

The xypdf package

Daniel Müllner

Version 1.7, dated 2011/03/20

Abstract

The `xypdf` package improves the output quality of the `Xy-pic` package when PDF documents are generated. It produces generic PDF code for graphical elements like lines, curves and circles instead of approximating these elements with glyphs in special fonts as the original `Xy-pic` package does. The `xypdf` package works with both `TEX` and `LATEX` in the occurrences of `pdfTEX`, `XYTEX` and ϵ -`TEX` with `dvipdfm(x)` to generate PDF files. `xypdf` is being integrated and distributed together with `Xy-pic`, starting with `Xy-pic` version 3.8.

Contents

1	Introduction	1
2	Usage	4
3	Acknowledgements	5
4	To do	5
5	Differences between <code>xypdf</code> and the <code>dvips</code> backend	5
6	The fine print: curves with multiple segments	6
7	Troubleshooting	6
8	Copyright, license and disclaimer	7
9	Implementation	8
9.1	Straight lines	13
9.2	Squiggled lines	19
9.3	Circles	20
9.4	Rotation and scaling	22
9.5	Temporary registers	25
9.6	Bézier curves	29
9.7	New improved curve styles	33
9.8	Multiple solid curves	35
9.9	A Bézier curve offset algorithm	37
9.10	Multiple dashed curves	55
9.11	Multiple dotted curves	56
9.12	Squiggled curves	60
9.13	Spline continuation	65
9.14	Color	66
9.15	Frames	70
9.16	Line styles	83
10	Changelog	84

1 Introduction

The \Xypic package is a utility for typesetting diagrams in \TeX and \LaTeX documents. The authors of the \Xypic package put much effort into the feature that most graphical elements are coded within the limited possibilities of the device independent file format (DVI). The diagrams can thus be generated with even the most basic \TeX systems and displayed universally by all device drivers. For example, diagonal lines are composed of short dashes, which are glyphs in a special font. Since there are dashes in 127 discrete directions in the font `xydash10`, diagonal lines which do not match one of these slopes look slightly rugged when they are magnified.

For a better output quality in Postscript files, the authors of the \Xypic package provided a Postscript backend for DVI-to-Postscript drivers. These extensions draw lines and curves by generic Postscript commands, thus trading a much better output quality against universality of the produced DVI files.

The development of the present package was based on \Xypic version 3.7, from 1999, where there is no support for pdf \TeX . In order to produce PDF files with high-quality \Xypic diagrams, users had to use so far the Postscript file format as an intermediate step or embed the diagrams as external graphics. However, since many users directly generate PDF files from the \TeX or DVI files (with bookmarks, hyperlinks and other PDF features), it is highly desirable to also have the possibility of directly generating \Xypic diagrams with high-quality PDF graphics elements.

The PDF driver provided by the `xypdf` package adapts the output routines of the \Xypic package to generate high-quality graphics for PDF output. It works with both pdf \TeX and the two-step compilation $\text{\TeX} \rightarrow \text{dvipdfm}(x)$ with an intermediate DVI file. Thus, it also works with \XeTeX since this program internally uses a modification of `dvipdfmx`. Note that some version of $\varepsilon\text{-TeX}$ is needed (which is anyway used by default in modern \TeX installations). [Figure 1](#) compares the output quality of a small \Xypic diagram.

The `xypdf` package is very similar to the Postscript drivers for \Xypic . It does not have (yet) all of their features (see [Section 4](#)) but is much more powerful in other respects, e.g. when drawing multiple curves. In general, it greatly improves graphics quality. Currently, the following features are implemented:

- Both straight lines and curves (solid, dashed, dotted and squiggled) are drawn by generic PDF commands.
- \Xypic automatically draws the symbols of which lines and curves are composed at the very beginning and end of a segment. It then distributes the inner symbols evenly

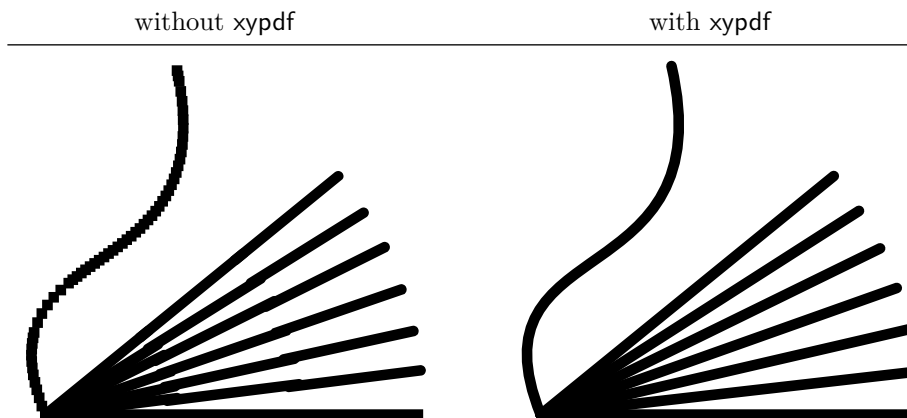
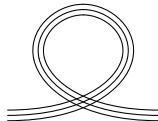


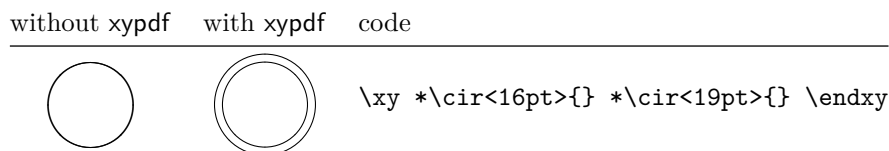
Figure 1: Comparison of \Xypic output, magnified 10 times.

across the segment. Since the arc length of a Bézier curve is normally not proportional to its parameter, this is a nontrivial task in the case of curves. The `xypdf` package handles this better than the original code. Compare the output in [Figure 2](#).

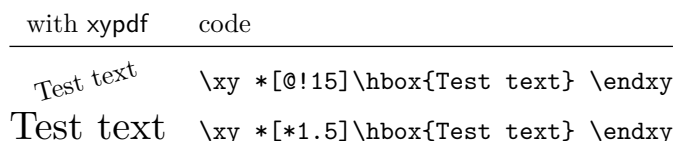
- As a highlight, `xypdf` features a Bézier curve offset algorithm, producing high-quality curves with two or three parallel strokes.



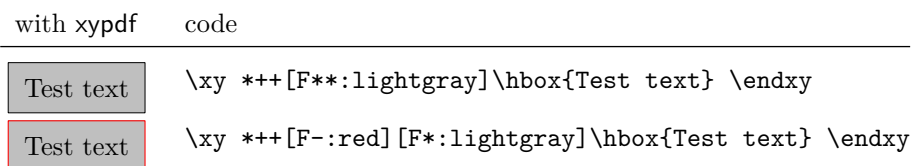
- The `\cir` object draws circles of arbitrary radius.



- `xypdf` supports the “rotate” extension of `Xy-pic`.



- `xypdf` supports the “frame” extension of `Xy-pic`. The only intended difference to the Postscript drivers is the appearance of the `{**}` fill style. The postscript drivers fill the frame and stroke it with a line of thickness 0. The PDF driver strokes the region instead with a black line which is half as thick as the normal lines. Tip: If you want a colored boundary, overlay two frames.



- `xypdf` supports the “color” extension of `Xy-pic`. As described in the `Xy-pic` Reference Manual, colors can be defined by the `\newcycolor` command, e. g.

```
\newxycolor{mygreen}{.5 0 1 .5 cmyk}
```

In addition, if the command `\color` is defined, e. g. if the `color`¹ or `xcolor`² package has been loaded, `xypdf` recognizes the named colors from these packages and uses the mechanisms from these packages to set colors in the output DVI or PDF file.































An example:



This was generated by `\usepackage{xcolor}` in the document preamble and the following code:

¹<http://www.ctan.org/tex-archive/macros/latex/required/graphics/>

²<http://www.ctan.org/tex-archive/macros/latex/contrib/xcolor/>

without xypdf	with xypdf	$\langle arrow\ style \rangle$
		$\{ - \}$
		$2\{ - \}$
		$3\{ - \}$
		$\{ -- \}$
		$2\{ -- \}$
		$3\{ -- \}$
		$\{ \cdot \}$
		$2\{ \cdot \}$
		$3\{ \cdot \}$
		$\{ \sim \}$
		$2\{ \sim \}$
		$3\{ \sim \}$
		$\{ \sim\sim \}$
		$2\{ \sim\sim \}$
		$3\{ \sim\sim \}$

Code: `\xy (0,0) \ar @{\langle arrow\ style \rangle}@{(20,20),(10,-20)} (30,0) \endxy`

Figure 2: Comparison of X_y-pic output for curves with various line styles.

GreenYellow:		Yellow:		Goldenrod:	
Dandelion:		Apricot:		Peach:	
Melon:		YellowOrange:		Orange:	
BurntOrange:		Bittersweet:		RedOrange:	
Mahogany:		Maroon:		BrickRed:	
Red:		OrangeRed:		RubineRed:	
WildStrawberry:		Salmon:		CarnationPink:	
Magenta:		VioletRed:		Rhodamine:	
Mulberry:		RedViolet:		Fuchsia:	
Lavender:		Thistle:		Orchid:	
DarkOrchid:		Purple:		Plum:	
Violet:		RoyalPurple:		BlueViolet:	
Periwinkle:		CadetBlue:		CornflowerBlue:	
MidnightBlue:		NavyBlue:		RoyalBlue:	
Blue:		Cerulean:		Cyan:	
ProcessBlue:		SkyBlue:		Turquoise:	
TealBlue:		Aquamarine:		BlueGreen:	
Emerald:		JungleGreen:		SeaGreen:	
Green:		ForestGreen:		PineGreen:	
LimeGreen:		YellowGreen:		SpringGreen:	
OliveGreen:		RawSienna:		Sepia:	
Brown:		Tan:		Gray:	
Black:		White:			

Figure 3: Additional color names provided by `\UseCrayolaColors`.

```

\definecolor{mygreen}{cmyk}{.5 0 1 .5}
\xymatrix{*[orange] [F-:blue] \hbox{Orange}
& *[mygreen] \hbox{Green}}

```

When a named color has been defined by both `\newxycolor` and by a color package command like `\definecolor`, the X_Y-pic definition overrides the general one.

The X_Y-pic command `\UseCrayolaColors` defines a set of color names, as explained in the X_Y-pic Reference Manual. [Figure 3](#) lists these colors.

If you notice any unwanted behavior, please generate a minimal example and e-mail it to the author of this package. Current contact details are available at <http://math.stanford.edu/~muellner>. Please report situations where the algorithms produce arithmetic overflows. Also, the code is not really optimized for speed but for accuracy, so feel free to report a significant slowdown of the compiling process for your thesis/paper/book.

2 Usage

Use `pdf` as an option to the X_Y-pic package, as in

```
\usepackage[pdf]{xy}
```

or

```

\usepackage{xy}
\xyoption{pdf}

```

for L^AT_EX and

```
\input xy
\xyoption{pdf}
```

for plain T_EX. Do not use one of the other driver options to X_Y-pic like **dvips**, since combining two drivers will most likely result in mutilated diagrams.

The **xypdf** functionality can be switched off and on within the document by **\xypdfoff** and **\xypdfon**.

When you use plain T_EX, make sure that **xypdf** is loaded after any global changes to the math fonts.

3 Acknowledgements

Since the **xypdf** package extends X_Y-pic, some ideas are adopted from this package and its Postscript backend, and the author gratefully acknowledges the service which Kristoffer H. Rose and Ross Moore did to the mathematical community with their original package.

4 To do

- Support for the “line” extension.

5 Differences between **xypdf** and the **dvips** backend

Most users of **xypdf** will have used the **dvips** backend before. In this section, we list the intentional differences between **xypdf**’s behavior and the **dvips** driver. If you see further differences which are not mentioned here, this could be a bug, and please notify the author, Daniel Müllner.



- **xypdf** draw frames always around the current point *c*.

with dvips	with xypdf	code
		<code>\xy *{+} **<2pc>!R[F-]{ABC}\endxy</code>
		<code>\xy *{+} **<2pc>!R[Fe]{ABC}\endxy</code>

- The center of rotation and scaling is always the current point *c*.

with dvips	with xypdf	code
		<code>\xy *[@!30]{ABCDEF} *{ABCDEF} \endxy</code>
		<code>\xy *!R[@!30]{ABCDEF} *!R{ABCDEF} \endxy</code>

- Line frames which are drawn as **\frm** objects are always black with the **dvips** backend, as opposed to filled frames. **xypdf** changes the color for both line and filled frames.

with dvips	with xypdf	code
		<pre>\xy **<2pc,1pc>[o]{} *[orange]\frm{e} \endxy \xy **<2pc,1pc>[o]{} *[orange]\frm{**} \endxy</pre>

The [F...] object modifiers give the same colors to frames with both drivers.

6 The fine print: curves with multiple segments

Since the dashes in Bézier segments are aligned to the boundary points, this would result in dashes of double length when a curve is composed of several Bézier segments, as shown in the upper left diagram in Figure 4. To avoid this, xypdf records the end point of each segment and adapts the dash pattern whenever the starting point of a segment coincides with the end point of the previous one (see the upper right diagram). Analogous improvements apply to the “dotted” and “squiggled” line styles.

Since this mechanism does not exist in the original Xy-pic, it can be switched on and off by `\xypdfcontpatternon` and `\xypdfcontpatternoff`. By default, it is switched on.

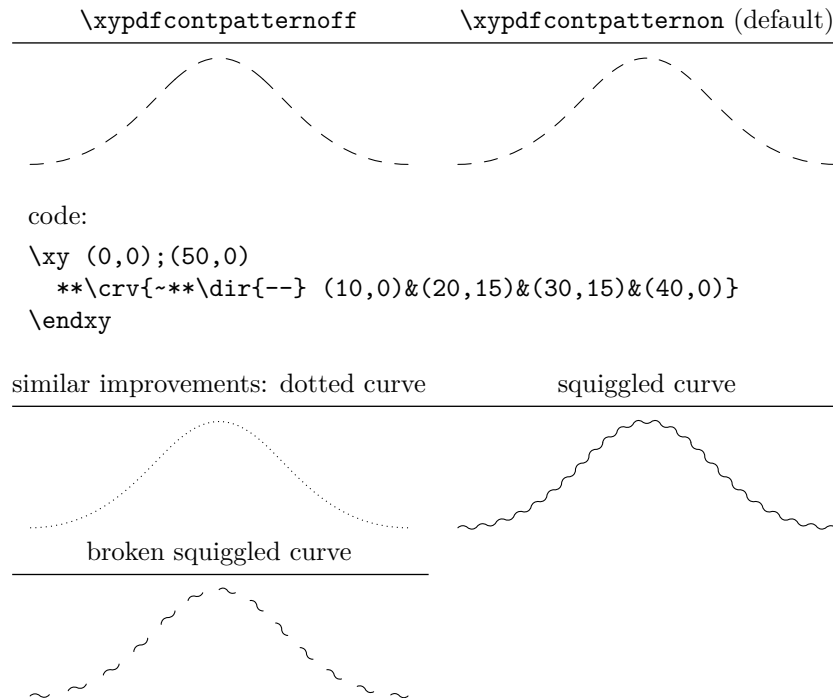


Figure 4: Seamless patterns for curves with multiple segments.

7 Troubleshooting

- I get the error message `pdfTeX version 1.40.0 or higher is needed for the xypdf package with PDF output`

You seem to use an old version of pdfTeX. If you cannot update your T_EX system for some reason, you may still use the xypdf package in DVI mode and produce a PDF file via `dvipdfm(x)`. The pathway T_EX → `dvipdfm(x)` is preferable in many cases anyway since it usually produces much smaller PDF files.

- I get the error message `eTeX is needed for the xypdf package`.

In your $\text{T}_{\text{E}}\text{X}$ installation, the $\varepsilon\text{-T}_{\text{E}}\text{X}$ features are not enabled, although they most certainly can be in any reasonably modern $\text{T}_{\text{E}}\text{X}$ installation. You must probably rebuild the $\text{T}_{\text{E}}\text{X}$ and \LaTeX format files with $\varepsilon\text{-T}_{\text{E}}\text{X}$ enabled. Please consult the documentation of your $\text{T}_{\text{E}}\text{X}$ distribution on how to rebuild the format files.

8 Copyright, license and disclaimer

The copyright for the `xypdf` package is by its author, Daniel Müllner. Current contact details will be maintained at <http://math.stanford.edu/~muellner>.

The `xypdf` package is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 2 of the License, or (at your option) any later version. This license is available at <http://www.gnu.org/licenses>.

This program is distributed in the hope that it will be useful, but without any warranty; without even the implied warranty of merchantability or fitness for a particular purpose. See the GNU General Public License for more details.

9 Implementation

This is the code for the file `xypdf.tex`. From version 1.4 on, it is loaded as the option `pdf` to `Xy-pic`.

```
\xypdfversion \xyprovide defines \xypdfversion.
\xypdfdate
1 <*basic>
2 \ifx\xyloaded\undefined\input xy \fi
3 \xyprovide{pdf}{PDF driver}{1.7}%
4 {Daniel M\"ullner\newline}%
5 {\url{http://math.stanford.edu/~muellner}}{}
6 \ifx\makeatletter\undefined\input miniltx \fi
7 \newcommand*\xypdfdate{2011/03/20}
8 \newdriver{%
9 \xyaddsupport{pdf}\xP@pdf@on
10 \xyaddsupport{color}\xP@color@on
11 \xyaddsupport{curve}\xP@curve@on
12 \xyaddsupport{frame}\xP@frame@on
13 % \xyaddsupport{line}\xP@line@on
14 \xyaddsupport{rotate}\xP@rotate@on
15 }
16 \xyaddunsupport{pdf}\xP@pdf@off
17 \xyaddunsupport{color}\xP@color@off
18 \xyaddunsupport{curve}\xP@curve@off
19 \xyaddunsupport{frame}\xP@frame@off
20 %\xyaddunsupport{line}\xP@line@off
21 \xyaddunsupport{rotate}\xP@rotate@off
```

See the end of the file for the code that loads the other files `xypdf-*.tex`.

```
\xypdfon Commands for switching the driver on and off.
\xypdfoff
22 \newcommand*\xypdfon{%
23 \xP@pdf@on
24 \xP@color@on
25 \xP@curve@on
26 \xP@frame@on
27 \xP@line@on
28 \xP@rotate@on
29 }
30 \newcommand*\xypdfoff{%
31 \xP@pdf@off
32 \xP@color@off
33 \xP@curve@off
34 \xP@frame@off
35 \xP@line@off
36 \xP@rotate@off
37 }
```

Test for ε -TEX

```
38 \ifx\unexpanded\@undefined
39 \PackageError{xypdf}{eTeX is needed for the xypdf package}{}
40 \fi
```

`\xP@testpdfsave` Test for `\pdfsave`, which was introduced in pdfTEX version 1.40.0.

```
41 \newcommand*\xP@testpdfsave{%
42 \ifpdf
43 \ifx\pdfsave\@undefined
```

```

44      \PackageError{xypdf}{pdfTeX version 1.40.0 or higher is needed for the %
45      xypdf^^J%
46      package with PDF output}{}%
47      \fi
48      \fi
49      \let\xP@testpdfsave\@undefined
50  }

\xP@warning Check for \PackageWarning.
51  \ifx\PackageWarning\@undefined
52    \newcommand*\xP@warning[2]{\{%
53      \newlinechar'\^^J%
54      \@warning{Package #1 Warning: #2\@empty.}%
55    }}
56  \else
57    \newcommand*\xP@warning{\PackageWarning}
58  \fi

\xP@pdf@on The following initializations are necessary for each supported extension, otherwise the
\xP@pdf@off \xP@hook commands will not work.
59  \newcommand*\xP@pdf@on{}
60  \newcommand*\xP@pdf@off{}

\xP@color@on
\xP@color@off 61  \newcommand*\xP@color@on{}
62  \newcommand*\xP@color@off{}

\xP@curve@on
\xP@curve@off 63  \newcommand*\xP@curve@on{}
64  \newcommand*\xP@curve@off{}

\xP@frame@on
\xP@frame@off 65  \newcommand*\xP@frame@on{}
66  \newcommand*\xP@frame@off{}

\xP@line@on
\xP@line@off 67  \newcommand*\xP@line@on{}
68  \newcommand*\xP@line@off{}

\xP@rotate@on
\xP@rotate@off 69  \newcommand*\xP@rotate@on{}
70  \newcommand*\xP@rotate@off{}

\xP@hook Commands for switching the driver on and off.
71  \newcommand*\xP@hook[2]{%
72    \edef\next@{%
73      \let\expandafter\noexpand\csname xP@old@#2\endcsname
74      \expandafter\noexpand\csname#2\endcsname}%
75    \next@
76    \expandafter\edef\csname xP@#1@on\endcsname{%
77      \unexpanded\expandafter\expandafter\expandafter{\csname xP@#1@on\endcsname}%
78      \let\expandafter\noexpand\csname#2\endcsname
79      \expandafter\noexpand\csname xP@#2\endcsname
80    }%
81    \expandafter\edef\csname xP@#1@off\endcsname{%

```

```

82   \unexpanded\expandafter\expandafter\expandafter{\csname xP@#1@off\endcsname}%
83   \let\expandafter\noexpand\csname#2\endcsname
84   \expandafter\noexpand\csname xP@old@#2\endcsname
85 }%
86 }

```

\xP@defpdfliteral Two possibilities to insert literal PDF commands, one for pdfTeX and one for dvipdfm(x).

\xP@literal The command \xP@cm changes the current transformation matrix.

```

\xP@cm      87 \newcommand*\xP@defpdfliteral{%
\xP@setcolor 88 \ifpdf
\xP@resetcolor 89 \newcommand*\xP@literal[1]{\pdfsave\pdfliteral{##1}\pdfrestore}
90 \newcommand*\xP@cm[5]{%
91   \pdfsave
92   \pdfsetmatrix{##1 ##2 ##3 ##4}%
93   ##5%
94   \pdfrestore
95 }

```

Mimick pdfTeX.def from the graphicx package.

```

96 \ifundefined{@pdfcolorstack}{%
97   \def\pdfcolorstack{\z@}%
98 }{%
99   \newcommand*\xP@setcolor[3]{%
100     \pdfcolorstack\pdfcolorstack push{##1 ##2 ##1 ##3}}
101   \newcommand*\xP@resetcolor{\pdfcolorstack\pdfcolorstack pop\relax}%
102 \else
103   \newcommand*\xP@literal{%
104     \xP@warning{xypdf}{%
105       The produced DVI file is NOT PORTABLE. Convert it with^^J%
106       dvipdfm(x) to the PDF format but do not expect the DVI file itself to be^^J%
107       displayed correctly\@gobble}%
108     \global\let\xP@literal\xP@literal@
109     \xP@literal
110   }
111   \newcommand*\xP@literal@[1]{\special{pdf:content ##1}}
112   \newcommand*\xP@cm[5]{%
113     \special{pdf:btrans matrix ##1 ##2 ##3 ##4 0 0}%
114     ##5%
115     \special{pdf:etrans}%
116   }
117   \newcommand*\xP@setcolor[3]{\special{pdf:bcolor[##1]}}
118   \newcommand*\xP@resetcolor{\special{pdf:ecolor}}%
119 \fi
120 \let\xP@defpdfliteral\@undefined
121 }

```

Rely on the ifpdf package to test for PDF output. The \AtEndOfPackage is necessary if xypdf is loaded as an option in \usepackage[*options*]{xy}. If it is called as a plain T_EX package, the commands below can be executed immediately.

```

122 \DN@{\@firstofone}
123 \DNii@{xy}
124 \ifx\@currname\nextii@
125   \ifx\AtEndOfPackage\@undefined
126   \else
127     \DN@{\AtEndOfPackage}%
128   \fi

```

```

129 \fi
130 \next@
131 {\RequirePackage{ifpdf}}%
132 \xP@testpdfsave
133 \xP@defpdfliteral}

\xP@digits Set the precision for dimension output according to pdfTeX's \pdfdecimaldigits. If this
number is not defined, use dvipdfm's default precision, which is two decimals.
134 \ifx\pdfdecimaldigits\undefined
135 \newcommand*\xP@digits{2}
136 \else
137 \@ifdefinable\xP@digits\relax
138 \xdef\xP@digits{\the\pdfdecimaldigits}
139 \ifnum\pdfdecimaldigits<2
140 \xP@warning{xypdf}{%
141 The precision in \string\pdfdecimaldigits\space is only \xP@digits\space
142 decimals.^~J%
143 It is recommended to set \string\pdfdecimaldigits\space to 2 or 3 for %
144 best output quality@gobble}
145 \fi
146 \fi

\xP@dim Conversion between TeX points (pt) and PDF/Postscript points (bp)
147 \newcommand*\xP@dim[1]{%
148 \expandafter\xP@removePT\the\dimexpr(#1)*800/803\relax\space}

\xP@precdim Precise conversion between TeX points (pt) and PDF/Postscript points (bp). No truncation.
149 \newcommand*\xP@precdim[1]{\xP@EARPT\dimexpr(#1)*800/803\relax\space}

\xP@EARPT
150 \newcommand*\xP@EARPT{\expandafter\removePT@the}

\xP@coor Coordinates: two dimensions
151 \newcommand*\xP@coor[1]{\xP@dim{#1}\xP@dim}

\xP@removePT The following two macros round and truncate a dimension to the desired number of decimal
digits.
152 \@ifdefinable\xP@removePT\relax
153 {\@makeother\p\@makeother\t\gdef\xP@removePT#1pt{\xP@removePT@#10000@}}

\xP@removePT@
154 \@ifdefinable\xP@removePT@\relax
155 \ifcase\xP@digits
0 decimals
156 \def\xP@removePT@#1.#2#3@{%
157 \ifnum#2<5
158 #1%
159 \else
160 \the\numexpr-\if-#1-\else-#1+\fi\@ne\relax
161 \fi
162 }
163 \or

```

1 decimal

```
164 \def\xP@removePT@#1#2.#3#4#5@{%
165 \ifnum#4<5
166 #1#2%
167 \if#30%
168 \else
169 .#3%
170 \fi
171 \else
172 \expandafter\xP@removePT
173 \the\dimexpr#1#2.#3pt+\if#1--\fi.12pt\relax
174 \fi
175 }
176 \or
```

2 decimals

```
177 \def\xP@removePT@#1#2.#3#4#5#6@{%
178 \ifnum#5<5
179 #1#2%
180 \if#40%
181 \if#30%
182 \else
183 .#3%
184 \fi
185 \else
186 .#3#4%
187 \fi
188 \else
189 \expandafter\xP@removePT
190 \the\dimexpr#1#2.#3#4pt+\if#1--\fi786sp\relax
191 \fi
192 }
193 \or
```

3 decimals

```
194 \def\xP@removePT@#1#2.#3#4#5#6#7@{%
195 \ifnum#6<5
196 #1#2%
197 \if#50%
198 \if#40%
199 \if#30%
200 \else
201 .#3%
202 \fi
203 \else
204 .#3#4%
205 \fi
206 \else
207 .#3#4#5%
208 \fi
209 \else
210 \expandafter\xP@removePT
211 \the\dimexpr#1#2.#3#4#5pt+\if#1--\fi79sp\relax
212 \fi
213 }
214 \or
```

4 decimals

```

215 \def\xP@removePT@#1#2.#3#4#5#6#7#8@{%
216   \ifnum#7<5
217     #1#2%
218     \if#60%
219       \if#50%
220         \if#40%
221           \if#30%
222             \else
223               .#3%
224             \fi
225           \else
226             .#3#4%
227           \fi
228         \else
229           .#3#4#5%
230         \fi
231       \else
232         .#3#4#5#6%
233       \fi
234     \else
235       \expandafter\xP@removePT
236       \the\dimexpr#1#2.#3#4#5#6pt+\if#1--\fi8sp\relax
237     \fi
238   }
239 \else

```

5 or more decimals: no truncation

```

240 \let\xP@dim\xP@precdim
241 \fi

```

`\xP@lw` Find out the default line width in the math fonts. This is done at the beginning of the document, when hopefully all potential changes to math fonts have taken place.

```

242 \AtBeginDocument{%
Initialize math fonts
243   {\setboxz@h{$ $}}%
244   \@ifdefinable\xP@lw\relax
245   \@ifdefinable\xP@preclw\relax
246   \edef\xP@preclw{\the\fontdimen8\textfont3}%
247   \edef\xP@lw{\xP@dim\xP@preclw}%
248   \PackageInfo{xypdf}{Line width: \xP@preclw}%
249 }

```

9.1 Straight lines

`\line@` Also change the code for `\dir{-}` as an object. Now these dashes are not drawn from the dash font any more but by generic PDF line commands.

```

250 \xP@hook{pdf}{\line@}
251 \newcommand*\xP@line@{%
252   \setboxz@h{%
253     \xP@setsolidpat
254     \xP@stroke{0 0 m \xP@coord{\cosDirection\xdashl@}{\sinDirection\xdashl@}1}%
255   }%
256   \U@c{\sinDirection\xdashl@
257   \D@c\z@
258   \ifdim\U@c<\z@
259     \multiply\U@c\m@ne

```

```

260 \xP@swapdim\U@c\D@c
261 \fi
262 \ht\z@\U@c
263 \dp\z@\D@c
264 \R@c\cosDirection\xydashl@
265 \L@c\z@
266 \ifdim\R@c<\z@
267 \multiply\R@c\m@ne
268 \xP@swapdim\L@c\R@c
269 \fi
270 \hskip\L@c\boxz@\hskip\R@c
271 \edef\tmp@{\egroup\U@c\the\U@c\D@c\the\D@c\L@c\the\L@c\R@c\the\R@c}%
272 \tmp@
273 \Edge@c={\rectangleEdge}%
274 \edef\Upness@{\ifdim\z@<\U@c1\else0\fi}%
275 \edef\Leftness@{\ifdim\z@<\L@c1\else0\fi}%
276 \def\Drop@@{\styledboxz@}%
277 \def\Connect@@{\solid@}%
278 }

\solid@ This is the hook for solid straight lines. Derived from \xyPSSolid@ in xyps.tex.
\xP@solid@ 279 \xP@hook{pdf}{\solid@}
280 \newcommand*\xP@solid@{\straight@\xP@solidSpread}

\xP@solidSpread
281 \@ifdefinable\xP@solidSpread\relax
282 \def\xP@solidSpread#1\repeat@{ {%
Neglect zero-length lines.
283 \@tempswatrue
284 \ifdim\X@p=\X@c
285 \ifdim\Y@p=\Y@c
286 \@tempswafalse
287 \fi
288 \fi
289 \if@tempswa
290 \xP@setsolidpat
291 \xP@stroke{\xP@coor\X@p\Y@p m \xP@coor\X@c\Y@c l}%
292 \fi
293 }}

\xP@pattern
294 \newcommand*\xP@pattern{}

\xP@setsolidpat Pattern for solid lines
295 \newcommand*\xP@setsolidpat{%
296 \def\xP@pattern{1 J 1 j []0 d}%
297 \global\let\xP@lastpattern\xP@solidmacro
298 }

\xP@stroke
299 \newcommand*\xP@stroke[1]{\xP@literal{\xP@lw w \xP@pattern\space#1 S}}

\dash@ This is the hook for dashed straight lines. Derived from \xyPSdashed@ in xyps.tex.
\xP@dash@ 300 \xP@hook{pdf}{\dash@}
301 \newcommand\xP@dash@{\line@\def\Connect@@{\straight@\xP@dashedSpread}}

```

`\xP@dashedSpread`

```

302 \ifdefinable\xP@dashedSpread\relax
303 \def\xP@dashedSpread#1\repeat@{ {%
304   \xP@vecLen
Neglect zero-length lines.
305   \ifdim\@tempdimb>\z@
306     \xP@setdashpat
307     \xP@savec
308     \xP@stroke{\xP@coord\X@p\Y@p m \xP@coord\X@c\Y@c l}%
309   \fi
310 }}
```

`\xP@setdashpat` The formula for the dash length is the same as in the `dashed` operator in `xypsdict.tex`:

$$(\text{dash length}) = \frac{l}{2 \cdot \text{round}\left(\frac{l+d}{2d}\right) - 1},$$

where l is the length of the line and d is the minimal dash length.

