Image processing - 046200

Homework #3 wet

Alexander Shender 328626114

Sahar Carmel 305554453

Technion - Israel Institute of Technology

The answers to the questions are the following: $R = 2000$, $S = 8$.
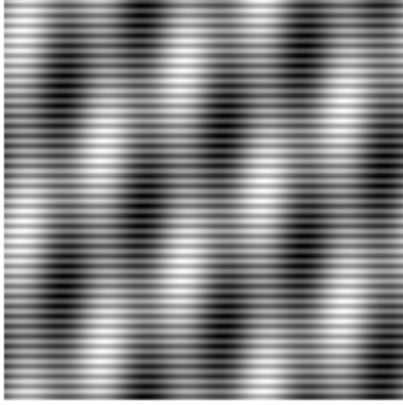
A.

The F_1 function is the following:

$$F_1 = sin(2\pi(400/R) \cdot y) + sin(2\pi(50/R) \cdot x) + sin(2\pi(20/R) \cdot (x+y))$$

$$F_1 = sin(2\pi \cdot \frac{y}{5}) + sin(2\pi \cdot \frac{x}{20}) + sin(2\pi \cdot \frac{x+y}{50})$$
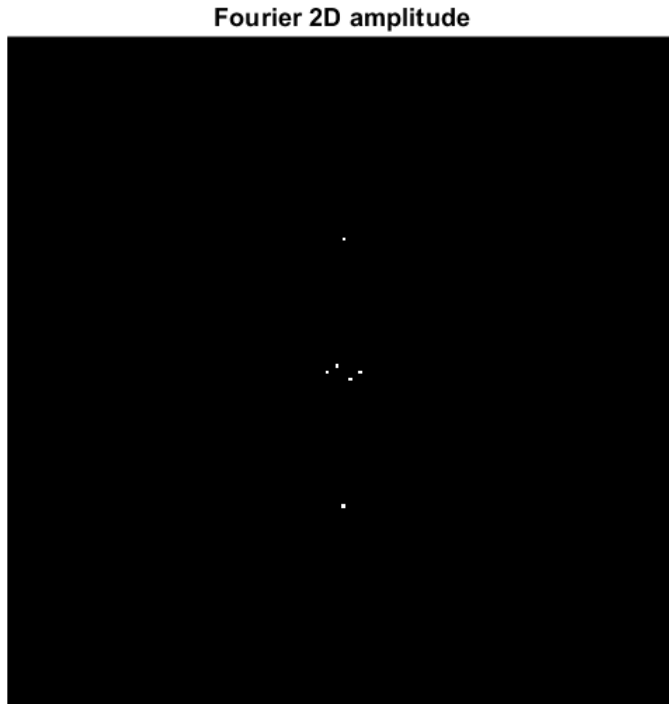
B.

The $F_1^{\Delta=1}$ was recreated using the above equation and we have received the result, identical to the one in the question. Size of the image: 200x200 px:


$F_1 (\Delta = 1)$

C.

The 2D FFT was used to create the Fourier representation of the image. The Amplitudes are the following:

Fourier 2D amplitude

As expected, we can see 6 deltas, according to the 3 frequencies that are present in the image:

1. Two dots on the vertical line → high frequency sine: $sin(2\pi \cdot \frac{y}{5})$

2. Two dots on the horizontal line → $sin(2\pi \cdot \frac{x}{20}))$

3. Two dots on the diagonal line (smallest frequencies) → $sin(2\pi \cdot \frac{x+y}{50})$

To understand it easier, we have colored those lines:

