Package 'trackdem'

September 24, 2021

Type Package

Title Particle Tracking and Demography

Version 0.6

Date 2021-09-24

Author Marjolein Bruijning, Marco D. Visser, Caspar A. Hallmann, Eelke Jongejans

Maintainer Marjolein Bruijning <mbruijning@princeton.edu>

Description Obtain population density and body size structure, using video material or image sequences as input. Functions assist in the creation of image sequences from videos, background detection and subtraction, particle identification and tracking. An artificial neural network can be trained for noise filtering. The goal is to supply accurate estimates of population size, structure and/or individual behavior, for use in evolutionary and ecological studies.

License GPL-2

URL https://github.com/marjoleinbruijning/trackdem

BugReports https://github.com/marjoleinbruijning/trackdem/issues

Encoding UTF-8

Depends

Imports png, neuralnet, raster, Rcpp, MASS, grDevices, graphics, stats, shiny

LinkingTo Rcpp, RcppArmadillo,

RoxygenNote 7.1.1

SystemRequirements Python (>=2.7), Libav, ExifTool

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-09-24 19:40:02 UTC

R topics documented:

createBackground . . . 2
createImageSeq . . . 3
findMaxCost . . . 5
findPixelRange . . . 6
findThreshold . . . 6
identifyParticles . . . 7
loadImages . . . 9
manuallySelect . . . 10
mergeTracks . . . 11
plot.TrDm . . . 12
print.summaryTrDm . . . 14
print.TrDm . . . 14
runBatch . . . 15
simulTrajec . . . 17
subtractBackground . . . 18
summary.TrDm . . . 19
testNN . . . 20
trackdem . . . 22
trackParticles . . . 22
update.particles . . . 24

Index **26**

`createBackground` *Background detection*

Description

`createBackground` detects the still background, containing all motionless pixels (non particles). Three different methods to detect the background can be used.

Usage

```
createBackground(colorimages, method = "mean")
```

Arguments

`colorimages`	Array of class 'TrDm' containing all images, obtained by `loadImages`.
`method`	Use `method='mean'` to calculate the mean value for each pixel and color. Use `method='powerroot'` to deflate dark values (note, this can only be used for dark particles on a light background). Use `method='filter'` to replace pixels in which movement has occurred with the mode of neighboring values. Note that `method='filter'` is computationally more intensive.

Value

Array of class 'TrDm' and 'colorimage' containing detected background.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain="square",
                    h=0.01,rho=0.9,
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
stillBack <- createBackground(allFullImages,method="mean")
plot(stillBack)

## End(Not run)
```

`createImageSeq` *Create image sequence*

Description

`createImageSeq` creates an image sequences (.png) using video files as input. All movies within a directory will be converted into an image sequence. For each movie, a new directory is created containing the recorded date and name of the movie.

Usage

```
createImageSeq(
  moviepath = "Movies",
  imagepath = "ImageSequences",
  x = 1920,
  y = 1080,
  fps = 15,
  nsec = 2,
  start = NULL,
  stop = NULL,
  ext = "MTS",
  libavpath = "avconv",
  exiftoolpath = "exiftool",
  pythonpath = "python",
  verbose = FALSE,
  logfile = FALSE
)
```

Arguments

moviepath	Path to existing directory containing the video files. By default, 'Movies' is used.
imagepath	Path to location of a directory in which image sequences should be saved. By default, 'ImageSequences' is used (and created if not existing).
x	Number of pixels in horizontal direction; default is 1920 (HD).
y	Number of pixels in vertical direction; default is 1080 (HD).
fps	Frames per second, default is 15.
nsec	Duration of movie that is exported, default is 2 seconds. When movie length is greater than nsec, the nsec seconds in the exact middle of the movie are exported.
start	Start time (in seconds) from where the video is converted (optional). By default, the nsec middle second of the video are used. If a a start time is specified and no stop time, nsec seconds starting from start are converted.
stop	End time (in seconds) from where the video is converted (optional). By default, the nsec middle second of the video are used. When an end time but no start time are specified, conversion starts at nsec seconds before end.
ext	The extension of the video. Default is 'MTS'. All formats supported by libav are accepted. To convert videos with different extensions, use for example c('MTS','mp4').
libavpath	Path to location where the executable file for libav can be found (named 'avconv.exe'), in case it is not found automatically, e.g. 'C:/Users/libav/usr/bin/avconv.exe'.
exiftoolpath	Path to location where the executable file for ExifTool can be found, in case it is not found automatically. For instance, use 'exiftool(-k).exe', if located in the working directory.
pythonpath	Path to location where the executable file for Python 2.7 can be found, in case it is not found automatically. For instance, use 'C:/Python27/python.exe'.
verbose	Logical. By default FALSE. Set to TRUE will print additional information.
logfile	Logical. By default FALSE. Set to TRUE will create a log file in the working directory.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
createImageSeq(moviepath="Movies",imagepath="ImageSequences",
               nsec=3,ext="AVI")

## End(Not run)
```

