EEE-6512 Image Processing and Computer Version

Homework #5

Wenxuan Wang

UF gatorlink username: wenxuanwang

Ufmail: wenxuanwang@ufl.edu

UFID:64118211

Textbook Question

Prob 7.3 Suppose we wish to construct a Gaussian pyramid with $n=3$ images per octave

(a) What is the downsampling factor?

(b) How should σ^2 be chosen to ensure that the overall smoothing between octaves is $\sigma^2 = 1.2$?

From book page P330 [1]

we know that each reduction by a factor of two is known as an octave

(a) If the downsampling factor is $\sqrt[n]{2} = 2^{\frac{1}{n}}$, then there are n images per octave [1]

From the problem, it is with $n=3$ images per octave

so the dampling factor is $2^{\frac{1}{3}} = 1.2599$

(b) Since repeated convolutions with a Gaussian are equivalent to a single convolution with a Gaussian whose variance is the sum of the individual variances, we define $\sigma'^2 = \frac{1}{n}\sigma^2$ to ensure that the overall smoothing between octaves is the same as between consecutive levels equation 7.2 (book P329)

so the variance $\sigma'^2 = \frac{1}{3}\sigma^2 = \frac{1}{3} \times 1.2 = 0.4$

Prob. 7.4 Suppose we wish to construct a Laplacian pyramid with $n=5$ images per octave

(a) What should be the variance ratio ρ in order to ensure that each octave is convolved with the same sequence of variances relative to the image size?

(b) What variance should be applied for pyramid levels 1, 2 and 3 (i.e. what are $\sigma_0^2\ \sigma_1^2\ \sigma_2^2$)?

From book page P332

(a) We know that to ensure that each octave is convolved with the same sequence of variances relative to the image size, the variance ratio should be set to $\rho = 2^{\frac{1}{n}}$ [1]

so $\rho = 2^{\frac{1}{5}} \approx 1.1487$

(b) A reasonable choice for the initial variance is $\sigma_0^2 = \frac{1}{n}(0.5)$

So let $\sigma_0^2 = \frac{0.5}{5} = 0.1$

$\rho^2 = (2^{\frac{1}{5}})^2 = 2^{\frac{2}{5}} = 1.3195$

so $\sigma_1^2 = \rho^2\sigma_0^2 = 1.3195 \times 0.1 = 0.13195$

$\sigma_2^2 = 1.3195 \times 0.13195 = 0.1741$

Prob 7.5 Explain why the casuality criterion is important in computing the scale space

Book P333

Casuality criterion is among scale-space axioms. It can ensure that the number of local extrema does not increase as we proceed to coarser levels of scale. In other words, the maxima are flattened while the minima are raised. [1]

That is, all local extrama found by the scale space computation are due to the image itself.

Prob. 7.7 Explain why the Canny edge detector fails at the intersection of two lines.

The assumption of the canny edge detector is that a single line within the immediate neighborhood. When it's at the intersection of two lines, there are multiple lines, which does not meet the assumption. Therefore, it fails at the intersection of two lines.

Prob. 7.8 — Perform non-maximal suppression on the following gradient magnitude and phase images. (Compute results only for the inner 3 × 3 array.)

3	3	3	3	3
3	10	9	5	3
3	20	8	7	3
3	5	30	10	3
3	3	3	3	3

magnitude

0	0	0	0	0
0	0	$-\frac{4\pi}{8}$	0	0
0	$\frac{10\pi}{8}$	$\frac{3\pi}{4}$	0	0
0	$\frac{4\pi}{8}$	0	$\frac{2\pi}{8}$	0
0	0	0	0	0

phase

① value 10, angle 0 => horizontal (red), and the neighbors 3 $\overset{max}{\underbrace{10}}$ 9

return Glocalmax [1] -> the maximum value 10

② value 9, angle $-\frac{4\pi}{8} \Rightarrow -\frac{\pi}{2} \Rightarrow -\frac{\pi}{2}+2\pi = \frac{3\pi}{2}$ => vertical (blue) 3 $\overset{max}{\underbrace{9}}$ 8

return the local maximam 9

③ value 5, angle 0 => horizontal, $\overset{max}{\underbrace{9}}$ 5 3 ∴ Glocalmax ← 0 [1]