Fourier 2D amplitude
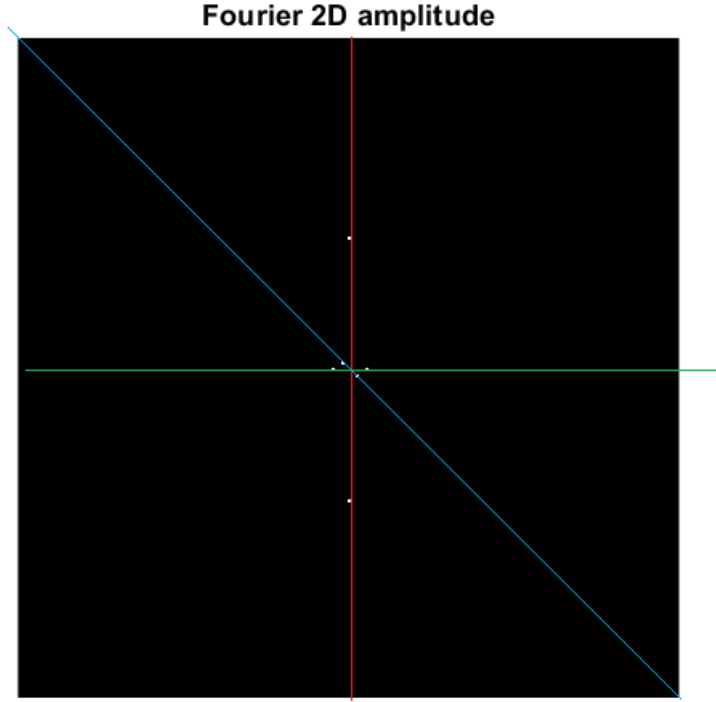
The exact coordinated of the deltas are:

|	Row	Col	Distance from center (px)	Frequency
Middle point	101	101		
$sin(2\pi \cdot \frac{y}{5})$	61	101	40	0.2
	141	101		
$sin(2\pi \cdot \frac{x}{20})$	101	96	5	0.05
	101	106		
$sin(2\pi \cdot \frac{x+y}{50})$	99	99	2.82	0.02
	103	103		

We can see that the distance from center fits the frequency. For example, the highest distance from center for y is 100 pixels, which corresponds to the highest frequency that the image can display with sample size of 1: $f = 0.5$. Thus, for frequency of 0.2, we obtain distance $100 \cdot \frac{0.2}{0.5} = 40$.

D.

The image was not sampled with a sampling size (Δ) of 9 ($S + 1$) = $8 + 1 = 9$. This sampleing size corresponds to a frequency of $\frac{1}{9} = 0.11111....$ After performing the same procedure, the Fourier transform was found to give the following amplitures map:

F₁ (△ = 9)   Fourier 2D amplitude
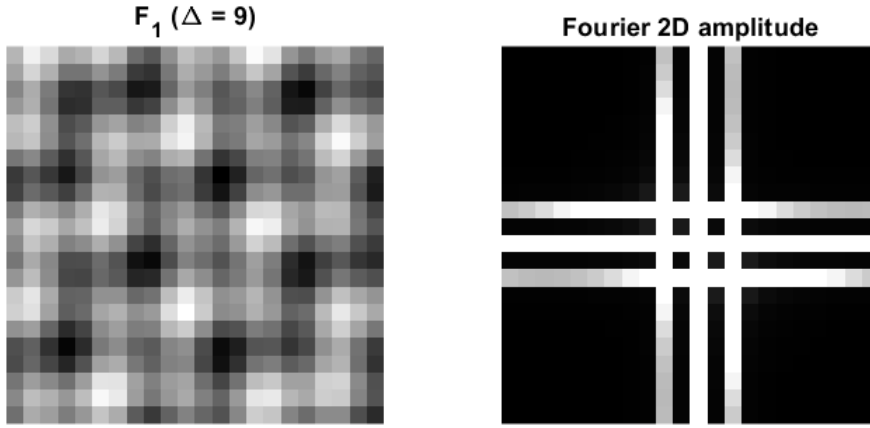
We can clearly see the aliasing here. But the aliasing only occurs on the first 1 terms in F_1. To avoid aliasing, we are required to sample at least at the frequency which is twice higher than the frequency of the signal, e.g. 0.4. But we sample at a much lower rate - 0.1111... The existance of this diversity of frequency (lines) shows that we sample at the frequency which does not align with the given sine frequency, not matter how we multiply both. For example, if we sample at the Sample Size of 4, which corresponds to the frequency of 0.25, we alias the $sin(2\pi \cdot \frac{y}{5})$ function another sine wave function of lower frequency which has consistent common intersections with the original sine wave. We can see this on the following FFT transform. Notice how the dots on the vertical lines came closer to the center:


