

# The curves Package\*

Ian Maclaine-cross

Internet: `i.maclaine-cross@unsw.edu.au`

22 August 2000

## Abstract

Draws curves in the  $\text{\LaTeX 2}\epsilon$  `picture` environment using parabolas between data points with continuous slope at joins. For circles and arcs uses up to 16 parabolas. Also draws symbols or dash patterns along curves. Equivalent to technical pens with compasses and French curves. Curves consist of short secants drawn by overlapping disks or line drawing `\specials` selected by package options.

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Installation</b>	<b>2</b>
<b>3</b>	<b>Curves</b>	<b>3</b>
<b>4</b>	<b>Scaling</b>	<b>5</b>
<b>5</b>	<b>Symbols</b>	<b>6</b>
<b>6</b>	<b>Dashes</b>	<b>8</b>
<b>7</b>	<b>Errors</b>	<b>9</b>
<b>8</b>	<b>Curves Summary</b>	<b>10</b>
8.1	Loading curves . . . . .	10
8.2	Arguments of Commands . . . . .	10
8.3	Lengths used by Commands . . . . .	10
8.4	Control Commands . . . . .	11
8.5	Parameter Setting Commands . . . . .	11
8.6	Curve Drawing Commands . . . . .	11

---

\*This file has version number 1.50, last revised 2000/08/22.

# 1 Introduction

The picture environment in the  $\LaTeX 2_{\epsilon}$ <sup>1</sup> macro package for  $\TeX$ <sup>2</sup> allows simple line drawing using characters. These characters include quadrant circular arcs, solid disks with diameters from 1 to 15pt<sup>3</sup> and short lines with a limited range of slopes in two thicknesses. A `\begin{picture}` command defines an area where following commands place these characters to draw a  $\LaTeX$  picture.

$\LaTeX$  pictures save disk space for source descriptions and computer time in producing documents compared with printer commands or bit mapped graphics. From initial pencil sketch on squared graph paper to final printout, they take half the time for manual pen drawings. The labor savings are higher for revisions and rewrites. Unfortunately standard  $\LaTeX$  cannot yet draw curves like a pen, compass and French curves can. Fortunately there are many macro packages which supplement  $\LaTeX 2_{\epsilon}$ 's capabilities and do marvellous graphical things for any printing need<sup>4</sup>. Curves just adds curve drawing to  $\LaTeX$  pictures.

Brief descriptions, simple examples and a command summary follow. They presume familiarity with relevant chapters of the  $\LaTeX$  manual<sup>1</sup>.

# 2 Installation

`curves.sty` To create the file `curves.sty` you need  $\LaTeX 2_{\epsilon}$  and a command like:

```
$ latex curves.ins
```

Put `curves.sty` and `curvesls.sty` in your default or a `texinput` directory. The package `curvesls` provides compatibility for old documents. Comprehensive  $\TeX$  distributions preinstall `curves` so for most users the above step is unnecessary.

Put `curves` in a `\usepackage` command at the top of your main `.tex` file for any document where you wish to use `curves` *e.g.*,

```
\documentclass[11pt]{article}
\usepackage{curves}
```

Do not combine `curves` with `bezier` in this command. `Curves` contains a fast powerful replacement for the `\bezier` command. Drawings using the `\bezier` command should not change their appearance.

`dvips` The `curves` package has options to save  $\TeX$  memory and runtime using  
`emtex` `\special` commands to draw the straight lines which approximate curves. Select  
`xdvi` an option only if your program for viewing or printing  $\TeX$ 's `dvi` files recognizes  
WML and uses the corresponding `\specials`. Otherwise the curves on your drawings will  
be invisible. The `dvips` option uses the  $\em\TeX$  `\specials` of `dvips` which draw  
lines with rounded ends. The `emtex` option uses the original `\specials` of  $\em\TeX$   
by Eberhard Mattes with disks added to hide their square ends. The `xdvi` option  
uses the PostScript `\specials` of Tomas Rokicki's `dvips` to draw lines which the  
`xdvi` viewer now implements. WML are new versions of the  $\em\TeX$  `\specials` in

---

<sup>1</sup>Leslie Lamport,  *$\LaTeX$  A Document Preparation System 2nd ed.*, Addison-Wesley, 1994.

<sup>2</sup>Donald E. Knuth, *The  $\TeX$ book*, Addison-Wesley, 1984.

<sup>3</sup>A printer's point, abbreviated pt, is approximately 0.351460 mm.

