

# Das Paket `pst-pdf`\*

Rolf Niepraschk<sup>†</sup>      Hubert Gäßlein

2008/10/09

## 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 Extraktionsmodus' 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` v1.1v vom 2008/10/09.

<sup>†</sup>`Rolf.Niepraschk@gmx.de`

**notightpage** Die Abmessung der zur Grafik gehörenden  $\TeX$ -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 pdf $\LaTeX$ -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`<sup>1</sup> 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 AMS $\LaTeX$ -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 erzeugen<sup>2</sup>. 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 Umdefinieren des Makros `\PDFcontainer` geändert werden.

## 2.3 Nutzeranweisungen

<code>pspicture</code>	<code>\begin{pspicture}[(keys)] (&lt;x0,x1&gt;)&lt;(y0,y1)&gt; ... \end{pspicture}</code>
	Die <code>pspicture</code> -Umgebung steht zur Verfügung, wenn nicht die Option “nopstricks” angegeben wurde. Sie wird so wie in PSTricks üblich verwendet. Im pdf $\LaTeX$ -Modus wird ihr Inhalt nur dann dargestellt, wenn vorher die Containerdatei erzeugt wurde.
<code>postscript</code>	<code>\begin{postscript}[(keys)] ... \end{postscript}</code>

<sup>1</sup>CTAN: `support/pdfcrop/`

<sup>2</sup>Die 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<sup>A</sup>T<sub>E</sub>X-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[<i>keys</i>]{<i>filename</i>}</code> Wie in <code>graphics/graphics</code> definiert zu verwenden. Zusätzlich ist es nun möglich, auch im pdfL <sup>A</sup> T <sub>E</sub> X-Modus EPS-Dateien als Argument anzugeben und ihren Inhalt darzustellen. Er wird dazu ebenfalls der Containerdatei entnommen.
<code>\includegraphicx</code>	<code>\includegraphicx[<i>keys</i>](<i>pxadd</i>)&lt;<i>ovpfgd</i>&gt;[<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> 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[<i>keys</i>]{<i>name</i>}</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>&amp;</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<sup>A</sup>T<sub>E</sub>X-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 Compiler tests

Es wird getestet, welcher  $\text{\TeX}$  compiler in welchem Modus läuft (siehe ‘graphics.cfg’ von  $\text{te}\text{\TeX}/\text{\TeX}Live$ ). 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 ( $\text{\TeX}$  with a dvi-to-ps converter)
24 \chardef\x=0 %
Check pdf $\text{\TeX}$ 
25 \@ifundefined{pdfoutput}{}{%
26 \ifcase\pdfoutput\else
27 \chardef\x=1 %
28 \fi
29 }%
Check V $\text{\TeX}$ 
30 \@ifundefined{OpMode}{}{\chardef\x=2 }%
31 \expandafter\endgroup
32 \ifcase\x
⇒ DVI mode
33 \def\ppf@TeX@mode{0}%
34 \or
```

⇒ pdf $\TeX$  is running in PDF mode

```
35 \def\ppf@TeX@mode{1}%
36 \else
    ⇒ VT $\TeX$  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 \endgroup
48 % ??? \@currentline
49 }{%
50 \begingroup
51 \def\@currentvir{pst-pdf-defs}%
52 }

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

57 \newif\if@ppf@pdftex@graphic
58 \newif\if@Gin@frame\Gin@framefalse
59 \newif\if@Gin@innerframe\Gin@innerframefalse
60 \newif\if@Gin@showname\Gin@shownamefalse
61 \newif\if@Gin@ignore\Gin@ignorefalse
```

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

```
62 \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.

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

Statt Einladen einer identifizierten Grafik nur Test der Grafikendung.

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

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

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

```
76 \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.

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

Es werden auch die im pdf<sup>L</sup>TeX-Modus erlaubten Endungen von Grafikdateien benötigt.

