check. (RN/HjG) . . .	8	v1.1r	
v1.1k		General: Fixed values for \PreviewBbAdjustbecause \paperwidthis not always defined (suggested by Will Robertson).	7
General: New environment pst-pdf-			

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.

18

<code>\pdfastximagepages</code>	305	<code>\ppf@namefont</code> 263, 264, 268, 295	<code>\PreviewMacro</code> 157, 170, 184, 230
<code>\pdfliteral</code> 185	<code>\ppf@nofiles</code>	.. 206, 208	<code>\psmatrix</code> 148, 150, 152, 153, 381
<code>\pdfoutput</code> 26	<code>\ppf@other@extensions</code> 43, 91, 101, 177, 241, 279	<code>psmatrix</code>	(environ- ment) ... 133 , 340
<code>\pdfrefximage</code> 401	<code>\ppf@pr@psmatrix</code>	..	<code>\pspicture</code> 370
<code>\pdfTeXtext</code> 67, 70	149, 154	<code>pspicture</code>	(environ- ment) . 2 , 132 , 340
<code>\pdfTeXversion</code> 90	<code>\ppf@PreviewBbAdjust</code> 85, 87, 117	<code>pst-pdf-defs</code>	(environ- ment) .. 3 , 46 , 434
<code>\pdfximage</code> 304	<code>\ppf@psmatrix</code>	. 148, 152	<code>\pst@@picture</code> 377
<code>\pfx@includegraphics</code> 384	<code>\ppf@RestoreBbAdjust</code> 86, 127	<code>\pst@object</code> 163, 236, 357, 358
<code>\pfx@includegraphicsx</code> 170, 385	<code>\ppf@set@mode</code> 136, 140, 142, 145, 154, 155, 340, 344, 346, 349, 362, 364	<code>\PSTricksOff</code> 192
<code>\postscript</code>	... 377, 381	<code>\ppf@temp</code> 195, 196, 438, 441	R	
<code>postscript</code>	(environ- ment) . 2 , 115 , 340	<code>\ppf@temptoken</code> 203, 204, 209, 210	<code>\raisebox</code> 296
<code>\ppf@getpicture</code>	225, 364, 375, 385, 428	<code>\ppf@test@mmode</code> 137, 152, 341, 354	<code>\refstepcounter</code>	286, 430
<code>\ppf@container@max</code> 304 , 308, 310, 320, 420	<code>\ppf@TeX@mode</code>	. 2 , 7 , 8, 21, 22, 33, 35, 37, 77, 79, 187, 446	<code>\rule</code> 178
<code>\ppf@draft</code>	3, 13, 14, 424	<code>\pr@cleanup</code>	... 214, 226	S	
<code>\ppf@endpsmatrix</code> 151, 155	<code>\pr@endbox</code> 222	<code>\savepicture</code>	. 3 , 45, 388
<code>\ppf@filename</code>	. 267, 272, 296, 297, 400	<code>\pr@outerfalse</code> 73, 217, 359, 371	<code>\setkeys</code> 120, 276
<code>\ppf@getpicture</code> 394, 418 , 431	<code>\pr@shipout</code> 208	<code>\shipout</code> 208
<code>\ppf@Gin@extensions</code>	56	<code>\PreviewBbAdjust</code>	.. . 87, 110, 117, 121	<code>\string</code>	... 150, 153, 358
<code>\ppf@Gin@ii</code> 57, 291	<code>\PreviewEnvironment</code>	. 128, 132, 135, 369	<code>\strutbox</code> 296
<code>\ppf@Gin@keys</code>	. 269, 355, 356, 427, 431			T	
<code>\ppf@Gin@setfile</code> 270, 271			<code>\tabularx</code> 166
<code>\ppf@Gin@include@graphics</code> 55, 73, 179, 181, 184			U	
<code>\ppf@is@pdfTeX@graphic</code>	. 64 , 177, 279, 448			<code>\usepicture</code>	.. 3 , 44, 390
<code>\ppf@isnum</code>	... 337 , 393			V	
				<code>\voidb@x</code> 224
				X	
				<code>\XKV@err</code> 99