Package 'GenArt'

June 4, 2019

Title Genetic algorithm for image reproduction

Version 0.1.0

Description R package provides a genetic algorithm for image reproduction. It is based on forcing one or more errors during DNA replication (mutation) to keep the best candidate for another mutation. DNA consists of coordinates, degree of transparency, and RGB codes or gray scales for a given number of polygons.

Depends R (>= 3.5.1)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Maintainer Robert Herrmann <robertherrmann@mail.de>

Author Robert Herrmann

RoxygenNote 6.1.1

Import png

NeedsCompilation no

R topics documented:

dna_convert *Convert image DNA to PNG format*

Description

Function converts image DNA to array object including RGB or gray scale for each pixel.

Usage

```
dna_convert(dna, maxXY, tempf, pngWH, bg = "white")
```

Arguments

dna	matrix or character, untangled or tangled image DNA of any size.
tempf	temporate file generated by default or given as file path.
pngWH	vector, width and height of reconstructed image. If missing, width and height of original image are used.
bg	character, color or RGB code indicating the background color of PNG.

Details

See example...

Examples

```
dna <- dna_untangle(dna_in(rgb = FALSE))
for(i in 1:20){
dna <- dna_mutate(dna)
}
test <- dna_convert(dna)
grid::grid.raster(test)
test[1,1,]
```

dna_in *Generate image DNA*

Description

Function to generate a random image DNA.

Usage

```
dna_in(n = 10, vertex = 3, maxXY = c(200, 200), rgb = TRUE)
```

Arguments

n	numeric, number of polygons to be drawn to your image, Default = 10.
vertex	numeric, number of vertices each polygen should be constructed of, Default = 3.
maxXY	vector, sets the pixels given by the original image, Default = c(200, 200).
rgb	logic, if TRUE, generates an RGB code (default), if FALLS, generates a gray scale.

Details

Is generally the first step of starting an image reproduction. Use maxXY = dim(pic)[2:1] to get the number of pixels in the original image.

Examples

```
dna_in(n = 5, vertex = 2, rgb = FALSE)
```

---

dna_mutate *Mutate image DNA*

Description

Function to mutate an image DNA.

Usage

```
dna_mutate(dna, degree = "soft", maxXY)
```

Arguments

dna	matrix or character, untangled or tangled image DNA of any size.
degree	numeric, number nucleotides to be modified per mutation, Default = 1.

Details

It is recommended to use the softest mutation rate as given by default.

Examples

```
dna <- dna_untangle(dna_in(rgb = FALSE))
test <- dna_mutate(dna, degree = 20)
pic_compare(dna, test)
```

---

dna_print *Plot image DNA*

Description

Function to plot an image DNA.

Usage

```
dna_print(dna, maxXY)
```

Arguments

dna	matrix or character, untangled or tangled image DNA of any size.

Details

Are comming up soon...

Examples

```
#Is comming up soon...
```

`dna_read` *Read image DNA*

Description

Function to read file.txt including a tangled or untangled image DNA.

Usage

```
dna_read(log, tangle = TRUE)
```

Arguments

`log`	filepath.
`tangle`	logic, TRUE if tangled (default), FALSE if untangled.

Details

After your calculation is done, you may want to improve the results by reloading the existing image DNA, and start the calculation again.

`dna_tangle` *Tangle image DNA*

Description

Function to tangle an untangled image DNA.

Usage

```
dna_tangle(dna)
```

Arguments

`dna`	matrix, untangled image DNA of any size.

Details

See example...

Examples

```
dna <- dna_untangle(dna_in(n = 2, vertex = 3, rgb = FALSE))
dna_tangle(dna)
```

dna_untangle *Untangle image DNA*

Description

Function to untangle an image DNA to get color code and coordinates of each polygon.

Usage

```
dna_untangle(dna)
```

Arguments

dna	character, tangled image DNA of any size.

Details

See example...

Examples

```
dna_untangle(dna_in())
dna_untangle(dna_in(rgb = FALSE))
```

gen_art *Genetic algorithm*

Description

Algorithm that continues to mutate (improve) the image DNA until manually stoped or a given maximum of iterations is reached

Usage

```
gen_art(pic, dna, iter = Inf, fname = "ReconPic.png",
  fnamelog = "Log.txt", tempf, degree = "Soft")
```

Arguments

pic	object of class array, original image in PNG format.
dna	matrix or character, untangled or tangled image DNA of any size.
iter	numeric, number of iterations. Default = Inf.
fname	character, filename for the reconstructed image. Default = "ReconPic.png".
tempf	temporate file generated by default or given as file path. See ...
degree	numeric, how many nucleotides should be modified per one iteration, Default = 1.
fanmelog	character, filename for file.txt including the tangled image DNA. Default = "Log.txt".

Details

It is recommended to use the softest mutation rate as given by default.

See example...

Examples

```
#Is coming up soon...
```

---

pic_compare *Calculate image fitness*

Description

Function to compare two images to calculate fitness.

Usage

```
pic_compare(imgONE, imgTWO)
```

Arguments

imgONE	first array to compare with second array.
imgTWO	second array to compare with first array.

Details

...

---

pic_convert *Convert RGB code to gray scale*

Description

Function to convert RGB to gray scale.

Usage

```
pic_convert(pic)
```

Arguments

pic	array including RGB.

Details

...

pic_randomtiles *Randomly crop an image*

Description

Function to tile an image into 4 different pieces.

Usage

```
pic_randomtiles(w, h, plot = TRUE)
```

Arguments

w	numeric, width of the original picture.
h	numeric, height of the original picture.
plot	logic, function plots layout if TRUE (default).

Details

Try multiple times to get the required image layout. Assign data frame to an object for subsequent use in gen_art. See example.

Examples

```
## Image width and height
h <- 100
w <- 120

## Three trials as an example
for(i in 1:3){
   pic_randomtiles(w, h, plot = TRUE)
   title("Split your image for better results...")
   text(w/2, h/2, paste("Trial", i), cex = 4, col = "white")
   Sys.sleep(1.5)
}
```

pic_unitiles *Uniformly crop an image*

Description

Function to uniformly crop an image into n pieces.

Usage

```
pic_unitiles(wh, nn, plot = TRUE)
```

Arguments

wh	vector, width and height of the original image.
nn	vector, number of columns and rows to split the image in.
plot	logic, function plots layout if TRUE (default).

Details

Function cuts remaining pixels if width/height is indivisible by columns/rows. Assign data frame to an object for subsequent use in gen_art. See example.

Examples

```
## Image width/height and number of colums/rows
wh <- c(89, 70)
nn <- c(6, 4)

## Calculation and plotting
pic_unitiles(wh, nn)
```

Index