The length l must be in `\@tempdimb`. The dash length is returned in `\@tempdima`.

```

311 \newcommand*\xP@setdashpat{%
312   \xP@testcont\xP@dashmacro
313   \ifxP@splinecont
Special pattern in case this line continues another dashed segment.
314     {\count@ \numexpr2*((\@tempdimb-\xydashl@/3)/(2*\xydashl@))\relax
315     \global\dimen@i
316     \ifnum\count@>\z@
317       \dimexpr\@tempdimb/\count@\relax
318     \else
319       \z@
320     \fi
321   }%
322   \@tempdima\dimen@i
323   \edef\xP@pattern{1 J 1 j [%
324     \ifdim\@tempdima>\z@
325       \xP@precdim\@tempdima]\xP@precdim\@tempdima
326     \else
327       ] 0 %
328     \fi
329   d}%
330 \else
331   \@tempdima
332   \ifdim\@tempdimb>\xydashl@
333     \dimexpr\@tempdimb/(2*((\@tempdimb+\xydashl@)/(2*\xydashl@))-1)\relax
334   \else
335     \z@
336   \fi
337   \edef\xP@pattern{1 J 1 j [%
338     \ifdim\@tempdima>\z@\xP@precdim\@tempdima\fi
339   ] 0 d}%
340 \fi
341 \global\let\xP@lastpattern\xP@dashmacro
342 }
```

`\xP@setcldashpat` Dash pattern for closed paths. Offset is half of the dash length to avoid artifacts.


```

343 \newcommand*\xP@setcldashpat{%
344   {\count@\numexpr2*((\@tempdimb-\xydashl@/3)/(2*\xydashl@))\relax
345   \xdef\@gtempa{1 J 1 j [\ifnum\count@>\z@\xP@precdim{\@tempdimb/\count@}\fi}%
346   \ifnum\count@>\z@\xP@precdim{\@tempdimb*3/2/\count@}\else0 \fi d}%
347 }%
348 \edef\xP@pattern{\@gtempa}%
349 }

```

\point@ This is the hook for points. Derived from \xyPSPoint@ in xyps.tex.

```

\xP@point@ 350 \xP@hook{pdf}{point@}
351 \newcommand*\xP@point@{\xP@zerodot\egroup\Invisible@false
352   \Hidden@false\def\Leftness@{.5}\def\Upness@{.5}\ctipEdge@
353   \def\Drop@@{\stylenboxz}%
354   \def\Connect@@{\straight@\xP@dottedSpread}%
355 }

```

\xP@zerodot

```

356 \newcommand*\xP@zerodot{%
357   \hb@xt@\z@{\hss
358     \vbox to\z@{\vss\hrule\@width\xP@preclw\@height\xP@preclw\vss}%
359     \hss}%
360 }

```

\xP@dottedSpread

```

361 \@ifdefinable\xP@dottedSpread\relax
362 \def\xP@dottedSpread#1\repeat@{ {%
363   \xP@veclen
364   \ifdim\@tempdimb>\z@
365     \xP@setdottedpat
366     \xP@savvec
367     \xP@stroke{\xP@coor\X@p\Y@p m \xP@coor\X@c\Y@c l}%
368     \fi
369 }}

```

\xP@setdottedpat The formula for the distance between dots is the same as in the dotted operator in xypsdict.tex:

$$(\text{dot distance}) = \frac{l}{\text{round}\left(\frac{l}{2\text{pt}}\right) + 1},$$

where l is the length of the line.

The length l must be in \@tempdimb.

```

370 \newcommand*\xP@setdottedpat{%
371   \xP@testcont\xP@dotmacro
372   \ifxP@splinecont
373     \@tempdima\dimexpr\@tempdimb/(\@tempdimb/131072+1)-\xP@preclw\relax
374     \edef\xP@pattern{%
375       0 J [%

```

Produce a dot pattern only when the segment is long enough.

```

376     \ifdim\@tempdima>\z@
377       \xP@precdim\xP@preclw\xP@precdim\@tempdima
378       \fi

```

Advance the offset very slightly by 1sp to really hide the first dot in the viewer. (This improves the display at least in the author's PDF-Xchange viewer.)

```

379       ]\xP@precdim{\xP@preclw+1sp}d}%

```

```

380 \else
381   \advance\@tempdimb-\xP@preclw
382   \ifdim\@tempdimb<\z@\@tempdimb\z@\fi
383   \@tempdima\dimexpr\@tempdimb/(\@tempdimb/131072+1)-\xP@preclw\relax
384   \edef\xP@pattern{%
385     0 J [%
Produce a dot pattern only when the segment is long enough.
386     \ifdim\@tempdima>\z@
387       \xP@precdim\xP@preclw\xP@precdim\@tempdima
388       \fi
389     ]0 d}%
390 \fi
391 \global\let\xP@lastpattern\xP@dotmacro
392 }

```

`\xP@setcldottedpat` Dotted pattern for closed paths. Offset is half of the dot distance to avoid artifacts.

```

393 \newcommand*\xP@setcldottedpat{%
394   \@tempdima\dimexpr\@tempdimb/(\@tempdimb/131072+1)-\xP@preclw\relax
395   \edef\xP@pattern{%
396     0 J [%
397     \ifdim\@tempdima>\z@
398       \xP@precdim\xP@preclw\xP@precdim\@tempdima
399       \fi
400     ]\xP@precdim{\dimexpr\xP@preclw+\@tempdima/2\relax}d}%
401 }

```

In contrast to the Postscript drivers for $\text{\texttt{Xy-pic}}$, where some computations are left to the Postscript code, all arithmetic for the PDF output must be done by $\text{\texttt{T\textsubscript{E}X}}$ itself. With $\text{\texttt{T\textsubscript{E}X}}$'s rudimentary fixed-point arithmetic, it is still a pain to compute even the length of a line segment, but things have become considerably easier with $\varepsilon\text{\texttt{T\textsubscript{E}X}}$.

`\xP@abs` Absolute value

```

402 \newcommand*\xP@abs[1]{\ifdim#1<\z@\multiply#1\m@ne\fi}

```

`\xP@ifabsless`

```

403 \newcommand*\xP@ifabsless[2]{\ifpdfabsdim#1<#2}
404 \ifx\ifpdfabsdim\undefined
405   \renewcommand*\xP@ifabsless[2]{\ifdim\ifdim#1<\z@-\fi#1<\ifdim#2<\z@-\fi#2}
406   \@gobble\fi
407 \fi

```

`\xP@swapdim` Works unless parameter #2 is `\@tempdima`.

```

408 \newcommand*\xP@swapdim[2]{\@tempdima#1#2\@tempdima}

```

`\xP@swapnum` Works unless parameter #2 is `\@tempcnta`.

```

409 \newcommand*\xP@swapnum[2]{\@tempcnta#1#2\@tempcnta}

```

`\xP@min` Maximum of two lengths

```

410 \newcommand*\xP@min[2]{\ifdim#1<#2#1\else#2\fi}

```

`\xP@max` Maximum of two lengths

```

411 \newcommand*\xP@max[2]{\ifdim#1>#2#1\else#2\fi}

```

`\xP@Max` Assigns #1 the maximum of #1 and the absolute value of #2.

```

412 \newcommand*\xP@Max[2]{#1\ifdim#2<\z@\xP@max#1{-#2}\else\xP@max#1#2\fi}

```

`\xP@sqrt` Square root algorithm. The argument is in `\@tempdima`, and the start value for the iteration in `\@tempdimc`. The result goes into `\@tempdimb`.

```
413 \newcommand*\xP@sqrt{%
414   \loop
415     \@tempdimb\dimexpr(\@tempdimc+(\@tempdima*\p@/\@tempdimc))/2\relax
```

ε -TeX's `\unless` instead of `\else` since the plain TeX `\loop` cannot deal with `\else`.

```
416   \unless\ifdim\@tempdimc=\@tempdimb
```

iterate: (old approx.) := (new approx.)

```
417     \@tempdimc\@tempdimb\relax
418   \repeat
419 }
```

`\xP@veclen` Absolute length of the vector (`\d@X`, `\d@Y`). The result goes into the register `\@tempdimb`. Several L^AT_EX registers are used as temporary registers, so this function is called safely within a group.

(Maybe it is not necessary to scale the coordinates so much as it is done here, and a simpler code would be fine as well.)

```
420 \newcommand*\xP@veclen{%
421   \xP@veclen@
422   \global\dimen@i\@tempdimb
423   }\@tempdimb\dimen@i
424 }
```

`\xP@veclen@`

```
425 \newcommand*\xP@veclen@{%
426   \xP@abs\d@Y
```

1) Strictly vertical vector

```
427   \ifdim\d@X=\z@
428     \@tempdimb\d@Y
429   \else
430     \xP@abs\d@X
```

2) Strictly horizontal vector

```
431   \ifdim\d@Y=\z@
432     \@tempdimb\d@X
433   \else
```

3) Diagonal vector. $5931642\text{sp} = \sqrt{\text{maxdimen}/2}$. Test whether the components are small enough so that their sum of squares does not generate an arithmetic overflow.

```
434     \@tempswatrue
435     \ifdim\d@X>5931641\text{sp}\relax\@tempswafalse\fi
436     \ifdim\d@Y>5931641\text{sp}\relax\@tempswafalse\fi
437     \if@tempswa
```

3a) Small vector. `\count@` contains a scaling factor for a precise fixed-point arithmetic.

```
438     \count@\@ne
439     \loop
440       \@tempdima\dimexpr\d@X*\d@X/\p@+\d@Y*\d@Y/\p@\relax
```

If the coordinates are small enough, scale them up to improve precision.

```
441     \ifdim\@tempdima<4096\text{pt}
442       \@tempcnta\ifdim\@tempdima<1024\text{pt}\ifdim\@tempdima<256\text{pt}8\else4\fi%
443       \else\tw@\fi
444       \multiply\d@X\@tempcnta
445       \multiply\d@Y\@tempcnta
```

```

446      \multiply\count@\@tempcnta
447      \repeat
Starting value for the square root algorithm
448      \@tempdimc\dimexpr(\d@X+\d@Y)*3/4\relax
449      \xP@sqrt
Rescale
450      \@tempdimb\dimexpr\@tempdimb/\count@\relax
451      \else
452      \ifdim\d@X>83042982sp\relax\@tempwattrue\fi
453      \ifdim\d@Y>83042982sp\relax\@tempwattrue\fi
454      \if@tempswa
3b) Large vector. Scale the coordinates down to avoid an overflow. 11927552sp = 182pt
455      \@tempdima\dimexpr\d@X/182*\d@X/11927552+\d@Y/182*\d@Y/11927552\relax
456      \@tempdimc\dimexpr(\d@X+\d@Y)*3/728\relax
457      \xP@sqrt
458      \multiply\@tempdimb182\relax
459      \else
3c) Medium vector. Also scale the coordinates down. 12845056sp = 196pt = 142pt
460      \@tempdima\dimexpr\d@X*\d@X/12845056+\d@Y*\d@Y/12845056\relax
461      \@tempdimc\dimexpr(\d@X+\d@Y)*3/56\relax
462      \xP@sqrt
463      \multiply\@tempdimb14\relax
464      \fi
465      \fi
466      \fi
467      \fi
468 }

```

9.2 Squiggled lines

```

\squiggledSpread@ This is the hook for squiggled straight lines.
\xP@squiggledSpread@
469 \xP@hook{pdf}{\squiggledSpread@}
470 \@ifdefinable\xP@squiggledSpread@\relax
471 \def\xP@squiggledSpread@#1\repeat@{
472   \xP@veclen

```

Neglect zero-length lines.

```

473   \ifdim\@tempdimb>\z@
474     \edef\@tempa{\xP@coord\X@p\Y@p m }%
475     \toks@\expandafter{\@tempa}%
\@tempcnta = number of squiggles
476   \@tempcnta\numexpr\@tempdimb/\xybsqll@\relax
477   \ifnum\@tempcnta<\tw@\@tempcnta\tw@\fi
478   \@tempdima\dimexpr\d@X/\@tempcnta\relax
479   \@tempdimc\dimexpr\d@Y/\@tempcnta\relax

```

Reverse the direction of the little arcs, if the last squiggle from the previous segment makes it necessary.

```

480   \xP@testcont\xP@oddsquigglemacro
481   \ifxP@splinecont
482     \def\xP@squigsign{-}%
483   \else
484     \let\xP@squigsign\@empty
485   \fi

```

```

486 \count@\z@
487 \loop

```

The fraction is the continuous fraction approximation for the best spline approximation to a quarter circle ($147546029/534618434 \approx \frac{1}{2} \cdot 0.55196760761152504532$).

```

488 \xP@append\toks@{%
489 \xP@coor{\X@p+\d@X*\count@/\@tempcnta+(\@tempdima
490 -\xP@squigsign\ifodd\count@-\fi\@tempdimc)*147546029/534618434}%
491 {\Y@p+\d@Y*\count@/\@tempcnta+(\@tempdimc
492 +\xP@squigsign\ifodd\count@-\fi\@tempdima)*147546029/534618434}%
493 }%
494 \advance\count@\@ne
495 \xP@append\toks@{%
496 \xP@coor{\X@p+\d@X*\count@/\@tempcnta-(\@tempdima
497 -\xP@squigsign\ifodd\count@-\fi\@tempdimc)*147546029/534618434}%
498 {\Y@p+\d@Y*\count@/\@tempcnta-(\@tempdimc
499 +\xP@squigsign\ifodd\count@-\fi\@tempdima)*147546029/534618434}%
500 \xP@coor{\X@p+\d@X*\count@/\@tempcnta}%
501 {\Y@p+\d@Y*\count@/\@tempcnta}%
502 c }%
503 \ifnum\count@<\@tempcnta
504 \repeat
505 \xP@setsolidpat

```

Record the direction of the last squiggle.

```

506 \global\expandafter\let\expandafter\xP@lastpattern
507 \ifodd\numexpr\count@\if\xP@squigsign-+1\fi\relax
508 \xP@oddsquigglemacro
509 \else
510 \xP@evensquigglemacro
511 \fi
512 \xP@savec
513 \xP@stroke{\the\toks@}%
514 \fi
515 }}

```

\xP@squigsign

```

516 \newcommand*\xP@squigsign{}

```

\xP@append

```

517 \newcommand*\xP@append[2]{\{
518 \edef\@tempa{\#1\the\#2}\}%
519 \expandafter\xP@tempa
520 }

```

9.3 Circles

\circhar@@ Replacement macro for the circle chars.

```

\xP@circhar@@ 521 \xP@hook{pdf}{\circhar@@}
522 \newcommand*\xP@circhar@@[1]{%
523 \expandafter\xP@circhar@@@ \ifcase#1 %

```

Bézier segments for 1/8 circle. Let

$$a := \sqrt{1/2} \approx .707106781,$$

$$b := \frac{8}{3}\sqrt{2} \cos(\pi/8) (1 - \cos(\pi/8)) \approx .2652164898,$$

$$c := \frac{1}{3}(-3 + 8 \cos(\pi/8) - 2 \cos^2(\pi/8)) \approx .8946431596,$$

$$d := \frac{1}{2}b(2 + 3 \cos(\pi/8) - \cos^2(\pi/8)) \approx .5195704027.$$

(We have $\cos(\pi/8) = \frac{1}{2}\sqrt{2 + \sqrt{2}}$.)

The fractions below are best possible rational approximations (obtained by continued fractions) to the following coordinates:

$(0, 0), (0, -b), (1 - c, -d), (1 - a, -a)$

```
524      00%
525      0{-173517671/654249180}%
526      {65307479/619869377}{-34221476/65864945}%
527      {225058681/768398401}{-543339720/768398401}%
528      \or
```

$(0, -a), (a - d, -c), (a - b, -1), (a, -1)$

```
529      0{-543339720/768398401}%
530      {181455824/967576667}{-554561898/619869377}%
531      {826676217/1870772527}{-1}%
532      {543339720/768398401}{-1}%
533      \or
```

$(0, -1), (b, -1), (d, -c), (a, -a)$

```
534      0{-1}%
535      {173517671/654249180}{-1}%
536      {34221476/65864945}{-554561898/619869377}%
537      {543339720/768398401}{-543339720/768398401}%
538      \or
```

$(0, -a), (c - a, -d), (1 - a, -b), (1 - a, 0)$

```
539      0{-543339720/768398401}%
540      {181455824/967576667}{-34221476/65864945}%
541      {225058681/768398401}{-173517671/654249180}%
542      {225058681/768398401}0%
543      \or
```

$(0, a), (c - a, d), (1 - a, b), (1 - a, 0)$

```
544      0{543339720/768398401}%
545      {181455824/967576667}{34221476/65864945}%
546      {225058681/768398401}{173517671/654249180}%
547      {225058681/768398401}0%
548      \or
```

$(0, 1), (b, 1), (d, c), (a, a)$

```
549      01%
550      {173517671/654249180}1%
551      {34221476/65864945}{554561898/619869377}%
552      {543339720/768398401}{543339720/768398401}%
553      \or
```

$(0, a), (a - d, c), (a - b, 1), (a, 1)$

```
554      0{543339720/768398401}%
555      {181455824/967576667}{554561898/619869377}%
556      {826676217/1870772527}1%
557      {543339720/768398401}1%
558      \or
```

$(0, 0), (0, b), (1 - c, d), (1 - a, a)$

```
559      00%
560      0{173517671/654249180}%

```

```

561 {65307479/619869377}{34221476/65864945}%
562 {225058681/768398401}{543339720/768398401}%
563 \fi}

```

`\xP@circhar@@` Draw the arc of $1/8$ circle and use the same space as the chars from the circle font do.

```

564 \newcommand\xP@circhar@@[8]{%
565 \xP@setsolidpat
566 \xP@stroke{\xP@coord{\R@*#1}{\R@*#2}m
567 \xP@coord{\R@*#3}{\R@*#4}\xP@coord{\R@*#5}{\R@*#6}%
568 \xP@coord{\R@*#7}{\R@*#8}c}%
569 \vrule width\z@ height\R@ depth\R@
570 \kern\dimexpr\R@*#7\relax
571 }

```

`\cirrestrict@@` Basically, `\cirrestrict@@` is turned into a no-op and does not change the radius.

```

\xP@cirrestrict@@ 572 \xP@hook{pdf}{\cirrestrict@@}
573 \newcommand*\xP@cirrestrict@@{\count@ \z@ \relax}
574 \</basic>

```

9.4 Rotation and scaling

```

575 \< *rotate>
576 \xycatcodes

```

`\xypdf-ro@loaded`

```

577 \expandafter\let\csname xypdf-ro@loaded\endcsname\@empty

```

`\xyscale@@` Scale the box 0 to the factors in #1 and #2.

```

\xP@xyscale@@ 578 \xP@hook{rotate}{\xyscale@@}
579 \newcommand*\xP@xyscale@@[2]{%
580 \setboxz@h{%
581 \hskip\L@c
582 \hskip-\R@p
583 \lower\U@p\hbox{\xP@cm{#1}00{#2}%
584 {\raise\U@p\hb@xt@ \z@{\hskip-\L@c\hskip\R@p\boxz@\hss}}}%
585 }%
586 }%
587 \global\let\xP@lastpattern\@empty
588 }

```

`\xyRotate@@` Rotation in the direction #1.

```

\xP@xyRotate@@ 589 \xP@hook{rotate}{\xyRotate@@}
590 \newcommand\xP@xyRotate@@{\xP@rotate@\xP@trigfromdir}

```

`\doSpecialRotate@@` Rotation by the angle in #1.

```

\xP@doSpecialRotate@@ 591 \xP@hook{rotate}{\doSpecialRotate@@}
592 \ifdefinable\xP@doSpecialRotate@@\relax
593 \def\xP@doSpecialRotate@@#1@@{\xP@rotate@\xP@trig{#1}}

```

`\xP@rotate@` Common code for both rotations: rotate the box 0.

```

594 \newcommand*\xP@rotate@[2]{%
595 \setboxz@h{%
596 #1{#2}%
597 \hskip\L@c
598 \hskip-\R@p
599 \lower\U@p\hbox{\xP@cm\cosDirection\sinDirection

```

```

600     {\if-\sinDirection\else-\sinDirection\fi}\cosDirection
601     {\raise\U@p\hb@xt@z@{\hskip-\L@c\hskip\R@p\boxz@hss}}}%
602   }%
603 }%
604 \global\let\xP@lastpattern\@empty
605 }

\xP@trig Calculate sine and cosine from the angle in #1.
606 \newcommand*\xP@trig[1]{%
607   \@tempdima\dimexpr#1pt\relax
Translate the argument into the interval [0pt, 360pt].
608   \@tempdimb\@tempdima

$$23592960 = 360 \cdot 65536$$

609   \divide\@tempdimb23592960
610   \advance\@tempdima-23592960\@tempdimb
611   \ifdim\@tempdima<z@\advance\@tempdima360pt\fi
612   \@tempdimb\@tempdima

$$5898240 = 90 \cdot 65536$$

613   \divide\@tempdimb5898240
It's enough to know sin between 0° and 90°. The cos and the values in the other quadrants
can be derived from that.
614   \ifcase\@tempdimb
615     \xP@sinpoly
616     \edef\sinDirection{\xP@EARPT\@tempdimb}%
617     \@tempdima\dimexpr90pt-\@tempdima\relax
618     \xP@sinpoly
619     \edef\cosDirection{\xP@EARPT\@tempdimb}%
620   \or
621     \@tempdima\dimexpr180pt-\@tempdima\relax
622     \xP@sinpoly
623     \edef\sinDirection{\xP@EARPT\@tempdimb}%
624     \@tempdima\dimexpr90pt-\@tempdima\relax
625     \xP@sinpoly
626     \edef\cosDirection{\xP@EARPT\dimexpr-\@tempdimb\relax}%
627   \or
628     \@tempdima\dimexpr\@tempdima-180pt\relax
629     \xP@sinpoly
630     \edef\sinDirection{\xP@EARPT\dimexpr-\@tempdimb\relax}%
631     \@tempdima\dimexpr90pt-\@tempdima\relax
632     \xP@sinpoly
633     \edef\cosDirection{\xP@EARPT\dimexpr-\@tempdimb\relax}%
634   \or
635     \@tempdima\dimexpr360pt-\@tempdima\relax
636     \xP@sinpoly
637     \edef\sinDirection{\xP@EARPT\dimexpr-\@tempdimb\relax}%
638     \@tempdima\dimexpr90pt-\@tempdima\relax
639     \xP@sinpoly
640     \edef\cosDirection{\xP@EARPT\@tempdimb}%
641   \else
642     \PackageError{xypdf}{Unexpected case in sin/cos calculation}%
643     {Feel free to contact the author of the xypdf package with a minimal %
644       example.}%
645   \fi
646 }

```


`\xP@sinpoly` Polynomial approximation to the sine in the interval [0pt,90pt]. The deviation should be $\pm 1\text{sp}$ maximal (but no guarantee). (3rd order, 4 subintervals, exact values for 0pt and 90pt)

```

647 \newcommand*\xP@sinpoly{%
648   \ifdim\@tempdima<49pt
649     \ifdim\@tempdima<27pt
650       \@tempdimb\dimexpr((\@tempdima*-529771058/16039085-1384933sp)%
651         * \@tempdima/268756075+10714164sp)*\@tempdima/613777813\relax
652     \else
653       \advance\@tempdima-27pt
654       \@tempdimb\dimexpr((\@tempdima*-743101305/20672414-238989613sp)%
655         * \@tempdima/80975565+42661556sp)*\@tempdima/622461739+2\p@)%
656       *157520747/693945047\relax
657     \fi
658   \else
659     \ifdim\@tempdima<70pt
660       \advance\@tempdima-49pt
661       \@tempdimb\dimexpr((\@tempdima*-348406699/107952940-55079229sp)%
662         * \@tempdima/866635628+408805sp)*\@tempdima/26926757+\p@)%
663       *135751711/179873976\relax
664     \else
665       \advance\@tempdima-70pt
666       \@tempdimb\dimexpr((\@tempdima*-1015850353/137849442-460519207sp)%
667         * \@tempdima/8742349+142263941sp)*\@tempdima/972432199+23\p@)%
668       *31253604/764969669\relax
669     \fi
670   \fi
671   \global\dimen@i\@tempdimb
672 } \@tempdimb\dimen@i
673 }
```

End of the section for X_y-pic’s “rotate” option. The macro `\xP@trigfromdir` below is also used for the `{-}` directional.

```

674 \xyendinput
675 \</rotate>
676 \<*basic>
```

`\xP@trigfromdir` Calculate sine and cosine from the direction number in #1.

```

677 \newcommand*\xP@trigfromdir[1]{%
678   \Direction#1\relax
\Direction mod 2048
679   \count@-\Direction
680   \advance\count@4096
681   \divide\count@2048
Assign the slope in the right way.
682   \ifcase\count@
683     \d@X\K@\p@
684     \d@Y\numexpr\Direction-3*\K@\relax\p@
685   \or
686     \d@X\numexpr\Direction-\K@\relax\p@
687     \d@Y-\K@\p@
688   \or
689     \d@X-\K@\p@
690     \d@Y\numexpr-\Direction-\K@\relax\p@
691   \or
692     \d@X\numexpr-\Direction-3*\K@\relax\p@
```

```

693 \d@Y\K@p@
694 \else
695 \PackageError{xypdf}{Unexpected case in direction calculation}%
696 {Feel free to contact the author of the xypdf package with a minimal %
697 example.}%
698 \fi

```

Bring the pair ($\d@X$, $\d@Y$) to norm 1.

```

699 \xP@vec{len}
700 \xdef\@gtempa{%
701 \def\noexpand\cosDirection{\xP@EARPT\dimexpr\d@X*\p@/\@tempdimb\relax}%
702 \def\noexpand\sinDirection{\xP@EARPT\dimexpr\d@Y*\p@/\@tempdimb\relax}%
703 }%
704 }\@gtempa
705 }

```

9.5 Temporary registers

`\xP@newdimen` Remove the `\outer` from `\newdimen`. (This applies for plain \TeX .)

```

706 \outer\def\@tempa{\alloc@1\dimen\dimendef\insc@unt}
707 \let\xP@newdimen\newdimen
708 \ifx\newdimen\@tempa
709 \def\xP@newdimen{\alloc@1\dimen\dimendef\insc@unt}
710 \fi
711 \outer\def\@tempa#1{\count@=\escapechar\escapechar=-1
712 \expandafter\expandafter\expandafter
713 \def\@if#1{true}{\let#1=\iftrue}%
714 \expandafter\expandafter\expandafter
715 \def\@if#1{false}{\let#1=\iffalse}%
716 \@if#1{false}\escapechar=\count@}
717 \let\@tempa\relax

```

The next section is for the “curve” extension!

```

718 </basic>
719 <*curve>
720 \xycatcodes

```

`\xypdf-cu@loaded`

```

721 \expandafter\let\csname xypdf-cu@loaded\endcsname\@empty

```

`\xP@tempvar` In order to save registers, `xypdf` shares \Xy-pic ’s and \LaTeX dimension and counter registers but uses different, more descriptive names. Every macro that uses these temporary variables must be safely encapsulated in a group so that the registers are not changed from the outside scope!