```
87 \begingroup
88 \let\AtBeginDocument\@gobble \let\PackageWarningNoLine\@gobbletwo
89 \chardef\pdftexversion=121 %
90 \newcount\pdfoutput
91 \pdfoutput=1 %
92 \input{pdftex.def}%
93 \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}
94 }%
95 \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.

```
96 \AtBeginDocument{%
97 \@ifpackageloaded{keyval}{%
98 \def\KV@errx#1{\PackageInfo{keyval}{#1}}%
99 }{}%
100 \@ifpackageloaded{xkeyval}{%
101 \def\XKV@err#1{\PackageInfo{xkeyval}{#1}}%
102 }{}%
```

In diesem Modus sollten undefinierte keys keinen Fehler bewirken.

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

In diesem Modus keine Funktion.

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

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

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

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

131 \PreviewEnvironment{postscript}%
132 \AtBeginDocument{%
133 \@ifundefined{PSTricksLoaded}{}%
134 {%
```

**pspicture** Originaldefinition preview bekannt machen.

```
135 \PreviewEnvironment{pspicture}%
```

**psmatrix** Originaldefinition preview bekannt machen.

```
136 \@ifundefined{psmatrix}{}%
137 {%
138 \PreviewEnvironment{psmatrix}%
139 \newcommand*\ppf@set@mode{}%
140 \newcommand*\ppf@test@mmode{%
141 \ifmmode
142 \ifinner
143 \let\ppf@set@mode=$%
144 \else
```

```

145         \def\ppf@set@mode{$$}%
146         \fi
147     \else
148         \let\ppf@set@mode=\@empty
149     \fi
150 }%
151 \let\ppf@psmatrix=\psmatrix
152 \expandafter\let\expandafter\ppf@pr@psmatrix%
153     \expandafter=\csname pr@string\psmatrix\endcsname
154 \let\ppf@endpsmatrix=\endpsmatrix
155 \def\psmatrix{\ppf@test@mode\ppf@psmatrix}
156 \expandafter\def\csname pr@string\psmatrix\endcsname{%
157     \ppf@set@mode\ppf@pr@psmatrix}%
158 \def\endpsmatrix{\ppf@endpsmatrix\ppf@set@mode}%
159 }%

```

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>>(<o>)
(m = notwendig, * = optional, o = optional)

```

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

```

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

```

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

```

167     \@ifundefined{tabularx}{}{}%
168     \newcolumntype{X}{c}%
169     \expandafter\let\expandafter\tabularx\csname tabular*\endcsname
170     \expandafter\let\expandafter\endtabularx\csname endtabular*\endcsname
171     }%

```

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

```

172     \@ifundefined{pfx@includegraphicx}{}{}%
173     \PreviewMacro[{}{}]{\pfx@includegraphicx}%
174     }%

```

`\Gscale@@box` Skalieren verhindern.

```

175     \def\Gscale@@box#1#2#3{%
176         \toks@{\mbox}%
177     }

```

`\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.

```

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

```



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

```

180   \ppf@is@pdfTeX@graphic{#1}{\ppf@other@extensions}{\Gin@extensions}%
    Dummy-Box, um Division durch Null bei Skalierung/Rotation zu vermeiden. Wird
    ansonsten ignoriert.
181   {\rule{10pt}{10pt}}%
182   {\ppf@Gin@include@graphics{#1}}%
183   \else
    Innerhalb von PS-Umgebungen (pspicture usw.) muss sich \includegraphics
    wie die Originaldefinition verhalten (nur die DVIPS-Graphik-Typen sind gültig).
184   \ppf@Gin@include@graphics{#1}%
185   \fi
186   }%

187   \PreviewMacro[{}]{\ppf@Gin@include@graphics}%
188   \let\pdfliteral@gobble%
189 \or

```

### 3.4 pdf $\LaTeX$ 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.

```

190 \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (pdfTeX mode)}%
    Verhindert pdf $\TeX$ s Warnung Non-PDF special ignored!.
191 \if@ppf@PST@used
192   \let\ppf@temp\AtBeginDvi\let\AtBeginDvi@gobble
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}{}-%
    Für PSTricks < 2.0 nötig.
199     \PSTricksOff
200     \@ifundefined{c@lor@to@ps}{\def\c@lor@to@ps#1 #2\@{}}{}}%
    PostScript-Ausgabe jetzt verhindern und später noch einmal.
201   \the\@temptokena
202   \expandafter\AtBeginDocument\expandafter
203   {\the\@temptokena\@temptokena{}}%
204   \@ifundefined{PSTricksLoaded}{}-%

```

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.

```

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: Gegenüber Originaldefinition vereinfacht.