F₁ (△ = 4)   Fourier 2D amplitude
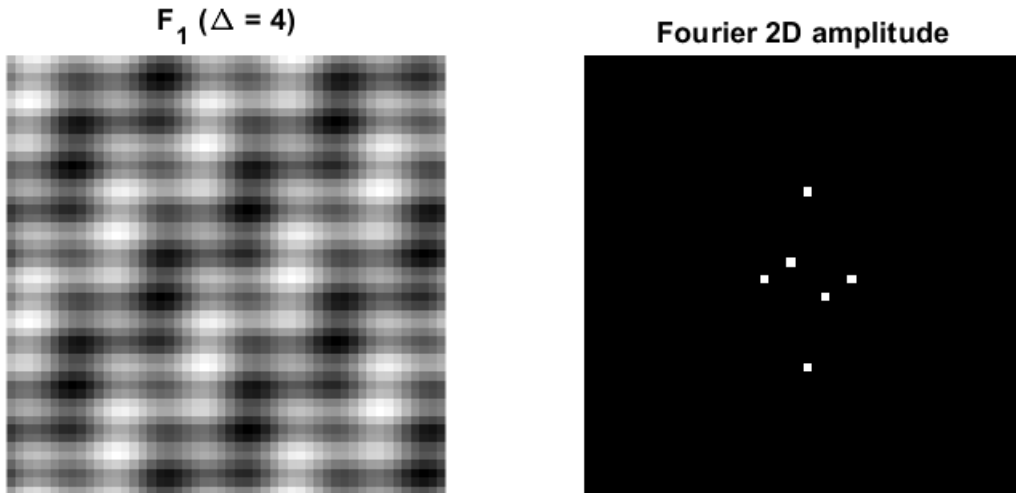
Or we can choose a sample rate which will alias this sine wave $(sin(2\pi \cdot \frac{y}{5}))$ function with a constant value! We have to choose a sample size, which corresponds exactly to the period of the sine wave, which is 10 pixels. This way we always fall onto '0' value of this function:

F$_1$ ($\triangle$ = 10)   **Fourier 2D amplitude**
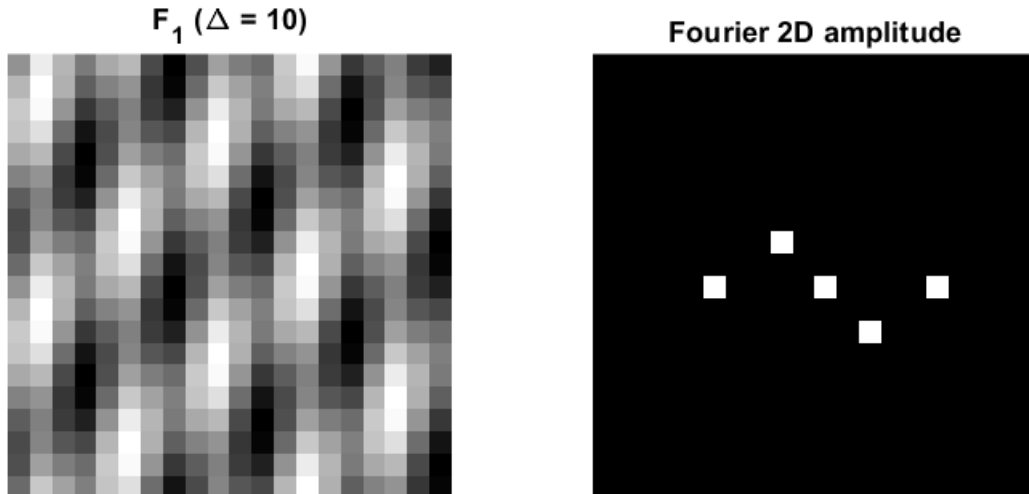
E.