findMaxCost *Find maximum tracking cost*

Description

This function can help to find a appropriate maximum value for linking a particle to another particle (parameter L in function `trackParticles`)

Usage

```
findMaxCost(particles, frame = 1, colorimages = NULL)
```

Arguments

`particles`	Object of class 'particles', obtained using `identifyParticles`.
`frame`	Number specifying which frame to use. Default is frame 1.
`colorimages`	Array containing original color images. By default, the original color images are obtained from the global environment.

Value

Returns the number that is interactively chosen by the user. Use this value in `trackParticles`.

Author(s)

Marjolein Bruijning

Examples

```
## Not run:
partIden <- identifyParticles(sbg=allImages,
                              threshold=-0.05)
maxcost <- findMaxCost(partIden,frame=1)
records <- trackParticles(partIden,L=maxcost,R=1)

## End(Not run)
```

findPixelRange *Find pixel range*

Description

This function can help to find the minimum and maximum particle size in pixels, to use in `identifyParticles`.

Usage

```
findPixelRange(colorimages, frame = 1)
```

Arguments

`colorimages`	Array containing original color images.
`frame`	Number specifying which frame to use. Default is frame 1.

Author(s)

Marjolein Bruijning

Examples

```
## Not run:
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
stillBack <- createBackground(allFullImages)
allImages <- subtractBackground(stillBack)
findPixelRange(allFullImages,frame=10)

## End(Not run)
```

findThreshold *Find threshold*

Description

This function can help to find a threshold value to distinguish noise from particles of interest.

Usage

```
findThreshold(images, frame = 1, colorimages = NULL)
```

Arguments

`images`	Array containing images containing all moving particles, as obtained from `subtractBackground`.
`frame`	Number specifying which frame to use. Default is frame 1.
`colorimages`	Array containing original color images. By default, the original color images are obtained from the global environment.

Value

Returns the number that is interactively chosen by the user. Use this threshold value in `identifyParticles`.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain="square",
                    h=0.01,rho=0.9,
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
stillBack <- createBackground(allFullImages,method="mean")
allImages <- subtractBackground(stillBack)
thr <- findThreshold(allImages,frame=10)

## End(Not run)
```

---

identifyParticles *Identify moving particles*

Description

`identifyParticles` identifies moving particles using the subtracted images obtained from `subtractBackground`. Function uses Connected Component Labeling and obtains particle statistics based on code developed for the orphaned package SDMTools (written by Jeremy VanDerWal).

Usage

```
identifyParticles(
  sbg,
  threshold = -0.1,
  pixelRange = NULL,
  qthreshold = NULL,
  select = "dark",
  colorimages = NULL,
  autoThres = FALSE,
  perFrame = FALSE,
  frames = NULL
)
```

Arguments

sbg	Array containing images containing all moving particles, as obtained from subtractBackground.
threshold	Thresholds for including particles. A numeric vector containing three values; one for each color. Otherwise, supply one value which is to be used for all three colors. For a chosen quantile for each frame, use qthreshold. Default is threshold=-0.1, which works for dark particles on a light background. Alternatively, set autoThres below for an automatic threshold.
pixelRange	Default is NULL. Numeric vector with minimum and maximum particle size (area), used as a first filter to identify particles. Use if particle of interest are of a known size range (in pixels).
qthreshold	Default is NULL. Supply a value, to do thresholding based on quantile. Quantile is calculated for each frame separately.
select	Select dark particles ('dark'), light particles ('light'), or both ('both'), compared to background. Default is 'dark'.
colorimages	Array containing original color images. By default, the original color images are obtained from global environment.
autoThres	Logical. TRUE to get an automated threshold for each color layer. Default is FALSE.
perFrame	Logical. If autoThres=TRUE, set at TRUE to calculate a threshold for each frame separately. Default is FALSE. Note that is can be computationally intensive to calculate a threshold for each frame.
frames	When autoThres=TRUE and allFrames=FALSE, supply a numeric vector specifying over which frames the automated threshold should be calculated on (e.g. c(1,3,5,7,9,11) for all odd frames from 1 to 11).

Value

Returns a dataframe of class 'TrDm' and 'particles', containing particle statistics with identified particles for each frame

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain="square",
                    h=0.01,rho=0.9,
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
stillBack <- createBackground(allFullImages,method="mean")
```

```
allImages <- subtractBackground(stillBack)
partIden <- identifyParticles(allImages,threshold=-0.1,
                                    pixelRange=c(3,400))
plot(partIden)
summary(partIden)

## End(Not run)
```