```

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}%

```

(Siehe auch identische Definition im DVI-Modus.)

```

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 }%

```

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

```

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

```

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

```

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@}{%
(Nur intern zu benutzen!)
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` Realen Dateinamen und ggf. Seitenzahl zur späteren Verwendung merken.

```

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

```

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

```

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

```

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

```

Leerer Dateiname beim Aufruf von `\usepicture` aus.

```

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

```

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

```

281 {%
282 \setkeys{Gin}{#1}%
283 \ifx\ppf@tempb\PDFcontainer
284 \@temptokena{page=\GPT@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” = innerer Rahmen, “frame@@=1” = äußerer Rahmen.
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}[Opt][Opt]{\llap{\ppf@filename}}%
299     \gdef\ppf@filename{}%
300     \fi
301     \fi
302     \endgroup
303 }%

304 \IfFileExists{\PDFcontainer}%
305 {%

```

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

```

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

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

    Warnung ist nur sinnvoll, wenn überhaupt Grafiken benötigt wurden.
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` Ist Parameter #1 numerisch, werden Anweisungen in #2 sonst die in #3 ausgeführt (siehe `bibtopic.sty`).

```

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

```

`psmatrix` 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.

`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\@currenvir
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@includegraphicx}{-}{%

```

Die im pdf<sub>T</sub>E<sub>X</sub>-Modus unnütze Umdefinition von `\includegraphics` (Paket `psfrag`) führt zu zweifachem Einfügen des Ergebnisses, weshalb die Originaldefinition wiederhergestellt wird.

```

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

```

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

```
377 \def\savepicture#1{%
378 \expandafter\xdef\csname ppf@@@#1\endcsname{\the\pdflastximage}}%
```

`\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`.

```
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@curx{\Gin@nat@width}%
394 \Gin@defaultbp\Gin@cury{\Gin@nat@height}%
395 \Gin@bboxtrue\Gin@viewport@code
396 \Gin@nat@height\Gin@cury bp%
397 \advance\Gin@nat@height-\Gin@lly bp%
398 \Gin@nat@width\Gin@curx 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` Fügt die Seite (Grafik) mit Nummer #2 aus der Containerdatei ein. Parameter #1: Optionen wie bei `\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` Fügt die nächste Seite (Grafik) aus der Containerdatei ein.

```
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` Umgebung, die keine eigene Gruppe aufmacht. Innerhalb der Umgebung bekommt das Zeichen `&` den Kategoriecode „other“. Gedacht für eigene Makrodefinitionen, die z. B. eine `psmatrix` enthalten.

