Computer Vision
Jacobs University Bremen
Dragi Kamov

Course: CO22-320671
October 20th, 2018

Homework 2

Problem 1

Solution:

Problem 2

Solution:

a) My code for this problem can be found in *crossCorrelation.m*.
Also here is the code added for this section:

```
a = normxcorr2(template, photo);
figure, surf(a), shading flat
correlationImg = [a];
```

The peak occurs there because that is the place on the image which Matlab finds the most similarities.

b) The code is the same from *a)* just that we are doing the correlation now with the slightly larger image. We can notice that the slightly larger image doesn't really correlate and that is because it contains more pixels and then when it compares it doesn't scale the pixels but it checks on that scale.
This tells us that cross-correlation has a limitation when it comes just to comparing because not always the scales of the images are going to be the same.
Cross-correlation would be good at pattern recognition, weather forecasting, signal detection, monitoring traffic.

Problem 3

Solution:

a) Let's say that the magnitude of the origin derivative is:
$M = \sqrt{dx^2 + dy^2} = |dx|$
Then the magnitude of rotated derivative is:
$M' = \sqrt{(dx\cos\theta)^2 + (dx\sin\theta)^2} = |dx|$
From this we can see that the magnitude doesn't change so they can be detected by the same Canny edge detector.

b) If fake edges appear, it means we come up with extra edges, so we should increase the high threshold. And, if long edges are broken into short segments, it means we lose some weak edges, so we should decrease the low threshold.

Problem 4

Solution:
The solution to this problem can be found in *difference_gaussian.m*, but also it can be seen here:

```
syms x
syms y
syms g(x,y)

g(x,y) = exp(-(x.^2) / (2 * y.^2)) / (y * sqrt(2 * pi));

% Part a)
g_2der = diff(g,x,2);

figure('Name','2nd Derivative'), fplot(g_2der(x,1),[-7 7]);

% Part b)
```

```
syms d(x,y,k)

d(x,y,k) = ((exp(-(x.^2) / (2 * y.^2 * k)) / (y * k * sqrt(2 * pi))) - (exp(-(x
    .^2)/(2 * y.^2)) / (y * sqrt(2 * pi))))/ (k * y -y)

figure('Name','k=1.2'), fplot(d(x,1,1.2),[-7 7])
figure('Name','k=1.4'), fplot(d(x,1,1.4),[-7 7])
figure('Name','k=1.6'), fplot(d(x,1,1.6),[-7 7])
figure('Name','k=1.8'), fplot(d(x,1,1.8),[-7 7])
figure('Name','k=2.0'), fplot(d(x,1,2.0),[-7 7])
```

Problem 5

Solution: