Package ‘aRtsy’

August 16, 2021

Title  Generative Art with 'ggplot2'
Description  Provides various algorithms for creating artworks in the 'ggplot2' language that incorporate some form of randomness.
Version  0.1.0
Date  2021-08-11
BugReports  https://github.com/koenderks/aRtsy/issues
URL  https://koenderks.github.io/aRtsy/,
     https://github.com/koenderks/aRtsy,
     https://twitter.com/aRtsy_package
Imports  dplyr, ggplot2, ggpubr, Rcpp, reshape2
LinkingTo  Rcpp, RcppArmadillo
Language  en-US
License  GPL-3
Encoding  UTF-8
RoxygenNote  7.1.1
Suggests  testthat (>= 3.0.0)
Config/testthat/edition  3
NeedsCompilation  yes
Author  Koen Derks [aut, cre] (<https://orcid.org/0000-0002-5533-9349>)
Maintainer  Koen Derks <koen-derks@hotmail.com>
Repository  CRAN
Date/Publication  2021-08-16 07:40:02 UTC

R topics documented:

aRtsy-package ................................................................. 2
canvas_ant ................................................................. 3
canvas_circlemap ......................................................... 4
aRtsy-package

Description

aRtsy is an attempt at making generative art available for the masses in a simple and standardized format. The package provides various algorithms for creating artworks in ggplot2 that incorporate some form of randomness (depending on the set seed). Each type of artwork is implemented in a separate function.

For documentation on aRtsy itself, including the manual and user guide for the package, worked examples, and other tutorial information visit the package website.

Author(s)

Koen Derks (maintainer, author)  <koen-derks@hotmail.com>

Please use the citation provided by R when citing this package. A BibTex entry is available from citation("aRtsy").

See Also

Useful links:

- The twitter feed to check the artwork of the day.
- The issue page to submit a bug report or feature request.
**canvas_ant**

**Paint Langton’s Ant on a Canvas**

**Description**

This function paints Langton’s Ant. Langton’s ant is a two-dimensional universal Turing machine with a very simple set of rules but complex emergent behavior.

**Usage**

```r
canvas_ant(colors, background = '#fafafa', iterations = 1e7, width = 200, height = 200)
```

**Arguments**

- `colors` a character (vector) specifying the colors for the ant.
- `background` a character specifying the color of the background.
- `iterations` the number of iterations of the ant.
- `width` the width of the artwork in pixels.
- `height` the height of the artwork in pixels.

**Value**

A ggplot object containing the artwork.

**Author(s)**

Koen Derks, <koen-derks@hotmail.com>

**References**


**Examples**

```r
canvas_ant(colors = '#000000', background = '#fafafa')
```
canvas_circlemap

Paint a Circle Map on a Canvas

Description

This function is my attempt at a circle map.

Usage

```r
canvas_circlemap(colors, x_min = 0, x_max = 12.56, y_min = 0, y_max = 1,
                iterations = 10, width = 1500, height = 1500)
```

Arguments

- `colors`: a character specifying the color used for the function shape.
- `x_min`: a numeric value specifying the minimum value for the x-axis.
- `x_max`: a numeric value specifying the maximum value for the x-axis.
- `y_min`: a numeric value specifying the minimum value for the y-axis.
- `y_max`: a numeric value specifying the maximum value for the y-axis.
- `iterations`: the number of iterations.
- `width`: the width of the artwork in pixels.
- `height`: the height of the artwork in pixels.

Value

A ggplot object containing the artwork.

Author(s)

Koen Derks, <koen-derks@hotmail.com>

References

https://linas.org/art-gallery/circle-map/circle-map.html

Examples

```r
canvas_circlemap(colors = colorPalette('tuscany2'))
```
canvas_collatz

Paint the Collatz Conjecture on Canvas

Description

This function draws the Collatz conjecture on the canvas.

Usage

canvas_collatz(colors, background = '#fafafa', n = 200,
   angle.even = 0.0075, angle.odd = 0.0145, side = FALSE)

Arguments

colors a character (vector) specifying the colors used for the artwork.
background a character specifying the color used for the background.
n the number of numbers to sample for the lines. Can also be a vector of numbers to use.
agele.even the angle (radials) to use after odd numbers.
agele.odd the angle (radials) to use after even numbers.
side logical. Whether to put the artwork on its side.

Value

A ggplot object containing the artwork.