The `xypdf` package uses several sets of temporary variable names for different modules. Since it is important that these assignments do not overlap and that the variables are only used encapsulated within groups, the macros which use temporary variables are marked by colored bullets ●1, ●2, ●3, ●4, ●5, ●6, ●7 with one color for each set of variables.

The table in Figure 5 lists all variable assignments in these sets. It can be seen from the table which sets of variables can be used together. For example, set ●1 consisting of `\xP@bigdim` can be used together with all other temporary variables, while ●2 and ●4 must never be used together.

```

722 \newcommand*\xP@tempvar[2]{%
723 \ifdefinable#1\relax
724 \let#1#2%
725 }

```

Xy-pic/L ^A T _E X variable	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7
\quotPTK@	\xP@bigdim						
\L@p	\xP@parA	\xP@A		(\L@p)		(\L@p)	
\U@p	\xP@velA	\xP@B		(\U@p)		(\U@p)	
\R@p	\xP@parB	\xP@C		(\R@p)		(\R@p)	
\D@p	\xP@velB	\xP@D		(\D@p)		(\D@p)	
\X@origin	\xP@parC	\xP@E					\xP@temppar
\Y@origin	\xP@velC	\xP@F					\xP@tempvel
\X@xbase	\xP@parD	\xP@G					\xP@posX
\Y@xbase	\xP@velD	\xP@H					\xP@posY
\X@ybase	\xP@parE	\xP@I = \xP@a	\xP@a				\xP@oldpar
\Y@ybase	\xP@velE	\xP@J = \xP@b	\xP@b				\xP@lastpar
\X@min	\xP@lenA	\xP@K	\xP@c				\xP@tempvel@
\Y@min	\xP@lenB	\xP@L	\xP@valA				\xP@parinc
\X@max	\xP@partlen	\xP@fa	\xP@valB				
\Y@max	\xP@oldpartlen	\xP@fd	\xP@devA				
\almostz@	\xP@tolerance	\xP@tm	\xP@devB				\xP@squiglen
\K@dXdY		\xP@xm	\xP@ti				
\K@dYdX		\xP@ym	\xP@tip				
\splineval@		\xP@off	(\xP@off)				
\splinedepth@		\xP@ta					
\splinetol@		\xP@tb					
\splinelength@		\xP@tc					
\L@		\xP@M					
\normallineskiplimit		\xP@oldobj					
\@savsk			\xP@Tax			\xP@sa	
\lower@bound			\xP@Tay			\xP@sb	
\upper@bound			\xP@Tdx			\xP@sc	
\leftmargin			\xP@Tdy			\xP@Ab	
\rightmargin			\xP@Tmx			\xP@AAb	
\listparindent			\xP@Tmy			\xP@Aba	
\itemindent			\xP@xa (\xP@xa)			\xP@Abb	
\labelwidth			\xP@ya (\xP@ya)			\xP@Abc	
\labelsep			\xP@xb (\xP@xb)			\xP@AAba	
\linewidth			\xP@yb (\xP@yb)			\xP@AAbb	
\@totalleftmargin			\xP@xc (\xP@xc)			\xP@AAbc	
\leftmargini			\xP@yc (\xP@yc)			\xP@dta	
\leftmarginii			\xP@xd (\xP@xd)			\xP@dtb	
\leftmarginiii			\xP@yd (\xP@yd)			\xP@dtc	

Figure 5: Temporary dimension registers in xypdf.

`\xP@oldornewdimen` Either reuse an existing dimension register if the name is defined or allocate a new dimension register. This makes it possible to selectively reuse L^AT_EX dimension registers or allocate new registers in plain T_EX mode.

```

726 \newcommand*\xP@oldornewdimen[2]{%
727   \ifundefined{#2}%
728     {\xP@newdimen#1}%
729     {\expandafter\xP@tempvar\expandafter#1\csname#2\endcsname}%
730 }
```

`\xP@bigdim` •1 A big constant less than $\frac{1}{3}\text{\maxdimen} \approx 5461\text{pt}$ and having many small prime factors.

```
731 \xP@tempvar\xP@bigdim\quotPTK@
```

`\xP@parA` •2 Second set of temporary variables: for the arc length algorithm.

```

\xP@velA 732 \xP@tempvar\xP@parA\L@p
\xP@parB 733 \xP@tempvar\xP@velA\U@p
\xP@velB 734 \xP@tempvar\xP@parB\R@p
\xP@parC 735 \xP@tempvar\xP@velB\D@p
\xP@velC 736 \xP@tempvar\xP@parC\X@origin
\xP@parD 737 \xP@tempvar\xP@velC\Y@origin
\xP@velD 738 \xP@tempvar\xP@parD\X@xbase
\xP@parE 739 \xP@tempvar\xP@velD\Y@xbase
\xP@velE 740 \xP@tempvar\xP@parE\X@ybase
\xP@lenA 741 \xP@tempvar\xP@velE\Y@ybase
\xP@lenB 742 \xP@tempvar\xP@lenA\X@min
          743 \xP@tempvar\xP@lenB\Y@min
\xP@partlen 744 \xP@tempvar\xP@partlen\X@max
\xP@oldpartlen 745 \xP@tempvar\xP@oldpartlen\Y@max
\xP@tolerance 746 \xP@tempvar\xP@tolerance\almostz@
```

`\xP@A` •3 Third set of temporary registers: Bézier offset algorithm and solving linear equations.

```

\xP@B 747 \xP@tempvar\xP@A\L@p
\xP@C 748 \xP@tempvar\xP@B\U@p
\xP@D 749 \xP@tempvar\xP@C\R@p
\xP@E 750 \xP@tempvar\xP@D\D@p
\xP@F 751 \xP@tempvar\xP@E\X@origin
\xP@G 752 \xP@tempvar\xP@F\Y@origin
\xP@H 753 \xP@tempvar\xP@G\X@xbase
\xP@I 754 \xP@tempvar\xP@H\Y@xbase
\xP@J 755 \xP@tempvar\xP@I\X@ybase
\xP@K 756 \xP@tempvar\xP@J\Y@ybase
\xP@L 757 \xP@tempvar\xP@K\X@min
\xP@fa 758 \xP@tempvar\xP@L\Y@min
\xP@fd 759 \xP@tempvar\xP@fa\X@max
\xP@tm 760 \xP@tempvar\xP@fd\Y@max
\xP@xm 761 \xP@tempvar\xP@tm\almostz@
\xP@ym 762 \xP@tempvar\xP@xm\K@dXdY
\xP@off 763 \xP@tempvar\xP@ym\K@dYdX
\xP@ta 764 \xP@tempvar\xP@off\splineval@
\xP@tb 765 \xP@tempvar\xP@ta\splinedepth@
\xP@tc 766 \xP@tempvar\xP@tb\splinetol@
\xP@M 767 \xP@tempvar\xP@tc\splinelength@
       768 \xP@tempvar\xP@M\L@
```

`\xP@oldobj` •3 We need 15 more temporary registers. If possible, we use existing L^AT_EX dimension registers. If Xy-pic is used in plain T_EX mode, there are hopefully still free slots for dimension

```

\xP@Tax
\xP@Tay
\xP@Tdx
\xP@Tdy
\xP@Tmx
\xP@Tmy
\xP@xa
\xP@ya
\xP@xb
\xP@yb
```

registers. We take them for the temporary variables but release them afterwards so that other packages can use them.

```
769 \@tempcnta\count11\relax
770 \xP@oldornewdimen\xP@oldobj{normallineskiplimit}
771 \xP@oldornewdimen\xP@Tax{@savsk}
```

•4

```
772 \xP@oldornewdimen\xP@Tay{lower@bound}
773 \xP@oldornewdimen\xP@Tdx{upper@bound}
774 \xP@oldornewdimen\xP@Tdy{leftmargin}
775 \xP@oldornewdimen\xP@Tmx{rightmargin}
776 \xP@oldornewdimen\xP@Tmy{listparindent}
777 \xP@oldornewdimen\xP@xa{itemindent}
778 \xP@oldornewdimen\xP@ya{labelwidth}
779 \xP@oldornewdimen\xP@xb{labelsep}
780 \xP@oldornewdimen\xP@yb{linewidth}
781 \xP@oldornewdimen\xP@xc{@totalleftmargin}
782 \xP@oldornewdimen\xP@yc{leftmargini}
783 \xP@oldornewdimen\xP@xd{leftmarginii}
784 \xP@oldornewdimen\xP@yd{leftmarginiii}
785 \count11\@tempcnta
```

\xP@a •5 Fifth set of temporary variables: Parameters for drawing part of a spline segment.

```
\xP@b 786 \xP@tempvar\xP@a\X@ybase
\xP@c 787 \xP@tempvar\xP@b\Y@ybase
\xP@valA 788 \xP@tempvar\xP@c\X@min
\xP@valB 789 \xP@tempvar\xP@valA\Y@min
\xP@devA 790 \xP@tempvar\xP@valB\X@max
\xP@devB 791 \xP@tempvar\xP@devA\Y@max
\xP@ti 792 \xP@tempvar\xP@devB\almostz@
\xP@tip 793 \xP@tempvar\xP@ti\K@dXdY
794 \xP@tempvar\xP@tip\K@dYdX
```

\xP@sa •6 Sixth set of temporary variables: Solving a linear system approximately.

```
\xP@sb 795 \xP@tempvar\xP@sa\xP@Tax
\xP@sc 796 \xP@tempvar\xP@sb\xP@Tay
\xP@Ab 797 \xP@tempvar\xP@sc\xP@Tdx
\xP@AAb 798 \xP@tempvar\xP@Ab\xP@Tdy
\xP@Aba 799 \xP@tempvar\xP@AAb\xP@Tmx
\xP@Abb 800 \xP@tempvar\xP@Aba\xP@Tmy
\xP@Abc 801 \xP@tempvar\xP@Abb\xP@xa
802 \xP@tempvar\xP@Abc\xP@ya
\xP@AAba 803 \xP@tempvar\xP@AAba\xP@xb
\xP@AAbb 804 \xP@tempvar\xP@AAbb\xP@yb
\xP@AAbc 805 \xP@tempvar\xP@AAbc\xP@xc
\xP@dta 806 \xP@tempvar\xP@dta\xP@yc
\xP@dtb 807 \xP@tempvar\xP@dtb\xP@xd
\xP@dtc 808 \xP@tempvar\xP@dtc\xP@yd
```

\xP@temppar •7 Seventh set of temporary registers: For multiple dotted splines.

```
\xP@tempvel 809 \xP@tempvar\xP@temppar\X@origin
\xP@posX 810 \xP@tempvar\xP@tempvel\Y@origin
\xP@posY 811 \xP@tempvar\xP@posX\X@xbase
\xP@oldpar 812 \xP@tempvar\xP@posY\Y@xbase
\xP@lastpar 813 \xP@tempvar\xP@oldpar\X@ybase
\xP@tempvel@ 814 \xP@tempvar\xP@lastpar\Y@ybase
\xP@parinc
\xP@squiglen
```

```

815 \xP@tempvar\xP@tempvel@X@min
816 \xP@tempvar\xP@parincY@min
817 \xP@tempvar\xP@squiglen\almostz@

\xP@scaleone We also use temporary numerical registers for scaling factors in \xP@solvelinearsystem.
\xP@scaletwo 818 \xP@tempvar\xP@scaleone\K@
\xP@scalethree 819 \xP@tempvar\xP@scaletwo\KK@
820 \xP@tempvar\xP@scalethree\Direction

```

9.6 Bézier curves

```

\splinesolid@ These are the hooks for single-stroke splines (solid, dashed and dotted).
\splinedashed@ 821 \xP@hook{curve}{splinesolid@}
\splinedotted@ 822 \newcommand*\xP@splinesolid@{\xP@spline\xP@setsolidpat}
823 \xP@hook{curve}{splinedashed@}
824 \newcommand*\xP@splinedashed@{\xP@spline\xP@setdashpat}
825 \xP@hook{curve}{splinedotted@}
826 \newcommand*\xP@splinedotted@{\xP@spline\xP@setdottedpat}

\xP@spline Output a spline segment. Parameter: Macro for the dash pattern generation.
827 \newcommand*\xP@spline[1]{%
828   \readsplineparams@

Neglect splines which are drawn “backwards”. Somehow Xy-pic draws curves forward and
backward, but we need it to be drawn only once.

829   \ifdim\dimen5<\dimen7
830     \xP@preparespline

Neglect splines of length zero.

831     \ifdim\@tempdimb>\z@

Set the dash pattern.

832     #1%

Draw the spline.

833     \xP@stroke{\xP@coorX@pY@p m %
834       \xP@coorL@cU@c\xP@coorR@cD@c\xP@coorX@cY@c c}%

Record the end point for pattern continuation.

835     \xP@savvec
836     \fi
837     \fi
838   }

\xP@preparespline

839 \newcommand*\xP@preparespline{%

If we have a quadratic Bézier segment, convert it to a cubic one.

840   \ifx\splineinfo@\squineinfo@
841     \L@c\dimexpr(\X@p+2\A@)/3\relax
842     \U@c\dimexpr(\Y@p+2\B@)/3\relax
843     \R@c\dimexpr(\X@c+2\A@)/3\relax
844     \D@c\dimexpr(\Y@c+2\B@)/3\relax
845     \fi

Cut the spline according to that start and end parameters in \dimen5 and \dimen7.

846     \xP@shavespline

```

Determine the spline length (for the pattern generation; unnecessary for solid splines).

```
847 \xP@bezierlength
848 }
```

`\xP@inibigdim` •1 Initialize `\xP@bigdim` every time a macro that uses this register is called. See e.g. `\xP@shaveprec`.

```
849 \newcommand*\xP@inibigdim{\xP@bigdim5040pt}
```

`\xP@shavespline` Shave a cubic spline at both ends at the parameter values in `\dimen5` and `\dimen7`. For normal use, the parameters fulfill $0pt \leq \dimen5 < \dimen7 \leq 1pt$.

(Note that `\xP@bigdim` only occurs in the arguments to `\xP@shaveprec`, so this use is safe.)

```
850 \newcommand*\xP@shavespline{%
851 \xP@shaveprec{\dimen5*\xP@bigdim/\p@}{\dimen7*\xP@bigdim/\p@}%
852 }
```

`\xP@shaveprec` •1 Shave a cubic spline at both ends at the parameter values in #1 and #2. For normal use, the parameters fulfill $0pt \leq \#1 < \#2 \leq \xP@bigdim$. The control points for the cubic Bézier curve are $(\X@p, \Y@p)$, $(\L@c, \U@c)$, $(\R@c, \D@c)$, $(\X@c, \Y@c)$. The `Xy-pic` registers `\A@`, `\B@`, `\L@p`, `\U@p`, `\R@p`, `\D@p`, `\X@min` and `\Y@min` are used as temporary registers, but safely encapsulated in a group.

```
853 \newcommand*\xP@shaveprec[2]{%
854 \xP@inibigdim
855 \A@\dimexpr#1\relax
856 \B@\dimexpr#2\relax
```

Shortcut in case the spline is not changed.

```
857 \@tempwattrue
858 \ifdim\A@=\B@\ifdim\B@=\xP@bigdim\@tempwafalse\fi\fi
859 \if@tempwa
860 \L@p\dimexpr\L@c-\X@p\relax
861 \U@p\dimexpr\R@c-\L@p-\L@c\relax
862 \R@p\dimexpr\X@c-3\R@c+3\L@c-\X@p\relax
863 \D@p\dimexpr\U@c-\Y@p\relax
864 \X@min\dimexpr\D@c-\D@p-\U@c\relax
865 \Y@min\dimexpr\Y@c-3\D@c+3\U@c-\Y@p\relax
866 \xdef\@gtempa{%
867 \X@p\the\dimexpr\X@p+(3\L@p+(3\U@p+\R@p*\A@/\xP@bigdim)%
868 *\A@/\xP@bigdim)*\A@/\xP@bigdim\relax
869 \Y@p\the\dimexpr\Y@p+(3\D@p+(3\X@min+\Y@min*\A@/\xP@bigdim)%
870 *\A@/\xP@bigdim)*\A@/\xP@bigdim\relax
871 \L@c\the\dimexpr\X@p+(2\A@+\B@)*\L@p/\xP@bigdim+((\A@+2\B@)%
872 *\U@p/\xP@bigdim+\R@p*\A@/\xP@bigdim*\B@/\xP@bigdim)%
873 *\A@/\xP@bigdim\relax
874 \U@c\the\dimexpr\Y@p+(2\A@+\B@)*\D@p/\xP@bigdim+((\A@+2\B@)%
875 *\X@min/\xP@bigdim+\Y@min*\A@/\xP@bigdim*\B@/\xP@bigdim)%
876 *\A@/\xP@bigdim\relax
877 \R@c\the\dimexpr\X@p+(2\B@+\A@)*\L@p/\xP@bigdim+((\B@+2\A@)%
878 *\U@p/\xP@bigdim+\R@p*\B@/\xP@bigdim*\A@/\xP@bigdim)%
879 *\B@/\xP@bigdim\relax
880 \D@c\the\dimexpr\Y@p+(2\B@+\A@)*\D@p/\xP@bigdim+((\B@+2\A@)%
881 *\X@min/\xP@bigdim+\Y@min*\B@/\xP@bigdim*\A@/\xP@bigdim)%
882 *\B@/\xP@bigdim\relax
883 \X@c\the\dimexpr\X@p+(3\L@p+(3\U@p+\R@p*\B@/\xP@bigdim)%
884 *\B@/\xP@bigdim)*\B@/\xP@bigdim\relax
885 \Y@c\the\dimexpr\Y@p+(3\D@p+(3\X@min+\Y@min*\B@/\xP@bigdim)%
```

```

886      *\B@/\xP@bigdim)*\B@/\xP@bigdim\relax}%
887 \else
888   \global\let\@gtempa\relax
889 \fi
890 }\@gtempa
891 }

```

`\xP@bezierlength` ●1 ●2 Compute the arc length of a cubic Bézier segment.

The following algorithm is used: The velocity for a partial segment is fitted at three points (A-C-E) by a quadratic function, and the arc length is approximated by the integral over this quadratic function.

Each interval is recursively divided in halves (A-B-C, C-D-E) as long as the result for the arc length changes more than the precision parameter `\xP@tolerance`. If the desired precision is reached, the arc length in the small interval is added to the total arc length, and the next interval is considered.

The result goes into `\@tempdimb`.

```

892 \newcommand*\xP@bezierlength{%
893   \xP@inibigdim
894   \@tempdimb\z@
895   \xP@parA\z@
896   \xP@velocity\z@\xP@velA
897   \xP@parC.5\xP@bigdim
898   \xP@velocity\xP@parC\xP@velC
899   \xP@velocity\xP@bigdim\xP@velE

```

Arc length (integral over the quadratic approximation)

```

900   \xP@oldpartlen\dimexpr(\xP@velA+4\xP@velC+\xP@velE)/6\relax

```

Tolerance parameter: It is set to 1/100000 of the approximate arc length, but at least 1sp.

```

901   \xP@tolerance\xP@max{1sp}\dimexpr\xP@oldpartlen/100000\relax}%

```

Initiate the recursive algorithm with the interval [0, 1].

```

902   \xP@arclength\xP@parC\xP@velC\xP@bigdim\xP@velE\xP@oldpartlen

```

Pass the result to outside the group.

```

903   \global\dimen@i\@tempdimb
904   }\@tempdimb\dimen@i
905 }

```

`\xP@velocity` ●1 Compute the velocity at the point #1 on a cubic Bézier curve. Needs: Bézier control points `\X@p,...,\Y@c`. Parameter #2: dimension register for the result. Temporary: `\L@p`, `\U@p`, `\d@X`, `\d@Y`.

```

906 \newcommand*\xP@velocity[2]{%
907   \@tempdima\dimexpr#1\relax
908   \xP@tangent
909   \global\dimen@i\@tempdimb
910   }#2\dimen@i
911 }

```

`\xP@tangent` ●1

```

912 \newcommand*\xP@tangent{%
913   \d@X3\xP@precbeziertan\X@p\L@c\R@c\X@c\@tempdima
914   \d@Y3\xP@precbeziertan\Y@p\U@c\D@c\Y@c\@tempdima
915   \xP@vecclen
916 }

```


`\xP@tangentvec` ●1 Tangent vector on a Bézier curve. Parameter #1: Parameter on the segment. Needs: Bézier parameters `\X@c,...,\Y@c`. Returns: vector in $(\mathrm{d}X, \mathrm{d}Y)$, norm in `\@tempdimb`.

```

917 \newcommand*\xP@tangentvec[1]{%
918   \@tempdima#1\relax
919   \xP@tangent

```

If the velocity is zero at some point, take the second derivative for the tangent vector.

```

920   \ifdim\@tempdimb=\z@
921     \L@p\dimexpr\X@c-\X@p+(\L@c-\R@c)*3\relax
922     \U@p\dimexpr\Y@c-\Y@p+(\U@c-\D@c)*3\relax
923     \d@X\dimexpr\L@p*\@tempdima/\xP@bigdim+(\X@p-2\L@c+\R@c)\relax
924     \d@Y\dimexpr\U@p*\@tempdima/\xP@bigdim+(\Y@p-2\U@c+\D@c)\relax
925     \xP@vecclen

```

Or even the third derivative.

```

926     \ifdim\@tempdimb=\z@
927       \d@X\L@p
928       \d@Y\U@p
929       \xP@vecclen
930     \ifdim\@tempdimb=\z@
931       \xP@warning{xypdf}{Cannot determine a tangent vector to a curve}%
932       \@tempdimb\p@
933     \fi
934   \fi
935 \fi
936 \global\dimen@i\d@X
937 \global\dimen3\d@Y
938 \global\dimen5\@tempdimb
939 }%
940 \d@X\dimen@i
941 \d@Y\dimen3\relax
942 \@tempdimb\dimen5\relax
943 }

```

`\xP@arclength` ●2 The recursive step for the arc length computation.

Needs: `\xP@tolerance`, `\xP@parA`, `\xP@velA`. Parameter: #1 is the middle parameter, #2 the velocity at #1, #3 the third parameter, #4 the velocity at #3, #5 the approximate arc length in the interval from `\xP@parA` to #3.

```

944 \newcommand*\xP@arclength[5]{%
945   \xP@parE#3%
946   \xP@velE#4%
947   \xP@parC#1%
948   \xP@velC#2%
949   \xP@oldpartlen#5%

```

Compute two more pairs (parameter, velocity) at positions $\frac{1}{4}$ and $\frac{3}{4}$ of the interval.

```

950   \xP@parB\dimexpr(\xP@parC+\xP@parA)/2\relax
951   \xP@velocity\xP@parB\xP@velB
952   \xP@parD\dimexpr(\xP@parE+\xP@parC)/2\relax
953   \xP@velocity\xP@parD\xP@velD

```

Compute the approximations for the arc length on the two smaller parameter intervals (A-B-C) and (C-D-E).

```

954   \xP@lenA
955     \dimexpr(\xP@velA+4\xP@velB+\xP@velC)/6*(\xP@parC-\xP@parA)/\xP@bigdim\relax
956   \xP@lenB
957     \dimexpr(\xP@velC+4\xP@velD+\xP@velE)/6*(\xP@parE-\xP@parC)/\xP@bigdim\relax
958   \xP@partlen\dimexpr\xP@lenA+\xP@lenB\relax

```

Check whether the approximation for the arc length has changed more than the precision parameter. The code is a hack to compare the absolute value without occupying another dimension register.

```

959  {\@tempdima\dimexpr\xP@oldpartlen-\xP@partlen\relax
960  \expandafter}\ifdim\ifdim\@tempdima<\z@-\fi\@tempdima>\xP@tolerance

```

Yes? Subdivide the interval. The input queue serves as a LIFO stack here!

```

961  \edef\next@{%
962    \noexpand\xP@arclength\xP@parB\xP@velB\xP@parC\xP@velC\xP@lenA
963    \noexpand\xP@arclength{\the\xP@parD}{\the\xP@velD}{\the\xP@parE}%
964    {\the\xP@velE}{\the\xP@lenB}%
965  }%
966  \else

```

No? Proceed to the next parameter interval.

```

967  \xP@parA\xP@parE
968  \xP@velA\xP@velE
969  \advance\@tempdimb\xP@partlen
970  \DN@{}%
971  \fi
972  \next@
973 }

```

9.7 New improved curve styles

\@crv@ Extend the list of curve styles for which special routines exist. New curve styles: 3{.}, {~}, \xP@crv@ 2{~}, 3{~}, {~~}, 2{~~}, 3{~~}

```

974 \xP@hook{curve}{\@crv@}
975 \newcommand*\xP@crv@[2]{\DN@{#1#2}%
976   \ifx\next@\@empty \edef\next@\{crv@defaultshape}%
977   \ifx\bstartPLACE@\@empty\xdef\crvSTYLE@{\crv@defaultshape}}\fi
978   \else
979   \ifx\bstartPLACE@\@empty\gdef\crvSTYLE@{#1#2}}\fi
980   \fi
981   \ifx\next@\@empty\crv@noobject \DN@{\crv@}{\xy@crvaddstack@}%
982   \else\def\tmp@{-}\ifx\next@\tmp@ \DN@{\crv@}{\xy@crvaddstack@}%
983   \else\def\tmp@{=}\ifx\next@\tmp@
984     \DN@{\expandafter\crv@\crv@normaltemplate{\dir{=}}}%
985   \else\def\tmp@{2-}\ifx\next@\tmp@
986     \DN@{\expandafter\crv@\crv@normaltemplate{\dir{2.}}}%
987   \else\def\tmp@{3-}\ifx\next@\tmp@
988     \DN@{\expandafter\crv@\crv@normaltemplate{\dir{3.}}}%
989   \else\def\tmp@{--}\ifx\next@\tmp@
990     \DN@{\expandafter\crv@\crv@specialtemplate{--}}%
991   \else\def\tmp@{==}\ifx\next@\tmp@
992     \DN@{\expandafter\crv@\crv@normaltemplate{\dir2{--}}}%
993   \else\def\tmp@{2--}\ifx\next@\tmp@
994     \DN@{\expandafter\crv@\crv@normaltemplate{\dir2{--}}}%
995   \else\def\tmp@{3--}\ifx\next@\tmp@
996     \DN@{\expandafter\crv@\crv@normaltemplate{\dir3{--}}}%
997   \else\def\tmp@{.}\ifx\next@\tmp@
998     \DN@{\expandafter\crv@\crv@specialtemplate{.}}%
999   \else\def\tmp@{:}\ifx\next@\tmp@
1000     \DN@{\expandafter\crv@\crv@normaltemplate{\dir{:}}}%
1001   \else\def\tmp@{2.}\ifx\next@\tmp@
1002     \DN@{\expandafter\crv@\crv@normaltemplate{\dir{:}}}%

```

```

\yxsplinespecialcases@ New: \dir3{.}, \dir2{--}, \dir3{--}, \dir{~}, \dir2{~}, \dir3{~}, \dir{~~},
\xP@yxsplinespecialcases@ \dir2{~~}, \dir3{~~}

```

The next line does not occur in Xy-pic for an unknown reason. However, it seems reasonable to define the special pattern `\dir2{.}` in the same way as for straight lines.