<sup>4</sup>Michel Goossens, Frank Mittelbach and Sebastian Rahtz, *The  $\LaTeX$  Graphics Companion*, Addison-Wesley, 1997.

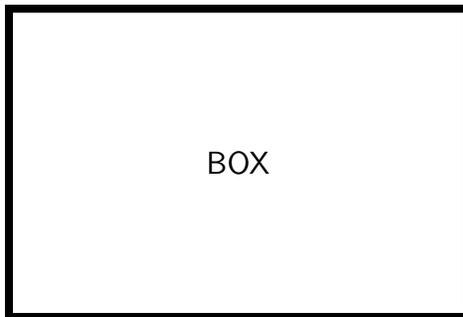


Figure 1: This is a box.

`dvips` with compact names. No options draws lines using disks from standard  $\LaTeX$  fonts. No options or `dvips` work with the `color` package but other drivers may or may not. Select package options when required by modifying `\usepackage` like:

```
\usepackage[dvips]{curves}
```

`BOX` A drawing frequently uses auxiliary commands to size, place, label and caption it. The following commands draw the box in Figure 1 on page 3:

```
\begin{figure}
\begin{center}
\setlength{\unitlength}{1mm}
\begin{picture}(100,50) \large\sf
\linethickness{1mm}
\put(20,5){\framebox(60,40){BOX}}
\end{picture}
\end{center}
\caption{This is a box.}
\label{box}
\end{figure}
```

Lamport<sup>1</sup> explains these commands. This example is for those unfamiliar with the  $\LaTeX$  `picture` environment. The following examples avoid the `figure` environment but it is often essential.

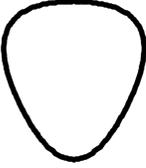
### 3 Curves

`\curve` The commands, `\curve`, `\closecurve` and `\tagcurve`, draw parabola segments  
`\closecurve` between coordinate points in the argument<sup>5</sup>. The segments' tangents at these  
`\tagcurve` points are parallel to each other and to straight lines through the points either  
side. Segments at `\curve` ends are parabolas with the point second from the end

---

<sup>5</sup>Please see Section 8 for full descriptions of `curves` commands.

a vertex. `\closecurve` adds a parabola between end points to close the curve. `\tagcurve` omits the first and last segments drawing curves with end tangents specified. When only three points are specified `\tagcurve` draws the last segment. The following table shows these features.

Example	Curve
<code>\curve(0,0, 50,100, 100,0)</code>	
<code>\closecurve(0,0, 50,100, 100,0)</code>	
<code>\tagcurve(100,0, 0,0, 50,100, 100,0, 0,0)</code>	

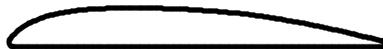
`\arc` Axial flow fans often use the RAF 6E aerofoil section. The section coordinates in  
`\curve` the following macro come directly from aerodynamic tables<sup>6</sup>. The `\arc` commands draw the leading and trailing radii and the two coordinate `\curve` the flat chord.

```
\newcommand{\RAFsixE}{
  \scaleput(1.25,1.25){\arc(0,-1.25){-135}}
  \scaleput(0,0){\curve(0.366,2.133, 1.25,3.19, 2.5,4.42,
    5.0,6.10, 7.5,7.24, 10,8.09, 15,9.28, 20,9.90, 30,10.3,
    40,10.22, 50,9.80, 60,8.98, 70,7.70, 80,5.91, 90,3.79,
    95,2.58, 99.24,1.52)}
  \scaleput(99.24,0.76){\arc(0,-0.76){180}}
  \scaleput(0,0){\curve(1.25,0, 99.24,0)}
}
```

In a picture environment like:

```
\begin{picture}(100,20)
  \RAFsixE
\end{picture}
```

this macro draws:



The RAF 6E has a flat undersurface.

`\bigcircle` The drawing command `\bigcircle` works similarly to `\circle` except there is no `\circle*` equivalent. The following section scales it to an ellipse.

<sup>6</sup>R.A. Wallis, *Axial Flow Fans*, Academic Press, 1961, p.335

## 4 Scaling

`\unitlength` The size of L<sup>A</sup>T<sub>E</sub>X picture objects may be uniformly scaled by preceding them with:  
`\put` `\setlength{\unitlength}{\scale\unitlength}`

the desired scale factor `\scale` is previously defined perhaps with `\newcommand` as a decimal number. The new coordinates of a point  $(x', y')$  relative to the current origin are related to the old coordinates  $(x, y)$  relative to the same origin by

$$\begin{aligned}x' &= x \times \text{\scale} \\y' &= y \times \text{\scale}\end{aligned}$$

If a `\put(x, y){...}` followed the change in `\unitlength` it would actually put the objects `{...}` at  $(x', y')$ . Objects defined by `\unitlength` in `{...}` would also be larger by `\scale`. Lamport<sup>1</sup> describes these commands.