Author(s)

Koen Derks, <koen-derks@hotmail.com>

Examples

set.seed(1)
canvas_collatz(colors = colorPalette('dark1'), n = 100)
canvas_diamonds  
*Paint A Diamond on Canvas*

---

**Description**

This function draws many diamonds on the canvas and places two lines behind them. The diamonds can be transparent or have a random color sampled from the input.

**Usage**

```r
canvas_diamonds(colors, background = '#fafa', col.line = 'black',
                 radius = 10, alpha = 1, p = 0.2,
                 width = 500, height = 500)
```

**Arguments**

- **colors**: a character (vector) specifying the colors used for the strokes.
- **background**: a character specifying the color used for the background.
- **col.line**: color of the lines.
- **radius**: radius of the diamonds.
- **alpha**: transparency of the diamonds. If NULL, added layers become increasingly more transparent.
- **p**: takeover probability.
- **width**: the width of the artwork in pixels.
- **height**: the height of the artwork in pixels.

**Value**

A `ggplot` object containing the artwork.

**Author(s)**

Koen Derks, <koen-derks@hotmail.com>

**Examples**

```r
set.seed(1)
canvas_diamonds(colors = colorPalette('house'), radius = 10)
```
canvas_function

Paint Functions on a Canvas

Description

This function paints functions with random parameters and mimics the functionality of the generativeart package.

Usage

canvas_function(color, background = '#fafafa')

Arguments

color  a character specifying the color used for the function shape.
background  a character specifying the color used for the background.

Value

A ggplot object containing the artwork.

Author(s)

Koen Derks, <koen-derks@hotmail.com>

References

https://github.com/cutterkom/generativeart

Examples

set.seed(10)
canvas_function(color = '#000000', background = '#fafafa')
canvas_mandelbrot  

Paint the Mandelbrot Set on Canvas

Description
This function draws the Mandelbrot set on the canvas.

Usage
canvas_mandelbrot(colors, n = 100, xmin = -1.7, xmax = -0.2, ymin = -0.2999, ymax = 0.8001, zoom = 1, width = 500, height = 500)

Arguments
- colors: a character (vector) specifying the colors used for the artwork.
- n: the number of iterations.
- xmin: the minimum x value.
- xmax: the maximum x value.
- ymin: the minimum y value.
- ymax: the maximum y value.
- zoom: the amount of zoom to apply.
- width: the width of the artwork in pixels.
- height: the height of the artwork in pixels.

Value
A ggplot object containing the artwork.

Author(s)
Koen Derks, <koen-derks@hotmail.com>

Examples
set.seed(1)
canvas_mandelbrot(colors = colorPalette('dark1'), n = 100)
canvas_planet

Paint a Planet on a Canvas

Description

This function paints one or multiple planets.

Usage

```
canvas_planet(colors, threshold = 4, iterations = 200,
  starprob = 0.01, fade = 0.2,
  radius = NULL, center.x = NULL, center.y = NULL,
  light.right = TRUE, width = 1500, height = 1500)
```

Arguments

- **colors**: a character specifying the colors used for the planet(s). Can also be a list where each entry is a vector of colors for each planet.
- **threshold**: a character specifying the threshold for a color take.
- **iterations**: the number of iterations of the planets.
- **starprob**: the probability of drawing a star in outer space.
- **fade**: the fading factor.
- **radius**: a numeric (vector) specifying the radius of the planet(s).
- **center.x**: the x-axis coordinate(s) for the center(s) of the planet(s).
- **center.y**: the y-axis coordinate(s) for the center(s) of the planet(s).
- **light.right**: whether to draw the light from the right or the left.
- **width**: the width of the artwork in pixels.
- **height**: the height of the artwork in pixels.

Value

A `ggplot` object containing the artwork.

Author(s)

Koen Derks, <koen-derks@hotmail.com>

Examples

```r
# Sun behind Earth and Moon
set.seed(1)
colors <- list(c("khaki1", "lightcoral", "lightsalmon"),
  c("dodgerblue", "forestgreen", "white"),
  c("gray", "darkgray", "beige"))
```
canvas_polylines(colors, radius = c(800, 400, 150),
              center.x = c(1, 500, 1100),
              center.y = c(1400, 500, 1000),
              starprob = 0.005)

canvas_polylines

Paint Polygons and Lines on Canvas

Description
This function draws many points on the canvas and connects these points into a polygon. After repeating this for all the colors, the edges of all polygons are drawn on top of the artwork.