35

9.8 Multiple solid curves

```

\XP@splinedoubled@
1053 \XP@hook{curve}{splinedoubled@}
1054 \newcommand*\XP@splinedoubled@{%
1055   \XP@checkspline\XP@splinemultsolid\XP@doublestroke}

\XP@splineribboned@
1056 \XP@hook{curve}{splineribboned@}
1057 \@ifdefinable\XP@splineribboned@\relax
1058 \let\XP@splineribboned@\XP@splinedoubled@

\XP@splinetrebled@
1059 \XP@hook{curve}{splinetrebled@}
1060 \newcommand*\XP@splinetrebled@{%
1061   \XP@checkspline\XP@splinemultsolid\XP@trblstroke}

\XP@doublestroke Offset parameters for double lines and curves
1062 \newcommand*\XP@doublestroke{\xydashh@/2,-\xydashh@/2}

\XP@trblstroke Offset parameters for treble lines and curves
1063 \newcommand*\XP@trblstroke{\xydashh@,\z@,-\xydashh@}

\XP@checkspline Get and check spline parameters before the macro in #1 is executed.
1064 \newcommand*\XP@checkspline[1]{%
1065   \readsplineparams@

Neglect splines which are drawn “backwards”. Somehow Xy-pic draws curves forward and
backward, but we need it to be drawn only once.
1066   \let\next@\@gobble
1067   \ifdim\dimen5<\dimen7
1068     \XP@preparespline
Neglect splines of zero length.
1069     \ifdim\@tempdimb>\z@
If the path length is less than twice the line width, just draw a solid path.
1070       \ifdim\@tempdimb<2\dimexpr\XP@preclw\relax
1071         \let\next@\XP@splinemultsolid
1072       \else
1073         \let\next@#1%
1074       \fi
1075     \fi
1076   \fi
1077   \next@
1078 }

\XP@splinemultsolid ●1
1079 \newcommand*\XP@splinemultsolid[1]{%
1080   \XP@inibigdim
1081   \@temptokena{%
1082     \XP@setsolidpat
The \@for loop does the multiple strokes. \@tempa records the respective offset distance.
1083   \@for\@tempa:=#1\do{\XP@paintsolid\z@\XP@bigdim}%
1084   \XP@stroke{\the\@temptokena}%
1085 }}

```

`\xP@paintsolid` ●1 ●5 Draw a solid spline in the parameter interval $[\#1, \#2] \subseteq [0pt, \xP@bigdim]$ with a certain offset. The offset distance is expected in `\@tempa`.

```

1086 \newcommand*\xP@paintsolid[2]{\%
Record the original anchor points.
1087 \xP@savepts
1088 \xP@a#1\relax
1089 \xP@c#2\relax
1090 \xP@movetotrue
1091 \xP@paintsolid@
1092 \xdef\@gtempa{\the\@temptokena}%
1093 }%
1094 \@temptokena\expandafter{\@gtempa}%
1095 }

```

`\xP@paintsolid@` ●1 ●5

```

1096 \newcommand*\xP@paintsolid@{\%
These parameters record which part of the spline is currently being offset. They are varied
as the spline may be subdivided for a precise offset curve.
1097 \xP@b\xP@c
Offset distance
1098 \xP@off\dimexpr\@tempa\relax
1099 \ifdim\xP@off=\z@
1100 \xP@shaveprec\xP@a\xP@c
1101 \else
1102 \loop
Restore the original anchor points.
1103 \xP@restorepts
Compute the approximate offset curve. Note that \xP@a and \xP@b contain the boundary
parameters for the partial spline.
1104 \xP@offsetsegment
Test if the offset curve is good enough.
1105 \xP@testoffset
If not, shorten the parameter interval by 30%.
1106 \unless\ifxP@offsetok
1107 \xP@b\dimexpr\xP@a+(\xP@b-\xP@a)*7/10\relax
1108 \repeat
1109 \fi
Append the new segment to the path.
1110 \xP@append\@temptokena{\ifxP@moveto\xP@coord\xP@p\Y@p m \fi
1111 \xP@coord\l@c\U@c\xP@coord\R@c\D@c\xP@coord\X@c\Y@c c }%
1112 \xP@movetofalse
Test if the end of the spline has been reached. If not, offset the rest of the curve.
1113 \ifdim\xP@b<\xP@c\relax
1114 \xP@a\xP@b
1115 \expandafter\xP@paintsolid@
1116 \fi
1117 }

```

\ifxP@moveto We need a PDF moveto operator only for the first partial segment. Additional segments connect seamlessly.

```
1118 \@ifdefinable\ifxP@moveto\relax
1119 \@ifdefinable\xP@movetotrue\relax
1120 \@ifdefinable\xP@movetofalse\relax
1121 \newif\ifxP@moveto
```

\xP@savepts •5 Save the anchor points to the second set of reserved variables.

```
1122 \newcommand*\xP@savepts{%
1123   \xP@xa\X@p
1124   \xP@ya\Y@p
1125   \xP@xb\L@c
1126   \xP@yb\U@c
1127   \xP@xc\R@c
1128   \xP@yc\D@c
1129   \xP@xd\X@c
1130   \xP@yd\Y@c
1131 }
```

\xP@restorepts •5 Restore the anchor points from the second set of reserved variables.

```
1132 \newcommand*\xP@restorepts{%
1133   \X@p\xP@xa
1134   \Y@p\xP@ya
1135   \L@c\xP@xb
1136   \U@c\xP@yb
1137   \R@c\xP@xc
1138   \D@c\xP@yc
1139   \X@c\xP@xd
1140   \Y@c\xP@yd
1141 }
```

9.9 A Bézier curve offset algorithm

First, all control points are offset by the desired distance and in the direction of the normal vectors at the boundary points of the curve. We then adjust the distance of the inner two control points to the boundary control points along the tangents at the boundary points: $x_b = x_a + f_a T_{ax}$, $x_c = x_d + f_d T_{dx}$, and likewise for the y -coordinates. In nondegenerate cases, we have $T_{ax} = x_b - x_a$ and $T_{dx} = x_c - x_d$.

Let $P(a, b, c, d, t)$ denote the Bézier polynomial $a(1-t)^3 + 3bt(1-t)^2 + 3ct^2(1-t) + dt^3$. In order to determine the factors f_a and f_d , we set up a system of three equations.

- Two equations: The old point at parameter $\frac{1}{2}$ plus offset, (x_m, y_m) , is the new point at parameter t_m .

$$\begin{aligned} x_m &= P(x_a, x_a + f_a T_{ax}, x_d + f_d T_{dx}, x_d, t_m) \\ y_m &= P(y_a, y_a + f_a T_{ay}, y_d + f_d T_{dy}, y_d, t_m) \end{aligned}$$

- Third equation: The old tangent at parameter $\frac{1}{2}$ is in the same direction as the new tangent at t_m .

$$\begin{aligned} &\frac{\partial}{\partial t_m} P(x_a, x_a + f_a T_{ax}, x_d + f_d T_{dx}, x_d, t_m) \cdot T_{my} \\ &= \frac{\partial}{\partial t_m} P(y_a, y_a + f_a T_{ay}, y_d + f_d T_{dy}, y_d, t_m) \cdot T_{mx} \end{aligned}$$

Up to a scalar factor of $-3/4$, (T_{mx}, T_{my}) is the velocity vector to the original curve at parameter $\frac{1}{2}$. We have $T_{mx} = (X_a + X_b - X_c - X_d)/2$ (in the old coordinates!) and T_{my} analogously. The system above is a nonlinear system of three equations in three variables, which we solve by Newton's method. Let f_a , f_d , and t_m be approximate solutions, and denote by Δf_a , Δf_d , and Δt_m the increments to the next approximation. In the first order, the three equations become:

$$\begin{aligned}
x_m &= P(x_a, x_b, x_c, x_d, t_m) + \Delta f_a \cdot T_{ax} \cdot 3t_m(1 - t_m)^2 + \Delta f_d \cdot T_{dx} \cdot 3t_m^2(1 - t_m) \\
&\quad + \Delta t_m \frac{\partial}{\partial t_m} P(x_a, x_b, x_c, x_d, t_m) \\
y_m &= P(y_a, y_b, y_c, y_d, t_m) + \Delta f_a \cdot T_{ay} \cdot 3t_m(1 - t_m)^2 + \Delta f_d \cdot T_{dy} \cdot 3t_m^2(1 - t_m) \\
&\quad + \Delta t_m \frac{\partial}{\partial t_m} P(y_a, y_b, y_c, y_d, t_m) \\
&\left(\frac{\partial}{\partial t_m} P(x_a, x_b, x_c, x_d, t_m) + \Delta f_a \cdot T_{ax} \cdot 3(1 - 4t_m + 3t_m^2) + \Delta f_d \cdot T_{dx} \cdot 3(2t_m - 3t_m^2) \right. \\
&\quad \left. + \Delta t_m \cdot 6((x_a - 2x_b + x_c) + t_m(x_d - x_a + 3(x_b - x_c))) \right) \cdot T_{my} \\
&= \left(\frac{\partial}{\partial t_m} P(y_a, y_b, y_c, y_d, t_m) + \Delta f_a \cdot T_{ay} \cdot 3(1 - 4t_m + 3t_m^2) + \Delta f_d \cdot T_{dy} \cdot 3(2t_m - 3t_m^2) \right. \\
&\quad \left. + \Delta t_m \cdot 6((y_a - 2y_b + y_c) + t_m(y_d - y_a + 3(y_b - y_c))) \right) \cdot T_{mx}
\end{aligned}$$

Rewrite the equations so that they resemble the \TeX code.

$$\begin{aligned}
8P(x_a, x_b, x_c, x_d, t_m) - 8x_m &= -\Delta f_a \cdot 3T_{ax} \cdot 2t_m \cdot (2(1 - t_m))^2 \\
&\quad - \Delta f_d \cdot 3T_{dx} \cdot 4t_m^2 \cdot 2(1 - t_m) - \Delta t_m \cdot 8 \frac{\partial}{\partial t_m} P(x_a, x_b, x_c, x_d, t_m) \\
8P(y_a, y_b, y_c, y_d, t_m) - 8y_m &= -\Delta f_a \cdot 3T_{ay} \cdot 2t_m \cdot (2(1 - t_m))^2 \\
&\quad - \Delta f_d \cdot 3T_{dy} \cdot 4t_m^2 \cdot 2(1 - t_m) - \Delta t_m \cdot 8 \frac{\partial}{\partial t_m} P(y_a, y_b, y_c, y_d, t_m) \\
T_{mx} \cdot 8 \frac{\partial}{\partial t_m} P(y_a, y_b, y_c, y_d, t_m) - T_{my} \cdot 8 \frac{\partial}{\partial t_m} P(x_a, x_b, x_c, x_d, t_m) \\
&= -\Delta f_a \cdot (3T_{ay} \cdot 2T_{mx} - 3T_{ax} \cdot 2T_{my}) \cdot 2(1 - 3t_m) \cdot 2(1 - t_m) \\
&\quad - \Delta f_d \cdot (3T_{dy} \cdot 2T_{mx} - 3T_{dx} \cdot 2T_{my}) \cdot 2(2 - 3t_m) \cdot 2t_m \\
&\quad - \Delta t_m \cdot (((y_d - y_a + 3(y_b - y_c)) \cdot 2t_m + 2(y_a - 2y_b + y_c)) \cdot 3 \cdot 8T_{mx} \\
&\quad - ((x_d - x_a + 3(x_b - x_c)) \cdot 2t_m + 2(x_a - 2x_b + x_c)) \cdot 3 \cdot 8T_{my})
\end{aligned}$$

Substitute $2t_m = \tau_m$.

$$\begin{aligned}
8P(x_a, x_b, x_c, x_d, t_m) - 8x_m &= -\Delta f_a \cdot 3T_{ax} \cdot \tau_m(2 - \tau_m)^2 \\
&\quad - \Delta f_d \cdot 3T_{dx} \cdot \tau_m^2(2 - \tau_m) - \frac{1}{2} \Delta \tau_m \cdot 8 \frac{\partial}{\partial t_m} P(x_a, x_b, x_c, x_d, t_m) \\
8P(y_a, y_b, y_c, y_d, t_m) - 8y_m &= -\Delta f_a \cdot 3T_{ay} \cdot \tau_m \cdot (2 - \tau_m)^2 \\
&\quad - \Delta f_d \cdot 3T_{dy} \cdot \tau_m^2(2 - \tau_m) - \frac{1}{2} \Delta \tau_m \cdot 8 \frac{\partial}{\partial t_m} P(y_a, y_b, y_c, y_d, t_m) \\
T_{mx} \cdot 8 \frac{\partial}{\partial t_m} P(y_a, y_b, y_c, y_d, t_m) - T_{my} \cdot 8 \frac{\partial}{\partial t_m} P(x_a, x_b, x_c, x_d, t_m) \\
&= -\Delta f_a \cdot (3T_{ay} \cdot 2T_{mx} - 3T_{ax} \cdot 2T_{my}) \cdot (2 - 3\tau_m)(2 - \tau_m) \\
&\quad - \Delta f_d \cdot (3T_{dy} \cdot 2T_{mx} - 3T_{dx} \cdot 2T_{my}) \cdot (4 - 3\tau_m)\tau_m \\
&\quad - \Delta \tau_m \cdot (((y_d - y_a + 3(y_b - y_c)) \cdot \tau_m + 2(y_a - 2y_b + y_c)) \cdot 12T_{mx} \\
&\quad - ((x_d - x_a + 3(x_b - x_c)) \cdot \tau_m + 2(x_a - 2x_b + x_c)) \cdot 12T_{my})
\end{aligned}$$

The translation into \TeX dimensions:

- $f_a = \text{\xP@fa}$, $f_d = \text{\xP@fd}$
- $\tau_m = \text{\xP@tm}$
- $x_a = \text{\xP@xa}, \dots, x_d = \text{\xP@xd}, \dots, y_d = \text{\xP@yd}$
- $8P(x_1, x_2, x_3, x_4, \frac{1}{2}x_5) = \text{\xP@bezierpoly\#1\#2\#3\#4\#5}$
- $8x_m = \text{\xP@xm}$, $8y_m = \text{\xP@ym}$
- $3T_{ax} = \text{\xP@Tax}$, $3T_{dx} = \text{\xP@Tdx}$, $3T_{ay} = \text{\xP@Tay}$, $3T_{dy} = \text{\xP@Tdy}$
- $8\frac{\partial}{\partial x_5}P(x_1, x_2, x_3, x_4, \frac{1}{2}x_5) = \text{\xP@beziertan\#1\#2\#3\#4\#5}$

Temporary:

- $2 - \tau_m = \text{\xP@ta}$
- $\tau_m(2 - \tau_m) = \text{\xP@tb}$
- $T_{mx} = \text{\xP@Tmx}$, $T_{my} = \text{\xP@Tmy}$
- $2 - 3\tau_m = \text{\xP@tb}$
- $4 - 3\tau_m = \text{\xP@tc}$

Since the linear system above tends to be singular or ill-conditioned (think about the frequent case when all control points are nearly collinear!), the Gauss algorithm `\xP@solvelinearsystem` does not always return a valid solution. In these cases, the system is not solved exactly but approximated iteratively in `\xP@applinsys`.

```
\xP@tmx
\xP@tmy 1142 \ifdefinable\xP@tmx\relax
1143 \ifdefinable\xP@tmy\relax
```

```
\xP@Tmxy ●4
\xP@Tmyx 1144 \newcommand*\xP@Tmxy{* \xP@Tmx / \xP@Tmy}
1145 \newcommand*\xP@Tmyx{* \xP@Tmy / \xP@Tmx}
```

```
\xP@Tmzero
1146 \newcommand*\xP@Tmzero{* \z@}
```

`\xP@offsetsegment` ●1 ●3 ●4 Offset a cubic segment. The offset distance is given in `\xP@off`. The anchor points are given in `\X@p, \Y@c`. The partial spline in the parameter interval $[\text{\xP@a}, \text{\xP@b}] \subseteq [0pt, \text{\xP@bigdim}]$ is offset. The new Bézier curve is returned in `\xP@xa, \dots, \xP@yd`.

```
1147 \newcommand*\xP@offsetsegment{%
New first anchor point and tangent vector at 0
1148 \xP@tangentvec\xP@a
1149 \xP@xa\dimexpr\xP@precbezierpoly\X@p\L@c\R@c\X@c\xP@a/8%
1150 +\d@Y*\xP@off/\@tempdimb\relax
1151 \xP@ya\dimexpr\xP@precbezierpoly\Y@p\U@c\D@c\Y@c\xP@a/8%
1152 -\d@X*\xP@off/\@tempdimb\relax
1153 \xP@scaleT
1154 \xP@Tax\d@X
1155 \xP@Tay\d@Y
1156 \xP@E\@tempdimb
```


New last anchor point and tangent vector at 1

```

1157 \xP@tangentvec\xP@b
1158 \xP@d\dimexpr\xP@precbezierpoly\X@p\L@c\R@c\X@c\xP@b/8%
1159 +\d@Y*\xP@off/\@tempdimb\relax
1160 \xP@yd\dimexpr\xP@precbezierpoly\Y@p\U@c\D@c\Y@c\xP@b/8%
1161 -\d@X*\xP@off/\@tempdimb\relax
1162 \xP@scaleT
1163 \xP@Tdx-\d@X
1164 \xP@Tdy-\d@Y
1165 \xP@F\@tempdimb

```

Scalar product of the tangent vectors

```

1166 \xP@M\z@
1167 \xP@Max\xP@M\xP@Tdx
1168 \xP@Max\xP@M\xP@Tdy
1169 \xP@L\dimexpr\xP@Tax*\xP@Tdx/\xP@M+\xP@Tay*\xP@Tdy/\xP@M\relax
1170 \xP@tm\dimexpr(\xP@a+\xP@b)/2\relax
1171 \ifdim\xP@L>\dimexpr\xP@E*\xP@F/\xP@M*49/50\relax

```

Trick to improve the offset algorithm near sharp bends and cusps: If the tangent vectors (T_{ax}, T_{ay}) and (T_{dx}, T_{dy}) point nearly in the same direction, we do not use the true tangent vector for (T_{mx}, T_{my}) at the middle point but a fake one. (The exact condition is that their normed scalar product is greater than $49/50$. For a straight line, the vectors would point in opposite directions.) The fake tangent vector is defined to be $(T_{ax} + T_{dx}, T_{ay} + T_{dy})$ rotated by $\pm 90^\circ$. Its direction is chosen such that the scalar product with $(X_d - X_a, Y_d - Y_a)$ is nonnegative. (Use $(X_c - X_b, Y_c - Y_b)$ in the degenerate case $(X_d - X_a, Y_d - Y_a) = (0, 0)$.)

Rationale: In the presence of a sharp bend or cusp, the offset algorithm will hardly meet the tip. Since the tangent/normal at the tip is needed for a good offset curve, we provide this artificially.

```

1172 \d@X-\dimexpr\xP@Tay+\xP@Tdy\relax
1173 \d@Y\dimexpr\xP@Tax+\xP@Tdx\relax
1174 \xP@vecLen
1175 \xP@A\dimexpr\X@c-\X@p\relax
1176 \xP@B\dimexpr\Y@c-\Y@p\relax
1177 \xP@M\z@
1178 \xP@Max\xP@M\xP@A
1179 \xP@Max\xP@M\xP@B
1180 \ifdim\xP@M=\z@
1181 \xP@A\dimexpr\R@c-\L@c\relax
1182 \xP@B\dimexpr\D@c-\U@c\relax
1183 \xP@Max\xP@M\xP@A
1184 \xP@Max\xP@M\xP@B
1185 \fi
1186 \xP@M\dimexpr\d@X*\xP@A/\xP@M+\d@Y*\xP@B/\xP@M\relax
1187 \ifdim\xP@M<\z@
1188 \multiply\d@X\m@ne
1189 \multiply\d@Y\m@ne
1190 \fi
1191 \else

```

Normal case: tangent vector at the middle point.

```

1192 \xP@tangentvec\xP@tm
1193 \fi

```

From here on, $\xP@a$ and $\xP@b$ will not be used any more, so these variables can be used under their other names $\xP@I$, $\xP@J$ for the linear systems below.

8 times (middle point plus offset)

```

1194 \xP@xm\dimexpr\xP@precbezierpoly\X@p\L@c\R@c\X@c\xP@tm%
1195 +8\d@Y*\xP@off/\@tempdimb\relax
1196 \xP@ym\dimexpr\xP@precbezierpoly\Y@p\U@c\D@c\Y@c\xP@tm%
1197 -8\d@X*\xP@off/\@tempdimb\relax

```

Tangent at middle point

```

1198 \xP@Tmx\d@X
1199 \xP@Tmy\d@Y
1200 \xP@ifabsless\xP@Tmy\xP@Tmx
1201 \let\xP@tmy\xP@Tmyx
1202 \let\xP@tmx\@empty
1203 \else
1204 \ifdim\xP@Tmy=\z@
1205 \let\xP@tmx\xP@Tmzero
1206 \let\xP@tmy\xP@Tmzero
1207 \else
1208 \let\xP@tmy\@empty
1209 \let\xP@tmx\xP@Tmxy
1210 \fi
1211 \fi

```

Initial guesses for the tangent vector scalings \xP@fa, \xP@fd and the near-middle position \xP@tm

```

1212 \xP@fa\p@
1213 \xP@fd\p@
1214 \xP@tm\p@

```

The main loop for finding the offset curve

```

1215 \count@\z@
1216 \loop

```

Set the new control points up.

```

1217 \xP@offsetpoints
1218 \@tempswafalse

```

At most 10 iterations

```

1219 \ifnum10>\count@

```

Determine the quality of the approximation by an objective function.

```

1220 \xP@objfun\xP@oldobj
1221 \ifdim\xP@oldobj>\xP@maxobjfun\relax\@tempwatruer\fi
1222 \fi
1223 \if@tempswa
1224 \xP@offsetloop
1225 \repeat

```

Return the new anchor points.

```

1226 \xdef\@gtempa{\X@p\the\xP@xa\Y@p\the\xP@ya
1227 \L@c\the\xP@xb\U@c\the\xP@yb\R@c\the\xP@xc\D@c\the\xP@yc
1228 \X@c\the\xP@xd\Y@c\the\xP@yd\relax}%
1229 }%
1230 \@gtempa
1231 }

```

\xP@scaleT ●134 This macro contains another trick to improve the offset algorithm around sharp bends and cusps. It adjusts the length of the tangent/velocity vectors. Let $(\d@X, \d@Y)$ be the velocity vector to the original curve at some point with velocity v_0 . The velocity at the same point, considered on a partial segment scales linearly with the length of the parameter interval. Hence, the velocity v_1 in the partial segment is $v_1 = v_0 \cdot (\xP@b -$

$\backslash\text{P@a})/\backslash\text{P@bigdim}$. Additionally the offset curve goes with a radius of $r + \backslash\text{P@off}$ around bends with radius r in the original curve. As an approximation to the velocity in the offset curve, we therefore scale the velocity vector in the end to the norm $v_1 + 2\pi \cdot |\backslash\text{P@off}|$.

```
1232 \newcommand*\xP@scaleT{%
1233   \xP@B6.28\xP@off
1234   \xP@abs\xP@B
1235   \xP@C\dimexpr\d@X*\xP@B/\@tempdimb\relax
1236   \xP@D\dimexpr\d@Y*\xP@B/\@tempdimb\relax
1237   \xP@A\dimexpr\xP@b-\xP@a\relax
1238   \d@X\dimexpr\xP@C+\d@X*\xP@a/\xP@bigdim\relax
1239   \d@Y\dimexpr\xP@D+\d@Y*\xP@a/\xP@bigdim\relax
```

Also record the change to the norm of the vector.

```
1240   \@tempdimb\dimexpr\xP@B+\@tempdimb*\xP@a/\xP@bigdim\relax
1241 }
```

$\backslash\text{P@offsetloop}$ ●1 ●3 ●4 The iteration in the offset loop: set up and solve (or approximate) the linear system.

```
1242 \newcommand*\xP@offsetloop{%
1243   \xP@C\dimexpr\xP@C/2\relax
1244   \xP@G\dimexpr\xP@G/2\relax
1245   \xP@ta\dimexpr2\p@-\xP@tm\relax
1246   \xP@tb\dimexpr\xP@tm*\xP@ta/\p@\relax
1247   \xP@A\dimexpr\xP@Tax*\xP@tb/\p@*\xP@ta/\p@\relax
1248   \xP@B\dimexpr\xP@Tdx*\xP@tb/\p@*\xP@tm/\p@\relax
```

1st linear equation

```
1249   \xP@E\dimexpr\xP@Tay*\xP@tb/\p@*\xP@ta/\p@\relax
1250   \xP@F\dimexpr\xP@Tdy*\xP@tb/\p@*\xP@tm/\p@\relax
```

2nd linear equation

```
1251   \xP@tb\dimexpr2\p@-3\xP@tm\relax
1252   \xP@tc\dimexpr\xP@tb+2\p@\relax
1253   \xP@I\dimexpr(2\xP@Tay\xP@tmx-2\xP@Tax\xP@tmy)*\xP@tb/\p@*\xP@ta/\p@\relax
1254   \xP@J\dimexpr(2\xP@Tdy\xP@tmx-2\xP@Tdx\xP@tmy)*\xP@tc/\p@*\xP@tm/\p@\relax
1255   \xP@K\dimexpr((\xP@yd-\xP@ya+(\xP@yb-\xP@yc)*3)
1256     *\xP@tm/\p@+(\xP@yc-2\xP@yb+\xP@ya)*2)*12\xP@tmx
1257     -((\xP@xd-\xP@xa+(\xP@xb-\xP@xc)*3)
1258     *\xP@tm/\p@+(\xP@xc-2\xP@xb+\xP@xa)*2)*12\xP@tmy\relax
```

Solve the system.

```
1259   \xP@solvelinearsystem
1260   \ifxP@validsol
```

Check whether the result is feasible and whether it actually improves the approximation.

```
1261   \xP@correctsol
1262   \ifdim\xP@ta=\z@
1263   \ifdim\xP@tb=\z@
1264   \ifdim\xP@tc=\z@
1265     \xP@validsolfalse
1266   \fi\fi\fi
1267   \fi
```

If the exact solution is not valid, try to at least approximate a solution.

```
1268   \ifxP@validsol
1269   \else
1270     \xP@applinsys
```

This time, the solution is not checked but applied immediately.

```
1271 \advance\xP@fa-\xP@ta
1272 \advance\xP@fd-\xP@tb
1273 \advance\xP@tm-\xP@tc
```

The near-middle parameter on the curve must not lie outside the segment.

```
1274 \ifdim\xP@tm<\z@\xP@tm\z@\fi
1275 \ifdim\xP@tm>2\p@\xP@tm2\p@\fi
1276 \fi
1277 \advance\count@\@ne
1278 }
```

\xP@maxsol Heuristic: maximal solution so that no arithmetic overflow is produced.

```
1279 \newcommand*\xP@maxsol{3pt}
```

\xP@correctsol ●3 ●4 Check whether the solution is feasible and actually improves the objective function.

```
1280 \newcommand*\xP@correctsol{%
```

If the solution is too big, scale all variables uniformly.

```
1281 \xP@M\z@
1282 \xP@Max\xP@M\xP@ta
1283 \xP@Max\xP@M\xP@tb
1284 \xP@Max\xP@M\xP@tc
1285 \ifdim\xP@M>\xP@maxsol
1286 \xP@ta\dimexpr\xP@maxsol*\xP@ta/\xP@M\relax
1287 \xP@tb\dimexpr\xP@maxsol*\xP@tb/\xP@M\relax
1288 \xP@tc\dimexpr\xP@maxsol*\xP@tc/\xP@M\relax
1289 \fi
```