```

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 Inactiver Modus

Es werden nur die Pakete `pstricks` und `graphicx` geladen – keine weitere Einflussnahme. Die Paketoption „inactive“ sowie der  $\text{V}\text{T}\text{E}\text{X}$ -Compiler erzwingen diesen Modus.

```

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	General: Initial version. . . . .	1	und 1. Now using of eps graphics directly in pdf $\text{\LaTeX}$ is possible. (RN) . . . . .	1	
v1.0b	General: Some code and documentation cleaning. (RN) . . . . .	1	v1.0e	<code>postscript</code> : “trim” option added. (RN) . . . . .	7
v1.0c	General: New options “pstricks”, “nopstricks”, “draft” and “final”. (RN) . . . . .	4	v1.0f	General: Config file loading added. (RN) . . . . .	15
v1.0d	General: Redefinition of <code>\includegraphics</code> in modes 0		<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		<code>\ppf@is@pdfTeX@graphic</code> . Now pdf $\TeX$ graphics are preferred. (RN) ..... 5
v1.0g	General: Definition of <code>\PDFcontainer</code> now with <code>\edef</code> . (RN) ..... 5	v1.0s	General: Scaling e.g. of PostScript pictures now only in extraction mode. Some code cleaning. (RN) ..... 1
	<code>\usepicture</code> : Now <code>\usepspicture</code> does accept a numerical parameter. (RN) ..... 14		<code>\Gin@ii</code> : Rewritten. (RN) ..... 11
v1.0h	<code>psmatrix</code> : Based no more on the comment environment from the verbatim package. (RN) .... 13	v1.1a	General: Support for the internal PSTricks macro <code>\pst@object</code> . (HjG/RN) ..... 8
v1.0i	<code>\ppf@is@pdfTeX@graphic</code> : No more errors for given files without extensions. (RN) ..... 5	v1.1b	General: Ignore the call of <code>\nofiles</code> inside of <code>preview</code> . (RN) ..... 9
v1.0j	General: Check <code>AtBeginDocument</code> for package ‘ <code>pstricks</code> ’ even if ‘ <code>nopstricks</code> ’ is given. (RN) ... 1		Some code and documentation cleaning. (RN) ..... 1
v1.0k	<code>\Gin@setfile</code> : Show also the page number if exists. (RN) .... 11	v1.1c	General: New package option ‘ <code>tightpage</code> ’ added. (RN) ..... 1
	<code>\Gin@include@graphics</code> : Prevent division by zero. (RN) ..... 8		Special support for ‘ <code>tabularx</code> ’. (RN) ..... 8
v1.0l	General: Options ‘ <code>framesep</code> ’, ‘ <code>framerule</code> ’, ‘ <code>linewidth</code> ’ removed, ‘ <code>fname</code> ’ and ‘ <code>innerframe</code> ’ added. (RN) ..... 1		Supress handling of pdf $\LaTeX$ graphic formats in DVI mode. (RN) ..... 6
v1.0m	General: New package option ‘ <code>notightpage</code> ’ added. (RN) ... 1	v1.1d	<code>postscript</code> : Support for PSTricks environment ‘ <code>psmatrix</code> ’. (RN) 13
v1.0n	General: Changed macro names ( <code>\savepicture</code> and <code>\usepicture</code> ). (RN) ..... 1	v1.1e	General: New option ‘ <code>displaymath</code> ’ (see <code>preview</code> package). (HjG/RN) ..... 4
	Some code cleaning. (RN) ..... 1	v1.1f	General: Package option ‘ <code>ignore</code> ’ reimplemented. Now the compilation of the dtx file in $\LaTeX$ mode is possible. (RN) ..... 4
v1.0o	General: New code for ‘ <code>notightpage</code> ’. (RN) ..... 7	v1.1g	<code>postscript</code> : ‘ <code>psmatrix</code> ’ environment (preserve math mode). (RN/HjG) ..... 13
	Option ‘ <code>fname</code> ’ renamed to ‘ <code>showname</code> ’. (RN) ..... 1		<code>pspicture</code> : <code>pspicture</code> environment must still parse its arguments. (RN/HjG) ..... 13
v1.0p	General: Some code and documentation cleaning. (RN) ..... 1	v1.1h	<code>\Gin@include@graphics</code> : Check if inside of a PS-related environment (correct graphic inclusion). (RN) ..... 8
v1.0q	<code>\usepicture</code> : Now <code>\usepspicture</code> works for all kind of graphics. (RN) ..... 14	v1.1i	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		