`\scaleput` The scale factors `\xscale`, `\xscaley`, `\yscale` and `\yscalex` are initially defined to be 1, 0, 1 and 0 respectively but may be redefined to any decimal number using `\renewcommand`. After they are redefined the new coordinates of a point  $(x', y')$  relative to the current origin are related to the old coordinates  $(x, y)$  relative to the same origin by

$$\begin{aligned}x' &= x \times \text{\xscale} + y \times \text{\xscaley} \\y' &= x \times \text{\yscalex} + y \times \text{\yscale}\end{aligned}$$

If a `\scaleput(x, y){...}` followed the change in these factors it would actual put the objects `{...}` at  $(x', y')$ . All the drawing commands in `curves` use the current values of these four scale factors in placing disks and secants.

These factors can rotate pictures which like `\RAFsixE` are made solely with `curves` commands. The factors following rotate the RAF 6E through 12° clockwise about its (0,0) co-ordinate:

```
\renewcommand{\xscale}{0.9781}
\renewcommand{\xscaley}{0.2079}
\renewcommand{\yscale}{0.9781}
\renewcommand{\yscalex}{-0.2079}
\put(0,20){\RAFsixE}
```

This draws:



The RAF 6E has maximum lift at angles of attack over 12°.

Note that  $\cos 12^\circ \approx 0.9781$  and  $\sin 12^\circ \approx 0.2079$

`\arc` Axonometric projection is another scaling application. Circles become ellipses  
`\bigcircle` and circular arcs become elliptical arcs. The commands drawing the ellipse and arc in the following washer are:

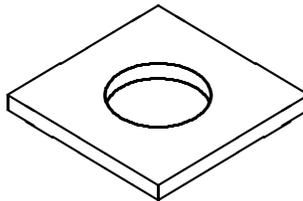
```
\put(20,5){
```

```

\renewcommand{\xscale}{1}
\renewcommand{\yscale}{-1}
\renewcommand{\yscale}{0.6}
\renewcommand{\yscalex}{0.6}
\scaleput(10,10){\bigcirc{10}}
\put(0,-2){
  \scaleput(10,10){\arc(5,0){121}}
  \scaleput(10,10){\arc(5,0){-31}}
}
}

```

(20,5) are the drawing coordinates of the upper vertex of the washer closest to the reader. The angles for the `\arcs` were found by trial and error.



Square washers are sometimes preferred for soft materials.

## 5 Symbols

`\curvesymbol` Curves can also place symbols. `\curvesymbol` must first define the symbol as anything a `\put` or `\multiput` may draw. A negative symbol count between drawing command and coordinates *e.g.*, `\tagcurve[-3](0,100,...)` fixes the number of symbols per curve segment.

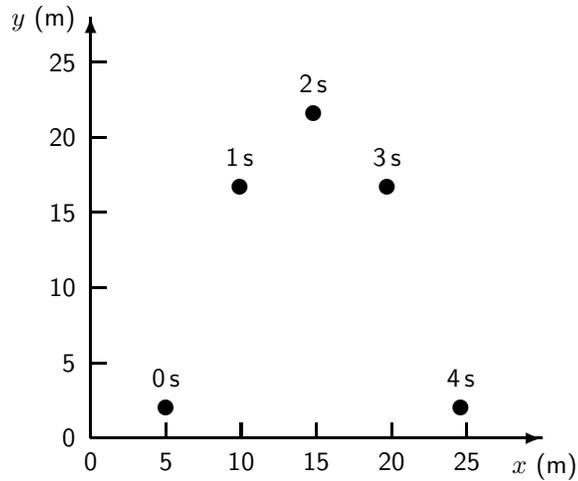
These commands draw flight times and successive positions in the following drawing:

```

\newcounter{time}
\curvesymbol{\thetime\s\addtocounter{time}{1}}
\put(5,4){\curve[-2](0,0, 9.8,19.6, 19.6,0)}
\curvesymbol{\phantom{\circle*{1}}\circle*{1}}
\put(5,2){\curve[-2](0,0, 9.8,19.6, 19.6,0)}

```

where `\phantom` is a plain `TEX` command from the `TEXbook`<sup>2</sup>.