④ value 20, angle $\frac{10\pi}{8} = \frac{5\pi}{4}$ => down-right (yellow) 3 20 $\underbrace{30}_{max}$ ∴ 0

⑤ value 8, angle $\frac{3\pi}{4}$ => down-left (green) 5 $\overset{max}{\underbrace{8}}$ 5 => return local max 8

⑥ value 7, angle 0 => horizontal (red) $\underbrace{8}_{max}$ 7 3 => Glocalmax ← 0

⑦ value 5, angle $\frac{4\pi}{8} = \frac{\pi}{2}$ => vertical (blue) => $\overset{max}{\underbrace{20}}$ 5 3 => Glocalmax ← 0

⑧ value 30, angle 0 => horizontal (red) => 5 $\overset{max}{\underbrace{30}}$ 10 => return local max 30

⑨ value 10, angle $\frac{2\pi}{8} = \frac{\pi}{4}$ => down-right (yellow) => 8 $\underbrace{10}_{max}$ 3 => return local max 10

so the computed result is
$$\begin{bmatrix} 10 & 9 & 0 \\ 0 & 8 & 0 \\ 0 & 30 & 10 \end{bmatrix}$$

Prob 7.9 Explain the localization - detection tradeoff

Book P337-338

The Location - detection tradeoff is a dilemma. [1]

The question is how to select filter for computing the gradient or, if the Gaussian derivative is used, what value to choose for the standard deviation. As it turns out, a large sigma yields a better SNR, but a smaller sigma yields a more accurate location for the edge. [1] In other words, a large region of support allows an edge detector to be robust to noise, but it distracts the detector from the most important pixels near the edge itself.

Prob 7.10 Why is the Marr - Hildreth operator a bad edge detector?

Book P340

The primary drawback is that it's isotropic, meaning that it smooths across as well as along edges, as opposed to the gradient vector, which can be used to treat pixels differently across and along the edge. [1] This reduces its ability to precisely locate the edge.

Prob 7.14 How is the Harris corner detector better than the Moravec interest operator?

Book P344 [1]

Harris features are largely invariant to rotation. It is isotropic, but Moravec interest operator is not.

Prob 7.15 Explain why you would want to perform non-maximum suppression after computing the Harris cornerness measure on the pixels of an image. How would you modify the non-maximum suppression produce of Algorithm 7.2 to apply to Harris?

① Non-maximum suppression maintains only the pixels with a large coreness measure compared with their neighbors. Pixels with small coreness are generally not of interest. ② Modify in two ways: a. ignore the direction, this step is required b. this way is optional. Increasing the window size within which pixels are compared. This aims to reduce the number of pixels retained by the computation.

Prob. 7.18 — Given the following directional derivatives, compute Harris and Tomasi-Kanade cornerness measures of the central pixel, assuming a 3×3 window and uniform weighting for all the pixels in the window.

$$I_x = \begin{bmatrix} -5 & -9 & 5 \\ 7 & 3 & -8 \\ -6 & 9 & 3 \end{bmatrix} \qquad I_y = \begin{bmatrix} 2 & -7 & -6 \\ -1 & 8 & 9 \\ -5 & 2 & 3 \end{bmatrix}$$

Book P344 - 345

$\Sigma x = 25 + 81 + 25 + 49 + 9 + 64 + 36 + 81 + 9 = 379$

$\Sigma y = 4 + 49 + 36 + 1 + 64 + 81 + 25 + 4 + 9 = 273$

$\Sigma xy = -5 \times 2 + (-9) \times (-7) + 5 \times (-6) + 7 \times (-1) + 3 \times 8 + (-8) \times 9 + (-6) \times (-5) + 2 \times 9 + 3 \times 3$

$= 25$

from (7.30) $\quad$ cornerness $= \det(\Sigma) - k(\mathrm{trace}(\Sigma))^2 \quad$ (Harris)

$= \Sigma x \Sigma y - \Sigma^2 xy - k(\Sigma x + \Sigma y)^2$

Let $k = 0.04$

$= 379 \times 273 - 25^2 - (379 + 273)^2 \times 0.04$

$= 102842 - 0.04 \times 425104$

$= 85838$

$\lambda = \frac{1}{2}\left((\Sigma x + \Sigma y) \pm \sqrt{(\Sigma x - \Sigma y)^2 + 4\Sigma^2 xy}\right)$