Package option “final” also for “graphicx”. (RN) . . . . .	4	v1.1p1	General: <code>\let\output@gobble</code> before loading of “preview” added. (RN) . . . . .	9
<code>\Ginclude@graphics</code> : Correction of the inside check. (RN/HjG) . . . . .	8	v1.1q	General: Problem with “tabularx” and “threeparttable” solved. (RN) . . . . .	8
v1.1k		v1.1r	General: Fixed values for <code>\PreviewBbAdjust</code> because <code>\paperwidth</code> is not always defined (suggested by Will Robertson). . . . .	7
General: New environment <code>pst-pdf-defs</code> : Support for PSTricks environment “psmatrix” inside user definitions. (RN,HjG) . . . . .	1	v1.1s	General: Dummy definition of the page key in DVI mode. . . . .	7
v1.1l		v1.1t	General: Remove the line “ <code>\let\output@gobble</code> ” because of bad side effects. (RN) . . . . .	9
General: Support for the package “psfragx”. (RN) . . . . .	8	<code>postscript</code> : Using <code>environ</code> the environment <code>postscript</code> is now simple and more robust. (RN) . . . . .	13	
v1.1m		v1.1u	General: <code>\pdfoutput</code> must be set when loading “ <code>pdftex.def</code> ” in DVI mode. (RN) . . . . .	6
General: Merge english and german version of the documentation. (RN) . . . . .	1			
v1.1n				
General: <code>\nofiles</code> added (suggestion of Torsten Bronger). . . . .	6			
v1.1o				
<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. . . . .	6			



303, 305, 311, 313, 324, 326, 413	\ppf@is@pdfTeX@graphic . 63, 179, 280, 436	\PreviewMacro . . . . . . 159, 172, 186, 231
\pdflastximage . . . . 377	\ppf@isnum . . . 338, 381	\psmatrix . . . . 150, 152, 154, 155, 368
\pdflastximagepages 306	\ppf@namefont . . . . . . 264, 265, 269, 296	psmatrix (environ- ment) . . . 135, 341
\pdfliteral . . . . . 187	\ppf@nofiles . . 206, 210	pspicture (environ- ment) . 2, 134, 341
\pdfoutput . . . . . 26, 90	\ppf@other@extensions . . . . . 43, 92, 102, 179, 242, 280	pst-pdf-defs (environ- ment) . . 3, 46, 422
\pdfrefximage . . . . . 389	\ppf@output 204, 205, 211	\pst@@@picture . . . . 365
\pdfTeXtext . . . . . 66, 69	\ppf@pr@psmatrix . . . . . . . 151, 156	\pst@object . . . 165, 237
\pdfTeXversion . . . . 89	\ppf@PreviewBbAdjust . . . . . 84, 86, 119	\PSTricksOff . . . . . 198
\pdfximage . . . . . 305	\ppf@psmatrix . 150, 154	
\pfx@includegraphics . . . . . 372	\ppf@RestoreBbAdjust . . . . . 85, 129	<b>R</b>
\pfx@includegraphicx . . . . . 172, 373	\ppf@set@mode . . . . . . 138, 142, 144, 147, 156, 157, 341, 345, 347, 350	\raisebox . . . . . 297
\postscript . . . 365, 368	\ppf@shipout . . 207, 209	\refstepcounter 287, 418
postscript (environ- ment) . 2, 117, 341	\ppf@temp . . . . . . 191, 192, 426, 429	\rule . . . . . 180
\ppf@getpicture . . . 226, 361, 373, 416	\ppf@test@mode . . . . . . . 139, 154, 342	<b>S</b>
\ppf@container@max . . . . . . 305, 309, 311, 321, 408	\ppf@TeX@mode . 2, 7, 8, 21, 22, 33, 35, 37, 76, 78, 189, 434	\savepicture . 3, 45, 376
\ppf@draft 3, 13, 14, 412	\pr@cleanup . . . 215, 227	\setkeys . . . . . 122, 277
\ppf@endpsmatrix . . . . . . . 153, 157	\pr@endbox . . . . . 223	\shipout . . . . . 207, 209
\ppf@filename . 268, 273, 297, 298, 388	\pr@outerfalse . 72, 218	\string . . . . . 152, 155
\ppf@getpicture . . . . . . . 382, 406, 419	\pr@startbox . . . . . 212	\strutbox . . . . . 297
\ppf@Gin@extensions 55	\PreviewBbAdjust . . . 86, 112, 119, 123	<b>T</b>
\ppf@Gin@ii . . . . 56, 292	\PreviewEnvironment . . . . . 130, 134, 137	\tabularx . . . . . 168
\ppf@Gin@keys . 270, 357, 359, 415, 419		<b>U</b>
\ppf@Gin@setfile . . . . . . . 271, 272		\usepicture . . 3, 44, 378
\ppf@Gin@include@graphics . . . . . 54, 72, 181, 183, 186		<b>V</b>
		\voidb@x . . . . . 225
		<b>X</b>
		\XKV@err . . . . . 100