Successive positions of a sphere with initial position (5, 2) m, initial velocity (4.9, 9.8) m/s, and acceleration (0, -9.8) m/s<sup>2</sup>. The flight time is recorded above each sphere position.

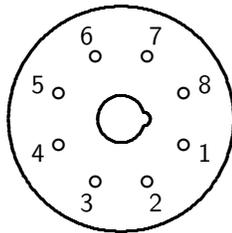
`\curvedashes` Fixed spacing of symbols at lengths other than the segment's requires more commands. Empty `\curvedashes`, empty `\curvesymbol` and negative symbol count stops drawing so a drawing command will calculate `\curvelength` only. `\curvesymbol` then resets the symbol and `\curvedashes` sets the spacing to its pattern length. If there are no symbols at the ends, `\overhang` pulls symbols along the curve. The last command with no symbol count draws the symbols.

`\arc` `\bigcircle` `\arc` and `\bigcircle` use sixteen segments for a circle so if eight symbols are required the fixed spacing technique is necessary. The following commands draw the pin numbers on a relay base:

```

\newcounter{pin}
\curvedashes{}
\curvesymbol{}
\put(60,60){\arc[-1](40,0){-360}}
\setlength{\curvelength}{0.125\curvelength}
\curvedashes[\curvelength]{1}
\setlength{\overhang}{0.5\curvelength}
\curvesymbol{\addtocounter{pin}{1}\thepin}
\put(60,60){\arc(40,0){-360}}

```



The pin numbering of plug-in relays is clockwise from the spigot key when viewed from below.

`\patternresolution`      If symbols and dash pattern exist and `\overhang` is 0pt, curves draw the first position blank. For equal spacing they draw the last position blank if rounding error causes the last pattern to be slightly short. If `\renewcommand` changes `\patternresolution`, rounding error changes and the final symbol may reappear. To avoid fiddling with `\patternresolution` for closed curves with symbols equally spaced, use an `\overhang` which is a fraction of a pattern length as in the previous example.

## 6 Dashes

`\curvedashes`      `\curvedashes` must first define a dash pattern with length greater than 0pt. Many symbol and pattern combinations are possible. The fixed number and fixed spacing methods of symbol drawing described in Section 5 work with three methods of drawing dashes which are:

1. if there is no symbol count and no symbol, a dash pattern with its length reduced by `\csdiameter` is drawn between symbols spaces of width close to `\csdiameter` to give an overall spacing equal to the pattern length specified by the `\curvedashes` command;
2. if there is a symbol count but no symbol, the dash patterns drawn have their length equal to that defined by `\curvedashes` with `\csdiameter` gaps at symbol positions;
3. if there is a symbol count and a symbol, the dash patterns drawn have their length adjusted slightly so an integral number of patterns fit between symbol positions.

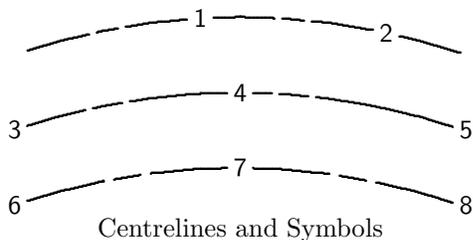
Dash pattern commands for centrelines<sup>7</sup> follow for the three techniques above in order:

```
\linethickness{0.25mm}
\curvedashes[1.2mm]{0,8,1,3,1,8}
\settowidth{\csdiameter}{00}
\put(0,20){\curve(0,0, 30,5, 60,0)}
\put(0,10){\curve[1](0,0, 30,5, 60,0)}
\curvesymbol{\thepin\addtocounter{pin}{1}}
\setlength{\csdiameter}{2\csdiameter}
\put(0,0){\curve[1](0,0, 30,5, 60,0)}
```

The following figure shows the resulting dash patterns. The upper line has first position blank because the `\overhang` is 0pt. It has patterns shrunk to scale between symbol spaces *e.g.*, 1 to 2, and symbol space centres one pattern length apart. The middle line has patterns close to defined length but with the first dash part blanked by half of symbol space 3 and the second pattern broken in its first dash by symbol space 4. The lower line patterns are stretched between symbol spaces. Which pattern is appropriate depends on picture meaning and function.

---

<sup>7</sup>R.N. Roth and I.A. van Haeringen, *The Australian Engineering Drawing Handbook, Part 1 Basic Principles and Techniques*, The Institution of Engineers, Australia, Canberra, 1986.



