COMP 558 - Report of Assignment 1

Manoosh Samiei
McGill University

6 October 2019

1 Question 1

Part a

We know that every function *x(t)* can be written as:

$$x(t) = \int_{-\infty}^{\infty} x(\tau)\delta(t-\tau)d\tau \tag{1}$$

The above equation is easy to interpret. The $\delta(t)$ function has a value of 1 when t is 0, and is equal to 0 for other values of t. Hence $\delta(t-\tau)$ is equal to 1 only if t is equal to τ. Therefore the integral can be written as: $\int_{-\infty}^{\infty} x(\tau)d\tau$, which is clearly equal to $x(t)$.

Now, we assume that the response of a linear time invariant system to impulse function $\delta(t)$ is $h(t)$. Also we assume that the response of the same LTI system to input $x(t)$ is $y(t)$. Therefore, if we pass the left side of equation(1) through the system, we reach to $y(t)$, and if we pass the right side of equation(1) through the system we reach to: $\int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$. Note that in the integral, $x(\tau)$ is not changed since it is a constant value multiplied by impulse function for each value of τ and does not vary with t. Also since the system is linear we can say that the output of $Ax(t)$ is equal to $Ay(t)$, which clarifies why we multiplied $x(\tau)$ by $h(t-\tau)$, after passing it through the system. Now we can rewrite the equation(1) after it is passed through the system as:

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau \tag{2}$$

The above integral is equal to the convolution of $x(t)$ and $h(t)$. Hence:

$$y(t) = x(t) * h(t) \tag{3}$$

Therefore we can obtain the response of an LTI system to an arbitrary input $x(t)$ using the impulse response of that system.

Part b

To prove that Laplace operator is rotation invariant, we first need to introduce two other orthogonal directions in addition to Cartesian coordinates x and y. The first direction can be $r = x\cos(\theta) + y\sin(\theta)$. r is a vector starting from the origin of coordinate system and having an angle of theta from the positive direction of x axis. We also define another vector $r' = x\cos(\theta - \pi/2) + y\sin(\theta - \pi/2)$ which is orthogonal to r. By using trigonometric equations, we can also write r' as: $r' = x\sin(\theta) - y\cos(\theta)$. Now we would like to prove the following:

$$\frac{\partial^2 I}{\partial y^2} + \frac{\partial^2 I}{\partial x^2} = \frac{\partial^2 I}{\partial r^2} + \frac{\partial^2 I}{\partial r'^2} \tag{4}$$

The step by step proof is as follows:

$$\begin{aligned} \frac{\partial I}{\partial y} &= \frac{\partial I}{\partial r}\frac{\partial r}{\partial y} + \frac{\partial I}{\partial r'}\frac{\partial r'}{\partial y} \\ \frac{\partial I}{\partial x} &= \frac{\partial I}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial I}{\partial r'}\frac{\partial r'}{\partial x} \end{aligned} \tag{5}$$

$$\begin{aligned} \frac{\partial I}{\partial y} &= \frac{\partial I}{\partial r}\sin\theta - \frac{\partial I}{\partial r'}\cos\theta \\ \frac{\partial I}{\partial x} &= \frac{\partial I}{\partial r}\cos\theta + \frac{\partial I}{\partial r'}\sin\theta \end{aligned} \tag{6}$$

$$\begin{aligned} \frac{\partial^2 I}{\partial y^2} &= \frac{\partial}{\partial y}(\frac{\partial I}{\partial r}\sin\theta - \frac{\partial I}{\partial r'}\cos\theta) \\ \frac{\partial^2 I}{\partial x^2} &= \frac{\partial I}{\partial x}(\frac{\partial I}{\partial r}\cos\theta + \frac{\partial I}{\partial r'}\sin\theta) \end{aligned} \tag{7}$$

$$\begin{aligned}
\frac{\partial}{\partial r}\frac{\partial I}{\partial y} &= \frac{\partial}{\partial r}(\frac{\partial I}{\partial r}\sin\theta - \frac{\partial I}{\partial r'}\cos\theta)\\
\frac{\partial}{\partial r'}\frac{\partial I}{\partial y} &= \frac{\partial I}{\partial r'}(\frac{\partial I}{\partial r}\sin\theta - \frac{\partial I}{\partial r'}\cos\theta)\\
\frac{\partial}{\partial r}\frac{\partial I}{\partial x} &= \frac{\partial}{\partial r}(\frac{\partial I}{\partial r}\cos\theta + \frac{\partial I}{\partial r'}\sin\theta)\\
\frac{\partial}{\partial r'}\frac{\partial I}{\partial x} &= \frac{\partial I}{\partial r'}(\frac{\partial I}{\partial r}\cos\theta + \frac{\partial I}{\partial r'}\sin\theta)
\end{aligned} \tag{8}$$

Now by replacing equations(8) into equations(7) we will have the following:

$$\begin{aligned}
\frac{\partial^2 I}{\partial y^2} &= \frac{\partial^2 I}{\partial r^2}\sin^2\theta - 2\frac{\partial^2 I}{\partial r\partial r'}\cos\theta\sin\theta + \frac{\partial^2 I}{\partial r'^2}\cos^2\theta\\
\frac{\partial^2 I}{\partial x^2} &= \frac{\partial^2 I}{\partial r^2}\cos^2\theta + 2\frac{\partial^2 I}{\partial r\partial r'}\cos\theta\sin\theta + \frac{\partial^2 I}{\partial r'^2}\sin^2\theta
\end{aligned} \tag{9}$$

Adding two above equations:

$$\frac{\partial^2 I}{\partial y^2} + \frac{\partial^2 I}{\partial x^2} = (\sin^2\theta + \cos^2\theta)(\frac{\partial^2 I}{\partial r^2} + \frac{\partial^2 I}{\partial r'^2}) \tag{10}$$

We know that $\sin^2\theta + \cos^2\theta = 1$, therefore we reach to:

$$\frac{\partial^2 I}{\partial y^2} + \frac{\partial^2 I}{\partial x^2} = \frac{\partial^2 I}{\partial r^2} + \frac{\partial^2 I}{\partial r'^2} \tag{11}$$

Therefore we proved that Laplace operator is rotation invariant.