`loadImages` *Load .png images*

Description

`loadImages` loads png images as three dimensional arrays. The objects created through the function can be used for image analysis.

Usage

```
loadImages(
  dirPictures,
  filenames = NULL,
  nImages = 1:30,
  xranges = NULL,
  yranges = NULL
)
```

Arguments

`dirPictures`	The path of the folder where the images can be found.
`filenames`	Default is `NULL`, or all files. If all files should NOT be loaded, here specify which files to use, as a character string.
`nImages`	Numeric vector specifying which images in the directory should be loaded; default is `1:30`.
`xranges`	By default the full image is loaded; specify to subset the number of columns.
`yranges`	By default the full image is loaded; specify to subset the number of rows.

Value

Array of class 'TrDm' and 'colorimages' containing all loaded images.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain="square",
                    h=0.01,rho=0.9,
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
plot(allFullImages)

## End(Not run)
```

`manuallySelect` *Manually identify true and false positives with a GUI.*

Description

`manuallySelect` opens a graphic user interface to create training data for a neural net by manually selecting true and false positives (i.e. correctly identified particles and noise, respectively).

Usage

```
manuallySelect(particles, colorimages = NULL, frames = NULL)
```

Arguments

`particles`	A data frame of class 'TrDm' with particle statistics for each frame, obtained by `identifyParticles`.
`colorimages`	An array with the original full color images, in order to plot on the original images. If `NULL`, the original color images are used, obtained from the global environment.
`frames`	A vector defining the frame(s) that should be used. Default is `NULL`; in that case the frame with the maximum number of identified particles is used.

Value

List containing three elements: true positives, false positives, and the evaluated frame.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain='square',
                    h=0.01,rho=0.9,movingNoise=TRUE,
                    parsMoving = list(density=20, duration=10, size=1,
                                      speed = 10, colRange = c(0,1)),
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
stillBack <- createBackground(allFullImages,method="mean")
allImages <- subtractBackground(stillBack)
partIden <- identifyParticles(allImages,threshold=-0.1,
                                    pixelRange=c(3,400))
# select the nframes with the most identified particles
nframes <- 3
frames <- order(tapply(partIden$patchID,partIden$frame,length),
                decreasing=TRUE)[1:nframes]
mId <- manuallySelect(particles=partIden,frame=frames)

## End(Not run)
```

`mergeTracks` *Merge track records*

Description

`mergeTracks` attempts to merge to two track objects as obtained by `trackParticles`.

Usage

```
mergeTracks(records1, records2, L = NULL, weight = NULL, logsizes = FALSE)
```

Arguments

`records1`	Object of class 'tracked', obtained using `trackParticles`.
`records2`	Object of class 'tracked', obtained using `trackParticles` that should be linked to `records1`.
`L`	Numeric. Maximum cost for linking a particle to another particle. When the cost is larger, particles will be not be linked (resulting in the begin or end of a segment). If `NULL`, the same `L` as used to create `records2` is used.
`weight`	Vector containing 3 weights to calculate costs. Depending on the study system user may want to value certain elements over others. Weights are ordered as follows; first number gives the weight for differences in x and y coordinates; second number gives the weight for particle size differences. Note that the difference

between the predicted location and the observed location is not taken into account in this function. If NULL, the same weights as used to create records2 is used.

logsizes Logical. Default is FALSE. Set to TRUE to take the natural logarithm of body sizes, when calculating the cost of linking two particles.