## 7 Errors

Syntax errors like incorrect or missing punctuation while using `curves` will result in  $\TeX$  or  $\LaTeX$  error messages. The  $\TeX$ book<sup>2</sup> and  $\LaTeX$  manual<sup>1</sup> explain the meaning and correction of these errors. The previous examples and Section 8 should make the correct syntax for `curves` commands clear.

`Curves` will write a `Package curves Error:...` message to the screen and `log` file if you supply an incorrect number of coordinates.

If four sequential points in a drawing command argument have the line through the first and third parallel to the line through the second and fourth:

- exactly or closely, `curves` knows it cannot draw a parabola tangent to two parallel lines, issues to the screen and `log` file:  
`Package curve Warning: \curve straight from ...`  
and draws a straight line;
- or approximately, `curves` may draw an unexpected curve with no warning.

If four sequential points in a drawing command argument have the line through the first and second parallel to the line through the third and fourth:

- `curves` draws a parabola which may be nowhere near the curve.

`\curvewarnfalse` If the four points were on a straight line, removing one or more points is a remedy. If they are not on a straight line, adding points may help. Specifying many points will give you a satisfactory curve with perhaps an annoying number of `\curve straight` warnings. After a `\curvewarnfalse`, `curves` still uses the straight lines but does not tell you.

`\tagcurve` Curvature changes sign on curves like  $y = \sin x$ . Specifying inflection points as `curve` coordinates will reduce error and specifying sufficient coordinates will then give satisfactory results. For discontinuous tangents splitting a curve into pieces is unavoidable. Splitting a curve into pieces with curvature the same sign can give satisfactory results with fewer coordinates. `\tagcurve` can prevent tangent discontinuities. If an inflexion point's exact location is unknown, try the midpoint of the straight line through the ends of its segment.

`\diskpitchstretch` Curves appear rougher than horizontal and vertical lines. Picture digitization causes this not inaccuracy in  $\TeX$  or `curves.sty`. Setting `\diskpitchstretch` to a value less than one with `\renewcommand` may smooth an unusually rough curve without package options.

Symbols and symbol spaces misaligned are usually due to rounding error. Adjusting `\patternresolution` below one can reduce rounding error and increase alignment accuracy. This should be limited to the misaligned curve with `{ }`<sup>1</sup>.

The replacement `\bezier` does not give exactly the same results as the original in `bezier.sty` or in `LaTeX2e`. The difference is extremely small but if it is important to you comment out the five lines of code for `\bezier` and `\@bezier` near the start of `curves.sty`. You now have a `\bezier` which is slower and needs more memory but has only its original capabilities and gives only its original results.

Please email [i.maclaine-cross@unsw.edu.au](mailto:i.maclaine-cross@unsw.edu.au) examples of any errors not covered above. You may have found a bug in the code or documentation.

## 8 Curves Summary

The commands following are for the picture environment in the `LATEX 2ε` manual<sup>1</sup>.

### 8.1 Loading curves

`\usepackage` A `\usepackage{curves}` between `\documentclass` and `\begin{document}` commands loads `curves`. If you have a `TEX` printing or viewing program which accepts the following `\special` commands you may optionally use them to allow larger pictures faster. Support for `color` may be lost.

You use `\usepackage[option]{curves}` to load a `\special` option for drawing the straight line secants which make curves where *option* is one of:

`dvips` uses the `emTEX \specials` with rounded line ends supported by `dvips`. Works with `color`.

`emtex` uses the original `emTEX \specials` with rectangular line ends. `Curves` adds a disk to round them.

`xdvi` uses the PostScript `\specials` of `dvips`.

`WML` the same as `dvips` but with single character names `W`, `M` and `L` to minimize `TEX` memory with large pictures.

### 8.2 Arguments of Commands

*blank length* decimal number of *unit len* blanks. Not negative.

*character or symbol* is anything which a `\put` or `\multiput` may draw.

*coordinates* are decimal numbers giving alternate *x* and *y* coordinates of the curve as multiples of `\unitlength`, comma separated.

*[,dash...]* optional continuation of alternating dash and blank numbers of unit lengths, comma separated. Not negative. Allows decimal points.

*diameter* is a decimal number giving the diameter in `\unitlengths`.

*symbol count* is the number of symbols or patterns to be drawn, default 0.

*unit len* unit length dimension *e.g.*, 2.5mm, 10pt, used in measuring blanks and dashes. Not negative. Default value is `\unitlength`.