Part c

At first, Marr and Hildreth stated in their paper [1] that to detect edges the zero crossings of a second derivative taken in the direction perpendicular to the local direction of the edge should be used. Then they stated that Under specific conditions this direction coincides with that in which the zero crossing has maximum slope.(I think the reason why we are trying to find maximum slope is that it shows a stronger intensity change or a stronger edge. since in a stronger edge we have a bigger and sharper maximum in first derivative and a sharper slope in zero crossing of second derivative.) This condition was called by them: "the condition of linear variation". The condition states that the intensity variation near and parallel to the line of zero-crossings should locally be linear. This condition can be written as a mathematical theorem and be proved as follows:

Theorem 1: Let l be an open line segment of the y-axis, containing the origin 0 . Suppose that $f(x, y)$ is twice continuously differentiable and that $N(l)$ is an open two dimensional neighbourhood of l. Assume that $\frac{\partial^2 f}{\partial x^2} = 0$ on l. Then, if $\frac{\partial f}{\partial y}$ is constant in $N(l)$, the slope of the second directional derivative taken perpendicular to l (i.e., the slope of $\frac{\partial^2 f}{\partial y^2}$) is greater than the slope of the zero-crossing along any other line through 0.

Proof: Consider the line segment $\Omega = (r\sin\theta, r\cos\theta)$ for fixed θ and values of r sufficiently small that Ω lies entirely within $N(l)$ Now writing f_{xx} for $\frac{\partial^2 f}{\partial x^2}$ etc., we have:

$$\begin{aligned}
(\frac{\partial^2 f}{\partial \Omega^2})_{r,\theta} &= (f_{xx}\cos^2\theta + f_{xy}2\sin\theta\cos\theta + f_{yy}\sin^2\theta)_{r,\theta}\\
&= (f_{xx}\cos^2\theta)_{r,\theta}
\end{aligned} \tag{12}$$

since the condition of the theorem that f_y be constant implies that f_{xy} and f_{yy} are both zero. As required, therefore, the above quantity is zero at $r = 0$ and has maximum slope when $\theta = 0$.

In the second step (which is **the main solution of assignment's question**), Marr and Hildreth discuss that convolutions are relatively expensive, and it would lessen the computational burden if their number could be reduced by using just one orientation independent operator. The only orientation-independent second-order differential operator is the Laplacian. The conditions that using Laplacian is mathematically sound is as follows:

Theorem 2: Let $f(x, y)$ be a real-valued, twice continuously differentiable function on the plane. Let l be an open line segment along the axis $x = 0$. Then the two conditions:
1) $\nabla^2(f) = 0$ on l, and
2) $\frac{\partial^2 f}{\partial x^2} = 0$ on l
are equivalent if and only if $f(0, y)$ is constant or linear on l.

Proof: If $f(0,y)$ is linear on l, $\frac{\partial^2 f}{\partial y^2} = 0$ on l. Hence, $\nabla^2(f) = 0$, there implies that $\frac{\partial^2 f}{\partial x^2} = 0$ on l too. Conversely, if $\frac{\partial^2 f}{\partial x^2} = \nabla^2(f) = 0$ on l. Then, $\frac{\partial^2 f}{\partial y^2} = 0$ on l, and so $f(0,y)$ varies at most linearly on l.

These conditions are weaker than the condition of linear variation, which was met in theorem 1, and they state that if the intensity variations in G*I (smoothed image) is linear along but not necessarily near to a line of zero-crossings, then the zero-crossing will be detected and accurately located by the zero values of Laplacian. Thus, if intensity varies along a segment in a very non-linear way, the Laplacian, and hence the operator $\nabla^2(G)$ will see the zero-crossing displaced to one side.

2 Question 2

Part a

The 2D Gaussian function is as follows:

$$G(x,y) = \frac{1}{2\pi\sigma^2} \exp -\frac{x^2+y^2}{2\sigma^2} \tag{13}$$

I visualized this 2D Gaussian for two choices of sigma using surface plotting function "surf" in MATLAB.

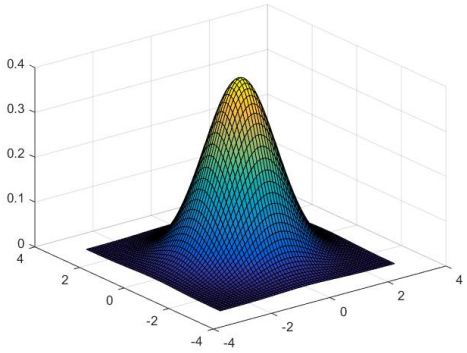

Figure 1. 2D Gaussian function with sigma=1 and N=5.

