

Package ‘image.CornerDetectionF9’

October 13, 2022

Type Package

Title Find Corners in Digital Images with FAST-9

Version 0.1.0

Maintainer Jan Wijffels <jwijffels@bnosac.be>

Description An implementation of the “FAST-9” corner detection algorithm explained in the paper ‘FASTER and better: A machine learning approach to corner detection’ by Rosen E., Porter R. and Drummond T. (2008), available at <[arXiv:0810.2434](https://arxiv.org/abs/0810.2434)>. The package allows to detect corners in digital images.

License BSD_2_clause + file LICENSE

URL <https://github.com/bnosac/image>

Imports Rcpp (>= 0.12.8)

Suggests pixmap, magick

LinkingTo Rcpp

RoxygenNote 7.1.0

NeedsCompilation yes

Author Jan Wijffels [aut, cre, cph],
BNOSAC [cph],
Julien Cayzac [ctb, cph]

Repository CRAN

Date/Publication 2020-07-27 12:30:21 UTC

R topics documented:

image.CornerDetectionF9-package	2
image_detect_corners	2
Index	4

image.CornerDetectionF9-package

Find Corners in Digital Images with FAST-9.

Description

An implementation of the "FAST-9" corner detection algorithm explained at <http://www.edwardrosten.com/work/fast.html>.
The package allows to detect corners in digital images.

See Also

[image_detect_corners](#)

image_detect_corners *Find Corners in Digital Images with FAST-9.*

Description

An implementation of the "FAST-9" corner detection algorithm explained at <http://www.edwardrosten.com/work/fast.html>.

Usage

```
image_detect_corners(x, threshold = 50L, suppress_non_max = FALSE)
```

Arguments

x	a matrix of image pixel values in the 0-255 range.
threshold	positive integer where threshold is the threshold below which differences in luminosity between adjacent pixels are ignored. Think of it as a smoothing parameter.
suppress_non_max	logical

Value

as list of the found corners with the x/y locations

Examples

```
library(pixmap)
imagelocation <- system.file("extdata", "chairs.pgm", package="image.CornerDetectionF9")
image <- read.pnm(file = imagelocation, cellres = 1)
x <- image@grey * 255
corners <- image_detect_corners(x, 80)
plot(image)
points(corners$x, corners$y, col = "red", pch = 20, lwd = 0.5)
```

```
##
## image_detect_corners expects a matrix as input
## if you have a jpg/png/... convert it to pgm first or take the r/g/b channel
library(magick)
x <- image_read(system.file("extdata", "hall.jpg", package="image.CornerDetectionF9"))
x
image <- image_data(x, channels = "Gray")
image <- as.integer(image, transpose = TRUE)
image <- drop(image)
corners <- image_detect_corners(image, threshold = 80)

plt <- image_draw(x)
points(corners$x, image_info(x)$height - corners$y, col = "red", pch = 20, lwd = 0.5)
dev.off()
plt

## same but now converting to portable grey map
f <- tempfile(fileext = ".pgm")
library(magick)
x <- image_read(system.file("extdata", "hall.jpg", package="image.CornerDetectionF9"))
x <- image_convert(x, format = "pgm", depth = 8)
image_write(x, path = f, format = "pgm")

image <- read.pnm(f, cellres = 1)
corners <- image_detect_corners(image@grey * 255, 80)
plot(image)
points(corners$x, corners$y, col = "red", pch = 20, lwd = 0.5)

file.remove(f)
```

Index

`image.CornerDetectionF9`-package, [2](#)
`image_detect_corners`, [2](#), [2](#)