### 8.3 Lengths used by Commands

`\csdiameter` is the size of the space left for a symbol and can be increased or set with `\settowidth{\csdiameter}{character or symbol}`.

`\curvelength` is the total length of the curve calculated before drawing by using Simpson's rule once between each pair of coordinate points.

`\overhang` length of as drawn dash pattern overlapping start of patterns.

## 8.4 Control Commands

`\curvewarntrue` turns warning of parabola replacement by straight lines on (default).  
`\curvewarnfalse` turns warning of parabola replacement by straight lines off.

## 8.5 Parameter Setting Commands

`\curvesymbol`  $\{\langle character\ or\ symbol\rangle\}$  sets symbol and `\csdiameter`.  
`\curvedashes`  $[\langle unit\ len\rangle][\langle blank\ length\rangle[\langle dash\dots\rangle]]$  A drawing command not following a `\curvedashes` or following one with an empty or zero length pattern will draw:  
if  $\langle symbol\ count\rangle$  is zero or missing, a continuous curve;  
else if  $\langle symbol\ count\rangle$  is positive,  $\langle symbol\ count\rangle-1$  squares of line thickness size between and additional squares at coordinates or bezier end points;  
else if no  $\langle character\ or\ symbol\rangle$  exists, nothing;  
else,  $-\langle symbol\ count\rangle-1$  characters or symbols between coordinates and additional ones at coordinates or bezier end points.

After a `\curvedashes` command defining a pattern whose length exceeds zero, commands draw:

if  $\langle symbol\ count\rangle$  is zero or missing then at a spacing equal to the specified pattern length,

if no  $\langle character\ or\ symbol\rangle$  exists, a dash pattern reduced in length by `\csdiameter` to fit between symbol spaces of `\csdiameter`,

else if `\overhang` is not 0pt, a  $\langle character\ or\ symbol\rangle$  at all positions,  
else a  $\langle character\ or\ symbol\rangle$  with the first position blank;

else, `\csdiameter` wide symbol spaces, one at and  $\langle symbol\ count\rangle-1$  between coordinate points with dash pattern lengths,

if no  $\langle character\ or\ symbol\rangle$  exists, exact but broken by the spaces,  
else, adjusted to give a whole number of patterns between spaces.

`\diskpitchstretch` is initially 1 but `\renewcommand` can change it to a higher value like 5 to save memory in drafts of complex documents or a lower local value like 0.5 to smooth curve digitization.

`\linethickness`  $\{\langle len\rangle\}$  sets line or dash thicknesses to  $\langle len\rangle$  from 0.5pt up to 15pt (0.17mm to 5mm). `\thicklines` and `\thinlines` also set thickness.

`\patternresolution` is initially 1 but `\renewcommand` can change it to a higher value like 5 to save memory in drafts of complex documents or a lower local value like 0.5 for greater dash pattern accuracy.

`\xscale`  
`\xscaley` are scale factors initially set to 1, 0, 1 and 0 respectively which `\renewcommand` can reset.

`\yscale`

`\yscalex`

## 8.6 Curve Drawing Commands

Curves drawn consist of parabolic arcs between coordinate points with tangents at each point parallel to the straight line through adjacent points.

`\arc`  $[\langle symbol\ count\rangle](X1,Y1)\{\langle angle\rangle\}$  draws a circular arc centred on current position, starting from (X1,Y1) and proceeding counterclockwise for  $\langle angle\rangle$  degrees.

`\bezier`  $\{\langle symbol\ count\rangle\}(X1,Y1)(X2,Y2)(X3,Y3)$  draws a curve through the end points

(X1,Y1) and (X3,Y3) tangent to the straight lines joining each of them to (X2,Y2). Extended faster replacement for `bezier.sty` version.

- `\bigcircle` [*symbol count*]{*diameter*} draws a circle of diameter equal to *diameter* times `\unitlength`.
- `\closecurve` [*symbol count*](*coordinates*) draws a closed curve with continuous tangents at all points. At least 6 coordinates required.
- `\curve` [*symbol count*](*coordinates*) draws a curve through the *coordinates* specified. For 4 coordinates this is a straight line.
- `\scaleput` (X1,Y1){*picture object*} places a picture object in a position scaled by `\xscale`, `\xscaley`, `\yscale` and `\yscalex` for axonometric projection or rotations.
- `\tagcurve` [*symbol count*](*coordinates*) draws a curve without its first and last segments but if only 6 coordinates draws the last segment only.