$= \frac{1}{2}\left((379 + 273) \pm \sqrt{(379 - 273)^2 + 4 \times 625}\right)$

$= \frac{1}{2}\left(652 \pm \sqrt{13736}\right) \quad \Rightarrow$ get $\lambda_1 = 384.6$, $\lambda_2 = 267.4$

from (7.33) $\quad$ cornerness $= \min(\{\lambda_1, \lambda_2\}) \quad$ (Tomasi-Kanade)

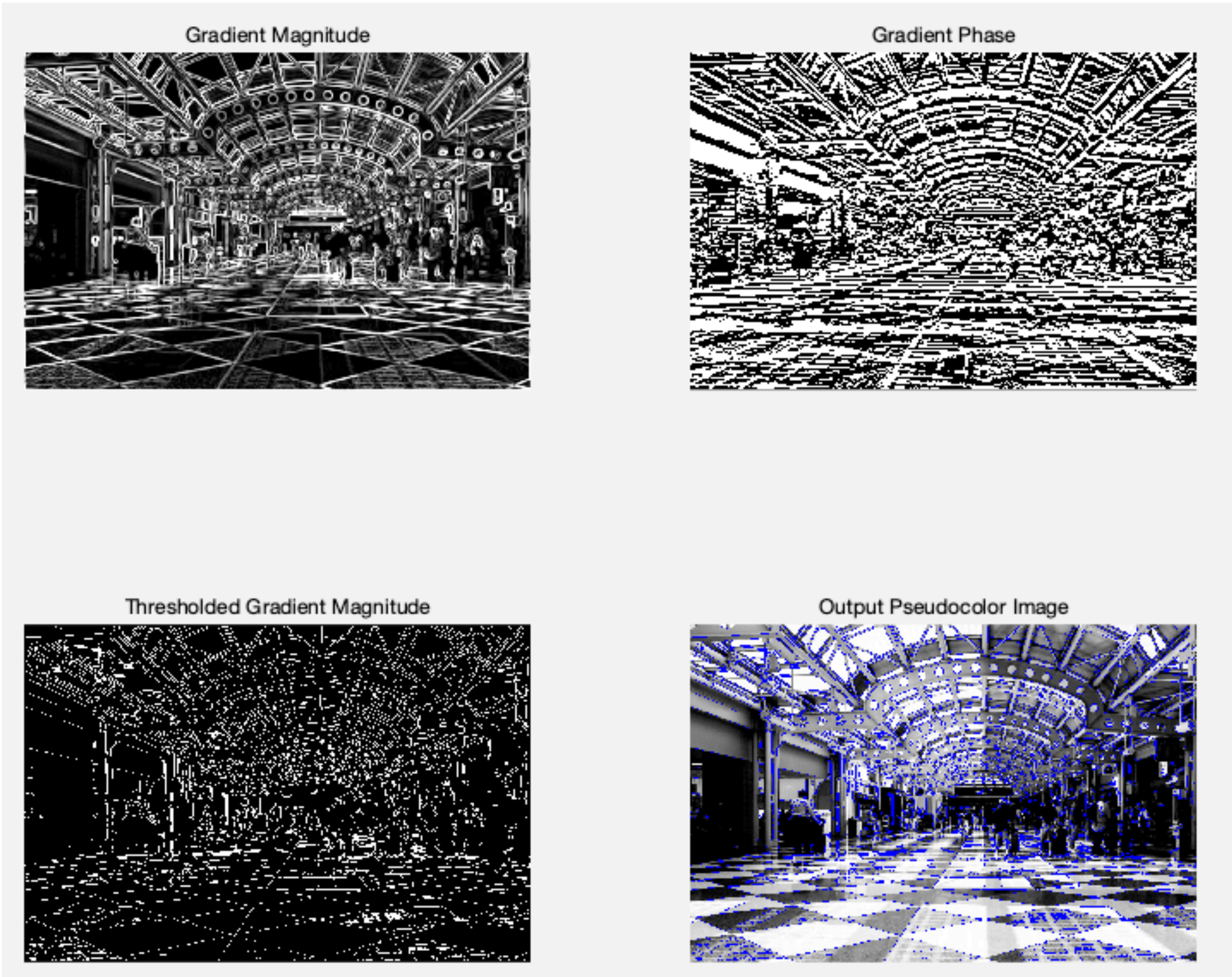
$\therefore$ cornerness $= 267.4$

[1]