Usage
canvas_polylines(colors, background = '\#fafafa', ratio = 0.5, iterations = 1000,
              alpha = NULL, size = 0.1, width = 500, height = 500)

Arguments
  colors          a character (vector) specifying the colors used for the strokes.
  background      a character specifying the color used for the borders.
  ratio           width of the polygons. Larger ratios cause more overlap.
  iterations      the number of points for each polygon.
  alpha           transparency of the polygons. If NULL, added layers become increasingly more transparent.
  size            size of the borders.
  width           the width of the artwork in pixels.
  height          the height of the artwork in pixels.

Value
A ggplot object containing the artwork.

Author(s)
Koen Derks, <koen-derks@hotmail.com>

Examples
set.seed(1)
canvas_polylines(colors = colorPalette('retro2'))
canvas_ribbons

Paint Ribbons on a Canvas

Description
This function paints ribbons and (optionally) a triangle in the middle.

Usage
canvas_ribbons(colors, background = '#fdf5e6', triangle = TRUE)

Arguments
- colors: a character (vector) specifying the colors for the ribbons. Colors determine the number of ribbons.
- background: a character specifying the color of the background.
- triangle: logical. Whether to draw the triangle that breaks the ribbon polygons.

Value
A ggplot object containing the artwork.

Author(s)
Koen Derks, <koen-derks@hotmail.com>

Examples
set.seed(1)
canvas_ribbons(colors = colorPalette('tuscany1'))

canvas_segments

Paint Line Segments on Canvas

Description
This function draws many line segments on the canvas.

Usage
canvas_segments(colors, background = '#fafafa', n = 100, p = 0.5, H = 0.1, size = 0.2)
canvas_squares

Arguments

- **colors**: a character (vector) specifying the colors used for the line segments.
- **background**: a character specifying the color used for the background.
- **n**: the number of line segments to draw.
- **p**: probability of drawing a vertical line segment.
- **H**: scaling factor for the line segments.
- **size**: line width of the segments.

Value

A ggplot object containing the artwork.

Author(s)

Koen Derks, <koen-derks@hotmail.com>

Examples

```r
set.seed(1)
canvas_segments(colors = 'black', background = '#fafafa')
```

---

canvas_squares  

*Paint Squares on a Canvas*

Description

This function paints a squares. It works by repeatedly cutting into the canvas at random locations and coloring the area that these cuts create.

Usage

```r
canvas_squares(colors, background = '#000000', cuts = 50, ratio = 1.618, width = 100, height = 100)
```

Arguments

- **colors**: a character vector specifying the colors used in the squares.
- **background**: a character specifying the color used for the background (borders).
- **cuts**: the number of cuts to make.
- **ratio**: the 1:1 ratio for each cut.
- **width**: the width of the artwork in pixels.
- **height**: the height of the artwork in pixels.
canvas_strokes

Value
A ggplot object containing the artwork.

Author(s)
Koen Derks, <koen-derks@hotmail.com>

Examples

```r
set.seed(6)
canvas_squares(colors = colorPalette('tuscany1'))
```

canvas_strokes  Paint Strokes on a Canvas

Description
This function creates an artwork that resembles paints strokes. The algorithm is based on the simple idea that each next point on the grid has a chance to take over the color of an adjacent colored point but also has a change of generating a new color.

Usage

```r
canvas_strokes(colors, neighbors = 1, p = 0.01, iterations = 1,
width = 500, height = 500, side = FALSE)
```

Arguments

colors  a character (vector) specifying the colors used for the strokes.
neighbors  the number of neighbors a block considers when taking over a color. More neighbors fades the artwork.
p  the probability of selecting a new color at each block. A higher probability adds more noise to the artwork.
iterations  the number of iterations on the artwork. More iterations fade the artwork.
width  the width of the artwork in pixels.
height  the height of the artwork in pixels.
side  whether to turn the artwork on its side.

Value
A ggplot object containing the artwork.
Author(s)
Koen Derks, <koen-derks@hotmail.com>

Examples

```r
set.seed(1)
canvas_strokes(colors = colorPalette("tuscany3"))
```

---

<table>
<thead>
<tr>
<th>canvas_turmite</th>
<th>Paint a Turmite on a Canvas</th>
</tr>
</thead>
</table>

Description

This function paints a turmite. A turmite is a Turing machine which has an orientation in addition to a current state and a "tape" that consists of a two-dimensional grid of cells. The algorithm is simple: 1) turn on the spot (left, right, up, down) 2) change the color of the square 3) move forward one square.