A real image was taken, from the internet. The image contains various elements, windows at buildings, and other things which have high frequencies. Image was resized to 400x400 px. Plotting the image and its 2D FFT amplitudes gives:

Image ($\triangle$ = 1)   **Fourier 2D log amplitude**
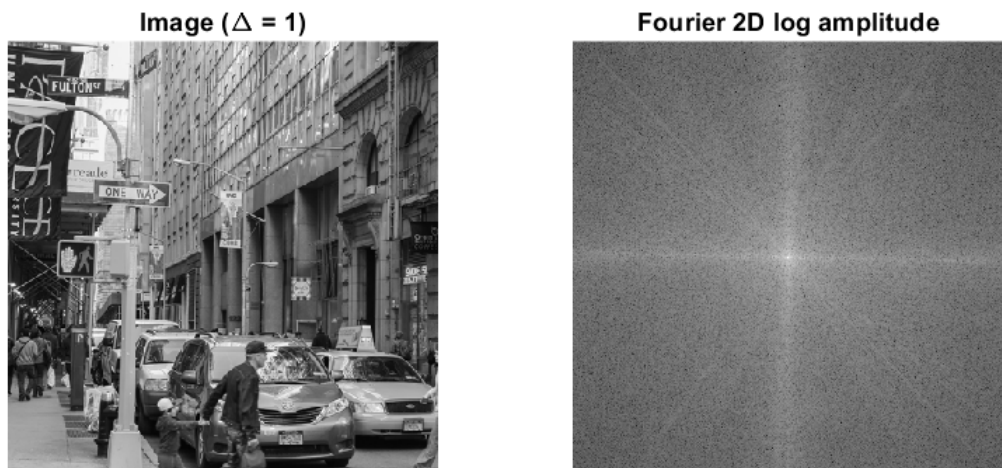
F.

The sampling with the a sampling size of 4 was performed. The resulting image is of size 100x100 px. Plotting the image and its 2D FFT log amplitude. To answer the question, we plot the images together with the original image, for convenience.

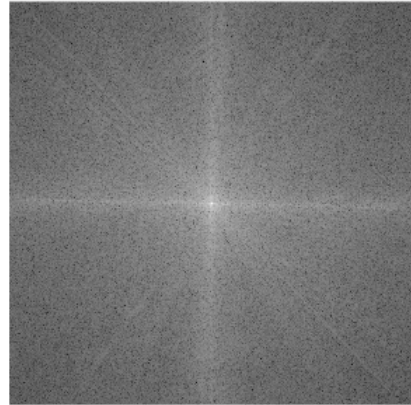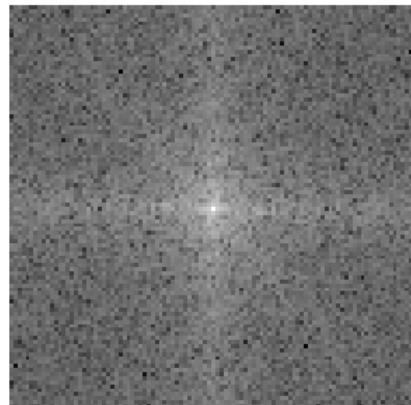Image ($\Delta = 1$) Fourier 2D log amplitude ($\Delta = 1$)

Image ($\Delta = 4$) Fourier 2D log amplitude ($\Delta = 4$)

Differences:

1. In spatial domain: we can see that the details that were small are not visible anymore, disappeared, because the sampling rate was too small to preserve them, and they got aliased with frequencies, which are smaller than Nyquist frequencies. Thus, those details are not visible anymore.