Apply the solution. Save the old value of \xP@tm to be able to restore it.

```
1290 \advance\xP@fa-\xP@ta
1291 \advance\xP@fd-\xP@tb
1292 \xP@M\xP@tm
1293 \advance\xP@tm-\xP@tc
```

The near-middle parameter must lie on the segment.

```
1294 \ifdim\xP@tm<\z@\xP@tm\z@\fi
1295 \ifdim\xP@tm>2\p@\xP@tm2\p@\fi
```

Check whether the solution actually improves the objective function.

```
1296 {\xP@offsetpoints
1297 \xP@objfun\xP@M
1298 \expandafter}%
```

If not, restore the old values and declare the solution invalid.

```
1299 \ifdim\xP@M>\xP@oldobj
1300 \advance\xP@fa\xP@ta
1301 \advance\xP@fd\xP@tb
1302 \xP@tm\xP@M
1303 \xP@validsolfalse
1304 \fi
1305 }
```

\xP@objfun ●3 ●4 The objective function: sum of squares of the deviation in x - and y -direction and the angular deviation at the middle point. We also compute some terms which will be used in the linear system.

```
1306 \newcommand*\xP@objfun[1]{%
1307 \xP@D\dimexpr\xP@bezierpoly\xP@xa\xP@xb\xP@xc\xP@xd\xP@tm-\xP@xm\relax
```

```

1308 \xP@H\dimexpr\xP@bezierpoly\xP@ya\xP@yb\xP@yc\xP@yd\xP@tm-\xP@ym\relax
1309 \xP@C\xP@beziertan\xP@xa\xP@xb\xP@xc\xP@xd\xP@tm
1310 \xP@G\xP@beziertan\xP@ya\xP@yb\xP@yc\xP@yd\xP@tm
1311 \xP@L\dimexpr\xP@G\xP@tmx-\xP@C\xP@tmy\relax

```

If the deviation is too big, let the objective function be `\maxdimen`. Otherwise, compute the sum of squares.

```

1312 #1\z@
1313 \xP@Max#1\xP@D
1314 \xP@Max#1\xP@H
1315 \xP@Max#1\xP@L
1316 #1\ifdim#1>4843165sp
1317 \maxdimen
1318 \else
1319 \dimexpr\xP@D*\xP@D/\p@+\xP@H*\xP@H/\p@+\xP@L*\xP@L/\p@\relax
1320 \fi
1321 }

```

`\xP@offsetpoints` ●3 ●4 Compute the new control points from the factors `\xP@fa`, `\xP@fd`.

```

1322 \newcommand*\xP@offsetpoints{%
1323 \xP@xb\dimexpr\xP@xa+\xP@Tax*\xP@fa/196608\relax
1324 \xP@yb\dimexpr\xP@ya+\xP@Tay*\xP@fa/196608\relax
1325 \xP@xc\dimexpr\xP@xd+\xP@Tdx*\xP@fd/196608\relax
1326 \xP@yc\dimexpr\xP@yd+\xP@Tdy*\xP@fd/196608\relax
1327 }

```

`\xP@bezierpoly` Formula for the polynomial $8(\#1 \cdot (1-t)^3 + 3 \cdot \#2 \cdot t(1-t)^2 + 3 \cdot \#3 \cdot t^2(1-t) + \#4 \cdot t^3)$, $t = \frac{1}{2}\#5$.

```

1328 \newcommand*\xP@bezierpoly[5]{%
1329 \dimexpr((\#4-\#1+(\#2-\#3)*3)*\#5/\p@+(\#1-2\#2+\#3)*6)*\#5/\p@+(\#2-\#1)*12)*\#5/\p@
1330 +\#1*8\relax
1331 }

```

`\xP@precbezierpoly` Formula for the polynomial $8(\#1 \cdot (1-t)^3 + 3 \cdot \#2 \cdot t(1-t)^2 + 3 \cdot \#3 \cdot t^2(1-t) + \#4 \cdot t^3)$, $t = \#5/\xP@bigdim$.

```

1332 \newcommand*\xP@precbezierpoly[5]{%
1333 \dimexpr((\#4-\#1+(\#2-\#3)*3)*2*\#5/\xP@bigdim+(\#1-2\#2+\#3)*6)*2*\#5/\xP@bigdim
1334 +(\#2-\#1)*12)*2*\#5/\xP@bigdim+\#1*8\relax
1335 }

```

`\xP@beziertan` Formula for the polynomial

$$24(-\#1 \cdot (1-t)^2 + \#2 \cdot (3t^2 - 4t + 1) + \#3 \cdot (-3t^2 + 2t) + \#4 \cdot t^2), \quad t = \frac{1}{2}\#5.$$

Up to a scalar factor, this is the derivative of the third order Bézier polynomial above.

```

1336 \newcommand*\xP@beziertan[5]{%
1337 \dimexpr((\#4-\#1+(\#2-\#3)*3)*3*\#5/32768+(\#1-2\#2+\#3)*24)*\#5/\p@+(\#2-\#1)*24\relax
1338 }

```

`\xP@precbeziertan` Formula for the polynomial

$$(-\#1 \cdot (1-t)^2 + \#2 \cdot (3t^2 - 4t + 1) + \#3 \cdot (-3t^2 + 2t) + \#4 \cdot t^2), \quad t = \#5/\xP@bigdim.$$

This is $\frac{1}{3}$ times the derivative of the third order Bézier polynomial.

```

1339 \newcommand*\xP@precbeziertan[5]{%
1340 \dimexpr((\#4-\#1+(\#2-\#3)*3)*\#5/\xP@bigdim+(\#1-2\#2+\#3)*2)*\#5/\xP@bigdim
1341 +\#2-\#1\relax
1342 }

```

`\xP@solvelinearsystem` ●3 The macro `\xP@solvelinearsystem` solves a system of three linear equations by the Gauss algorithm. The coefficients and desired values are passed in the extended matrix

$$\left(\begin{array}{ccc|c} \xP@A & \xP@B & \xP@C & \xP@D \\ \xP@E & \xP@F & \xP@G & \xP@H \\ \xP@I & \xP@J & \xP@K & \xP@L \end{array} \right)$$

The solution is returned in the vector $(\xP@ta, \xP@tb, \xP@tc)$.

`\xP@varone` With column swapping in the Gauss algorithm, variable names might be changed. These
`\xP@vartwo` macros record the variables.
`\xP@varthree` 1343 `\@ifdefinable\xP@varone\relax`
1344 `\@ifdefinable\xP@vartwo\relax`
1345 `\@ifdefinable\xP@varthree\relax`

`\ifxP@validsol` Records if a valid solution to the linear system is returned.

1346 `\@ifdefinable\ifxP@validsol\relax`
1347 `\@ifdefinable\xP@validsoltrue\relax`
1348 `\@ifdefinable\xP@validsolfalse\relax`
1349 `\newif\ifxP@validsol`

1350 `\newcommand*\xP@solvelinearsystem{%`

Scale the matrix so that the highest absolute value in each row and each column is $\geq 2048\text{pt}$ and $< 4096\text{pt}$.

1351 `\xP@scalerow\xP@A\xP@B\xP@C\xP@D`
1352 `\xP@scalerow\xP@E\xP@F\xP@G\xP@H`
1353 `\xP@scalerow\xP@I\xP@J\xP@K\xP@L`
1354 `\xP@scalectol\xP@A\xP@E\xP@I\xP@scalectone`
1355 `\xP@scalectol\xP@B\xP@F\xP@J\xP@scalectwo`
1356 `\xP@scalectol\xP@C\xP@G\xP@K\xP@scalecthree`

Record the initial variable-to-column assignment.

1357 `\let\xP@varone\xP@ta`
1358 `\let\xP@vartwo\xP@tb`
1359 `\let\xP@varthree\xP@tc`

Find the pivot position. `\xP@M` is used temporarily.

1360 `\count@m@ne`
1361 `\@tempcnta@m@ne`
1362 `\xP@ifabsless\xP@A\xP@B\@tempcnta\z@\xP@M\xP@B`
1363 `\else\xP@M\xP@A\fi`
1364 `\xP@ifabsless\xP@M\xP@C\@tempcnta\@ne\xP@M\xP@C\fi`
1365 `\xP@ifabsless\xP@M\xP@E\@tempcnta@m@ne\count@\z@\xP@M\xP@E\fi`
1366 `\xP@ifabsless\xP@M\xP@F\@tempcnta\z@\count@\z@\xP@M\xP@F\fi`
1367 `\xP@ifabsless\xP@M\xP@G\@tempcnta\@ne\count@\z@\xP@M\xP@G\fi`
1368 `\xP@ifabsless\xP@M\xP@I\@tempcnta@m@ne\count@\@ne\xP@M\xP@I\fi`
1369 `\xP@ifabsless\xP@M\xP@J\@tempcnta\z@\count@\@ne\xP@M\xP@J\fi`
1370 `\xP@ifabsless\xP@M\xP@K\@tempcnta\@ne\count@\@ne\fi`

Swap rows

1371 `\ifcase\count@`
1372 `\xP@swapdim\xP@A\xP@E`
1373 `\xP@swapdim\xP@B\xP@F`
1374 `\xP@swapdim\xP@C\xP@G`
1375 `\xP@swapdim\xP@D\xP@H`
1376 `\or`
1377 `\xP@swapdim\xP@A\xP@I`

```

1378 \xP@swapdim\xP@B\xP@J
1379 \xP@swapdim\xP@C\xP@K
1380 \xP@swapdim\xP@D\xP@L
1381 \fi

```

Swap columns

```

1382 \ifcase\@tempcnta
1383 \xP@swapdim\xP@A\xP@B
1384 \xP@swapdim\xP@E\xP@F
1385 \xP@swapdim\xP@I\xP@J
1386 \let\xP@varone\xP@tb
1387 \let\xP@vartwo\xP@ta
1388 \xP@swapnum\xP@scaleone\xP@scaletwo
1389 \or
1390 \xP@swapdim\xP@A\xP@C
1391 \xP@swapdim\xP@E\xP@G
1392 \xP@swapdim\xP@I\xP@K
1393 \let\xP@varone\xP@tc
1394 \let\xP@varthree\xP@ta
1395 \xP@swapnum\xP@scaleone\xP@scaletwo
1396 \fi

```

First elimination

```

1397 \multiply\xP@E\m@ne
1398 \multiply\xP@I\m@ne

```

Absolute values below are < 8192pt.

```

1399 \ifdim\xP@A=\z@
1400 \else
1401 \advance\xP@F\dimexpr\xP@B*\xP@E/\xP@A\relax
1402 \advance\xP@G\dimexpr\xP@C*\xP@E/\xP@A\relax
1403 \advance\xP@H\dimexpr\xP@D*\xP@E/\xP@A\relax
1404 \advance\xP@J\dimexpr\xP@B*\xP@I/\xP@A\relax
1405 \advance\xP@K\dimexpr\xP@C*\xP@I/\xP@A\relax
1406 \advance\xP@L\dimexpr\xP@D*\xP@I/\xP@A\relax
1407 \fi

```

Find the second pivot element. \xP@M is used temporarily.

```

1408 \count@\m@ne
1409 \xP@ifabsless\xP@F\xP@G\@tempcnta\z@\xP@M\xP@G
1410 \else\@tempcnta\m@ne\xP@M\xP@F\fi
1411 \xP@ifabsless\xP@M\xP@J\@tempcnta\m@ne\count@\z@\xP@M\xP@J\fi
1412 \xP@ifabsless\xP@M\xP@K\@tempcnta\z@\count@\z@\fi

```

Swap rows

```

1413 \ifnum\count@=\z@
1414 \xP@swapdim\xP@F\xP@J
1415 \xP@swapdim\xP@G\xP@K
1416 \xP@swapdim\xP@H\xP@L
1417 \fi

```

Swap columns

```

1418 \ifnum\@tempcnta=\z@
1419 \xP@swapdim\xP@B\xP@C
1420 \xP@swapdim\xP@F\xP@G
1421 \xP@swapdim\xP@J\xP@K
1422 \let\@tempa\xP@varthree
1423 \let\xP@varthree\xP@vartwo
1424 \let\xP@vartwo\@tempa

```

```

1425 \xP@swapnum\xP@scaletwo\xP@scaletthree
1426 \fi
Second elimination. Absolute values are < 16384pt.
1427 \ifdim\xP@F=\z@
1428 \else
1429 \advance\xP@K\dimexpr-\xP@G*\xP@J/\xP@F\relax
1430 \advance\xP@L\dimexpr-\xP@H*\xP@J/\xP@F\relax
1431 \fi

```

Compute the result from the upper triagonal form. Since the matrix can be singular, we have to ensure in every step that no overflow occurs. In general, we do not allow any solution greater than 60pt.

```

1432 \xP@ifabsless{\dimexpr\xP@L/60\relax}{\dimexpr\xP@K/\xP@scaletthree\relax}%
1433 \xP@validsoltrue
1434 \xP@varthree\dimexpr\xP@L*(\xP@scaletthree*\p@)/\xP@K\relax
1435 \else
1436 \xP@validsolfalse
1437 \fi
1438 \xP@checkabs{\xP@H/8191}{\xP@F/\xP@scalettwo}%
1439 \xP@checkabs{\xP@G/\xP@scaletthree/136}{\xP@F/\xP@scalettwo}%
1440 \ifxP@validsol
1441 \xP@vartwo\dimexpr\xP@H*(\xP@scalettwo*\p@)/\xP@F
1442 -\xP@varthree*\xP@scalettwo/\xP@scaletthree*\xP@G/\xP@F\relax
1443 \xP@checkabs\xP@vartwo{60pt}%
1444 \fi
1445 \xP@checkabs{\xP@D/5461}{\xP@A/\xP@scaletone}%
1446 \xP@checkabs{\xP@B/\xP@scalettwo/91}{\xP@A/\xP@scaletone}%
1447 \xP@checkabs{\xP@C/\xP@scaletthree/91}{\xP@A/\xP@scaletone}%
1448 \ifxP@validsol
1449 \xP@varone\dimexpr\xP@D*(\xP@scaletone*\p@)/\xP@A
1450 -\xP@vartwo*\xP@scaletone/\xP@scalettwo*\xP@B/\xP@A
1451 -\xP@varthree*\xP@scaletone/\xP@scaletthree*\xP@C/\xP@A\relax
1452 \xP@checkabs\xP@varone{60pt}%
1453 \fi

```

Return the result.

```

1454 \xdef\@gtempa{%
1455 \ifxP@validsol
1456 \xP@ta\the\xP@ta\relax
1457 \xP@tb\the\xP@tb\relax
1458 \xP@tc\the\xP@tc\relax
1459 \noexpand\xP@validsoltrue
1460 \else
1461 \noexpand\xP@validsolfalse
1462 \fi
1463 }%
1464 }\@gtempa
1465 }

```

`\xP@scalerow` ●3 Scale a row of the matrix to improve numerical precision. We scale by a power of two such that the maximal length is between 2048pt and 4096pt.

```

1466 \newcommand*\xP@scalerow[4]{%
1467 \xP@M\z@
1468 \xP@Max\xP@M#1%
1469 \xP@Max\xP@M#2%
1470 \xP@Max\xP@M#3%
1471 \xP@Max\xP@M#4%

```


$$134217727 = 2048 \cdot 65536 - 1$$

```

1472 \count@134217727
1473 \loop
1474   \divide\xP@M\tw@
1475 \ifdim\xP@M>\z@
1476   \divide\count@\tw@
1477 \repeat
1478 \advance\count@\@ne
1479 \multiply#1\count@
1480 \multiply#2\count@
1481 \multiply#3\count@
1482 \multiply#4\count@
1483 }

```

`\xP@scalecol` ●3 Scale a column of the matrix to improve numerical precision. The scaling factor has to be recorded for the solution assignment later.

```

1484 \newcommand*\xP@scalecol[4]{%
1485   \xP@M\z@
1486   \xP@Max\xP@M#1%
1487   \xP@Max\xP@M#2%
1488   \xP@Max\xP@M#3%
16777215 = 2048 · 8192 − 1
1489   #416777215
1490   \loop
1491     \divide\xP@M\tw@
1492   \ifdim\xP@M>\z@
1493     \divide#4\tw@
1494   \repeat
1495   \advance#4\@ne
1496   \multiply#1#4%
1497   \multiply#2#4%
1498   \multiply#3#4%
1499 }

```

`\xP@checkabs`

```

1500 \newcommand*\xP@checkabs[2]{%
1501   \xP@ifabsless{\dimexpr#1\relax}{\dimexpr#2\relax}\else\xP@validsolfalse\fi}

```

`\xP@applinsys` ●1 ●3 ●6 This is the second, alternative algorithm for Newton's method in the offset algorithm. Approximate a solution x for the linear system $Ax = b$ for a (3×3) -matrix A . The aim is to make the norm $\|Ax - b\|$ small with small values of $\|x\|$. The approach: Set $x = \lambda A^t b$ since the normed scalar product $\langle Ax, b \rangle / \|x\|$ is maximal in this case. The norm $\|Ax - b\|$ is then minimal for $\lambda = \|A^t b\|^2 / \|AA^t b\|^2$.

This approximation is performed between one and three times.

```

1502 \newcommand*\xP@applinsys{%

```

First iteration: approximate a solution and record the result.

```

1503 \xP@applinsys@
1504 \xP@ta\xP@dta
1505 \xP@tb\xP@dtb
1506 \xP@tc\xP@dtc

```

If the result is nonzero...

```

1507 \xP@checkapp
1508 \if@tempswa

```

...modify the objective function by the estimated change, approximate again,...

```
1509 \xP@modobj
1510 \xP@applinsys@
```

...and test for a nonzero result. If it is nonzero, repeat it a third time.

```
1511 \xP@checkapp
1512 \if@tempwa
1513 \xP@modsol
1514 \xP@modobj
1515 \xP@applinsys@
1516 \xP@modsol
1517 \fi
1518 \fi
```

Return the accumulated approximation from one to three iterations.

```
1519 \xdef\@gtempa{%
1520 \xP@ta\the\xP@ta\relax
1521 \xP@tb\the\xP@tb\relax
1522 \xP@tc\the\xP@tc\relax
1523 }%\@gtempa
1524 }
```

\xP@checkapp •6 Check whether the solution is nonzero.

```
1525 \newcommand*\xP@checkapp{%
1526 \@tempswatrue
1527 \ifdim\xP@dta=\z@
1528 \ifdim\xP@dtb=\z@
1529 \ifdim\xP@dtc=\z@
1530 \@tempswafalse
1531 \fi\fi\fi
1532 }
```

\xP@modobj •3 •6 Modify the objective function by the estimated difference, according to the first-order approximation.

```
1533 \newcommand*\xP@modobj{%
1534 \advance\xP@D
1535 \dimexpr-\xP@A*\xP@dta/\p@-\xP@B*\xP@dtb/\p@-\xP@C*\xP@dtc/\p@\relax
1536 \advance\xP@H
1537 \dimexpr-\xP@E*\xP@dta/\p@-\xP@F*\xP@dtb/\p@-\xP@G*\xP@dtc/\p@\relax
1538 \advance\xP@L
1539 \dimexpr-\xP@I*\xP@dta/\p@-\xP@J*\xP@dtb/\p@-\xP@K*\xP@dtc/\p@\relax
1540 }
```

\xP@modsol •3 •6 Modify the solution vector by the approximation.

```
1541 \newcommand*\xP@modsol{%
1542 \advance\xP@ta\xP@dta
1543 \advance\xP@tb\xP@dtb
1544 \advance\xP@tc\xP@dtc
1545 }
```

\xP@applinsys@ •1 •3 •6 The heart of the approximation routine.

```
1546 \newcommand*\xP@applinsys@{%
```

Determine scaling factors \xP@sa and \xP@sb to improve numerical precision.

```
1547 \xP@sa\z@
1548 \xP@Max\xP@sa\xP@A
1549 \xP@Max\xP@sa\xP@B
```

```

1550 \xP@Max\xP@sa\xP@C
1551 \xP@Max\xP@sa\xP@E
1552 \xP@Max\xP@sa\xP@F
1553 \xP@Max\xP@sa\xP@G
1554 \xP@Max\xP@sa\xP@I
1555 \xP@Max\xP@sa\xP@J
1556 \xP@Max\xP@sa\xP@K
1557 \xP@sa\ifdim\xP@sa<5460pt\thr@@\xP@sa\else\maxdimen\fi
1558 \xP@sb\z@
1559 \xP@Max\xP@sb\xP@D
1560 \xP@Max\xP@sb\xP@H
1561 \xP@Max\xP@sb\xP@L

```

Scale the vector b .

```

1562 \ifdim\xP@sb>\z@
1563 \xP@D\dimexpr\xP@D*\maxdimen/\xP@sb\relax
1564 \xP@H\dimexpr\xP@H*\maxdimen/\xP@sb\relax
1565 \xP@L\dimexpr\xP@L*\maxdimen/\xP@sb\relax
1566 \fi

```

Vector $A^t b$ (scaled)

```

1567 \xP@Aba\dimexpr\xP@A*\xP@D/\xP@sa+\xP@E*\xP@H/\xP@sa+\xP@I*\xP@L/\xP@sa\relax
1568 \xP@Abb\dimexpr\xP@B*\xP@D/\xP@sa+\xP@F*\xP@H/\xP@sa+\xP@J*\xP@L/\xP@sa\relax
1569 \xP@Abc\dimexpr\xP@C*\xP@D/\xP@sa+\xP@G*\xP@H/\xP@sa+\xP@K*\xP@L/\xP@sa\relax

```

Vector $AA^t b$ (scaled)

```

1570 \xP@AAba\dimexpr\xP@A*\xP@Aba/\xP@sa+\xP@B*\xP@Abb/\xP@sa
1571 +\xP@C*\xP@Abc/\xP@sa\relax
1572 \xP@AAbb\dimexpr\xP@E*\xP@Aba/\xP@sa+\xP@F*\xP@Abb/\xP@sa
1573 +\xP@G*\xP@Abc/\xP@sa\relax
1574 \xP@AAbc\dimexpr\xP@I*\xP@Aba/\xP@sa+\xP@J*\xP@Abb/\xP@sa
1575 +\xP@K*\xP@Abc/\xP@sa\relax

```

Another scaling factor.

```

1576 \xP@sc\z@
1577 \xP@Max\xP@sc\xP@Aba
1578 \xP@Max\xP@sc\xP@Abb
1579 \xP@Max\xP@sc\xP@Abc
1580 \xP@Max\xP@sc\xP@AAba
1581 \xP@Max\xP@sc\xP@AAbb
1582 \xP@Max\xP@sc\xP@AAbc

```

$\|A^t b\|^2$ and $\|AA^t b\|^2$

```

1583 \ifdim\xP@sc=\z@
1584 \xP@AAb\z@
1585 \else
1586 \xP@Ab\dimexpr\xP@Aba*\xP@bigdim/\xP@sc*\xP@Aba/\xP@sc
1587 +\xP@Abb*\xP@bigdim/\xP@sc*\xP@Abb/\xP@sc
1588 +\xP@Abc*\xP@bigdim/\xP@sc*\xP@Abc/\xP@sc
1589 \relax
1590 \xP@AAb\dimexpr\xP@AAba*\xP@bigdim/\xP@sc*\xP@AAba/\xP@sc
1591 +\xP@AAbb*\xP@bigdim/\xP@sc*\xP@AAbb/\xP@sc
1592 +\xP@AAbc*\xP@bigdim/\xP@sc*\xP@AAbc/\xP@sc
1593 \relax
1594 \fi

```

The approximation $x = \lambda A^t b$ with $\lambda = \|A^t b\|^2 / \|AA^t b\|^2$.

```

1595 \xdef\@gtempa{\%
1596 \ifdim\xP@AAb=\z@

```

```

1597      \xP@dta\z@
1598      \xP@dtb\z@
1599      \xP@dtc\z@
1600  \else
1601      \xP@dta\the\dimexpr\xP@Aba*\xP@sb/\xP@sa*\p@/\xP@AAb*\xP@Ab/\maxdimen
1602      \relax
1603      \xP@dtb\the\dimexpr\xP@Abb*\xP@sb/\xP@sa*\p@/\xP@AAb*\xP@Ab/\maxdimen
1604      \relax
1605      \xP@dtc\the\dimexpr\xP@Abc*\xP@sb/\xP@sa*\p@/\xP@AAb*\xP@Ab/\maxdimen
1606      \relax
1607  \fi
1608 }%
1609 }\@gtempa
1610 }

\ifxP@offsetok Switch whether the offset curve is enough
1611 \@ifdefinable\ifxP@offsetok\relax
1612 \@ifdefinable\xP@offsetoktrue\relax
1613 \@ifdefinable\xP@offsetokfalse\relax
1614 \newif\ifxP@offsetok

\xP@maxdev Maximal deviation, measured at 19 points on the curve. The actual tolerance is 1/8 of
\xP@maxdev. With the current value 0.1pt, the tolerance is 0.0125pt, which is about 1/32
of the line width for the Computer Modern fonts.
1615 \newcommand*\xP@maxdev{.1pt}

\xP@maxobjfun Tolerance for the objective function. Recommended value is  $\frac{1}{2}(\xP@maxdev)^2$ .
1616 \newcommand*\xP@maxobjfun{.005pt}

\xP@testoffset 1 5 Test procedure for the offset curve. It tests whether the Bézier curve defined by the
control points  $X@p, \dots, Y@c$  is a good approximation for the offset curve of the partial curve
defined by  $xP@a, \dots, xP@y$  in the parameter interval  $[\xP@a, \xP@b] \subseteq [0pt, \xP@bigdim]$ .
The parameter interval is uniformly divided by 20, and the deviation is measured at the
19 inner positions. (Since the boundary points are offset exactly by the algorithm, they do
not need to be checked.)
For simplicity, the parameter interval for both curves is normalized to  $[0, 1]$  in the fol-
lowing explanations. Denote the original curve by  $c_1 : [0, 1] \rightarrow \mathbb{R}^2$  and the offset curve
by  $c_2$ . The quality test is passed if the offset curve fulfills at each of the 19 test points
 $t_i \in \{\frac{1}{20}, \dots, \frac{19}{20}\}$  one of the following two conditions:


- Let  $v$  be the tangent vector  $c_1'(t_i)$ . For  $w := c_1(t_i) - c_2(t_i)$ , denote by  $w_{par}$  the
component parallel to  $v$  and by  $w_{orth}$  the component orthogonal to  $v$ . The test is
passed if  $|w_{par}| + |w_{orth} - \xP@off| \leq \frac{1}{8}\xP@maxdev$ . If  $\|v\|$  is very small so that
the direction cannot be determined precisely, the condition is  $\|c_1(t_i) - c_2(t_i)\| -
|\xP@off| \leq \frac{1}{8}\xP@maxdev$ .
- Compute the normal line at  $t_i$  to the curve  $c_1$  and intersect it with  $c_2$ . The intersection
point is allowed to have a different parameter  $\tilde{t}_i \in [t_i - 0.5, t_i + 0.5] \cap [0, 1]$ . Then
let  $w := c_1(t_i) - c_2(\tilde{t}_i)$  and test whether  $|w_{par}| + |w_{orth} - \xP@off| \leq \frac{1}{8}\xP@maxdev$ .
( $|w_{par}|$  is very small in this case and is nonzero only because of limited precision, in
particular since  $\tilde{t}_i$  is determined with an error of  $\approx 2^{-17}$  ( $= \frac{1}{2}sp$ ).)


1617 \newcommand*\xP@testoffset{%
```

