Package 'gridBezier'

October 13, 2022

Type Package

Title Bezier Curves in 'grid'

Version 1.1-1

Author Paul Murrell

Maintainer Paul Murrell <paul@stat.auckland.ac.nz>

Description Functions for rendering Bezier
curves (Pomax, 2018) <https://pomax.github.io/bezierinfo/>
in 'grid'.
There is support for both quadratic and cubic Bezier curves.
There are also functions for calculating points on curves,
tangents to curves, and normals to curves.

Depends grid

URL https://github.com/pmur002/gridbezier,https://stattech.wordpress.fos.auckland.ac.nz/2018/11/02/2018-11-variable-width-bezier-splines-in-r/

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-22 13:10:03 UTC

R topics documented:

BezierPoints *Calculate Points on a Bezier Curve*

Description

Calculate points on a Bezier curve and/or tangents and/or normals to the curve at those points.

Usage

```
BezierPoints(x, range = NULL)
BezierTangent(x, range = NULL)
BezierNormal(x, range = NULL)
```

Arguments

x	A "BezierGrob" object.
range	The range of t values within which to calculate points (or tangents or normals). A numeric vector of length 2.

Details

The tangents and normals are 1 inch in length.

Value

All functions return a list with components x and y. For BezierPoints these are locations on the curve. For BezierTangent and BezierNormal, these are the distances to the end points of tangent or normal line segments. All values are in inches.

Author(s)

Paul Murrell

Examples

```
x <- BezierGrob(c(.2, .2, .8, .8), c(.2, .8, .8, .2),
                stepFn=function(...) seq(0, 1, length.out=10))
grid.draw(x)
pts <- BezierPoints(x)
grid.circle(pts$x, pts$y, default.units="in", r=unit(.5, "mm"),
            gp=gpar(fill="black"))
tan <- BezierTangent(x)
grid.segments(pts$x, pts$y, pts$x + tan$x, pts$y + tan$y,
              default.units="in", gp=gpar(col="green"))
norm <- BezierNormal(x)
grid.segments(pts$x, pts$y, pts$x + norm$x, pts$y + norm$y,
              default.units="in", gp=gpar(col="red"))
```

grid.Bezier *Draw a Bezier Spline*

Description

Draw a Bezier curve or Bezier spline (multiple Bezier curves strung together).

Usage

```
grid.Bezier(...)
BezierGrob(x, y, default.units="npc",
           open=TRUE, stepFn=nSteps(100), gp=gpar(), name=NULL)
```

Arguments

x, y	Locations of control points. There should be four, or seven (or six if not open), or ten (or nine), etc. Locations can be numeric or **grid** "unit" objects.
default.units	The coordinate system to use if control point locations are just numeric.
open	Whether to reuse the first control point as the last control point. If closed, the shape may also be filled.
stepFn	A function to generate values of t at which the curve will be evaluated for drawing. The default is 100 equal-sized steps from 0 to 1. This function is called for each Bezier curve within the Bezier spline, with arguments x, y (the control points), and range (indicating the range of t to generate values for).
gp	A **grid** "gpar" object, as produced by gpar(), or NULL.
name	A name for the grob that is generated.
...	Arguments passed from grid.Bezier() to BezierGrob().

Details

This function will produce a nicer result than the grid.bezier function from **grid** (because the latter is just an approximation using X-splines). This function also supports Bezier splines.

Value

BezierGrob produces a "BezierGrob" object.

Author(s)

Paul Murrell

Examples

```
grid.Bezier(c(.2, .2, .8, .8), c(.2, .8, .8, .2))
```

grid.quad *Draw a Quadratic Bezier Spline*

Description

Draw a quadratic Bezier curve or quadratic Bezier spline (multiple quadratic Bezier curves strung together).

Usage

```
grid.quad(...)
quadGrob(x, y, default.units="npc",
           open=TRUE, stepFn=nSteps(100), gp=gpar(), name=NULL)
```

Arguments

x, y	Locations of control points. There should be three, or five (or four if not open), or seven (or eight), etc. Locations can be numeric or **grid** "unit" objects.
default.units	The coordinate system to use if control point locations are just numeric.
open	Whether to reuse the first control point as the last control point. If closed, the shape may also be filled.
stepFn	A function to generate values of t at which the curve will be evaluated for drawing. The default is 100 equal-sized steps from 0 to 1. This function is called for each Bezier curve within the Bezier spline, with arguments x, y (the control points), and range (indicating the range of t to generate values for).
gp	A **grid** "gpar" object, as produced by gpar(), or NULL.
name	A name for the grob that is generated.
...	Arguments passed from grid.quad() to quadGrob().

Value

quadGrob produces a "quadgrob" object.

Author(s)

Paul Murrell

Examples

```
grid.quad(c(.2, .5, .8), c(.2, .8, .2))
```

nSteps *Calculate Steps for a Bezier Spline*

Description

Calculate steps in t for drawing each Bezier curve within a Bezier spline.

Usage

```
nSteps(n)
```

Arguments

n	The number of steps (assuming a range of t from 0 to 1).

Details

This function generates a function that can be used as the `stepFn` argument to `grid.Bezier`. It will simply generate `n` values in the range 0 to 1, though if `range` is also provided, the number of steps is reduced (see the examples below). A minimum of 2 steps will be generated.

Value

`BezierGrob` produces a `"BezierGrob"` object.

Author(s)

Paul Murrell

Examples

```
nSteps(100)
nSteps(100)(range=0:1)
nSteps(100)(range=0:1/2)
```

quadPoints *Calculate Points on a Bezier Curve*

Description

Calculate points on a Bezier curve and/or tangents and/or normals to the curve at those points.

Usage

```
quadPoints(x, range = NULL)
quadTangent(x, range = NULL)
quadNormal(x, range = NULL)
```

Arguments

x	A "quadGrob" object.
range	The range of t values within which to calculate points (or tangents or normals). A numeric vector of length 2.

Details

The tangents and normals are 1 inch in length.

Value

All functions return a list with components x and y. For quadPoints these are locations on the curve. For quadTangent and quadNormal, these are the distances to the end points of tangent or normal line segments. All values are in inches.

Author(s)

Paul Murrell

Examples

```
x <- quadGrob(c(.2, .5, .8), c(.2, .8, .2),
                stepFn=function(...) seq(0, 1, length.out=10))
grid.draw(x)
pts <- quadPoints(x)
grid.circle(pts$x, pts$y, default.units="in", r=unit(.5, "mm"),
            gp=gpar(fill="black"))
tan <- quadTangent(x)
grid.segments(pts$x, pts$y, pts$x + tan$x, pts$y + tan$y,
              default.units="in", gp=gpar(col="green"))
norm <- quadNormal(x)
grid.segments(pts$x, pts$y, pts$x + norm$x, pts$y + norm$y,
              default.units="in", gp=gpar(col="red"))
```

Index