2. In frequency domain: We can see that the high frequencies got less intense, and most of them disappeared. We can see a shift towards the center of the frequencies plot, where the frequencies are lower. It happened because some high frequencies got aliased by the low frequencies, so their intensity rised.
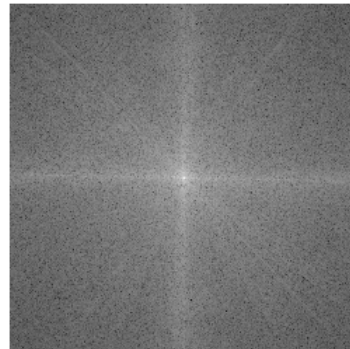
G.

In this subsection we first apply the Gaussian filter on the original image, and only then subsample. The results are the following:
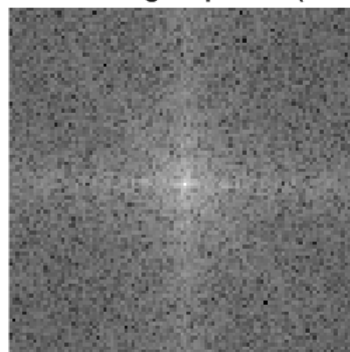

F_1 ($\Delta = 1$)

Fourier 2D log amplitude ($\Delta = 1$)
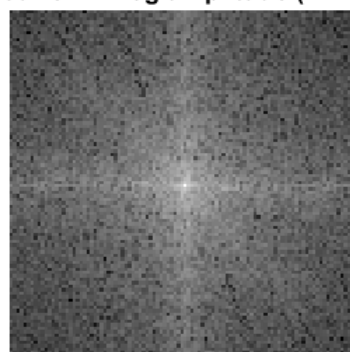
F_1 ($\Delta = 4$)

Fourier 2D log amplitude ($\Delta = 4$)

F_1 (gauss filtered) ($\Delta = 4$)

Fourier 2D log amplitude ($\Delta = 4$)

We can observe that here, the high frequencies were better preverved! This is indeed

one of the methods to reduce the aliasing effect. Because we have applied the Gaussian filter, which is the LPF, the high frequencies were reduced, thus not creating the aliasing effect.

II.   Question 4

1.

The function was created. See MATLAB code.

2.