Value

A list of class 'TrDm' and 'records'. Use 'summary' and 'plot'.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
## Create image sequence
dir.create("images")
traj <- simulTrajec(path="images",
                    nframes=60,nIndividuals=20,domain="square",
                    h=0.01,rho=0.9,sizes=runif(20,0.004,0.006))
## Analyse first part
dir <- "images"
allFullImages1 <- loadImages (dirPictures=dir,nImages=1:30)
stillBack1 <- createBackground(allFullImages1)
allImages1 <- subtractBackground(bg=stillBack1)
partIden1 <- identifyParticles(sbg=allImages1,
                               pixelRange=c(1,500),
                               threshold=-0.1)
records1 <- trackParticles(partIden1,L=20,R=2)
## Analyse second part
allFullImages2 <- loadImages (dirPictures=dir,nImages=31:60)
stillBack2 <- createBackground(allFullImages2)
allImages2 <- subtractBackground(bg=stillBack2)
partIden2 <- identifyParticles(sbg=allImages2,
                               pixelRange=c(1,500),
                               threshold=-0.1)
records2 <- trackParticles(partIden2,L=20,R=2)
## Merge tracks
records <- mergeTracks(records1,records2)
plot(records,colorimages=allFullImages1,type="trajectories",incThres=10)

## End(Not run)
```

---

plot.TrDm plot *methods for class 'TrDm'.*

Description

plot methods for class 'TrDm'.

Usage

```
## S3 method for class 'TrDm'
plot(
  x,
  frame = 1,
  type = NULL,
  incThres = NULL,
  colorimages = NULL,
  cl = 1,
  path = NULL,
  name = "animation",
  libavpath = NULL,
  ...
)
```

Arguments

x	An object of class 'TrDm'.
frame	Choose which frame to be plotted. By default, frame=1.
type	Only for 'tracked' objects. By default, both trajectories and size distribution are plotted. Choose 'trajectories' to plot only trajectories on the original color image. Choose 'sizes' to only plot the particle size distribution. Choose 'animation' to create an .mp4 animation. Here, images are temporarily saved in path. Set name of file with argument name.
incThres	Minimum length of tracked segments for particles to be included. By default an automated threshold is calculated. Only for 'tracked' objects.
colorimages	Original color images. By default, original color images are obtained from the global environment.
cl	When plotting a subtracted background image, choose which color layer is plotted. By default, cl=1.
path	When creating an animation, choose directory in which images are saved temporarily, and where the animation should be saved.
name	of animation; by default 'animation'.
libavpath	Path to location where the executable file for libav can be found (named 'avconv.exe'), in case it is not found automatically, e.g. 'C:/Users/libav/usr/bin/avconv.exe'. Only required when creating an animation.
...	further arguments passed to plotRGB

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

print.summaryTrDm print *methods for class 'TrDm'.*

Description

print methods for class 'TrDm'.

Usage

```
## S3 method for class 'summaryTrDm'
print(x, ...)
```

Arguments

x	Object of class 'summaryTrDm'.
...	Further arguments passed to or from other methods.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

print.TrDm print *methods for class 'TrDm'.*

Description

print methods for class 'TrDm'.

Usage

```
## S3 method for class 'TrDm'
print(x, ...)
```

Arguments

x	Object of class 'TrDm'.
...	Further arguments passed to or from other methods.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

runBatch *Batch analysis*

Description

runBatch analyzes all image sequences in a specified directory. Use this function when settings have been optimized previously on a single or selection of movies/image sequences.

Usage