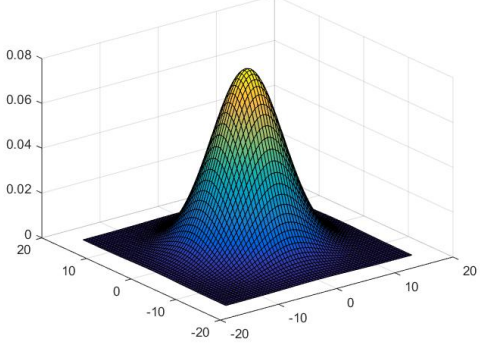

Figure 2. 2D Gaussian function with sigma=5 and N=31.

We can observe from the pictures that as we increase the sigma (standard deviation) the Gaussian function starts to flatten, meaning that its pick value starts to decrease (from around 0.3 to 0.06), and cover a larger area on x-y plane. Also, N was selected in a way to cover roughly +/-3 sigma around the pick location, to include 98 percent of the area under a Gaussian function. Therefore for sigma=1 and sigma=5, N=2M+1 is chosen roughly 6 times sigma.

Part b The Laplacian of Gaussian function is as follows:

$$\begin{aligned} \nabla^2 G(x,y,\sigma) &= \frac{\partial^2 G(x,y,\sigma)}{\partial x^2} + \frac{\partial^2 G(x,y,\sigma)}{\partial y^2} \\ \nabla^2 G &= \frac{-1}{\pi\sigma^4}(1 - \frac{x^2+y^2}{2\sigma^2}) \exp -\frac{x^2+y^2}{2\sigma^2} \end{aligned} \tag{14}$$

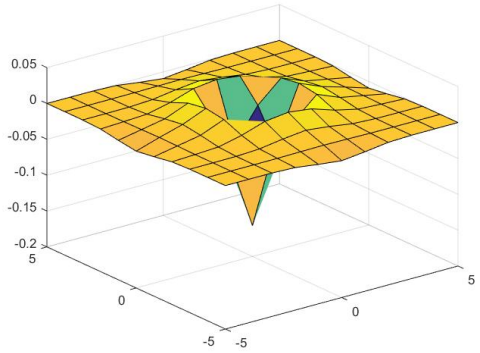

Figure 1. Laplacian of Gaussian function with sigma=1 and N=11.

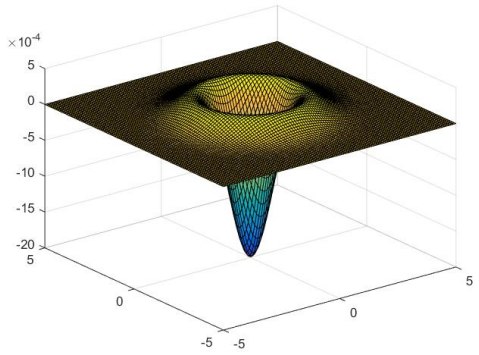

Figure 3. Laplacian of Gaussian function plotted by more points(smaller step)(sigma=1 and N=11.)

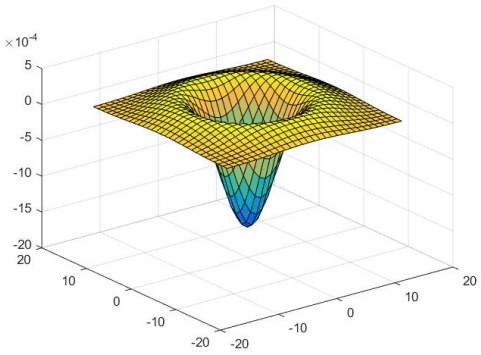

Figure 2. Laplacian of Gaussian function with sigma=5 and N=31.

As we increase sigma of Gaussian, the area under Laplacian of Gaussian filter increases.(It covers a larger area on y=0 plane)

Part c

The function of a Gabor filter is defined by a sinusoidal wave multiplied by a Gaussian function. These two components may be formed into a complex number or used individually. The complex format of a 2D Gabor filter is as follows [2]:

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \exp\left(i(2\pi\frac{x'}{\lambda} + \psi)\right) \quad (15)$$

In which x' and y' are as follows:

$$x' = x\cos\theta + y\sin\theta \; y' = -x\sin\theta + y\cos\theta \quad (16)$$

Also, λ represents the wavelength of the sinusoidal factor,θ represents the orientation of Gabor function, ψ is the phase offset, σ is the standard deviation of the Gaussian envelope and γ is the spatial aspect ratio, and specifies the ellipticity of the support of the Gabor function.

In order to visualize this filter, we need to separate the real and imaginary parts of the function or in another terms the odd and even parts of it:

The real(even) component:

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi\frac{x'}{\lambda} + \psi\right) \quad (17)$$

The imaginary(odd) component:

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \sin\left(2\pi\frac{x'}{\lambda} + \psi\right) \quad (18)$$

To simplify these formulas, we assume that the phase of sine and cosine is zero and omit ψ term from equations. Also we can assume that for both x' and y' we have the same sigma, thus removing spatial aspect ratio (γ) from equations.

As question says, We will set the sigma equal to wavelength λ.

Now for Lambda=2, Angle= $\pi/4$, and N=9, we plot both even and odd parts of a 2D Gabor filter:

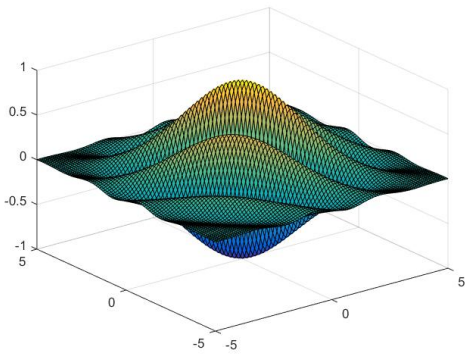

Figure 1. Even part of 2D Gabor filter, front view (Lambda=2, Angle=$\pi/4$, and N=9).