The quantization was performed on the selected image, and the following results have been obtained. The graphs are displayed in all cased in the following order: (1) The quantized image (2) the distortion levels (3) the relative improvement
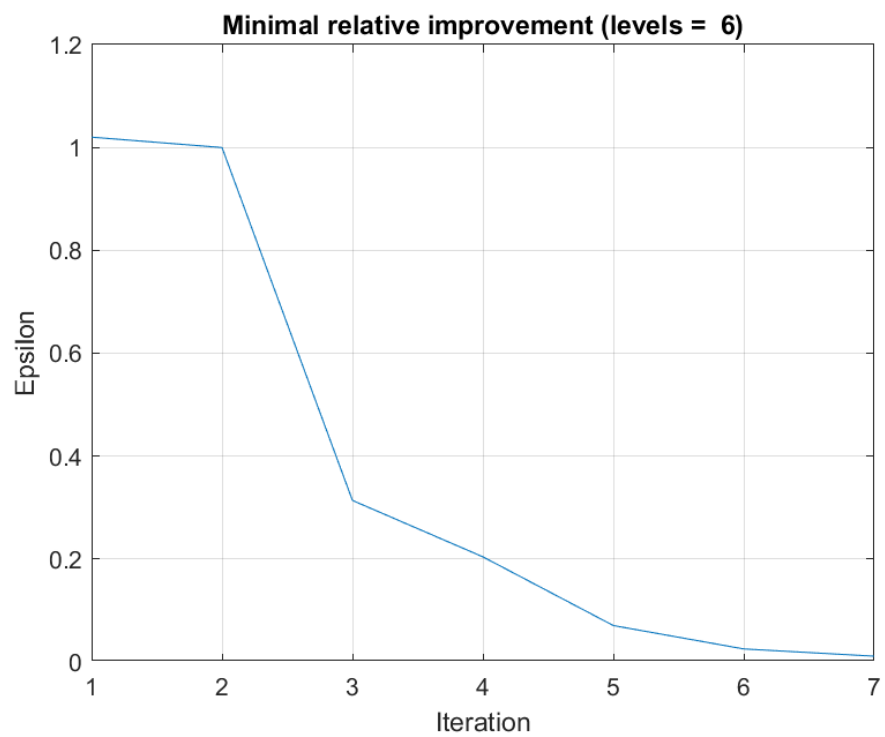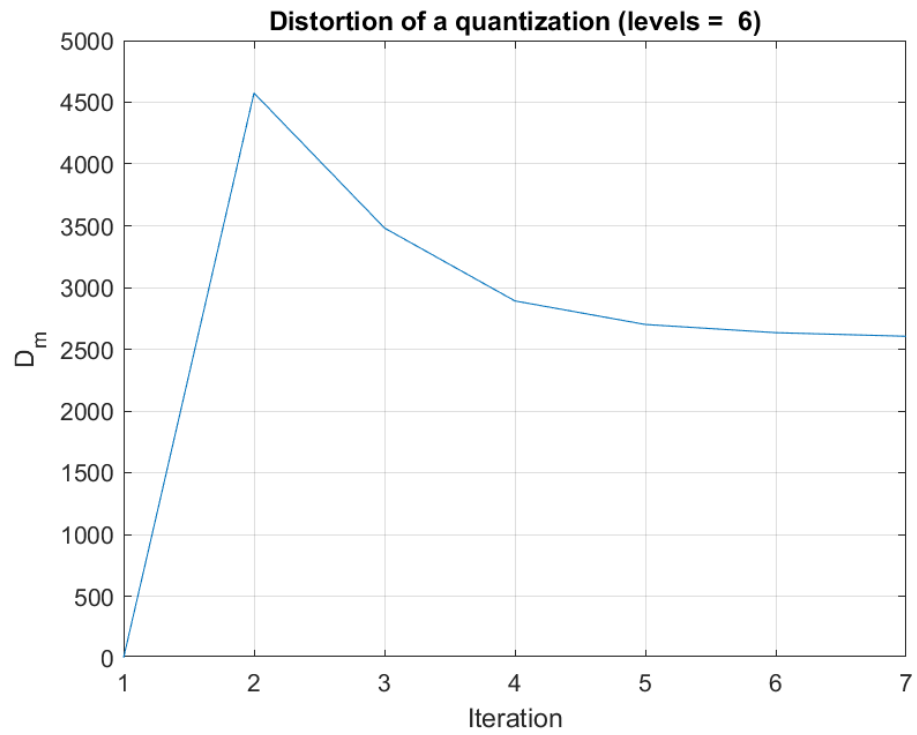
For the 6 levels quantization:
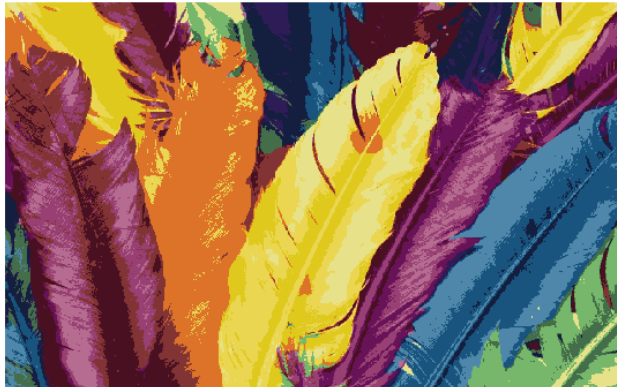

