FAST algorithm for corner detection

FAST algorithm is used to detect corner in a image in real-time. FAST algorithm is proposed by Edward Rosten and Tom Drummond in their paper [1] (all images are taken from this paper).

On the follow picture, we see the result of the FAST algorithm.

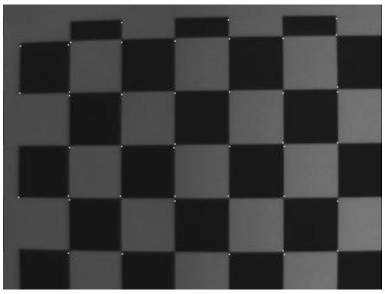


Figure 1: Fast - Corner detection

Properties

Name	Type	Description
enable	bool	Enable the processing

Equivalence

Matlab

```
I = imread('yourimage.jpg');
I = rgb2gray(I);
corners = detectFASTFeatures(I);
imshow(I); hold on;
plot(corners.selectStrongest(50));
```

https://nl.mathworks.com/help/vision/ref/detectfastfeatures.html

OpenCV

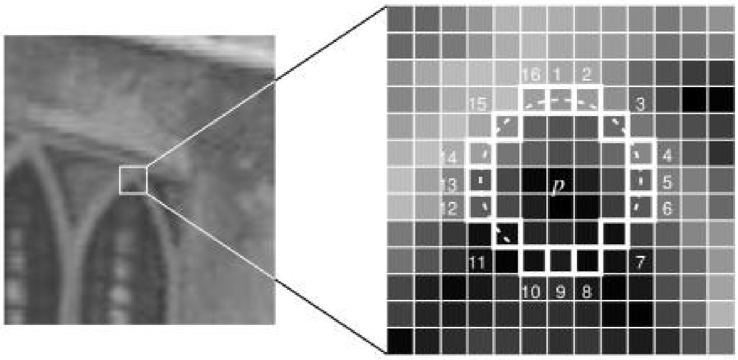
```
import numpy as np
import cv2
from matplotlib import pyplot as plt

img = cv2.imread('simple.jpg',0)

# Initiate FAST object with default values
fast = cv2.FastFeatureDetector()

# find and draw the keypoints
kp = fast.detect(img,None)
img2 = cv2.drawKeypoints(img, kp, color=(255,0,0))

# Print all default params
print "Threshold: ", fast.getInt('threshold')
print "nonmaxSuppression: ", fast.getBool('
   nonmaxSuppression')
print "neighborhood: ", fast.getInt('type')
print "Total Keypoints with nonmaxSuppression: ", len(kp)

cv2.imwrite('fast_true.png',img2)
```

Explanation of the FAST Algorithm[1]

1. Select a pixel p in the image which is to be identified as an interest point or not. Let its intensity be I_p.

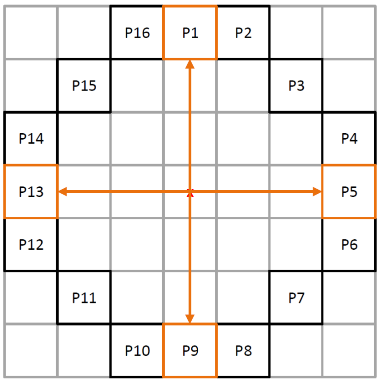
2. Select appropriate threshold value t.

3. Consider a circle of 16 pixels around the pixel under test.



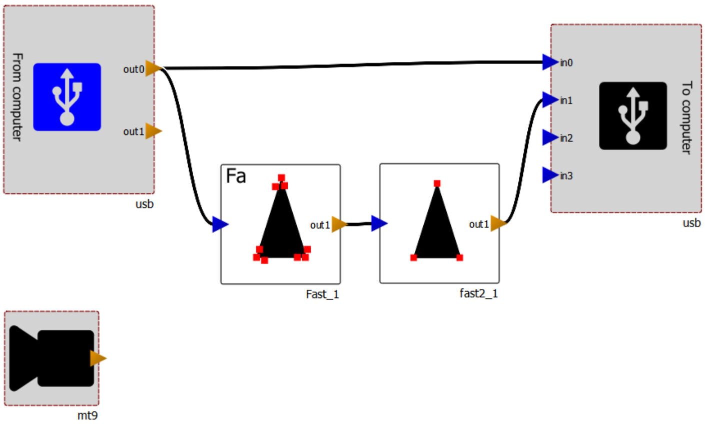
4. Now the pixel p is a corner if there exists a set of n contiguous pixels in the circle (of 16 pixels) which are all brighter than $I_p + t$ or all darker than $I_p + t$. n was chosen to be 12.

5. A high-speed test was proposed to exclude a large number of non-corners. This test examines only the four pixels at 1, 9, 5 and 13 (First 1 and 9 are tested if they are too brighter or darker. If so, then checks 5 and 13). If p is a corner, then at least three of these must all be brighter than $I_p + t$ or darker than $I_p - t$. If neither of these is the case, then p cannot be a corner. The full segment test criterion can then be applied to the passed candidates by examining all pixels in the circle. This detector in itself exhibits high performance, but there are several weaknesses :



[1] http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_fast/py_fast.html

after this, let's applying a filter to decrease the number of corner candidates. The full project on GPStudio is shown on the follow figure.


From computer
out0
out1
usb
Fa
out1
Fast_1
out1
fast2_1
in0
in1
in2
in3
To computer
usb
mt9