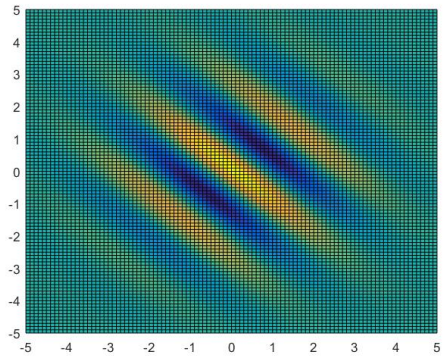

Figure 2. Even part of 2D Gabor filter, top view (Lambda=2, Angle=$\pi/4$, and N=9).

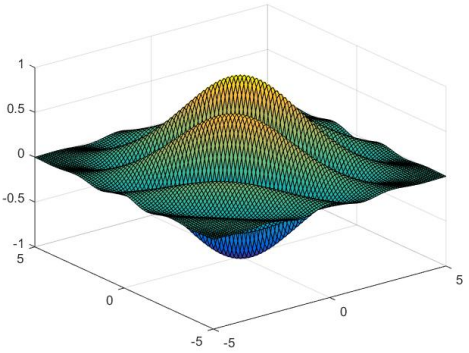

Figure 3. Odd part of 2D Gabor filter, front view (Lambda=2, Angle=$\pi/4$, and N=9).

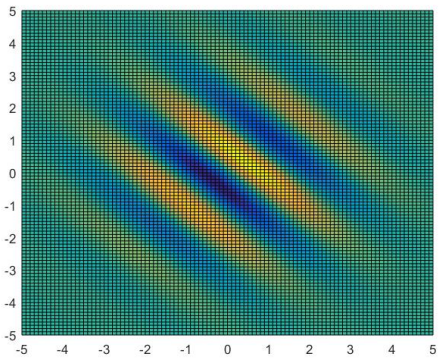

Figure 4. Odd part of 2D Gabor filter, top view (Lambda=2, Angle=$\pi/4$, and N=9).

Now we change angles (to $\pi/2$ and π) while keeping the lambda constant to realize the effect of changing the angle:

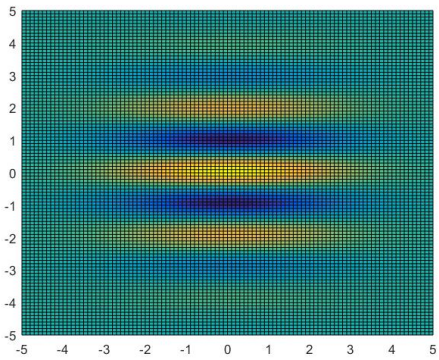

Figure 5. Even part of 2D Gabor filter, top view (Lambda=2, Angle=$\pi/2$, and N=9).

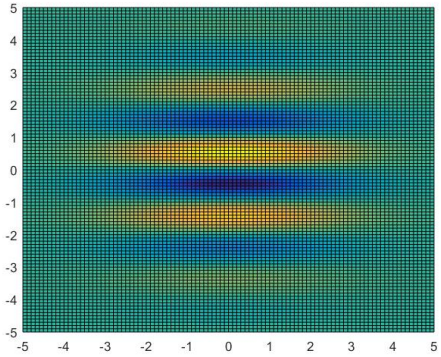

Figure 6. Odd part of 2D Gabor filter, top view (Lambda=2, Angle=$\pi/2$, and N=9).

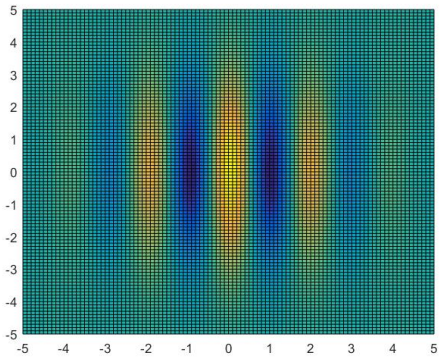

Figure 7. Even part of 2D Gabor filter, top view (Lambda=2, Angle=π, and N=9).

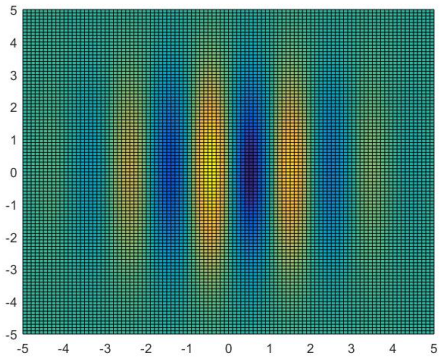

Figure 8. Odd part of 2D Gabor filter, top view (Lambda=2, Angle=π , and N=9).

We can easily observe from the pictures that changing angle will rotate the filter. Also, the filter has the same rotation in angles zero, π and 2π.

Now we change values of λ to 5 and 10, while keeping the angle equal to $\pi/4$:

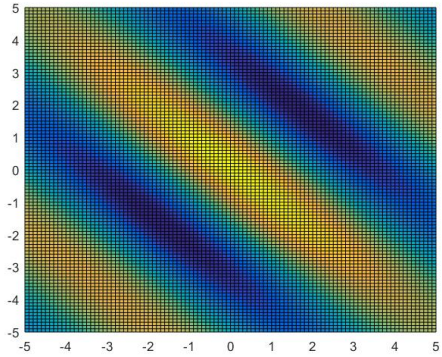

Figure 9. Even part of 2D Gabor filter, top view (Lambda=5, Angle=$\pi/4$, and N=9).