```
runBatch(
  path,
  settings = NULL,
  dirnames = NULL,
  nImages = 1:30,
  pixelRange = NULL,
  threshold = -0.1,
  qthreshold = NULL,
  select = "dark",
  nn = NULL,
  incThres = NULL,
  plotOutput = FALSE,
  plotType = "trajectories",
  L = 20,
  R = 2,
  weight = c(1, 1, 1),
  autoThres = FALSE,
  perFrame = FALSE,
  methodBg = "mean",
  frames = NULL,
  saveAll = FALSE
)
```

Arguments

path	A character vector of path name that contains all directories with image sequences.
settings	Object of class 'tracked' containing all optimized settings in attributes, as obtained from trackParticles. Alternatively, settings can be specified using arguments described below.
dirnames	If not all image sequences should be analyzed, specify which files to use as a character string.
nImages	See loadImages
pixelRange	See identifyParticles
threshold	See identifyParticles

qthreshold	See identifyParticles
select	See identifyParticles
nn	Name of artificial neural net if apply it to images. Default is NULL, resulting in no neural net being applied.
incThres	Minimum number of frames that a particle must be present. By default, automated estimate is used.
plotOutput	Default is FALSE. Set TRUE to plot results.
plotType	Default is 'trajectories'. Other options are 'sizes' and 'animation'.
L	See trackParticles
R	See trackParticles
weight	See trackParticles
autoThres	See identifyParticles
perFrame	See identifyParticles
methodBg	See createBackground
frames	See identifyParticles
saveAll	Logical. Set TRUE to save for each image sequence the full object obtained from loadImages. Default is FALSE.

Value

Dataframe with estimated population size for each image sequence.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

See Also

loadImages, createBackground, subtractBackground, identifyParticles, trackParticles.

Examples

```
## Not run:
## Simulate 3 image sequences
wd <- getwd()
folders <- paste0(rep("images",3),1:3)
populations <- c(15,25,50)
dir.create("./batchTest")
setwd("./batchTest")
for(i in 1:length(folders)){
  dir.create(folders[i])
  traj <- simulTrajec(path=folders[i],
                      nframes=30,nIndividuals=populations[i],
                      h=0.01,rho=0.9,
                      sizes=runif(populations[i],0.004,0.006))
}
setwd(wd)
```

```
batchpath <- "./batchTest"
results <- runBatch(path=batchpath,
                    nImages=1:30,threshold=-0.1,select='dark',
                    pixelRange=c(1,100),L=50,R=3,
                    incThres=10)
results

## End(Not run)
```

---

simulTrajec *Simulate trajectories and save as png files.*

Description

simulTrajec simulates movement trajectories within a bounded space, movements are set with speed (h) and may be correlated in direction (rho). Function simulates movement of particles in a video sequence of certain number of frames (nframes) in length. Images are saved as png files.

Usage

```
simulTrajec(
  nframes = 20,
  nIndividuals = 10,
  h = 0.02,
  rho = 0,
  domain = "square",
  correctBoundary = TRUE,
  sizes = stats::runif(nIndividuals) * 0.012 + 0.01,
  staticNoise = FALSE,
  movingNoise = FALSE,
  name = "trajectory",
  path = NULL,
 parsMoving = list(density = 10, duration = 10, size = 1, speed = 10, colRange = c(0,
    1)),
 parsStatic = list(density = 10, blur = TRUE, blurCoef = 0.025, sizes = NULL, col =
    "red"),
  width = 480,
  height = NULL
)
```

Arguments

nframes	Number of time frames(steps).
nIndividuals	Number of individual trajectories.
h	Displacement speed in pixels.
rho	Correlation parameter for angle of displacement.

domain	One of "square" or "circle", imposing a [0-1,0-1] rectangle domain, or a circular domain of radius 1, respectively. correct boundary ensure individual trajectories do not cross the domain
correctBoundary	Logical. TRUE to make sure that individuals cannot leave the image.
sizes	Vector of sizes for each simulated particle of length nIndividuals.
staticNoise	Logical. If TRUE, static noise is added.
movingNoise	Logical. If TRUE, moving noise is added.
name	Stem of the filename.
path	to location where the created images should be saved. By default the working directory is used.
parsMoving	List of parameters used to generate moving noise these include the density of noise particles (density), their duration (in n frames; duration), their size (size=1), their speed (speed=10) and the range or colors they are randomly drawn from (colRange=c(0,1)).
parsStatic	List of parameters used to generate static noise. These include the density (per image) of noise particles (density), whether spots look blurry (blur=TRUE or blur=FALSE), a blurring coefficient (blurCoef=0.025), and the size of the spots (with sizes).
width	of created png image. By default 480.
height	of create png image. If NULL, width is used.

Author(s)

Caspar A. Hallmann, Marjolein Bruijning & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence and save as png's in the working directory.
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain="square",
                    h=0.01,rho=0.9,
                    sizes=runif(20,0.004,0.006))

## End(Not run)
```

---

subtractBackground *Background subtraction*

Description

subtractBackground subtracts each image from a previously created still background. The objects created through the function contain all changing pixels (i.e. movement).