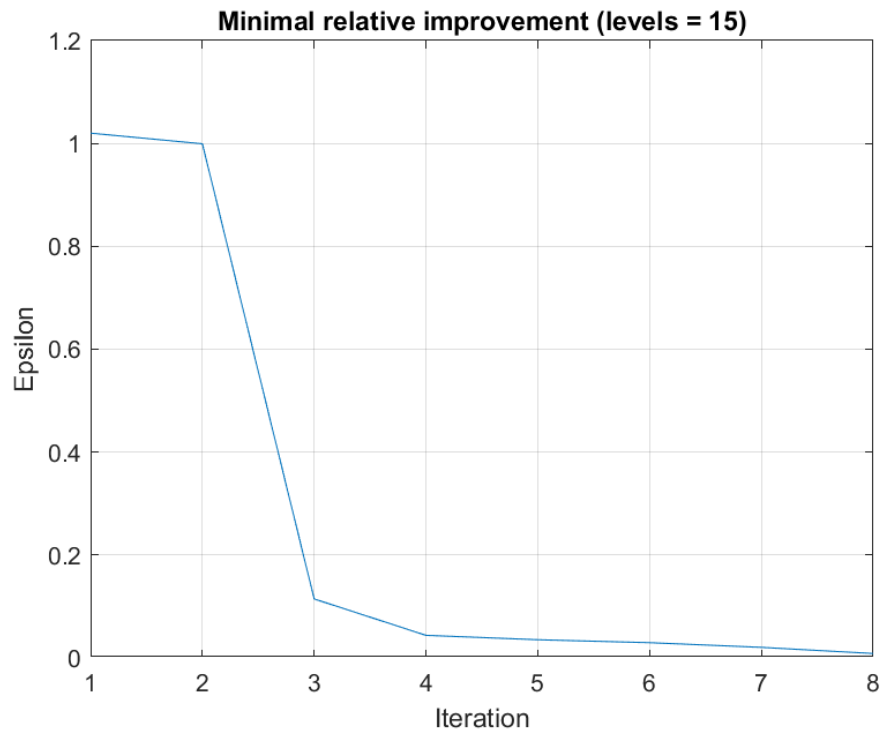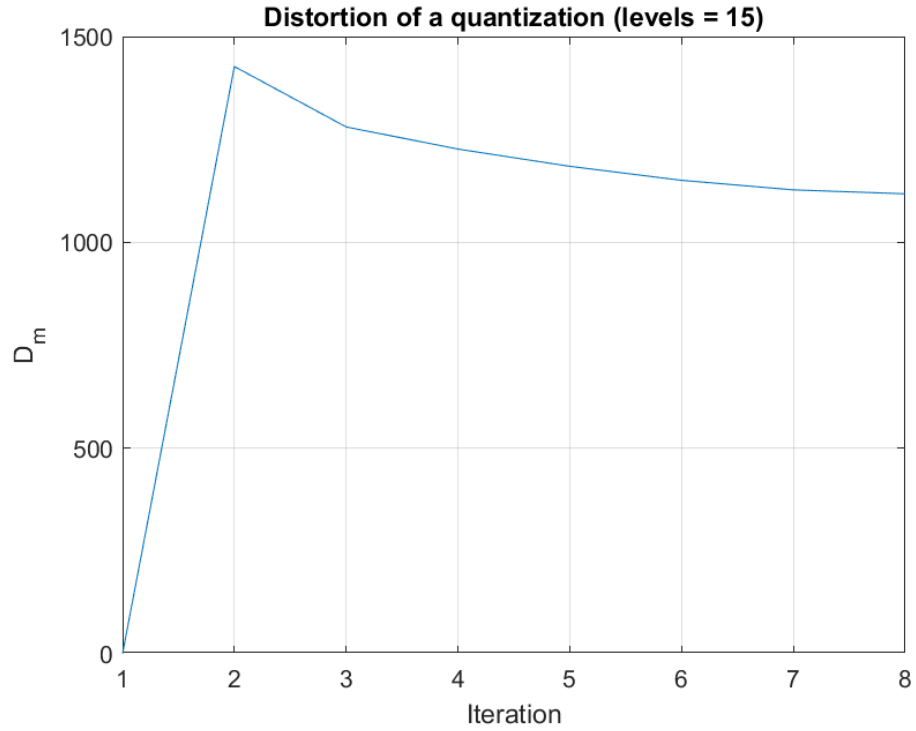Original image

Quantized image (levels = 6)

For the 15 levels quantization:

Original image

Quantized image (levels = 15)

11

Distortion of a quantization (levels = 15)

Minimal relative improvement (levels = 15)

3.