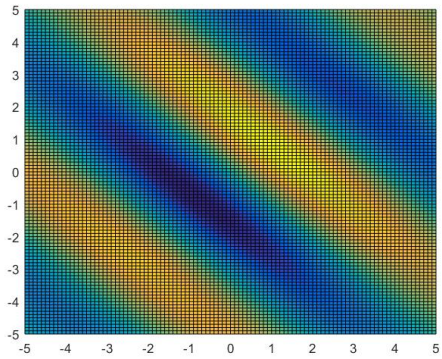

Figure 10. Odd part of 2D Gabor filter, top view (Lambda=5, Angle=$\pi/4$, and N=9).

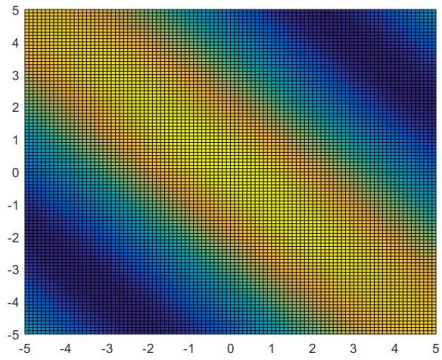

Figure 11. Even part of 2D Gabor filter, top view (Lambda=10, Angle=$\pi/4$, and N=9).

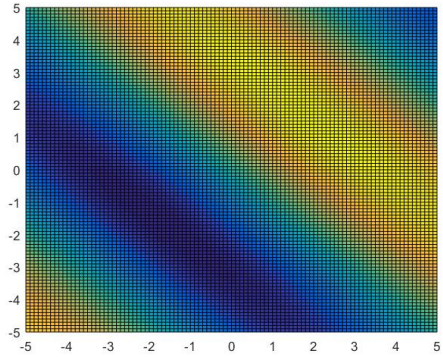

Figure 12. Odd part of 2D Gabor filter, top view (Lambda=10, Angle=$\pi/4$, and N=9).

It is observable from the pictures that increasing lambda, which is the wavelength of the sinusoidal factor, will make picks and valleys of the filter to be distributed over a larger area. Hence the area under picks and valleys of the filter increases.

Note: I removed some of the front-view pictures form this part to make the report more concise. But removed pictures are present in the project folder.

3 Question 3

Part a

In this problem, we first use zero padding to handle pixels at the image boundary, which adds a black rectangular frame around our image.

After convolving the padded image with the laplacian of the Gaussian, we will see that: when we increase sigma, the edges become thicker and we are less able to locate edges accurately. Also we will miss smaller edges in the picture. I tested different amounts for sigma and realized that with sigmas more than 1.5, edges become undesirably thick. As we decrease sigma (smaller than roughly 0.6) we become more locally sensitive to intensity changes, and we may have some false positives in our results. Meaning that we are too sensitive to intensity change that we have selected more edges than actually exist.

Another point is that as we increase N which is the kernel size(width x height of the filter mask), we detect more edges(features) from the input image. The size of kernel depends on the sigma of filter and therefore for each sigma we should adjust the

appropriate kernel size. Sometimes increasing the kernel size makes our filter too sensitive to intensity change.

Effect of Sigma on the thickness of the edges:

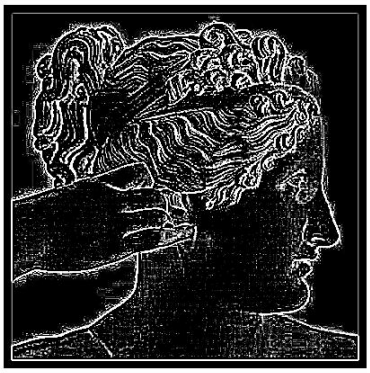

Figure 1. The output of convolving Laplacian of a Gaussian with an image(sigma=0.6, N=11)

Figure 2. The output of convolving Laplacian of a Gaussian with an image(sigma=2, N=31)

Effect of Kernel size on edge detection:

Figure 3. The output of convolving Laplacian of a Gaussian with an image(sigma=1 N=9)

Figure 4. The output of convolving Laplacian of a Gaussian with an image(sigma=1, N=13)

Figure 5. The output of convolving Laplacian of a Gaussian with an image(sigma=1, N=19)

Applying on another image:

Figure 6. original image

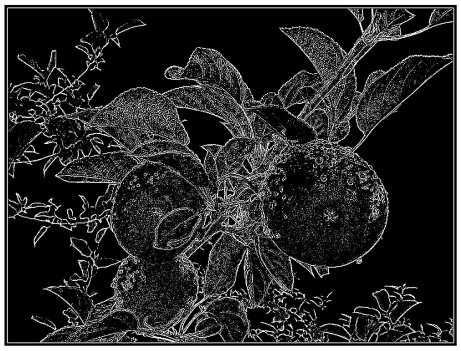

Figure 7. The output of convolving Laplacian of a Gaussian with an image(sigma=1, N=13)

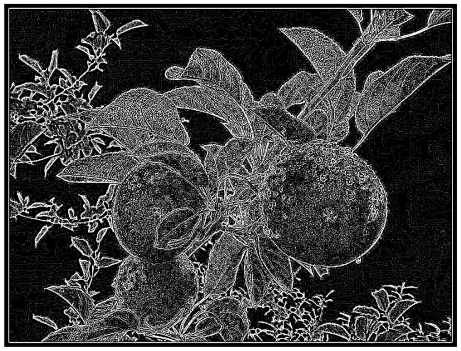

Figure 7. The output of convolving Laplacian of a Gaussian with an image(sigma=1, N=19)