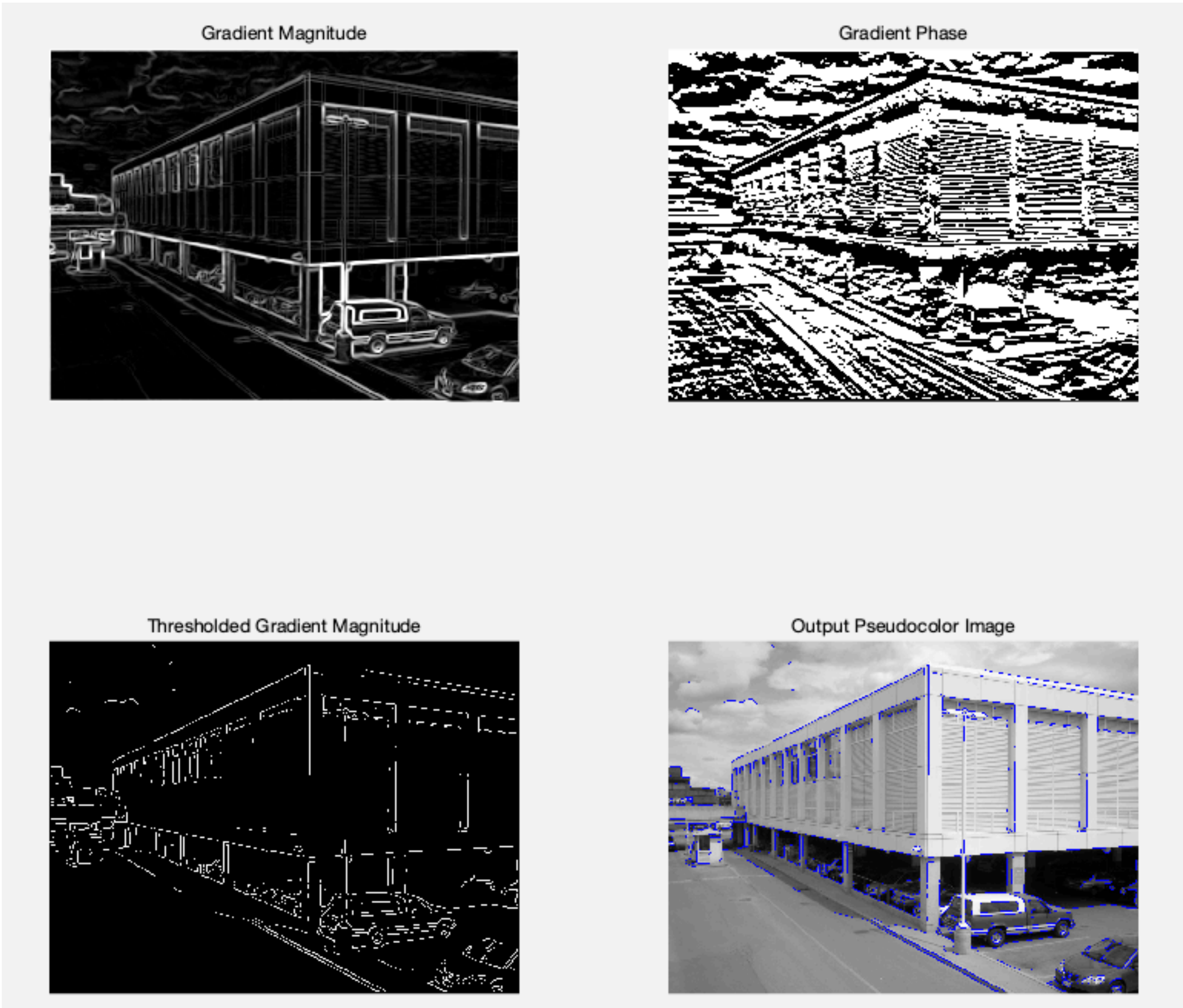
Coding part

myCannyEdgeDetector

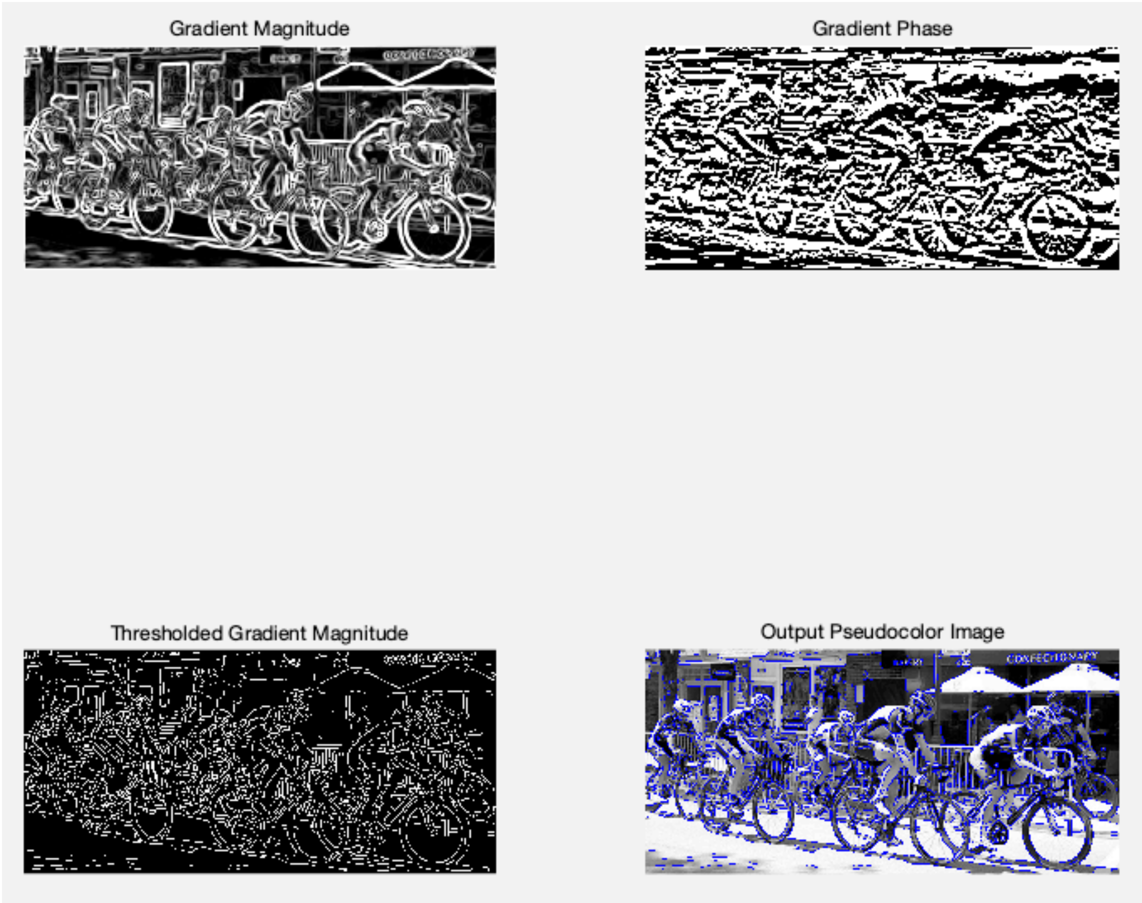
img1



Img2

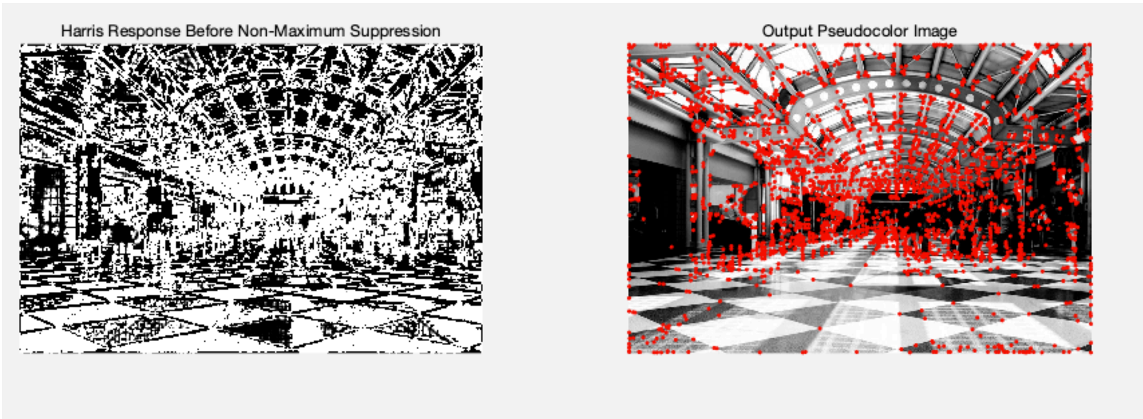

Gradient Magnitude
Gradient Phase
Thresholded Gradient Magnitude
Output Pseudocolor Image

Img3

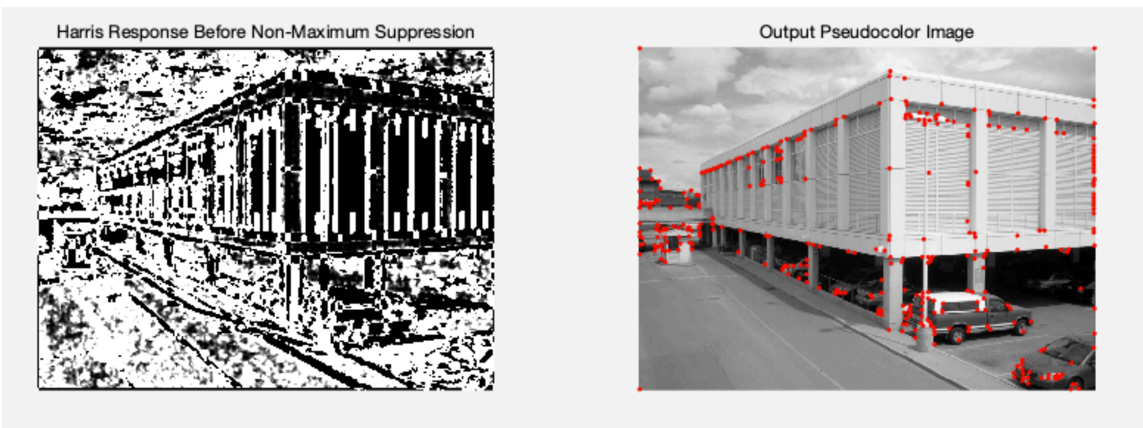

Gradient Magnitude
Gradient Phase
Thresholded Gradient Magnitude
Output Pseudocolor Image

myHarrisCornerDetector

img 1



Img2



img3

Coding question

My Canny Edge Detector

1. Parameters accept: low-bound threshold and high-bound threshold.
   Purpose: Both of them help find the edge. If the result < low-bound threshold, the pixel keep original color. If the result > high-bound threshold, the pixel turn blue. If it is located between these two bounds, check neighbor.