Usage

```r
canvas_turmite(color, background = "#fafafa", p = 0.5, iterations = 1e7, width = 1500, height = 1500)
```

Arguments

- **color**: a character specifying the color used for the turmite.
- **background**: a character specifying the color used for the background.
- **p**: the probability of a state switch within the turmite.
- **iterations**: the number of iterations of the turmite.
- **width**: the width of the artwork in pixels.
- **height**: the height of the artwork in pixels.

Value

A ggplot object containing the artwork.

Author(s)
Koen Derks, <koen-derks@hotmail.com>

References

https://en.wikipedia.org/wiki/Turmite
Examples

```r
set.seed(1)
canvas_turmite(color = "#000000", background = "#fafafa")
```

**colorPalette**  
*Color palette generator.*

**Description**

This function creates a random color palette, or allows the user to select a pre-implemented palette.

**Usage**

```
colorPalette(name, n = NULL)
```

**Arguments**

- `name`  
  name of the color palette. Can be `random` for random colors, but can also be the name of a pre-implemented palette. See the `details` section for a list of pre-implemented palettes.

- `n`  
  the number of colors to select from the palette. Required if `name = 'random'`. Otherwise, if `NULL`, automatically selects all colors from the chosen palette.

**Details**

The following color palettes are implemented:
Value
A vector of colors.

Author(s)
Koen Derks, <koen-derks@hotmail.com>

Examples
colorPalette('random', 5)

```r
saveCanvas(plot, filename, width = 7, height = 7, resolution)
```

Description
This function is a wrapper around `ggplot2::ggsave`. It provides a suggested export with square dimensions for a canvas created using the aRtsy package.

Usage
```r
saveCanvas(plot, filename, width = 7, height = 7, resolution)
```
themeCanvas

Arguments

- plot: a ggplot2 object to be saved.
- filename: the filename of the export.
- width: the width of the artwork in cm.
- height: the height of the artwork in cm.
- resolution: the dpi of the export.

Value

No return value, called for saving plots.

Author(s)

Koen Derks, <koen-derks@hotmail.com>

---

themeCanvas

Canvas theme for ggplot2 objects

Description

Add a canvas theme to the plot. The canvas theme by default has no margins and fills any empty canvas with a background color.

Usage

themeCanvas(x, background = '#fafafa', margin = -1.25)

Arguments

- x: a ggplot2 object.
- background: a character specifying the color used for the empty canvas.
- margin: margins of the plot.

Value

A ggplot object containing the artwork.

Author(s)

Koen Derks, <koen-derks@hotmail.com>
Index

* aRtsy
  aRtsy-package, 2

* artwork
  canvas_ant, 3
  canvas_circlemap, 4
  canvas_collatz, 5
  canvas_diamonds, 6
  canvas_function, 7
  canvas_mandelbrot, 8
  canvas_planet, 9
  canvas_polylines, 10
  canvas_ribbons, 11
  canvas_segments, 11
  canvas_squares, 12
  canvas_strokes, 13
  canvas_turmite, 14

* canvas
  canvas_ant, 3
  canvas_circlemap, 4
  canvas_collatz, 5
  canvas_diamonds, 6
  canvas_function, 7
  canvas_mandelbrot, 8
  canvas_planet, 9
  canvas_polylines, 10
  canvas_ribbons, 11
  canvas_segments, 11
  canvas_squares, 12
  canvas_strokes, 13
  canvas_turmite, 14
  colorPalette, 15
  saveCanvas, 16
  themeCanvas, 17

* theme
  themeCanvas, 17

aRtsy (aRtsy-package), 2
aRtsy-package, 2

canvas_ant, 3
canvas_circlemap, 4
canvas_collatz, 5
canvas_diamonds, 6
canvas_function, 7
canvas_mandelbrot, 8
canvas_planet, 9
canvas_polylines, 10
canvas_ribbons, 11
canvas_segments, 11
canvas_squares, 12
canvas_strokes, 13
canvas_turmite, 14
colorPalette, 15
saveCanvas, 16
themeCanvas, 17