Figure 8. The output of convolving Laplacian of a Gaussian with an image(sigma=2, N=19)

Part b

In previous part, I plotted the result of convolving an image with laplacian of Gaussian. In this part, I detect and plot the zero crossings of the convolution. To detect zero crossing, I defined two loops that iterate over each pixel of the image and check whether each pair of pixels around it have different signs. When they have different signs, it means that the pixel has crossed zero and is an edge. Then I assigned the value of its neighbour pixel (same row, right column) to it. At the end, we have a matrix that shows the edges. While iterating over pixels, We only check the two pixels that can form a line with the center pixel. The below picture clarifies this concept:

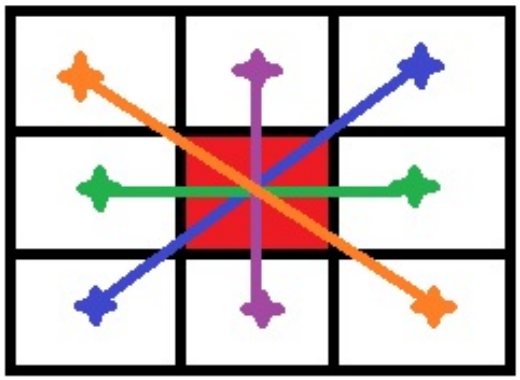

Figure 1. checking the sign of 4 pair of pixels around a particular pixel in the image

There is also an automatic way to find zero crossing by using this instruction in MATLAB: *edge(output,'zerocross',.5);*, (the algorithm seems to work better than this function.)

Plotting edges at different blurring scales (different sigmas) with zero crossing method,by both line drawing and overlaying edges on the original image(a threshold of 0.7 is defined to prevent excessive edges):

Figure 1. line drawing edges by finding zero-crossings (sigma=1, N=15)

Figure 2. overlaying edges in yellow color on the original image (sigma=1, N=15)

Figure 3. line drawing edges by finding zero-crossings (sigma=2, N=15)

Figure 4. overlaying edges in yellow color on the original image (sigma=2, N=15)

Figure 5. line drawing edges by finding zero-crossings (sigma=3, N=15)

Figure 6. overlaying edges in yellow color on the original image (sigma=3, N=15)

Figure 7. line drawing edges by finding zero-crossings (sigma=0.5, N=11)

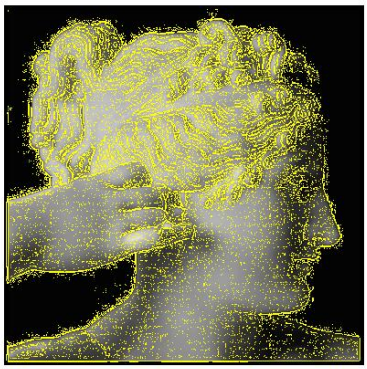

Figure 8. overlaying edges in yellow color on the original image (sigma=0.5, N=11)

Applying experiments to the other image:

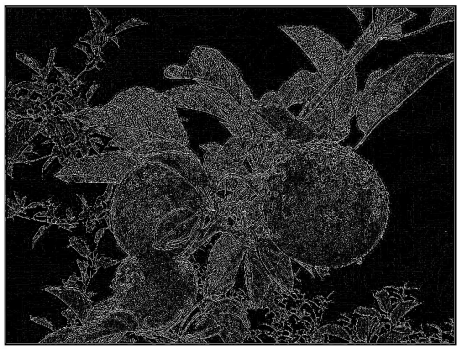

Figure 9. line drawing edges by finding zero-crossings (sigma=1, N=15)

Figure 10. overlaying edges in yellow color on the original image (sigma=1, N=15)

Figure 11. line drawing edges by finding zero-crossings (sigma=2, N=15)

Figure 12. overlaying edges in yellow color on the original image (sigma=2, N=15)

Part c

It is observable from pictures of section b that as we increase the blurring scale (using larger sigma) we can extract larger features(edges) from the image and we are less sensitive to minor details (edges)in an image.

Also, we can observe that when there is a linear variation in intensity along an edge, the edge is detected successfully; Otherwise, edges might not be com-

pletely detected. When all image intensity changes uniformly (e.g. adding "equally" a certain value to all image pixels to become brighter), this is known as linear intensity variation. This condition was an assumption that we made when we used Laplacian of Gaussian, instead of taking a second derivative from our image in an orthogonal direction to the edge.

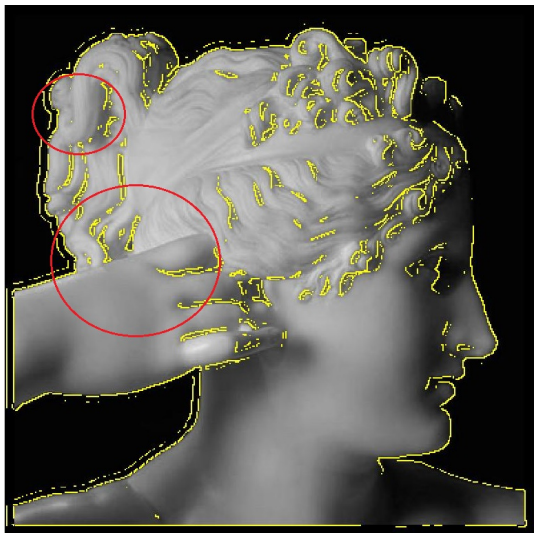

Figure 1. parts of the image that edge detection fails (sigma=2, N=15)

In Paolina image, we can see that in the upper intersection of the hand and head of Paolina, the hand boundary is not successfully detected. In this location, the variation of intensity along the edge is not linear. Since around the other parts of the hand, there is a black background or a dark shadow of the hand on the hair of Paolina, but in the upper boundary of hand, the intensity becomes so brighter. This is a non-linear intensity variation that brings failure to this method. This also happens in some other parts of the image, like Paolina's hair.