2. I choose Sobel. It can detect edges and their directions. It can provide with smoothing (which reduces noise) concurrently. And it's more robust.
3. Perform well: It has a high running seep. It does not cost much time. It can find most of the edges. Reason: thresholds are good.
   Perform bad: Sometimes, it cannot detect the weak edges. Or sometimes, it makes mistakes to find edges. It may find edges which are not exactly the edges. Reason: Thresholds may not be so precise.
4. How to improve
   Set the value according to the image pixels, so we can get better filtering;
   Improve the calculation of gradient's size and direction.
   Try the algorithm for more times to test different threshold.

my Harris Corner Detector:

1. Parameter accept: k. It is used in Harris coreness calculation equation. This equation measures coreness using the trace and determinant of the matrix. [1]
   Purpose: It is a small factor whose value is typically recommended to be the vicinity of 0.04. (And I set 0.04). [1]
2. Perform well: The running speed is high. It does not cost much time. The changes in brightness and contrast seem to have no effect on the corner points. Reson: correct code and constant k.
   Perform bad: It seems not to find all the corners. Reason: the filter may not be built well. Or something working not well with gradient.
3. How to improve: Improve Gaussian filter; Detect similarity between center point and other gray pixel values in n neighbors, and then check whether it can exactly be the corner.

Reference

[1] B. H. Brown, R. H. Smallwood, D. C. Barber, P. V Lawford, and D. R. Hose, *Image processing and analysis*. 2004.