Usage

```
subtractBackground(bg, colorimages = NULL)
```

Arguments

bg	Array containing still background, as returned from createBackground.
colorimages	Array containing all frames, obtained by loadImages. Default is NULL, in this case the original images are used from the global environment.

Value

Returns array of class 'TrDm' and 'sbg' with same size as images, subtracted from background.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain="square",
                    h=0.01,rho=0.9,
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
stillBack <- createBackground(allFullImages,method="mean")
allImages <- subtractBackground(stillBack)
plot(allImages)

## End(Not run)
```

---

summary.TrDm — summary *methods for class 'TrDm'.*

Description

summary methods for class 'TrDm'.

Usage

```
## S3 method for class 'TrDm'
summary(object, incThres = NULL, funSize = stats::median, ...)
```

Arguments

object	an object of class 'TrDm'.
incThres	Minimum length of tracked segments for particles to be included. By default an automated threshold is calculated. Only for 'tracked' objects.
funSize	Statistic to be calculated to obtain particle sizes. By default median is used.
...	further arguments passed to or from other methods.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

---

testNN *Train, validate and test artificial neural networks*

Description

Fits multiple neural networks to a dataset; data set has been randomly assigned to each of three categories: train, validate and test. A final neural net is selected based on a fit statistic (either precision, recall or the F1-score). All neural networks are trained to the training dataset. Neural network may vary in the number of hidden layers. Classification thresholds are selected based on the validation data, and then the final neural network is selected based on the test data.

Usage

```
testNN(
  dat,
  stat = "F",
  maxH = 5,
  repetitions = 3,
  prop = c(8, 1, 1),
  predictors = NULL,
  pca = TRUE,
  thr = 0.95,
  ...
)
```

Arguments

dat	a previously constructed dataset obtained from manuallySelect.
stat	Fit statistic. May be "precision", "recall", or "F" for the harmonic mean of precision and recall.
maxH	maximum number of hidden layers to test note that more layers will require more time to fit.
repetitions	the number of repetitions for the neural network's training.
prop	the proportion or ratio for each class c(training, validation,test).

predictors	Optional. A set of custom predictors for the neural network. Default uses all columns in dat.
pca	Logical. TRUE by default. Should the set of predictors be compressed to the most informative? In short, should a principal component analysis be conducted to select axis that explain at least a fraction thr (see below) of the variance in the full set of predictors?
thr	Threshold for pca (above).
...	additional parameters, passed to neuralnet.

Details

The neural networks may be selected based on precision, recall or a F1-score (default). In binary classification, precision is the number of correct positive results divided by the number of all positive predictions. Recall is the number of correct positive results divided by the number of positive results that could have been returned if the algorithm was perfect. A F1 score (F-score/ F-measure) is a statistical measure of accuracy. F1 scores considers both the precision and the recall. A F1 score may be seen as a weighted average (harmonic mean) of the precision and recall. Precision, recall and F1 scores are at best 1 and at worst 0.

Value

Returns trained artificial neural net.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain='square',
                    h=0.01,rho=0.9,movingNoise=TRUE,
                    parsMoving = list(density=20, duration=10, size=1,
                                      speed = 10, colRange = c(0,1)),
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
stillBack <- createBackground(allFullImages,method="mean")
allImages <- subtractBackground(stillBack)
partIden <- identifyParticles(allImages,threshold=-0.1,
                                  pixelRange=c(3,400))
nframes <- 3
frames <- order(tapply(partIden$patchID,partIden$frame,length),
                decreasing=TRUE)[1:nframes]
mId <- manuallySelect(particles=partIden,frame=frames)
finalNN <- testNN(dat=mId,repetitions=10,maxH=4,prop=c(6,2,2))
```

```
summary(finalNN)

## End(Not run)
```

`trackdem` *trackdem - Particle Tracking and Demography*

Description

trackdem - Particle Tracking and Demography

Author(s)

Marjolein Bruijning, Caspar A. Hallmann, Marco D. Visser, Eelke Jongejans

---

`trackParticles` *Track particles*

Description

`trackParticles` reconstructs trajectories by linking particles.