However, when we blur image in a smaller scale, by using a smaller sigma, we can observe smaller edges in the image. Looking into a narrower region around an edge, sometimes brings more linearity to intensity changes along it. I think that is why we can detect hand's upper boundary when we use a smaller sigma.(with sigma=1, and N=15) (the picture is provided in the previous section.)

Figure 2. parts of the image that edge detection fails (sigma=2, N=15)

As for the paolina image, we can see in the apple picture when we have non linear intensity variation, we fail to detect the edges properly. This has happened in multiple locations in the above picture. The location that one apple covers some part of another apple in the bottom of the image is the most noticeable part that the edge is not successfully detected.

Part d

Gabor filters are orientation dependant. I applied 3 different orientation (0, $\pi/4$, $\pi/2$) of Gabor filters to Paolina image. Each orientation of the filter only detects edges that are placed in that orientation. After finding zero crossings of the convolved image with Gabor filter we find edges of different orientations as follows:

Figure 1. detected edges at angle 0 by gabor filter (lambda=3, N=15)

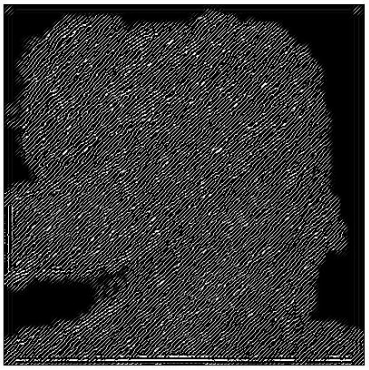

Figure 2. detected edges at angle $\pi/4$ by gabor filter (lambda=3, N=15)

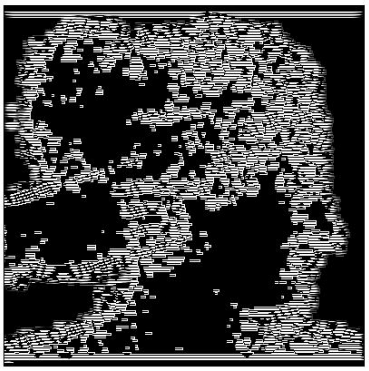

Figure 3. detected edges at angle $\pi/2$ by gabor filter (lambda=3, N=15)

Hence to detect edges in all directions, we should combine different orientations of Gabor filter.

The wavelength(λ) of Gabor filter also plays an important role in detecting features of an image. as we increase λ we detect larger features from an image, this is in correlation with standard deviation (σ) of laplacian of Gaussian filter. This was predictable since Gabor filter has a Gaussian component whose sigma is set equal to lambda in the function of Gabor filter. We can see the effect of sigma's value on feature extraction of filter:

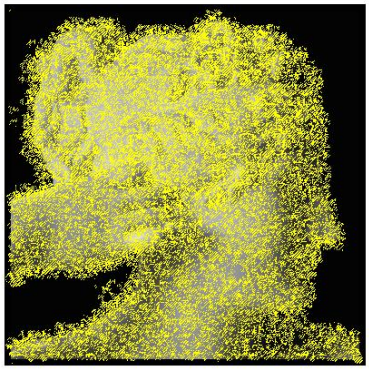

Figure 4. lambda=2 (edges detected by finding zero crossings - gabor filter)

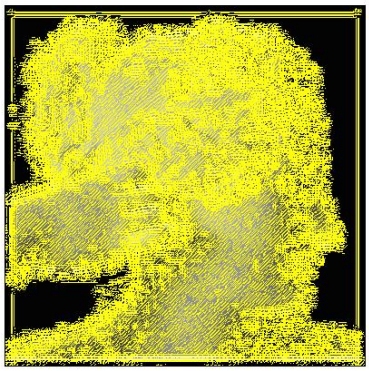

Figure 5. lambda=3 (edges detected by finding zero crossings - gabor filter)

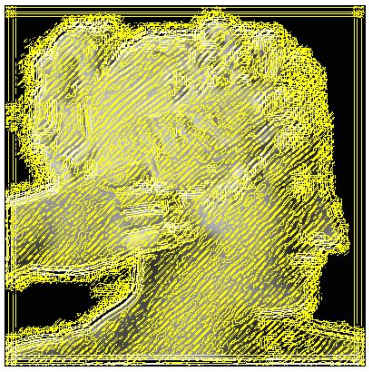

Figure 6. lambda=5 (edges detected by finding zero crossings - gabor filter)

Figure 7. lambda=6 (edges detected by finding zero crossings - gabor filter)

Figure 8. lambda=8 (edges detected by finding zero crossings - gabor filter)

Figure 9. lambda=10 (edges detected by finding zero crossings - gabor filter)

Figure 10. lambda=13 (edges detected by finding zero crossings - gabor filter)

Figure 11. lambda=20 (edges detected by finding zero crossings - gabor filter)

In order to detect edges with Gabor filter, we should extract edges with different filter orientations and wavelengths. I defined two arrays for filter's angle and lambda, and set some values for them. Then I defined two loops which iterate over these values and detect zero crossings for each lambda with all orientations. Finally, I combined all these detected edges in one matrix. My final result is a bit messy, though! It contains lots of edges extracted from different scales and orientations. Even with only one lambda we saw previously that the edges were not accurately detected for each lambda.