Default values for the return statement and the loop continuation.

```

1618 \gdef\xP@afteroffsetok{\xP@offsetoktrue}%
1619 \def\xP@offsetokif{\ifdim\xP@ti<1.85pt}%
1620 \xP@ti.1pt
1621 \loop
\xP@tip =  $t_i$ , denormalized for  $c_1$ 
1622 \xP@tip\dimexpr\xP@a+(\xP@b-\xP@a)*\xP@ti/131072\relax
Point on the original curve  $c_1$  (scaled by  $-8$ )
1623 \Lp\xP@precbezierpoly\xP@xa\xP@xb\xP@xc\xP@xd\xP@tip
1624 \Up\xP@precbezierpoly\xP@ya\xP@yb\xP@yc\xP@yd\xP@tip
 $8c_2(t_i) - 8c_1(t_i)$ 
1625 \xP@valA\dimexpr\xP@bezierpoly\Xp\Lc\Rc\Xc\xP@ti-\Lp\relax
1626 \xP@valB\dimexpr\xP@bezierpoly\Yp\Uc\Uc\Yc\xP@ti-\Up\relax
 $v$ 
1627 \d@X\xP@precbeziertan\xP@xa\xP@xb\xP@xc\xP@xd\xP@tip
1628 \d@Y\xP@precbeziertan\xP@ya\xP@yb\xP@yc\xP@yd\xP@tip
1629 \xP@vecLen
Decide if  $v$  is big enough (heuristically, may be changed in the future)
1630 \@tempdimc\dimexpr(\xP@b-\xP@a)*\@tempdimb/\xP@bigdim\relax
1631 \xP@abs\@tempdimc
1632 \ifdim.01pt<\@tempdimc
 $8w_{par}, 8w_{orth} - 8\xP@off$ ,
1633 \xP@devA\dimexpr\xP@valA*\d@X/\@tempdimb+\xP@valB*\d@Y/\@tempdimb\relax
1634 \xP@devB\dimexpr\xP@valA*\d@Y/\@tempdimb-\xP@valB*\d@X/\@tempdimb-8\xP@off
1635 \relax
1636 \xP@abs\xP@devA
1637 \xP@abs\xP@devB
1638 \@tempdima\dimexpr\xP@devA+\xP@devB\relax
1639 \else
If the velocity is zero, just pass the test.
1640 \ifdim\@tempdimc=\z@
1641 \@tempdima\z@
1642 \else
 $8\|c_1(t_i) - c_2(t_i)\|$ 
1643 {\%
1644 \d@X\xP@valA
1645 \d@Y\xP@valB
1646 \xP@vecLen@
1647 \global\dimen@i\@tempdimb
1648 }\@tempdima\dimen@i
1649 \advance\@tempdima\ifdim\xP@off>\z@-\fi8\xP@off
1650 \xP@abs\@tempdima
1651 \fi
1652 \fi
If the first condition is not fulfilled, test the second one.
1653 \ifdim\@tempdima>\xP@maxdev
 $-c_1(t_i)$ 
1654 \divide\Lp-8\relax
1655 \divide\Up-8\relax

```

Affine transformation of the offset curve: translate by $-c_1(t_i)$ and rotate so that the tangent v to $c_1(t_i)$ becomes the x -axis.

```

1656      {%
1657      \xP@transformcoor\X@p\Y@p
1658      \xP@transformcoor\L@c\U@c
1659      \xP@transformcoor\R@c\D@c
1660      \xP@transformcoor\X@c\Y@c

```

Find the parameter \tilde{t}_i and decide whether the approximation at \tilde{t}_i is good.

```

1661      \xP@findzero
1662      }%
1663      \fi
1664      \xP@offsetokif
1665      \advance\xP@ti.1pt
1666      \repeat
1667      \expandafter}\xP@afteroffsetok
1668  }

```

\xP@afteroffsetok

```

1669 \newcommand*\xP@afteroffsetok{}

```

\xP@offsetokif

```

1670 \newcommand*\xP@offsetokif{}

```

\xP@transformcoor

•5 Affine coordinate transformation. First, translate the coordinates in (#1,#2) by the vector $-(\text{\L@p}, \text{\U@p})$, then rotate by the angle between $v := (\text{\d@X}, \text{\d@Y})$ and $(1,0)$. The register \@tempdimb must contain the length $\|v\|$.

```

1671 \newcommand*\xP@transformcoor[2]{%
1672   \advance#1\L@p
1673   \advance#2\U@p
1674   \@tempdima\dimexpr#1*\d@X/\@tempdimb+#2*\d@Y/\@tempdimb\relax
1675   #2\dimexpr#2*\d@X/\@tempdimb-#1*\d@Y/\@tempdimb\relax
1676   #1\@tempdima
1677 }

```

\xP@findzero

•5 Find the parameter \tilde{t}_i by nested intervals/intermediate value theorem.

```

1678 \newcommand*\xP@findzero{%
1679   \xP@setleftvalue{.05}%
1680   \xP@setrightvalue{.05}%

```

Normalize: function value (x -coordinate) should be nonnegative at the upper end.

```

1681   \ifdim\xP@valB<\z@\xP@reversecoeff\fi

```

If the function value at the lower end is also positive, try a smaller parameter interval $t_i \pm \delta$ pt for $\delta \in \{.5, .35, .25, .2, .15, .1, .05\}$. Maybe we have different signs for the x -coordinate for the larger boundary parameters.

```

1682   \ifdim\xP@valA>\z@
1683     \@tempswatrue
1684     \for\@tempa:={.1,.15,.2,.25,.35,.5,1.1}\do{%
1685       \if@tempswa
1686         \xP@setleftvalue\@tempa
1687         \ifdim\xP@valA<\z@\@tempswafalse\fi
1688       \if@tempswa
1689         \xP@setrightvalue\@tempa
1690       \ifdim\xP@valB<\z@
1691         \@tempswafalse

```

```

1692         \xP@reversecoeff
1693     \fi
1694 \fi
1695 \fi
1696 }%

```

Last resort: Try the midpoint.

```

1697 \if@tempwa
1698 \L@p\xP@ti
1699 \xP@valA\xP@bezierpoly\X@p\L@c\R@c\X@c\L@p

```

If the midpoint leads to a negative value, we can proceed with a small interval. Otherwise, set both boundary points to the midpoint and effectively skip nested intervals.

```

1700 \ifdim\xP@valA<\z@

```

We had this before, so we know that the value is positive.

```

1701 \xP@setrightvalue{.05}%
1702 \else
1703 \U@p\L@p
1704 \xP@valB\xP@valA
1705 \fi
1706 \fi
1707 \fi

```

The actual nested interval algorithm

```

1708 \loop
1709 \ifnum\numexpr\U@p-\L@p\relax>\@ne
1710 \xP@ti\dimexpr(\L@p+\U@p)/2\relax
1711 \xP@devA\xP@bezierpoly\X@p\L@c\R@c\X@c\xP@ti
1712 \ifdim\xP@devA>\z@
1713 \U@p\xP@ti
1714 \xP@valB\xP@devA
1715 \else
1716 \L@p\xP@ti
1717 \xP@valA\xP@devA
1718 \fi
1719 \repeat

```

Take the left or right boundary point (only 1sp apart), depending on which one yields the smaller x -coordinate.

```

1720 \xP@ifabsless\xP@valB\xP@valA
1721 \L@p\U@p
1722 \xP@valA\xP@valB
1723 \fi

```

Compare the y -coordinate with $\xP@off$.

```

1724 \xP@valB\dimexpr\xP@bezierpoly\Y@p\U@c\D@c\Y@c\L@p+8\xP@off\relax
1725 \xP@abs\xP@valA
1726 \xP@abs\xP@valB
1727 \ifdim\dimexpr\xP@valA+\xP@valB\relax>\xP@maxdev\relax
1728 \xP@failed
1729 \fi
1730 }

```

$\xP@failed$ Break the loop for the t_i in $\xP@testoffset$. Set the return value to false.

```

1731 \newcommand*\xP@failed{%
1732 \global\let\xP@offsetokif\iffalse
1733 \gdef\xP@afteroffsetok{\xP@offsetokfalse}%
1734 }

```

`\xP@reversecoeff` Reverse the function for the nested interval algorithm.

```
1735 \newcommand*\xP@reversecoeff{%
1736   \multiply\xP@m@ne
1737   \multiply\L@c@m@ne
1738   \multiply\R@c@m@ne
1739   \multiply\X@c@m@ne
1740   \multiply\xP@valA@m@ne
1741   \multiply\xP@valB@m@ne
1742 }
```

`\xP@setleftvalue` ●5

```
1743 \newcommand*\xP@setleftvalue[1]{%
1744   \L@p\dimexpr\xP@ti-#1\p@\relax
1745   \ifdim\L@p<-.1pt\L@p-.1pt\fi
1746   \xP@valA\xP@bezierpoly\X@p\L@c\R@c\X@c\L@p
1747 }
```

`\xP@setrightvalue` ●5

```
1748 \newcommand*\xP@setrightvalue[1]{%
1749   \U@p\dimexpr\xP@ti+#1\p@\relax
1750   \ifdim\U@p>2.1\p@\U@p2.1\p@\fi
1751   \xP@valB\xP@bezierpoly\X@p\L@c\R@c\X@c\U@p
1752 }
```

9.10 Multiple dashed curves

`\xP@splinedbldashed`

```
1753 \newcommand*\xP@splinedbldashed{%
1754   \xP@checkspline\xP@splinemultdashed\xP@doublestroke}
```

`\xP@splinetrbldashed`

```
1755 \newcommand*\xP@splinetrbldashed{%
1756   \xP@checkspline\xP@splinemultdashed\xP@trblstroke}
```

`\xP@splinemultdashed`

```
1757 \newcommand*\xP@splinemultdashed[1]{%
```

Expected dash number. It is an even number if the spline is the continuation of the previous one, otherwise (default case) an odd number.

```
1758   \xP@testcont\xP@dashmacro
1759   \@tempcnta
1760   \ifxP@splinecont
1761     \numexpr2*((\@tempdimb-\xydashl@/3)/(2*\xydashl@))\relax
1762   \else
1763     \numexpr2*((\@tempdimb+\xydashl@)/(2*\xydashl@))-1\relax
1764   \fi
1765   \ifnum\@tempcnta>\@ne
1766     \xP@splinemultdashed@#1%
1767   \else
```

One dash: paint a solid line. Less than one dash: Leave the segment out, just record the end point.

```
1768   \ifnum\@tempcnta=\@ne
1769     \xP@splinemultsolid#1
1770   \else
1771     \xP@savec
```



```

1772     \fi
1773 \fi
1774 \global\let\xP@lastpattern\xP@dashmacro
1775 }

```

`\xP@splinemultdashed@` ●1 ●7 Make a list of parameter pairs for the start and end point of a dash.

```

1776 \newcommand*\xP@splinemultdashed@[1]{%
1777   \xP@inibigdim
Dash length
1778   \@tempdima\dimexpr\@tempdimb/\@tempcnta\relax
1779   \xP@temppar\z@
1780   \toks@{}%
1781   \xP@savec
1782   \ifodd\@tempcnta
1783   \else
1784     \xP@slide
1785   \fi
1786   \@tempcnta\z@
1787   \loop
1788     \advance\@tempcnta\@ne
1789     \xP@append\toks@{\ifodd\@tempcnta\noexpand\xP@paintdash\fi
1790       {\the\xP@temppar}}%
1791     \xP@oldpar\xP@temppar
1792     \xP@slide
1793   \ifdim\xP@temppar<\xP@bigdim
1794   \repeat

```

The last position is kept as a scaling factor so that the last dot can be drawn at exactly the parameter 1. Use the last or the next-to-last position, depending on the parity of segments.

```

1795   \xP@lastpar
1796   \ifodd\@tempcnta
1797     \xP@temppar
1798     \xP@append\toks@{\the\xP@temppar}%
1799   \else
1800     \xP@oldpar
1801   \fi

```

Convert the list of parameters to a list of PDF tokens.

```

1802   \@temptokena{}%
1803   \xP@setsolidpat
1804   \global\let\xP@lastpattern\xP@dashmacro
1805   \@for\@tempa:={#1}\do{\the\toks@}%
1806   \xP@stroke{\the\@temptokena}%
1807 }

```

`\xP@paintdash` ●1 ●7

```

1808 \newcommand*\xP@paintdash[2]{%
1809   \xP@paintsolid{\dimexpr#1*\xP@bigdim/\xP@lastpar\relax}%
1810   {\dimexpr#2*\xP@bigdim/\xP@lastpar\relax}%
1811 }

```

9.11 Multiple dotted curves

```

\splinedbldotted@
\xP@splinedbldotted@ 1812 \xP@hook{curve}{\splinedbldotted@}
1813 \newcommand*\xP@splinedbldotted@{%

```

```

1814 \let\xP@normalmult\@ne
1815 \xP@checkspline\xP@splinemultdotted\xP@doublestroke}

\xP@splinetrbldotted

1816 \newcommand*\xP@splinetrbldotted{%
1817 \let\xP@normalmult\tw@
1818 \xP@checkspline\xP@splinemultdotted\xP@trblstroke}

\xP@multdottedpat Dotted lines with multiple strokes are drawn in a different way from single-stroked lines.
They are composed of many small, straight lines normal to the curve at every dot position.
Hence, the dot pattern for multiple curves has dots which are spaced by the normal distance
between strokes.

1819 \newcommand*\xP@multdottedpat{%
1820 \def\xP@pattern{0 J [\xP@coord\xP@preclw{\xydashh@-\xP@preclw}]0 d}%
1821 \global\let\xP@lastpattern\xP@dotmacro
1822 }

\xP@normalmult

1823 \@ifdefinable\xP@normalmult\relax

\xP@splinemultdotted 1 7
1824 \newcommand\xP@splinemultdotted[1]{%
1825 \xP@inibigdim
Make a list of dot positions on the spline segment.
1826 \xP@temppar\z@
1827 \xP@testcont\xP@dotmacro
1828 \ifxP@splinecont
Expected dot distance (see the formula in \xP@setdottedpat)
1829 \@tempdimc\dimexpr\@tempdimb/(\@tempdimb/131072+1)\relax
1830 \@tempdima\dimexpr\@tempdimc-\xP@preclw/2\relax
1831 \xP@slide
1832 \@tempdima\@tempdimc
1833 \else
1834 \@tempdima\dimexpr\xP@preclw/2\relax
1835 \xP@slide
Expected dot distance (see the formula in \xP@setdottedpat)
1836 \@tempdima\dimexpr\@tempdimb-\xP@preclw\relax
1837 \ifdim\@tempdima<\z@\@tempdima\z@\fi
1838 \@tempdima\dimexpr\@tempdima/(\@tempdima/131072+1)\relax
1839 \fi
1840 \xP@savvec
1841 \toks@{}%
If the end of the segment is reached before the first dot position, leave the segment out.
1842 \ifdim\xP@temppar<\xP@bigdim
1843 \loop
1844 \xP@append\toks@{\noexpand\xP@paintdot{\the\xP@temppar}}%
1845 \xP@oldpar\xP@temppar
1846 \xP@slide
1847 \ifdim\xP@temppar<\xP@bigdim
1848 \repeat
1849 \xP@velocity\xP@bigdim\xP@tempvel

```

Test whether the last or the next-to-last dot is closer to `\xP@bigdim`. Measure from the end of the dot, hence the contribution of `\xP@preclw`. Also consider the case that the velocity at the end point is very small. In this case, always choose the next-to-last dot as the final one.

```

1850 \ifdim
1851 \ifdim\xP@preclw<\xP@tempvel
1852 \dimexpr2\xP@bigdim-\xP@oldpar-\xP@preclw*\xP@bigdim/\xP@tempvel\relax
1853 \else
1854 -\maxdimen
1855 \fi<\xP@temppar
1856 \xP@temppar\xP@oldpar
1857 \else
1858 \xP@append\toks@{\noexpand\xP@paintdot{\the\xP@temppar}}%
1859 \fi
1860 \@tempdima\dimexpr\xP@preclw/2\relax
1861 \xP@slide
1862 \xP@lastpar\xP@temppar

```

Convert the list of parameters to a list of PDF tokens.

```

1863 \@temptokena{}%
1864 \the\toks@

```

Actually draw the points in the list.

```

1865 \xP@multidottedpat
1866 \xP@stroke{\the\@temptokena}%
1867 \else

```

Leave the segment out because it is too short.

```

1868 \global\let\xP@lastpattern\@empty
1869 \fi
1870 }}

```

`\xP@slide` ●1 ●7 Slide along the Bézier segment by `\@tempdima`. Needs: `Xy-pic` spline parameter, current position parameter `\xP@temppar`, total spline length `\@tempdimb`.

```

1871 \newcommand*\xP@slide{%
1872 \xP@slide@

```

Return the new spline parameter after sliding.

```

1873 \global\dimen@i\xP@temppar
1874 }\xP@temppar\dimen@i
1875 }

```

`\xP@slide@` ●1 ●7

```

1876 \newcommand*\xP@slide@{%

```

Compute the velocity at two points, the starting point and an estimate for the end point.

```

1877 \xP@velocity\xP@temppar\xP@tempvel

```

The first estimate for the parameter increment is based on the total spline length.

```

1878 \@tempdimc\dimexpr\xP@bigdim*\@tempdima/\@tempdimb\relax
1879 \count@z@
1880 \@tempswatru

```

Improve the parameter increment iteratively.

```

1881 \loop

```

Velocity at the estimated end point.

```

1882 \xP@velocity{\xP@temppar+\@tempdimc}\xP@tempvel@

```

Prevent arithmetic overflow.

```
1883     \ifdim\dimexpr\@tempdima*4/13\relax>\xP@tempvel@
1884         \@tempswafalse
1885     \else
```

Difference to the old parameter increment. This is Newton's method, applied to the estimated spline length based on the velocities \xP@tempvel and \xP@tempvel@ at \xP@temppar and (\xP@temppar + \@tempdimc).

```
1886         \xP@parinc\dimexpr\@tempdima*\xP@bigdim/\xP@tempvel@
1887         -(\xP@tempvel+\xP@tempvel@)/2*\@tempdimc/\xP@tempvel@\relax
1888         \advance\@tempdimc\xP@parinc
```

If the estimated parameter increment is bigger than .12, increase the parameter by .1 and slide only partially. This increases the precision if the parameter increment is big.

```
1889         \ifdim\@tempdimc>.12\xP@bigdim
1890             \@tempswafalse
1891         \else
```

If the estimate is not improved, break the loop.

```
1892             \ifdim\xP@parinc=\z@
1893                 \@tempswafalse
1894             \else
```

Also break the loop after 10 iterations.

```
1895                 \ifnum\count@=9\relax
1896                     \@tempswafalse
1897                 \fi
1898             \fi
1899         \fi
1900     \fi
1901     \if@tempswa
1902         \advance\count@\@ne
1903     \repeat
```

Note that \if@tempswa is always false here.

If the parameter increment would be more than .1 and if the parameter is not too big already, increase the parameter by .1 and slide again.

```
1904     \ifdim\xP@temppar<5461pt
1905     \ifdim\@tempdimc>.1\xP@bigdim
1906         \@tempswatrue
1907     \fi
1908     \fi
1909     \if@tempswa
1910     {%
1911         \dimen5\xP@temppar
1912         \advance\xP@temppar.1\xP@bigdim
```

Cap the end parameter to prevent arithmetic overflows.

```
1913         \ifdim\xP@temppar>5461pt\xP@temppar5461pt\fi
1914         \dimen7\xP@temppar
```

Determine the exact distance of the partial slide.

```
1915         \xP@shaveprec{\dimen5}{\dimen7}%
1916         \xP@bezierlength
1917         \global\dimen@i\dimexpr\@tempdima-\@tempdimb\relax
1918         \global\dimen3\xP@temppar
1919     }%
1920     \@tempdima\dimen@i
1921     \xP@temppar\dimen3\relax
```

Slide again.

```
1922 \expandafter\xP@slide@
1923 \else
```

Finish the slide and return the new parameter.

```
1924 \advance\xP@temppar\@tempdimc
1925 \fi
1926 }
```

\xP@paintdot ●1 ●7

```
1927 \newcommand*\xP@paintdot[1]{%
```

Scale the parameter with a correction factor

```
1928 \@tempdima\dimexpr#1*\xP@bigdim/\xP@lastpar\relax
```

Position at parameter value \xP@temppar

```
1929 \xP@tangent
1930 \xP@posX\dimexpr\xP@precbezierpoly\X@p\L@c\R@c\X@c\@tempdima/8\relax
1931 \xP@posY\dimexpr\xP@precbezierpoly\Y@p\U@c\D@c\Y@c\@tempdima/8\relax
```

Normal vector to the curve with length \xydashhh@

```
1932 \@tempdima\dimexpr(\xydashhh@+\xP@preclw/\xP@normalmult)/2\relax
1933 \L@p\dimexpr\d@Y*\@tempdima/\@tempdimb\relax
1934 \U@p\dimexpr-\d@X*\@tempdima/\@tempdimb\relax
```

Append two points on both sides of the curve to the list. (The “multidottedpat” pattern is made to draw points with distance \xydashhh@.)

```
1935 \xP@append\@temptokena{\xP@coor{\xP@posX+\L@p*\xP@normalmult}}%
1936 {\xP@posY+\U@p*\xP@normalmult}m %
1937 \xP@coor{\xP@posX-\L@p*\xP@normalmult+\@ne}}%
1938 {\xP@posY-\U@p*\xP@normalmult+\@ne}}l }%
1939 }
```

9.12 Squiggled curves

\xP@splinesquiggled

```
1940 \newcommand*\xP@splinesquiggled{%
1941 \xP@checkspline\xP@splinesquiggled@z@}
```

\xP@splinedblsquiggled

```
1942 \newcommand*\xP@splinedblsquiggled{%
1943 \xP@checkspline\xP@splinesquiggled@\xP@doublestroke}
```

\xP@splinetrbldsquiggled

```
1944 \newcommand*\xP@splinetrbldsquiggled{%
1945 \xP@checkspline\xP@splinesquiggled@\xP@trblstroke}
```

\xP@splinesquiggled@ ●1 ●7

```
1946 \newcommand*\xP@splinesquiggled@[1]{%
1947 \xP@inibigdim
```

Reverse the direction of the little arcs, if the last squiggle from the previous segment makes it necessary.

```
1948 \xP@testcont\xP@oddsquigglemacro
1949 \ifxP@splinecont
1950 \def\xP@squigsign{-}%
1951 \else
```

```

1952      \let\xP@squigsign\@empty
1953      \fi
1954      \xP@savvec

```

Expected squiggle length

```

1955      \@tempcnta=\numexpr\@tempdimb/\xybsqll\@relax
1956      \ifnum\@tempcnta<\tw@\@tempcnta\tw@\fi
1957      \multiply\@tempcnta\tw@
1958      \@tempdima\dimexpr\@tempdimb/\@tempcnta\relax
1959      \xP@squiglen\@tempdima

```

Make a list of dot positions on the spline segment.

```

1960      \xP@temppar\z@
1961      \toks@{}\%
1962      \@tempcnta\z@
1963      \loop
1964      \advance\@tempcnta\@ne
1965      \xP@append\toks@{\noexpand\xP@paintsquiggle{\the\xP@temppar}}\%
1966      \xP@oldpar\xP@temppar
1967      \xP@slide
1968      \ifdim\xP@temppar<\xP@bigdim
1969      \repeat

```

The last position is kept as a scaling factor so that the last dot can be drawn at exactly the parameter 1. Use the last or the next-to-last position, on the parity of the number of positions.

```

1970      \xP@lastpar
1971      \ifodd\@tempcnta
1972      \xP@oldpar
1973      \advance\@tempcnta\m@ne
1974      \else
1975      \xP@temppar
1976      \xP@append\toks@{\noexpand\xP@paintsquiggle{\the\xP@temppar}}\%
1977      \fi

```

Convert the list of parameters to a list of PDF tokens.

```

1978      \@temptokena{}\%
1979      \xP@setsolidpat

```

Record the direction of the last squiggle.

```

1980      \global\expandafter\let\expandafter\xP@lastpattern
1981      \ifodd\numexpr\@tempcnta/2\if\xP@squigsign-+1\fi\relax
1982      \xP@oddsquigglemacro
1983      \else
1984      \xP@evensquigglemacro
1985      \fi

```

Draw the squiggles.

```

1986      \@for\@tempa:={#1}\do{%
1987      \let\xP@dosquiggle\xP@dosquiggle@
1988      \count@\z@
1989      \the\toks@
1990      }%
1991      \xP@stroke{\the\@temptokena}%
1992  }}

```

\xP@paintsquiggle ●1 ●7

```

1993 \newcommand*\xP@paintsquiggle[1]{%
1994 \xP@squigglevectors{#1}%

```

```

1995 \xP@dosquiggle
1996 \ifnum\count@=\thr@@\relax\count@\z@\else\advance\count@\@ne\fi
1997 }

\xP@squigglevectors ●1 ●7
1998 \newcommand*\xP@squigglevectors[1]{%
Scale the parameter with a correction factor
1999 \@tempdima\dimexpr#1*\xP@bigdim/\xP@lastpar\relax
Position at parameter value \xP@temppar, offset for multiple curves.
2000 \xP@tangent
2001 \xP@posX\dimexpr\xP@precbezierpoly\Xp\L@c\R@c\X@c\@tempdima/8%
2002 -\d@Y*(\@tempa)/\@tempdimb\relax
2003 \xP@posY\dimexpr\xP@precbezierpoly\Yp\U@c\D@c\Y@c\@tempdima/8%
2004 +\d@X*(\@tempa)/\@tempdimb\relax
Tangent vector to the curve with correct length
2005 \L@p\dimexpr\d@X*\xP@squiglen/\@tempdimb\relax
2006 \U@p\dimexpr\d@Y*\xP@squiglen/\@tempdimb\relax
2007 \R@p\dimexpr\L@p*543339720/1311738121\relax
2008 \D@p\dimexpr\U@p*543339720/1311738121\relax
2009 \X@min\dimexpr\L@p*362911648/967576667\relax
2010 \Y@min\dimexpr\U@p*362911648/967576667\relax
2011 \X@max\dimexpr(\L@p+\xP@squigsign\U@p)*173517671/654249180\relax
2012 \Y@max\dimexpr(\L@p-\xP@squigsign\U@p)*173517671/654249180\relax
2013 }

\xP@dosquiggle ●7
2014 \@ifdefinable\xP@dosquiggle@\relax

\xP@dosquiggle@ ●7
2015 \newcommand*\xP@dosquiggle@{%
2016 \edef\next@{\xP@coor{\xP@posX}{\xP@posY}m
2017 \xP@coor{\xP@posX+\Y@max}{\xP@posY+\xP@squigsign\X@max}%
2018 }%
2019 \let\xP@dosquiggle\xP@dosquiggle@@
2020 }

\xP@dosquiggle@@ ●7
2021 \newcommand*\xP@dosquiggle@@{%
2022 \xP@append\@temptokena{\next@\expandafter\xP@coor
2023 \ifcase\count@
2024 {\xP@posX-\Y@max}{\xP@posY-\xP@squigsign\X@max}%
2025 \xP@coor\xP@posX\xP@posY
2026 \or
2027 {\xP@posX-\xP@squigsign\D@p-\X@min}{\xP@posY+\xP@squigsign\R@p-\Y@min}%
2028 \xP@coor{\xP@posX-\xP@squigsign\D@p}{\xP@posY+\xP@squigsign\R@p}%
2029 \or
2030 {\xP@posX-\X@max}{\xP@posY+\xP@squigsign\Y@max}%
2031 \xP@coor\xP@posX\xP@posY
2032 \or
2033 {\xP@posX+\xP@squigsign\D@p-\X@min}{\xP@posY-\xP@squigsign\R@p-\Y@min}%
2034 \xP@coor{\xP@posX+\xP@squigsign\D@p}{\xP@posY-\xP@squigsign\R@p}%
2035 \fi c }%
2036 \edef\next@{\expandafter\xP@coor
2037 \ifcase\count@

```

```

2038      {\xP@posX+\Y@max}{\xP@posY+\xP@squigsign\X@max}%
2039      \or
2040      {\xP@posX-\xP@squigsign\D@p+\X@min}{\xP@posY+\xP@squigsign\R@p+\Y@min}%
2041      \or
2042      {\xP@posX+\X@max}{\xP@posY-\xP@squigsign\Y@max}%
2043      \or
2044      {\xP@posX+\xP@squigsign\D@p+\X@min}{\xP@posY-\xP@squigsign\R@p+\Y@min}%
2045      \fi
2046    }%
2047  }

```

\xP@splinebrokensquiggled

```

2048 \newcommand*\xP@splinebrokensquiggled{%
2049   \xP@checkspline\xP@splinebrokensquiggled@{z@}

```

\xP@splinebrokendsquiggled

```

2050 \newcommand*\xP@splinebrokendsquiggled{%
2051   \xP@checkspline\xP@splinebrokensquiggled@\xP@doublestroke}

```

\xP@splinebrokentrblesquiggled

```

2052 \newcommand*\xP@splinebrokentrblesquiggled{%
2053   \xP@checkspline\xP@splinebrokensquiggled@\xP@trblstroke}