Usage

```
trackParticles(
  particles,
  L = 50,
  R = 2,
  weight = c(1, 1, 1),
  costconstant = FALSE,
  logsizes = FALSE
)
```

Arguments

`particles`	Object of class 'particles', obtained using `identifyParticles`.
`L`	Numeric. Maximum cost for linking a particle to another particle. When the cost is larger, particles will be not be linked (resulting in the begin or end of a segment). Default set at `50`.
`R`	Integer. Link to how many subsequent frames? Default set at `2`.

weight	Vector containing 3 weights to calculate costs. Depending on the study system, users may want to value certain elements over others. For instance, when individuals can vary in size over frames (which happens when objects move away or towards a camera) the "size" weight may be decreased. Weights are ordered as follows; first number gives the weight for differences in x and y coordinates; second number gives the weight for particle size differences; third number gives the difference between the predicted location and the observed location. The latter is calculated using the location of the identified particle in the previous frame.
costconstant	Logical. Default is FALSE. This increases the maximum cost L proportional to the number of images in between two frames (when R > 1). Set to TRUE keeps maximum cost L constant for all 1:R frames.
logsizes	Logical. Default is FALSE. Set to TRUE to take the natural logarithm of body sizes, when calculating the cost of linking two particles.

Value

A list of class 'TrDm' and 'records'. Use 'summary' and 'plot'.

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain="square",
                    h=0.01,rho=0.9,
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
stillBack <- createBackground(allFullImages,method="mean")
allImages <- subtractBackground(stillBack)
partIden <- identifyParticles(allImages,threshold=-0.1,
                                  pixelRange=c(3,400))
records <- trackParticles(particles,L=40,R=2)
summary(records)
plot(records,type="trajectories")

## End(Not run)
```

`update.particles` *Update identified particles.*

Description

Apply trained artificial neural network to particleStat object.

Usage

```
## S3 method for class 'particles'
update(object, neuralnet, pca = TRUE, colorimages = NULL, sbg = NULL, ...)
```

Arguments

`object`	Object of class 'nnTrackdemObject'.
`neuralnet`	Trained neural net obtained from `testNN`
`pca`	Logical. By default `TRUE`, indicating that a principal component analysis is performed on the predictors.
`colorimages`	An array with the original full color images, in order to plot on the original images, obtained by `loadImages`. By default the original color images are used.
`sbg`	Images subtracted from background, as obtained by `subtractBackground`. By default, the original subtracted images are used.
`...`	further arguments passed to or from other methods.

Value

Data frame class 'particles', containing updated particle statistics (excluding particles that have been filtered out by the neural net).

Author(s)

Marjolein Bruijning, Caspar A. Hallmann & Marco D. Visser

Examples

```
## Not run:
dir.create("images")
## Create image sequence
traj <- simulTrajec(path="images",
                    nframes=30,nIndividuals=20,domain='square',
                    h=0.01,rho=0.9,movingNoise=TRUE,
                    parsMoving = list(density=20, duration=10, size=1,
                                      speed = 10, colRange = c(0,1)),
                    sizes=runif(20,0.004,0.006))
## Load images
dir <- "images"
allFullImages <- loadImages (dirPictures=dir,nImages=1:30)
```

```
stillBack <- createBackground(allFullImages,method="mean")
allImages <- subtractBackground(stillBack)
partIden <- identifyParticles(allImages,threshold=-0.1,
                                    pixelRange=c(3,400))
nframes <- 3
frames <- order(tapply(partIden$patchID,partIden$frame,length),
                decreasing=TRUE)[1:nframes]
mId <- manuallySelect(particles=partIden,frame=frames)
finalNN <- testNN(dat=mId,repetitions=10,maxH=4,prop=c(6,2,2))
partIdenNN <- update(particles=partIden,neuralnet=finalNN)

## End(Not run)
```

Index

createBackground, 2, *16*, *19*
createImageSeq, 3

findMaxCost, 5
findPixelRange, 6
findThreshold, 6

identifyParticles, 7, *10*, *15, 16*, *22*

loadImages, *2*, 9, *15, 16*, *19*, *24*

manuallySelect, 10
mergeTracks, 11

plot.TrDm, 12
plotRGB, *13*
print.summaryTrDm, 14
print.TrDm, 14

runBatch, 15

simulTrajec, 17
subtractBackground, *6–8*, *16*, 18, *24*
summary.TrDm, 19

testNN, 20, *24*
trackdem, 22
trackParticles, *11*, *15, 16*, 22

update.particles, 24