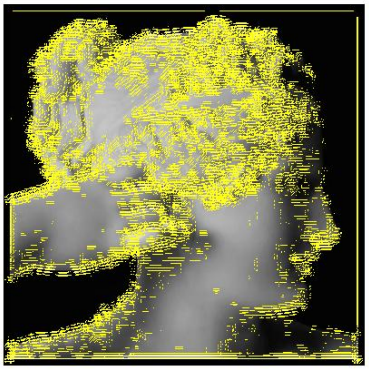

Figure 12. lambda=2 and 3 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

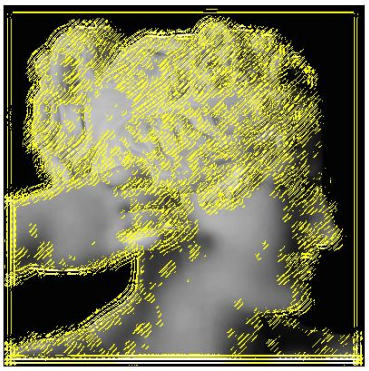

Figure 13. lambda=2 and 4 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

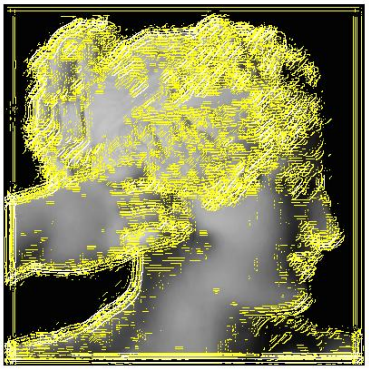

Figure 14. lambda=3 and 6 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

Figure 15. lambda=2 and 7 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

As far as I realized, Gabor filters are more used in medical imaging, in which pictures do not have a strict form. They are less used for detecting edges of a face. I also found that they are more used in feature extraction with different scales and orientations. (as seen here: https://stackoverflow.com/questions/20608458/gabor-feature-extraction). I find Gabor filter's edge detection to be less accurate than Laplacian of Gaussian.

Doing experiments for apple image:

Figure 16. lambda=2 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

Figure 17. lambda=4 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

Figure 18. lambda=7 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

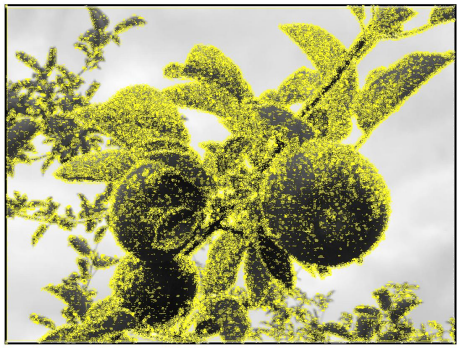

Figure 19. lambda=2 and 3 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

Figure 20. lambda=2 and 7 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

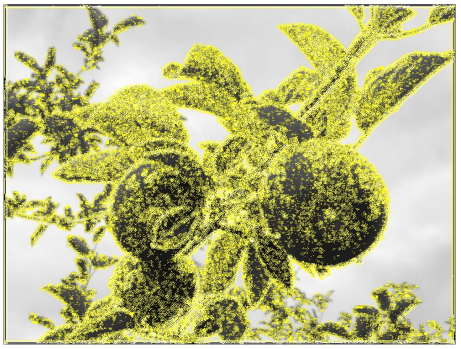

Figure 21. lambda=3 and 6 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

Linear variation theorem also plays a role in edge detection of Gabor filters. we can see in images that when the intensity along an edge changes non-linearly, the method fails to detect the edge properly.

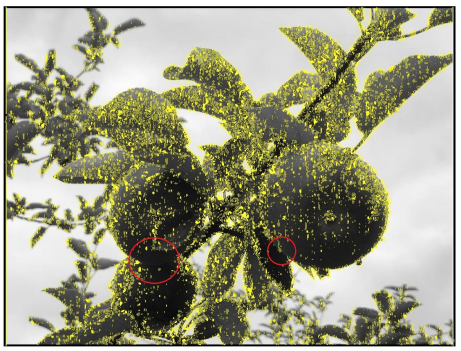

Figure 22. lambda=3 , angle=0 (edges detected by finding zero crossings - gabor filter)

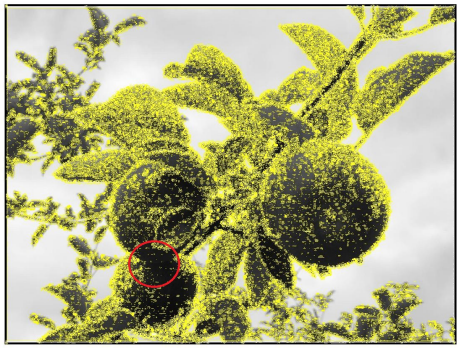

Figure 23. lambda=3 and 2 , in 3 orientations: 0, $\pi/4$, $\pi/2$ (edges detected by finding zero crossings - gabor filter)

References

[1] D. Marr & E. Hildreth *Theory of edge detection* (Royal Society Press, 1980).

[2] *en.wikipedia.org/wiki/Gabor*$_f$*ilter*

2