```

\xP@splinebrokensquiggled@ ●1 ●7

```

2054 \newcommand*\xP@splinebrokensquiggled@[1]{%
2055   \xP@inibigdim

```

Start with a space if the last squiggle from the previous segment makes it necessary.

```

2056   \xP@testcont\xP@brokensquigglemacro
2057   \ifxP@splinecont
2058     \let\xP@squigsign \@firstoftwo
2059   \else
2060     \let\xP@squigsign \@secondoftwo
2061   \fi
2062   \xP@savvec

```

Expected squiggle length

```

2063   \@tempcnta\numexpr(\@tempdimb\xP@squigsign\{+2*\xybsql1@})%
2064   /(4*\xybsql1@)\relax
2065   \ifnum\@tempcnta<\@ne\@tempcnta\@ne\fi
2066   \@tempdima\dimexpr\@tempdimb/(8*\@tempcnta\xP@squigsign\{-4})\relax
2067   \xP@squiglen\@tempdima

```

Make a list of dot positions on the spline segment.

```

2068   \xP@temppar\z@
2069   \xP@squigsign{\xP@slide\xP@slide\xP@slide\xP@slide}\}%
2070   \count@\z@
2071   \toks@{\}%
2072   \loop
2073     \xP@append\toks@{\noexpand\xP@paintbrokensquiggle{\the\xP@temppar}}%
2074     \xP@slide
2075     \xP@append\toks@{\the\xP@temppar}%
2076     \xP@slide
2077     \xP@append\toks@{\the\xP@temppar}%
2078     \xP@slide
2079     \xP@append\toks@{\the\xP@temppar}%
2080     \xP@slide

```



```

2081      \xP@append\toks@{\the\xP@temppar}}%
2082      \xP@lastpar\xP@temppar
2083      \advance\count@\@ne
2084      \ifnum\count@<\@tempcnta
2085      \xP@slide
2086      \xP@slide
2087      \xP@slide
2088      \xP@slide
2089      \repeat

```

Convert the list of parameters to a list of PDF tokens.

```

2090      \@temptokena{%
2091      \xP@setsolidpat
2092      \global\let\xP@lastpattern\xP@brokensquigglesmacro

```

Draw the squiggles.

```

2093      \let\xP@squigsig\@empty
2094      \@for\@tempa:={#1}\do{\the\toks@}%
2095      \xP@stroke{\the\@temptokena}%
2096  }}

```

\xP@paintbrokensquiggle ●1 ●7

```

2097 \newcommand*\xP@paintbrokensquiggle[5]{%
2098   \xP@squigglevectors{#1}%
2099   \xP@append\@temptokena{%
2100     \xP@coor\xP@posX\xP@posY m %
2101     \xP@coor{\xP@posX+\Y@max}{\xP@posY+\X@max}%
2102   }%
2103   \xP@squigglevectors{#2}%
2104   \xP@append\@temptokena{%
2105     \xP@coor{\xP@posX-\D@p-\X@min}{\xP@posY+\R@p-\Y@min}%
2106     \xP@coor{\xP@posX-\D@p}{\xP@posY+\R@p}c %
2107     \xP@coor{\xP@posX-\D@p+\X@min}{\xP@posY+\R@p+\Y@min}%
2108   }%
2109   \xP@squigglevectors{#3}%
2110   \xP@append\@temptokena{%
2111     \xP@coor{\xP@posX-\X@max}{\xP@posY+\Y@max}%
2112     \xP@coor\xP@posX\xP@posY c %
2113     \xP@coor{\xP@posX+\X@max}{\xP@posY-\Y@max}%
2114   }%
2115   \xP@squigglevectors{#4}%
2116   \xP@append\@temptokena{%
2117     \xP@coor{\xP@posX+\D@p-\X@min}{\xP@posY-\R@p-\Y@min}%
2118     \xP@coor{\xP@posX+\D@p}{\xP@posY-\R@p}c %
2119     \xP@coor{\xP@posX+\D@p+\X@min}{\xP@posY-\R@p+\Y@min}%
2120   }%
2121   \xP@squigglevectors{#5}%
2122   \xP@append\@temptokena{%
2123     \xP@coor{\xP@posX-\Y@max}{\xP@posY-\X@max}%
2124     \xP@coor\xP@posX\xP@posY c %
2125   }%
2126 }

```

End of the section for Xy-pic's “curve” option.

```

2127 \xyendinput
2128 </curve>
2129 <*basic>

```

9.13 Spline continuation

The following code handles the spline continuation (see [Section 6](#)). We introduce global macros which store the last end point of a Bézier segment. If the next segment continues at exactly the same coordinates, the dash/dot/squiggle patterns recognize the continuation.

```

\XP@lastX
\XP@lastY 2130 \newcommand*\XP@lastX{}
\XP@lastpattern 2131 \newcommand*\XP@lastY{}
2132 \newcommand*\XP@lastpattern{}

\XP@solidmacro
\XP@dotmacro 2133 \newcommand*\XP@solidmacro{solid}
\XP@dashmacro 2134 \newcommand*\XP@dotmacro{dot}
\XP@evensquigglemacro 2135 \newcommand*\XP@dashmacro{dash}
\XP@oddsquigglemacro 2136 \newcommand*\XP@evensquigglemacro{evensquiggle}
\XP@brokensquigglemacro 2137 \newcommand*\XP@oddsquigglemacro{oddsquiggle}
2138 \newcommand*\XP@brokensquigglemacro{brokensquiggle}

\xyinside@ Reset the last position with every new diagram.
2139 \renewcommand*\xyinside@{%
2140   \global\let\XP@lastpattern\@empty
2141   \saveXyStyle@ \aftergroup\xycheck@end
2142   \setboxz\h\bgroup
2143   \plainxy@
2144   \X@c=\z@\Y@c=\z@\czeroEdge@
2145   \X@p=\z@\Y@p=\z@\U@p=\z@\D@p=\z@\L@p=\z@\R@p=\z@\Edge@p={\zeroEdge}%
2146   \X@min=\hsize\X@max=-\hsize\Y@min=\hsize\Y@max=-\hsize
2147   \mathsurround=\z@
2148   \expandafter\POS\everyxy@@
2149 }

\XP@savec Save the current end point
2150 \newcommand*\XP@savec{%
2151   \xdef\XP@lastX{\the\X@c}%
2152   \xdef\XP@lastY{\the\Y@c}%
2153 }

\ifXP@splinecont Switch: does the next line/spline continue at the end point of the last one?
2154 \@ifdefinable\ifXP@splinecont\relax
2155 \@ifdefinable\XP@splineconttrue\relax
2156 \@ifdefinable\XP@splinecontfalse\relax
2157 \newif\ifXP@splinecont

\XP@testcont Test for \ifXP@splinecont
2158 \newcommand*\XP@testcont[1]{%
2159   \XP@splinecontfalse
2160   \ifXP@cont
2161     \ifx\XP@lastpattern#1%
2162       \ifdim\XP@lastX=\X@p
2163         \ifdim\XP@lastY=\Y@p
2164           \XP@splineconttrue
2165         \fi
2166       \fi
2167     \fi
2168   \fi
2169 }
```

```

\ifxP@cont Switch: shall the spline hack be applied?
2170 \@ifdefinable\ifxP@cont\relax
2171 \@ifdefinable\xP@conttrue\relax
2172 \@ifdefinable\xP@contfalse\relax
2173 \newif\ifxP@cont

\xypdfcontpatternon
\xypdfcontpatternoff 2174 \newcommand*\xypdfcontpatternon{\xP@conttrue}
2175 \newcommand*\xypdfcontpatternoff{\xP@contfalse}
2176 \xP@conttrue
2177 \</basic>

```

9.14 Color

```

2178 \< *color>
2179 \xycatcodes

\xypdf-co@loaded
2180 \expandafter\let\csname xypdf-co@loaded\endcsname\@empty

\xP@colorname
  \xP@colA 2181 \@ifdefinable\xP@colorname\relax
  \xP@colB 2182 \@ifdefinable\xP@colA\relax
  \xP@colC 2183 \@ifdefinable\xP@colB\relax
  \xP@colD 2184 \@ifdefinable\xP@colC\relax
2185 \@ifdefinable\xP@colD\relax

\newxycolor
\xP@newxycolor 2186 \xP@hook{color}{newxycolor}
2187 \newcommand*\xP@newxycolor[2]{%
2188   \def\xP@colorname{#1}%
2189   \xP@parsecolor#2 @%
2190 }

Notwithstanding the policy followed by the rest of the package that the xypdf macros are
enabled at \begin{document}, we overwrite the \newxycolor macro already here with the
new version. This makes it possible to define colors with \newxycolor in the preamble,
even if the rest of the driver is activated later.

2191 \xP@color@on

\xP@parsecolor
2192 \@ifdefinable\xP@parsecolor\relax
2193 \def\xP@parsecolor#1 #2 #3@{%
2194   \def\xP@colA{#1}%
2195   \def\xP@colB{#2}%
2196   \ifx\xP@colB\xP@gray
2197     \xP@newcolor\xP@colorname\xP@colA\xP@gray\xnewxycolor
2198   \else
2199     \xP@parsecolor@#3 @%
2200   \fi
2201 }

\xP@parsecolor@
2202 \@ifdefinable\xP@parsecolor@\relax
2203 \def\xP@parsecolor@#1 #2 #3 #4@{%

```

```

2204 \def\xP@colC{#1}%
2205 \def\xP@colD{#2}%
2206 \ifx\xP@colD\xP@rgb
2207   \xP@newcolor\xP@colorname{\xP@colA,\xP@colB,\xP@colC}\xP@rgb\newxycolor
2208 \else
2209   \def\@tempa{#3}%
2210   \ifx\@tempa\xP@cmyk
2211     \xP@newcolor\xP@colorname{\xP@colA,\xP@colB,\xP@colC,\xP@colD}{cmyk}%
2212     \newxycolor
2213   \else
2214     \PackageError{xypdf}{Syntax error in \string\newxycolor}{}%
2215   \fi
2216 \fi
2217 }

\xP@gray
\xP@rgb 2218 \newcommand*\xP@gray{gray}
\xP@cmyk 2219 \newcommand*\xP@rgb{rgb}
2220 \newcommand*\xP@cmyk{cmyk}

\OBJECT@shape
\xP@OBJECT@shape 2221 \xP@hook{color}{OBJECT@shape}
2222 \newcommand*\xP@OBJECT@shape[1]{\DN@{shape [1]}%
2223   \expandafter\let\expandafter\nextii@\csname\codeof\next@\endcsname
2224   \ifx\nextii@\relax\DN@{style [1]}%
2225     \expandafter\let\expandafter\nextii@\csname\codeof\next@\endcsname
2226     \ifx\nextii@\relax\DN@{\xP@checkcolor{#1}}%
2227     \else\DN@{\nextii@\xyFN@OBJECT@}%
2228     \fi
2229   \else\expandafter\addtotoks@\expandafter{\nextii@}%
2230   \DN@{\xyFN@OBJECT@}%
2231   \fi \next@}

\xP@checkcolor
2232 \newcommand*\xP@checkcolor[1]{%
2233   \@ifundefined{\string\color@detokenize{#1}}%
2234   {\OBJECT@shapei[1]}%
2235   {%
2236     \xP@append\toks@{\noexpand\xP@color{\detokenize{#1}}}%
2237     \xyFN@OBJECT@
2238   }%
2239 }

\xP@color
2240 \newcommand*\xP@color[1]{%
2241   \def\preStyle@{\addtostyletoks@{\bgroup\xP@protectedcolor#1}}%
2242   \def\postStyle@{\addtostyletoks@{\egroup}}%
2243   \modXYstyle@
2244 }

\xP@protectedcolor Somehow, \protect\color is not enough, so we use  $\varepsilon$ -TeX's way of \protected macro
definitions.
2245 \@ifdefinable\xP@protectedcolor\relax
2246 \protected\def\xP@protectedcolor{%
2247   \@ifundefined{color}\xP@pdfcolor\color
2248 }

```

\xP@pdfcolor

```
2249 \@ifdefinable\xP@pdfcolor\relax
2250 \def\xP@pdfcolor[#1]#2{%
2251   \edef\@tempa{#1}%
2252   \ifx\@tempa\xP@gray
2253     \DN@{\xP@graycolor{#2}}%
2254   \else\ifx\@tempa\xP@rgb
2255     \DN@{\xP@rgbcolor#2}%
2256   \else\ifx\@tempa\xP@cmyk
2257     \DN@{\xP@cmykcolor#2}%
2258   \fi\fi\fi
2259   \aftergroup\xP@resetcolor
2260   \next@
2261 }%
```

\xP@graycolor

```
2262 \newcommand*\xP@graycolor[1]{\xP@setcolor{#1}gG}%
```

\xP@rgbcolor

```
2263 \@ifdefinable\xP@rgbcolor\relax
2264 \def\xP@rgbcolor#1,#2,#3@{\xP@setcolor{#1 #2 #3}{rg}{RG}}
```

\xP@cmykcolor

```
2265 \@ifdefinable\xP@cmykcolor\relax
2266 \def\xP@cmykcolor#1,#2,#3,#4@{\xP@setcolor{#1 #2 #3 #4}{kK}}
```

\xP@newcolor

```
2267 \newcommand*\xP@newcolor[4]{%
2268   \expandafter\let\expandafter\next@\csname shape [1]\endcsname
2269   \ifx\next@\relax
2270     \@ifundefined{string\color@#1}\relax
2271     {\xP@warning{xypdf}{The color ‘#1’ is overridden by %
2272       \string#4}}%
2273     \edef\next@{\noexpand\newxystyle{#1}{\noexpand\xP@unnamedcolor{#2}{#3}}}%
2274   \else
2275     \DN@{}%
2276   \fi
2277   \next@\relax}
```

\xP@unnamedcolor

```
2278 \newcommand*\xP@unnamedcolor[2]{\xP@color{[2]{#1}}}
```

\xP@definecrayolacolor

```
2279 \newcommand\xP@definecrayolacolor[2]{%
2280   \xP@newcolor{#1}{#2}{cmyk}\UseCrayolaColors}%
```

\xP@installCrayolaColors

```
2281 \newcommand*\xP@installCrayolaColors{%
2282   \xP@definecrayolacolor{GreenYellow}{.15,0,.69,0}%
2283   \xP@definecrayolacolor{Yellow}{0,0,1,0}%
2284   \xP@definecrayolacolor{Goldenrod}{0,.1,.84,0}%
2285   \xP@definecrayolacolor{Dandelion}{0,.29,.84,0}%
2286   \xP@definecrayolacolor{Apricot}{0,.32,.52,0}%
2287   \xP@definecrayolacolor{Peach}{0,.5,.7,0}%
2288   \xP@definecrayolacolor{Melon}{0,.46,.5,0}%
```

```

2289 \xP@definecrayolacolor{YellowOrange}{0,.42,1,0}%
2290 \xP@definecrayolacolor{Orange}{0,.61,.87,0}%
2291 \xP@definecrayolacolor{BurntOrange}{0,.51,1,0}%
2292 \xP@definecrayolacolor{Bittersweet}{0,.75,1,.24}%
2293 \xP@definecrayolacolor{RedOrange}{0,.77,.87,0}%
2294 \xP@definecrayolacolor{Mahogany}{0,.85,.87,.35}%
2295 \xP@definecrayolacolor{Maroon}{0,.87,.68,.32}%
2296 \xP@definecrayolacolor{BrickRed}{0,.89,.94,.28}%
2297 \xP@definecrayolacolor{Red}{0,1,1,0}%
2298 \xP@definecrayolacolor{OrangeRed}{0,1,.5,0}%
2299 \xP@definecrayolacolor{RubineRed}{0,1,.13,0}%
2300 \xP@definecrayolacolor{WildStrawberry}{0,.96,.39,0}%
2301 \xP@definecrayolacolor{Salmon}{0,.53,.38,0}%
2302 \xP@definecrayolacolor{CarnationPink}{0,.63,0,0}%
2303 \xP@definecrayolacolor{Magenta}{0,1,0,0}%
2304 \xP@definecrayolacolor{VioletRed}{0,.81,0,0}%
2305 \xP@definecrayolacolor{Rhodamine}{0,.82,0,0}%
2306 \xP@definecrayolacolor{Mulberry}{.34,.9,0,.02}%
2307 \xP@definecrayolacolor{RedViolet}{.07,.9,0,.34}%
2308 \xP@definecrayolacolor{Fuchsia}{.47,.91,0,.08}%
2309 \xP@definecrayolacolor{Lavender}{0,.48,0,0}%
2310 \xP@definecrayolacolor{Thistle}{.12,.59,0,0}%
2311 \xP@definecrayolacolor{Orchid}{.32,.64,0,0}%
2312 \xP@definecrayolacolor{DarkOrchid}{.4,.8,.2,0}%
2313 \xP@definecrayolacolor{Purple}{.45,.86,0,0}%
2314 \xP@definecrayolacolor{Plum}{.5,1,0,0}%
2315 \xP@definecrayolacolor{Violet}{.79,.88,0,0}%
2316 \xP@definecrayolacolor{RoyalPurple}{.75,.9,0,0}%
2317 \xP@definecrayolacolor{BlueViolet}{.86,.91,0,.04}%
2318 \xP@definecrayolacolor{Periwinkle}{.57,.55,0,0}%
2319 \xP@definecrayolacolor{CadetBlue}{.62,.57,.23,0}%
2320 \xP@definecrayolacolor{CornflowerBlue}{.65,.13,0,0}%
2321 \xP@definecrayolacolor{MidnightBlue}{.98,.13,0,.43}%
2322 \xP@definecrayolacolor{NavyBlue}{.94,.54,0,0}%
2323 \xP@definecrayolacolor{RoyalBlue}{1,.5,0,0}%
2324 \xP@definecrayolacolor{Blue}{1,1,0,0}%
2325 \xP@definecrayolacolor{Cerulean}{.94,.11,0,0}%
2326 \xP@definecrayolacolor{Cyan}{1,0,0,0}%
2327 \xP@definecrayolacolor{ProcessBlue}{.96,0,0,0}%
2328 \xP@definecrayolacolor{SkyBlue}{.62,0,.12,0}%
2329 \xP@definecrayolacolor{Turquoise}{.85,0,.2,0}%
2330 \xP@definecrayolacolor{TealBlue}{.86,0,.34,.02}%
2331 \xP@definecrayolacolor{Aquamarine}{.82,0,.3,0}%
2332 \xP@definecrayolacolor{BlueGreen}{.85,0,.33,0}%
2333 \xP@definecrayolacolor{Emerald}{1,0,.5,0}%
2334 \xP@definecrayolacolor{JungleGreen}{.99,0,.52,0}%
2335 \xP@definecrayolacolor{SeaGreen}{.69,0,.5,0}%
2336 \xP@definecrayolacolor{Green}{1,0,1,0}%
2337 \xP@definecrayolacolor{ForestGreen}{.91,0,.88,.12}%
2338 \xP@definecrayolacolor{PineGreen}{.92,0,.59,.25}%
2339 \xP@definecrayolacolor{LimeGreen}{.5,0,1,0}%
2340 \xP@definecrayolacolor{YellowGreen}{.44,0,.74,0}%
2341 \xP@definecrayolacolor{SpringGreen}{.26,0,.76,0}%
2342 \xP@definecrayolacolor{OliveGreen}{.64,0,.95,.4}%
2343 \xP@definecrayolacolor{RawSienna}{0,.72,1,.45}%
2344 \xP@definecrayolacolor{Sepia}{0,.83,1,.7}%

```

```

2345 \xP@definecrayolacolor{Brown}{0,.81,1,.6}%
2346 \xP@definecrayolacolor{Tan}{.14,.42,.56,0}%
2347 \xP@definecrayolacolor{Gray}{0,0,0,.5}%
2348 \xP@definecrayolacolor{Black}{0,0,0,1}%
2349 \xP@definecrayolacolor{White}{0,0,0,0}%
2350 }
2351 \xywithoption{crayon}{%
2352 \xP@installCrayolaColors
2353 \renewcommand*\installCrayolaColors@{}%
2354 }

```

End of the section for Xy-pic's "color" option.

```

2355 \xyendinput
2356 </color>

```

9.15 Frames

```

2357 <{*frame}
2358 \xyrequire{curve}%
2359 \xycatcodes

```

\xypdf-fr@loaded

```

2360 \expandafter\let\csname xypdf-fr@loaded\endcsname\@empty

```

\xP@framedrop

```

2361 \newcommand*\xP@framedrop[1]{%
2362 \expandafter\frmDrop@\expandafter{%
2363 \expandafter\def\expandafter\prevEdge@@\expandafter{\prevEdge@@}%
2364 \setboxz@h{#1\frmradius@@}%
2365 \styledboxz@}%
2366 }

```

\frm{-}

```

\xP@frm{-} 2367 \xP@hook{frame}{\frm{-}}
2368 \expandafter\newcommand\expandafter*\csname xP@frm{-}\endcsname{%
2369 \xP@framedrop\xP@solidframe
2370 }

```

\xP@solidframe

```

2371 \newcommand*\xP@solidframe[1]{%
2372 \R@#1\relax
2373 \xP@setsolidpat
2374 \let\xP@fillorstroke\xP@stroke
2375 \xP@frameifnotzero\xP@oval
2376 }

```

\xP@ifzerosize

```

2377 \newcommand*\xP@ifzerosize[2]{%
2378 \@tempswafalse
2379 \ifdim\L@c=\z@\ifdim\R@c=\z@\ifdim\U@c=\z@\ifdim\D@c=\z@
2380 \@tempswatrue
2381 \fi\fi\fi\fi
2382 \if@tempswa#1\else#2\fi
2383 }

```

\xP@frameifnotzero

```

2384 \newcommand*\xP@frameifnotzero[1]{%
2385   \setboxz@h{%
2386     \hskip\X@c\raise\Y@c\hbox{\xP@ifzerosize{}\{#1\}}%
2387   }%
2388   \wd\z@\z@\ht\z@\z@\dp\z@\z@
2389   \boxz@
2390 }

```

\xP@oval

```

2391 \newcommand*\xP@oval{%
2392   \hskip-\L@c
2393   \lower\D@c\hbox{%
2394     \dimen@ \dimexpr\L@c+\R@c\relax
2395     \dimen@ii \dimexpr\U@c+\D@c\relax
2396     \R@\xP@min\R@{.5\dimen@}%
2397     \R@\xP@min\R@{.5\dimen@ii}%

```

Circumference. $696621973/405764219 \approx 8 - 2\pi$

```

2398   \@tempdimb \dimexpr2\dimen@+2\dimen@ii-\R@*696621973/405764219\relax
2399   \ifdim\R@=\z@

```

Sharp corners: draw a rectangle.

```

2400     \xP@fillorstroke{0 0 \xP@coor\dimen@\dimen@ii re}%
2401     \else

```

Rounded corners.

```

2402     \def\@tempa{*119763188/267309217}%
2403     \xP@fillorstroke{%
2404       \xP@dim\R@0 m \xP@dim{\R@\@tempa}0 0 \xP@dim{\R@\@tempa}%
2405       0 \xP@dim\R@c %
2406       \ifdim2\R@=\dimen@ii\else
2407         0 \xP@dim{\dimen@ii-\R@}1 %
2408       \fi
2409       0 \xP@dim{\dimen@ii-\R@\@tempa}\xP@coor{\R@\@tempa}\dimen@ii
2410       \xP@coor\R@\dimen@ii c %
2411       \ifdim2\R@=\dimen@ii\else
2412         \xP@coor{\dimen@-\R@}\dimen@ii 1 %
2413       \fi
2414       \xP@coor{\dimen@-\R@\@tempa}\dimen@ii
2415       \xP@coor\dimen@{\dimen@ii-\R@\@tempa}\xP@coor\dimen@{\dimen@ii-\R@} c %
2416       \ifdim2\R@=\dimen@ii\else
2417         \xP@coor\dimen@\R@ 1 %
2418       \fi
2419       \xP@coor\dimen@{\R@\@tempa}\xP@dim{\dimen@-\R@\@tempa}0 %
2420       \xP@dim{\dimen@-\R@}0 c h%
2421     }%
2422   \fi
2423 }%
2424 }

```

\frm[o]{-}

\xP@frm[o]{-}

```

2425 \xP@hook{frame}{frm[o]{-}}
2426 \expandafter\newcommand\expandafter*\csname xP@frm[o]{-}\endcsname{%
2427   \xP@framedrop{\xP@ellipseframe\xP@setsolidpat}%
2428 }

```


\xP@ellipseframe

```

2429 \newcommand*\xP@ellipseframe[2]{%
2430   \xP@getradii{#2}%
2431   \xP@ifzerosize{}{%
2432     \def\xP@fillorstroke{#1\xP@stroke}%
2433     \setboxz@h{\hskip\X@c\raise\Y@c\hbox{\xP@framedellipse}}%
2434     \wd\z@=\z@ht\z@=\z@dp\z@=\z@
2435     \boxz@
2436   }%
2437 }

```

\frm{.}
\xP@frm{.}

```

2438 \xP@hook{frame}{frm{.}}
2439 \expandafter\newcommand\expandafter*\csname xP@frm{.}\endcsname{%
2440   \xP@framedrop\xP@rectframedotted
2441 }

```

\xP@rectframedotted

```

2442 \newcommand*\xP@rectframedotted[1]{%
2443   \R@#1\relax
2444   \xP@frameifnotzero{%
2445     \ifdim\R@=\z@
2446       \xP@dottedirect
2447     \else
2448       \xP@dottedoval
2449     \fi
2450   }%
2451 }

```

\xP@dottedirect Make sure that there is a dot in every corner of the rectangle.

```

2452 \newcommand*\xP@dottedirect{%
2453   \hskip-\L@c
2454   \lower\D@c\hbox{%
2455     \dimen@ii\dimexpr\U@c+\D@c\relax
2456     \@tempdimc\dimexpr\xP@preclw/-2\relax
2457     \@tempdimb\dimexpr\L@c+\R@c+\xP@preclw\relax
2458     \xP@contfalse
2459     \xP@setdottedpat

```

Draw the horizontal lines $\frac{1}{2}$ whitespace longer to eliminate inaccuracies.

```

2460     \dimen@\dimexpr\@tempdimb+\@tempdima/2+\@tempdimc\relax
2461     \xP@stroke{\xP@dim\@tempdimc 0 m \xP@dim\dimen@ 0 1 %
2462       \xP@coor\@tempdimc\dimen@ii m \xP@coor\dimen@\dimen@ii 1}%

```

Ensure pattern continuation.

```

2463     \let\xP@testcont\xP@alwaysconttrue
2464     \@tempdimb\dimen@ii
2465     \xP@setdottedpat
2466     \dimen@\dimexpr\L@c+\R@c\relax
2467     \advance\dimen@ii\dimexpr\@tempdimc-\@tempdima/2\relax
2468     \multiply\@tempdimc\m@ne

```

Draw the vertical lines $\frac{1}{2}$ whitespace longer.

```

2469     \xP@stroke{0 \xP@dim\@tempdimc m 0 \xP@dim\dimen@ii 1 %
2470       \xP@coor\dimen@\@tempdimc m \xP@coor\dimen@\dimen@ii 1}%
2471   }%
2472 }

```

```

\xP@dottedoval
2473 \newcommand*\xP@dottedoval{%
2474   \def\xP@fillorstroke{\xP@setcldottedpat\xP@stroke}%
2475   \xP@oval
2476 }

\frm[o]{.}
\xP@frm[o]{.} 2477 \xP@hook{frame}{frm[o]{.}}
2478 \expandafter\newcommand\expandafter*\csname xP@frm[o]{.}\endcsname{%
2479   \xP@framedrop{\xP@ellipseframe\xP@setcldottedpat}%
2480 }

\frm{--}
\xP@frm{--} 2481 \xP@hook{frame}{frm{--}}
2482 \expandafter\newcommand\expandafter*\csname xP@frm{--}\endcsname{%
2483   \xP@framedrop\xP@rectframedashed
2484 }

\xP@rectframedashed
2485 \newcommand*\xP@rectframedashed[1]{%
2486   \R@#1\relax
2487   \xP@frameifnotzero{%
2488     \ifdim\R@=\z@
2489       \xP@dashedrect
2490     \else
2491       \xP@dashedoval
2492     \fi
2493   }%
2494 }

\xP@dashedrect
2495 \newcommand*\xP@dashedrect{%
2496   \hskip-\L@c
2497   \lower\D@c\hbox{%
2498     \dimen@ \dimexpr\L@c+\R@c\relax
2499     \dimen@ii \dimexpr\U@c+\D@c\relax
2500     \@tempdimb\dimen@
2501     \xP@contfalse
2502     \xP@setdashpat
2503     \xP@stroke{0 0 m \xP@dim\dimen@0 1 %
2504       0 \xP@dim\dimen@ii m \xP@coor\dimen@\dimen@ii l}%
2505     \@tempdimb\dimen@ii
2506     \xP@setdashpat
2507     \xP@stroke{0 0 m 0 \xP@dim\dimen@ii 1 %
2508       \xP@dim\dimen@0 m \xP@coor\dimen@\dimen@ii l}%
2509   }%
2510 }

\xP@dashedoval
2511 \newcommand*\xP@dashedoval{%
2512   \def\xP@fillorstroke{\xP@setcldashpat\xP@stroke}%
2513   \xP@oval
2514 }

\frm[o]{--}
\xP@frm[o]{--}

```

```

2515 \xP@hook{frame}{frm[o]{--}}
2516 \expandafter\newcommand\expandafter*\csname xP@frm[o]{--}\endcsname{%
2517 \xP@framedrop{\xP@ellipseframe\xP@setcldashpat}%
2518 }

\frm{,}
\xP@frm{,} 2519 \xP@hook{frame}{frm{,}}
2520 \expandafter\newcommand\expandafter*\csname xP@frm{,}\endcsname{%
2521 \xP@framedrop\xP@frameshadow
2522 }

\xP@frameshadow
2523 \newcommand*\xP@frameshadow[1]{%
2524 \R@#1\relax
2525 \ifdim\R@=\z@\R@1.2pt\relax\fi
2526 \xP@frameifnotzero\xP@shadow
2527 }

\xP@shadow
2528 \newcommand*\xP@shadow{%
2529 \hskip\dimexpr\R@+\R@/2\relax
2530 \lower\dimexpr\D@+\R@/2\relax\hbox{%
2531 \def\xP@pattern{0 J 0 j [] 0 d}%
2532 \edef\xP@lw{\xP@dim\R@}%
2533 \xP@stroke{\xP@dim{\R@/2-\L@c-\R@c} 0 m 0 0 1 0 \xP@dim{\D@+\U@c-\R@/2}l}%
2534 }%
2535 }

\frm{o-}
\xP@frm{o-} 2536 \xP@hook{frame}{frm{o-}}
2537 \expandafter\newcommand\expandafter*\csname xP@frm{o-}\endcsname{%
2538 \xP@framedrop\xP@roundedrectframe
2539 }

\xP@roundedrectframe
2540 \newcommand*\xP@roundedrectframe[1]{%
2541 \R@#1\relax
2542 \ifdim\R@=\z@\R@\xydashl@ \relax \fi
2543 \xP@frameifnotzero\xP@roundedrectangle
2544 }

\xP@roundedrectangle
2545 \newcommand*\xP@roundedrectangle{%
2546 \dimen@ \dimexpr\L@c+\R@c\relax
2547 \dimen@ii \dimexpr\U@c+\D@c\relax
2548 \R@\xP@min\R@{.5\dimen@}%
2549 \R@\xP@min\R@{.5\dimen@ii}%
2550 \hskip-\L@c
2551 \lower\D@c\hbox{%

Rounded corners
2552 \@tempdimb \dimexpr2\dimen@+2\dimen@ii-\R@*696621973/405764219\relax
2553 \def\@tempa{*119763188/267309217}%
2554 \xP@setsolidpat
2555 \xP@stroke{%
2556 \xP@dim\R@ 0 m \xP@dim{\R@\@tempa} 0 0 \xP@dim{\R@\@tempa}%

```

```

2557      0 \xP@dim\R@ c %
2558      \ifdim2\R@=\dimen@ii\else
2559      0 \xP@dim{\dimen@ii-\R@}m %
2560      \fi
2561      0 \xP@dim{\dimen@ii-\R@\@tempa}\xP@coor{\R@\@tempa}\dimen@ii
2562      \xP@coor\R@\dimen@ii c %
2563      \ifdim2\R@=\dimen@\else
2564      \xP@coor{\dimen@-\R@}\dimen@ii m %
2565      \fi
2566      \xP@coor{\dimen@-\R@\@tempa}\dimen@ii
2567      \xP@coor\dimen@{\dimen@ii-\R@\@tempa}\xP@coor\dimen@{\dimen@ii-\R@} c %
2568      \ifdim2\R@=\dimen@ii\else
2569      \xP@coor\dimen@\R@ m %
2570      \fi
2571      \xP@coor\dimen@\R@\@tempa}\xP@dim{\dimen@-\R@\@tempa}0 %
2572      \xP@dim{\dimen@-\R@}0 c%
2573      }%

```

Upper and lower horizontal dashes.

```

2574      \xP@contfalse
2575      \@tempdimb\dimexpr\l@c+\R@c-2\R@\relax
2576      \ifdim\@tempdimb<\z@\@tempdimb\z@\fi
2577      \xP@setdashpat
2578      \ifdim\@tempdima>\z@
2579      \dimen@\dimexpr\@tempdimb+\R@-\@tempdima/2\relax
2580      \dimen@ii\dimexpr\U@c+\D@c\relax
2581      \xP@stroke{%
2582      \xP@dim{\R@+\@tempdima}0 m \xP@dim\dimen@ 0 l %
2583      \xP@coor{\R@+\@tempdima}\dimen@ii m \xP@coor\dimen@\dimen@ii l%
2584      }%
2585      \fi

```

Left and right vertical dashes.

```

2586      \@tempdimb\dimexpr\U@c+\D@c-2\R@\relax
2587      \ifdim\@tempdimb<\z@\@tempdimb\z@\fi
2588      \xP@setdashpat
2589      \ifdim\@tempdima>\z@
2590      \dimen@\dimexpr\l@c+\R@c\relax
2591      \dimen@ii\dimexpr\@tempdimb+\R@-\@tempdima/2\relax
2592      \xP@stroke{%
2593      0 \xP@dim{\R@+\@tempdima}m 0 \xP@dim\dimen@ii l %
2594      \xP@coor\dimen@{\R@+\@tempdima}m \xP@coor\dimen@\dimen@ii l%
2595      }%
2596      \fi
2597      }%
2598 }

```

```

\frm{=}
\xP@frm{=} 2599 \xP@hook{frame}{frm{=}}
2600 \expandafter\newcommand\expandafter*\csname xP@frm{=}\endcsname{%
2601 \xP@framedrop\xP@dsframe
2602 }

\xP@dsframe
2603 \newcommand*\xP@dsframe[1]{%
2604 \R@#1\relax
2605 \xP@frameifnotzero\xP@dsoval
2606 }

```

```

\XP@dsoval
2607 \newcommand*\XP@dsoval{%
2608   \dimen@ \dimexpr (\L@c+\R@c)/2\relax
2609   \ifdim \dimen@ < \xydashh@ \dimen@ \xydashh@ \fi
2610   \dimen@ii \dimexpr (\U@c+\D@c)/2\relax
2611   \ifdim \dimen@ii < \xydashh@ \dimen@ii \xydashh@ \fi
2612   \R@\XP@min\R@\dimen@
2613   \R@\XP@min\R@\dimen@ii
2614   \XP@setsolidpat
2615   \let \XP@fillorstroke \XP@stroke
2616   \XP@oval
2617   \hskip \L@c
2618   \advance \L@c - \xydashh@
2619   \advance \R@c - \xydashh@
2620   \advance \U@c - \xydashh@
2621   \advance \D@c - \xydashh@
2622   \advance \R@ - \xydashh@
2623   \ifdim \R@ < \z@ \R@ \z@ \fi
2624   \XP@oval
2625 }

\frm[o]{=}
\XP@frm[o]{=} 2626 \XP@hook{frame}{\frm[o]{=}}
2627 \expandafter \newcommand \expandafter * \csname \XP@frm[o]{=}\endcsname {%
2628   \XP@framedrop \XP@dsellframe
2629 }

\frm{ee}
\XP@frm{ee} 2630 \XP@hook{frame}{\frm{ee}}
2631 \expandafter \newcommand \expandafter * \csname \XP@frm{ee}\endcsname {%
2632   \XP@framedrop \XP@dsellframe
2633 }

\XP@dsellframe
2634 \newcommand*\XP@dsellframe[1]{%
2635   \XP@getradii{#1}%
2636   \XP@frameifnotzero \XP@dsellipse
2637 }

\XP@temppath
2638 \@ifdefinable \XP@temppath \relax

\XP@dsellipse
2639 \newcommand*\XP@dsellipse{%
2640   \hskip \dimexpr (\R@c-\L@c)/2\relax
2641   \lower \dimexpr (\D@c-\U@c)/2\relax
2642   \hbox{%
2643     % Inner ellipse: true ellipse
2644     \advance \A@ - \xydashh@
2645     \advance \B@ - \xydashh@
2646     \ifdim \A@ < \z@ \A@ \z@ \fi
2647     \ifdim \B@ < \z@ \B@ \z@ \fi
2648     \def \XP@fillorstroke {\edef \XP@temppath}%
2649     \XP@ellipse@
2650     % Outer curve: offset ellipse
2651     \XP@inibigdim

```

```

2652 \let\@tempa\xydashh@
2653 \xP@offsetellipse
2654 \xP@setsolidpat
2655 \xP@stroke{\xP@temppath\space\the\@temptokena}%
2656 }%
2657 }

\xP@offsetellipse TeX grouping: Not necessary, it's in an \hbox anyway.
2658 \newcommand*\xP@offsetellipse{%
2659 \xP@movetotrue
2660 \@temptokena{}%
2661 \xP@offsetelliptseg\A@\z@\A@{\B@*173517671/654249180}%
2662 {\A@*554561898/619869377}{\B@*34221476/65864945}%
2663 {\A@*543339720/768398401}{\B@*543339720/768398401}%
2664 \xP@offsetelliptseg{\A@*543339720/768398401}{\B@*543339720/768398401}%
2665 {\A@*34221476/65864945}{\B@*554561898/619869377}%
2666 {\A@*173517671/654249180}\B@\z@\B@
2667 \xP@mirrorpath
2668 }

\xP@mirrorpath
2669 \newcommand*\xP@mirrorpath{%
2670 \edef\@tempa{\the\@temptokena\relax\space\space\space\space}%
2671 \let\@tempb\@empty
2672 \let\@tempc\@empty
2673 \expandafter\xP@mirrorpath@\@tempa
2674 }

\xP@mirrorpath@
2675 \@ifdefinable\xP@mirrorpath@\relax
2676 \def\xP@mirrorpath@#1 #2 #3 #4 #5 #6 #7 {%
2677 \ifx\relax#4%
2678 \xP@append\@temptokena{\@tempb\xP@minus#1 \xP@minus#2 #3 \@tempc h}%
2679 \else
2680 \edef\@tempb{\xP@minus#6 #7 \xP@minus#4 #5 \xP@minus#1 #2 c \@tempb%
2681 \if#3m\else\xP@minus#1 \xP@minus#2 #3 \fi%
2682 \xP@minus#4 \xP@minus#5 \xP@minus#6 \xP@minus#7 }%
2683 \edef\@tempc{#6 \xP@minus#7 #4 \xP@minus#5 #1 \xP@minus#2 c \@tempc}%
2684 \expandafter\xP@mirrorpath@
2685 \fi
2686 }

\xP@minus
2687 \@ifdefinable\xP@minus\relax
2688 \def\xP@minus#1 {\if-#1 \else\ifdim\dimexpr#1pt\relax=\z@\else-\fi#1 \fi}

\xP@insertbefore
2689 \newcommand*\xP@insertbefore[2]{\edef\@tempa{#1{#2\the#1}}\expandafter}\@tempa}

\xP@offsetelliptseg
2690 \newcommand*\xP@offsetelliptseg[8]{%
2691 \X@p\dimexpr#1\relax
2692 \Y@p\dimexpr#2\relax
2693 \L@c\dimexpr#3\relax
2694 \U@c\dimexpr#4\relax
2695 \R@c\dimexpr#5\relax

```

```

2696 \D@c\dimexpr#6\relax
2697 \X@c\dimexpr#7\relax
2698 \Y@c\dimexpr#8\relax
2699 \xP@savepts
2700 \xP@a\z@
2701 \xP@c\xP@bigdim
2702 \xP@paintsolid@
2703 }

\xP@getradii
2704 \newcommand*\xP@getradii[1]{%
2705 \edef\@tempa{#1}%
2706 \expandafter\xP@getradii@\@tempa,\maxdimen,%
2707 }

\xP@getradii@
2708 \@ifdefinable\xP@getradii@\relax
2709 \def\xP@getradii@#1,#2,#3{%
2710 \A@#1\relax
2711 \B@#2\relax
2712 \ifdim\B@=\maxdimen
2713 \A@\dimexpr(\L@c+\R@c)/2\relax
2714 \B@\dimexpr(\U@c+\D@c)/2\relax
2715 \fi
2716 }

\frm{o}
\xP@frm{o} 2717 \xP@hook{frame}{\frm{o}}
2718 \expandafter\newcommand\expandafter*\csname xP@frm{o}\endcsname{%
2719 \xP@framedrop{\xP@circleframe\xP@setsolidpat}%
2720 }

\frm{-o}
\xP@frm{-o} 2721 \xP@hook{frame}{\frm{-o}}
2722 \expandafter\newcommand\expandafter*\csname xP@frm{-o}\endcsname{%
2723 \xP@framedrop{\xP@circleframe\xP@setcldashpat}%
2724 }

\frm{.o}
\xP@frm{.o} 2725 \xP@hook{frame}{\frm{.o}}
2726 \expandafter\newcommand\expandafter*\csname xP@frm{.o}\endcsname{%
2727 \xP@framedrop{\xP@circleframe\xP@setcldottedpat}%
2728 }

\xP@circleframe
2729 \newcommand*\xP@circleframe[2]{%
2730 \R@#2\relax
2731 \def\xP@fillorstroke{#1\xP@stroke}%
2732 \xP@ifzerosize{%
2733 \ifdim\R@>\z@
2734 \xP@circleframe@
2735 \fi
2736 }{%
2737 \ifdim\R@=\z@
2738 \A@\dimexpr(\L@c+\R@c)/2\relax
2739 \B@\dimexpr(\U@c+\D@c)/2\relax

```

```

2740      \R@\xP@max\A@\B@
2741      \fi
2742      \xP@circleframe@
2743      }%
2744  }

\xP@circleframe@
2745  \newcommand*\xP@circleframe@{%
2746    \setboxz@h{\hskip\X@c\raise\Y@c\hbox{\xP@circle}}%
2747    \wd\z@\z@\ht\z@\z@\dp\z@\z@
2748    \boxz@
2749  }

  \frm{e}
\xP@frm{e} 2750 \xP@hook{frame}{\frm{e}}
2751 \expandafter\newcommand\expandafter*\csname xP@frm{e}\endcsname{%
2752   \xP@framedrop{\xP@ellipseframe\xP@setsolidpat}%
2753 }

  \frm{-e}
\xP@frm{-e} 2754 \xP@hook{frame}{\frm{-e}}
2755 \expandafter\newcommand\expandafter*\csname xP@frm{-e}\endcsname{%
2756   \xP@framedrop{\xP@ellipseframe\xP@setcldashpat}%
2757 }

  \frm{.e}
\xP@frm{.e} 2758 \xP@hook{frame}{\frm{.e}}
2759 \expandafter\newcommand\expandafter*\csname xP@frm{.e}\endcsname{%
2760   \xP@framedrop{\xP@ellipseframe\xP@setcldottedpat}%
2761 }

  \frm2{.e}
\xP@frm2{.e} 2762 \xP@hook{frame}{\frm2{.e}}
2763 \expandafter\newcommand\expandafter*\csname xP@frm2{.e}\endcsname{%
2764   \xP@framedrop\xP@dsdottedellframe
2765 }

\xP@dsdottedellframe
2766 \newcommand*\xP@dsdottedellframe[1]{%
2767   \xP@getradii{#1}%
2768   \xP@frameifnotzero\xP@dsdottedellipse
2769 }

\xP@dsdottedellipse
2770 \newcommand*\xP@dsdottedellipse{%
2771   \hskip\dimexpr(\R@c-\L@c)/2\relax
2772   \lower\dimexpr(\D@c-\U@c)/2\relax
2773   \hbox{%
Intermediate ellipse: true ellipse
2774     \@tempdima.5\xydashh@\relax
2775     \advance\A@-\@tempdima
2776     \advance\B@-\@tempdima
2777     \ifdim\A@<\@tempdima\A@\@tempdima\fi
2778     \ifdim\B@<\@tempdima\B@\@tempdima\fi
2779     \let\xP@normalmult\@ne

```



```

2780 \xP@specialellipse{\xP@splinemultdotted\xP@doublestroke}%
2781 }%
2782 }

```

\xP@specialellipse

```

2783 \newcommand*\xP@specialellipse[1]{%
2784 \def\@tempa{*147546029/267309217}%
2785 \X@p\A@
2786 \Y@p\z@
2787 \L@c\A@
2788 \U@c\dimexpr\B@\@tempa\relax
2789 \R@c\dimexpr\A@\@tempa\relax
2790 \D@c\B@
2791 \X@c\z@
2792 \Y@c\B@
2793 \xP@bezierlength
2794 \let\xP@testcont\xP@alwaysconttrue
2795 #1%
2796 \X@p\z@
2797 \Y@p\B@
2798 \L@c-\R@c
2799 \D@c\U@c
2800 \U@c\B@
2801 \R@c-\A@
2802 \X@c-\A@
2803 \Y@c\z@
2804 #1%
2805 \X@p-\A@
2806 \Y@p\z@
2807 \R@c\L@c
2808 \L@c-\A@
2809 \U@c-\D@c
2810 \D@c-\B@
2811 \X@c\z@
2812 \Y@c-\B@
2813 #1%
2814 \X@p\z@
2815 \Y@p-\B@
2816 \L@c-\R@c
2817 \D@c\U@c
2818 \U@c-\B@
2819 \R@c\A@
2820 \X@c\A@
2821 \Y@c\z@
2822 #1%
2823 }

```

\xP@alwaysconttrue

```

2824 \newcommand*\xP@alwaysconttrue[1]{\xP@splineconttrue}

```

```

\frm2{-e}
\xP@frm2{-e} 2825 \xP@hook{frame}{\frm2{-e}}
2826 \expandafter\newcommand\expandafter*\csname xP@frm2{-e}\endcsname{%
2827 \xP@framedrop\xP@dsdashedframe
2828 }

```

```

\xP@dashedframe
2829 \newcommand*\xP@dashedframe[1]{%
2830   \xP@getradii{#1}%
2831   \xP@frameifnotzero\xP@dashedellipse
2832 }

\xP@dashedellipse
2833 \newcommand*\xP@dashedellipse{%
2834   \hskip\dimexpr(\R@c-\L@c)/2\relax
2835   \lower\dimexpr(\D@c-\U@c)/2\relax
2836   \hbox{%
Inner ellipse: true ellipse
2837     \advance\A@-\xydashh@
2838     \advance\B@-\xydashh@
2839     \ifdim\A@<\z@\A@\z@\fi
2840     \ifdim\B@<\z@\B@\z@\fi
2841     \xP@specialellipse{\xP@splinemulldashed\xP@elldoublestroke}%
2842   }%
2843 }

\xP@elldoublestroke
2844 \newcommand*\xP@elldoublestroke{\z@,\xydashh@}

\xP@fill
2845 \newcommand*\xP@fill[1]{\xP@literal{#1 f}}

\xP@fillstroke
2846 \newcommand*\xP@fillstroke[1]{\xP@literal{\xP@dim{\xP@preclw/2}w 1 j 0 G #1 b}}

\xP@fillorstroke
2847 \newcommand*\xP@fillorstroke{

  \frm{*}
  \xP@frm{*} 2848 \xP@hook{frame}{frm{*}}
2849 \expandafter\newcommand\expandafter*\csname xP@frm{*}\endcsname{%
2850   \xP@framedrop{\let\xP@fillorstroke\xP@fill\xP@framefill}%
2851 }

  \frm{**}
  \xP@frm{**} 2852 \xP@hook{frame}{frm{**}}
2853 \expandafter\newcommand\expandafter*\csname xP@frm{**}\endcsname{%
2854   \xP@framedrop{\let\xP@fillorstroke\xP@fillstroke\xP@framefill}%
2855 }

\xP@framefill
2856 \newcommand*\xP@framefill[1]{%
2857   \R@#1\relax
2858   \xP@setsolidpat
2859   \setboxz\h{%
2860     \hskip\X@c\raise\Y@c\hbox{%

```

It would appear to make more sense to examine the token register `\Edge@c` instead of `\prevEdge@@`. If I (Daniel) had done so, the code `*+<1pc>[o]\frm{*}` had given a filled circle.

This certainly makes sense, but the `dvips` driver lets the frame be drawn in accordance with the object drawn before. Hence, I copied this behavior. E.g., the following code gives a filled ellipse:

```

\xy *[o]{a} *\frm{*} \endxy

2861 \DN@{\rectangleEdge}%
2862 \ifx\next@\prevEdge@@
2863 \DN@{\xP@oval}%
2864 \else
2865 \DN@{\circleEdge}%
2866 \ifx\next@\prevEdge@@
2867 \ifdim\R@=\z@
2868 \DN@{\xP@filledellipse}%
2869 \else
2870 \DN@{\restR@max\xP@circle}%
2871 \fi
2872 \else
2873 \ifdim\R@=\z@
2874 \DN@{\xP@oval}%
2875 \else
2876 \DN@{\xP@circle}%
2877 \fi
2878 \fi
2879 \fi
2880 \next@
2881 }%
2882 }%
2883 \wd\z@\z@\ht\z@\z@\dp\z@\z@
2884 \boxz@
2885 }

\xP@circle
2886 \newcommand*\xP@circle{%
2887 \xP@ellipse\R@\R@
2888 }

\xP@filledellipse
2889 \newcommand*\xP@filledellipse{%
2890 \xP@ellipse{\dimexpr(\L@+\R@)/2\relax}{\dimexpr(\U@+\D@)/2\relax}%
2891 }

\xP@framedellipse
2892 \newcommand*\xP@framedellipse{%
2893 \xP@ellipse\A@B@
2894 }

\xP@ellipse
2895 \newcommand*\xP@ellipse[2]{%
2896 \hskip\dimexpr(\R@-\L@)/2\relax
2897 \lower\dimexpr(\D@-\U@)/2\relax
2898 \hbox{%
2899 \A@#1\relax
2900 \B@#2\relax
2901 \xP@ellipse@
2902 }%
2903 }

```

```

\xP@ellipse@
2904 \newcommand*\xP@ellipse@{%
Perimeter, second segment
2905 \X@p\dimexpr\A@*543339720/768398401\relax
2906 \Y@p\dimexpr\B@*543339720/768398401\relax
2907 \L@c\dimexpr\A@*34221476/65864945\relax
2908 \U@c\dimexpr\B@*554561898/619869377\relax
2909 \R@c\dimexpr\A@*173517671/654249180\relax
2910 \D@c\B@
2911 \X@c\z@
2912 \Y@c\B@
2913 \xP@bezierlength
2914 \@tempdima\@tempdimb

Perimeter, first segment
2915 \X@p\A@
2916 \Y@p\z@
2917 \L@c\A@
2918 \U@c\dimexpr\B@*173517671/654249180\relax
2919 \R@c\dimexpr\A@*554561898/619869377\relax
2920 \D@c\dimexpr\B@*34221476/65864945\relax
2921 \X@c\dimexpr\A@*543339720/768398401\relax
2922 \Y@c\dimexpr\B@*543339720/768398401\relax
2923 \xP@bezierlength
2924 \@tempdimb4\dimexpr\@tempdima+\@tempdimb\relax
2925 \edef\@tempa{%
2926 \xP@dim\A@0 m
2927 \xP@coor\L@c\U@c
2928 \xP@coor\R@c\D@c
2929 \xP@coor\X@c\Y@c c %
2930 \xP@coor{\A@*34221476/65864945}\B@*554561898/619869377}%
2931 \xP@coor{\A@*173517671/654249180}\B@
2932 0 \xP@dim\B@ c }%
2933 \@temptokena\expandafter{\@tempa}%
2934 \xP@mirrorpath
2935 \xP@fillorstroke{\the\@temptokena}%
2936 }

End of the section for Xy-pic's "frame" option.
2937 \xyendinput
2938 </frame>

```

9.16 Line styles

```

2939 <*line>
2940 \xycatcodes
2941 \expandafter\let\csname xypdf-li@loaded\endcsname\@empty
2942 % Dummy file.

```

End of the section for Xy-pic's "line" option.

```

2943 \xyendinput
2944 </line>
2945 <*basic>

```

Finish the package initialization. The `\xywithoption` commands are wrapped into `\next@` so that they cannot change the catcodes for the next `\xywithoption` command.

```

2946 \let\@tempa\@undefined
2947 \let\nextii\@undefined
2948 \DN@{%
2949   \xywithoption{color}{%
2950     \message{Xy-pic pdf driver: 'color' extension support}%
2951     \@ifundefined{xypdf-co@loaded}{\input xypdf-co\relax}{\message{not reloaded}}}%
2952   }%
2953   \xywithoption{curve}{%
2954     \message{Xy-pic pdf driver: 'curve' extension support}%
2955     \@ifundefined{xypdf-cu@loaded}{\input xypdf-cu\relax}{\message{not reloaded}}}%
2956   }%
2957   \xywithoption{frame}{%
2958     \message{Xy-pic pdf driver: 'frame' extension support}%
2959     \@ifundefined{xypdf-fr@loaded}{\input xypdf-fr\relax}{\message{not reloaded}}}%
2960   }%
2961   \xywithoption{line}{%
2962     \message{Xy-pic pdf driver: 'line' extension support}%
2963     \@ifundefined{xypdf-li@loaded}{\input xypdf-li\relax}{\message{not reloaded}}}%
2964   }%
2965   \xywithoption{rotate}{%
2966     \message{Xy-pic pdf driver: 'rotate' extension support}%
2967     \@ifundefined{xypdf-ro@loaded}{\input xypdf-ro\relax}{\message{not reloaded}}}%
2968   }%
2969 }
2970 \next@
2971 \xyendinput
2972 </basic>

```

10 Changelog

v1.0 2010/03/24

Initial version

v1.1 2010/03/30

- Added support for the Xy-pic “rotate” extension.
- The parts of the style file dealing with Xy-pic extensions (currently “curve” and “rotate”) are only executed when those extension were loaded.
- xypdf does not give an error message when used with Xy-pic options which query the Postscript drivers (e. g. “all” or “color”).
- In DVI mode, a warning is issued that the DVI file is not portable, like Xy-pic does when a Postscript driver is in use.

v1.2 2010/04/08

- Improved precision and numerical stability for the offset algorithm around cusps.
- Improved slide algorithm \xP@slide@
- Respect \pdfdecimaldigits when dimensions are written to the PDF file.
- Correct continuation for dashed/dotted/squiggled curves consisting of more than one segment.
- Code cleanup

v1.3 2010/04/12

- Bug fix: No “`Extra \fi`” if `\ifpdfabsdim` is not defined.
- Bug fix: Moved the code for the spline continuation out of the optional section for curves since it is also needed for straight lines.
- Check the version of pdfTeX since `\pdfsave` is not defined prior to pdfTeX 1.40.0.
- “**Troubleshooting**” paragraph for TeX Live without the ε -TeX features enabled.
- Generic PDF code for the `{-}` directional object.

v1.4 2010/05/13

- Support for both plain TeX and L^ATeX, reorganization of the code and splitting into several files. The L^ATeX style file `xypdf.sty` has been replaced by `xypdf.tex`, which is recognized by X_Y-pic as a driver.
- Integration into the X_Y-pic distribution.
- Support for the “color” and “frame” extensions.
- New supported curve style `{~~}` (broken squiggled curves).

v1.5 2010/10/11

- Bug fix: Colored curves.
- Bug fix: PDF syntax in double elliptical frames.

v1.6 2011/02/09

The `xypdf` package was made even more frugal with dimension registers. If it is used as a L^ATeX package, it can now be loaded even if there is not a single free dimension register. The primary reason for this improvement is that `xypdf` now works with the “beamer” document class.

v1.7 2011/03/20

- The `\newxycolor` command is overwritten with the `xypdf` code in the preamble, not at `\begin{document}` as before, so that new colors can be defined anytime.
- Improved logic to detect zero-sized frames.
- `\newxycolor` bug fix.
- Corrected the position of the center of rotation and scaling.
- Colored frame objects.
- New section “**Differences between xypdf and the dvips backend**” in the documentation.