Indeed, if we re-run the algorithm, the results will be different each time. This happens because we instantiate the algorithm randomly, picking random pixels and taking their representations as representative levels. We may find numerous local minimas where the algorithm will be unable to find a more optimal solution. The algorithm stops when the

improvement is very little, meaning it has reached its local minimum point. Numerous local minimums can give numerous representations, and their distortion is not the same.
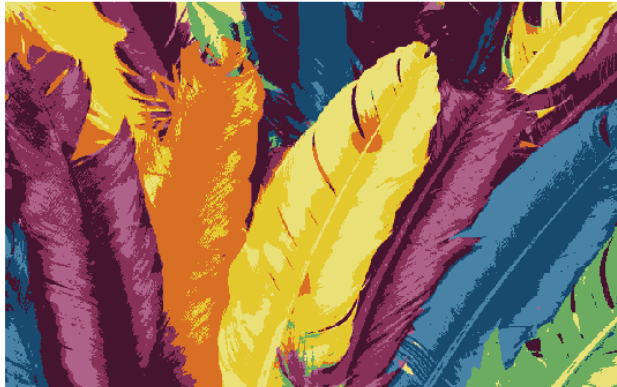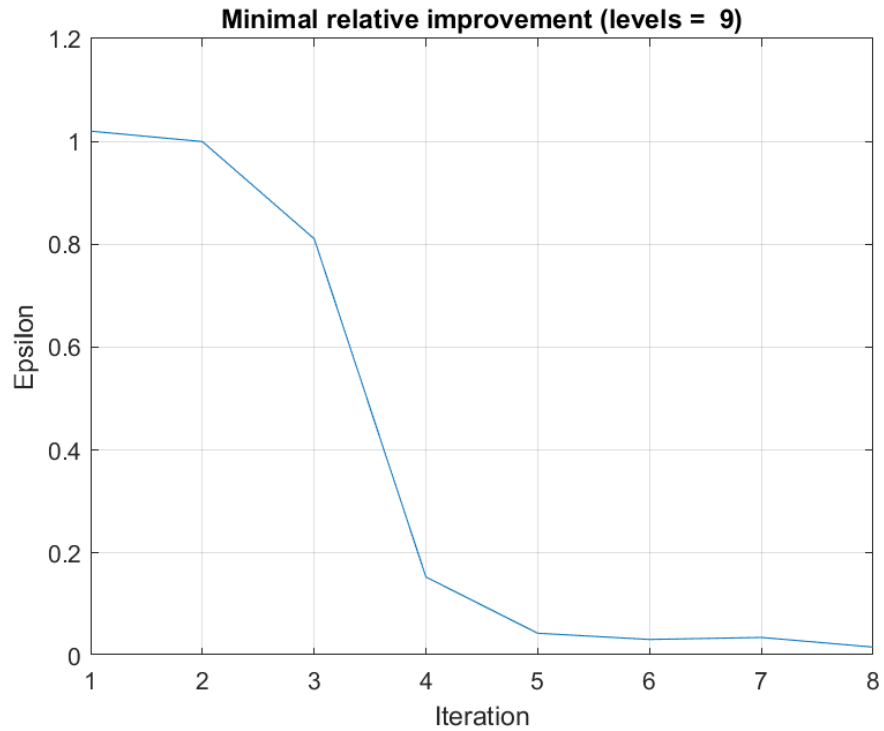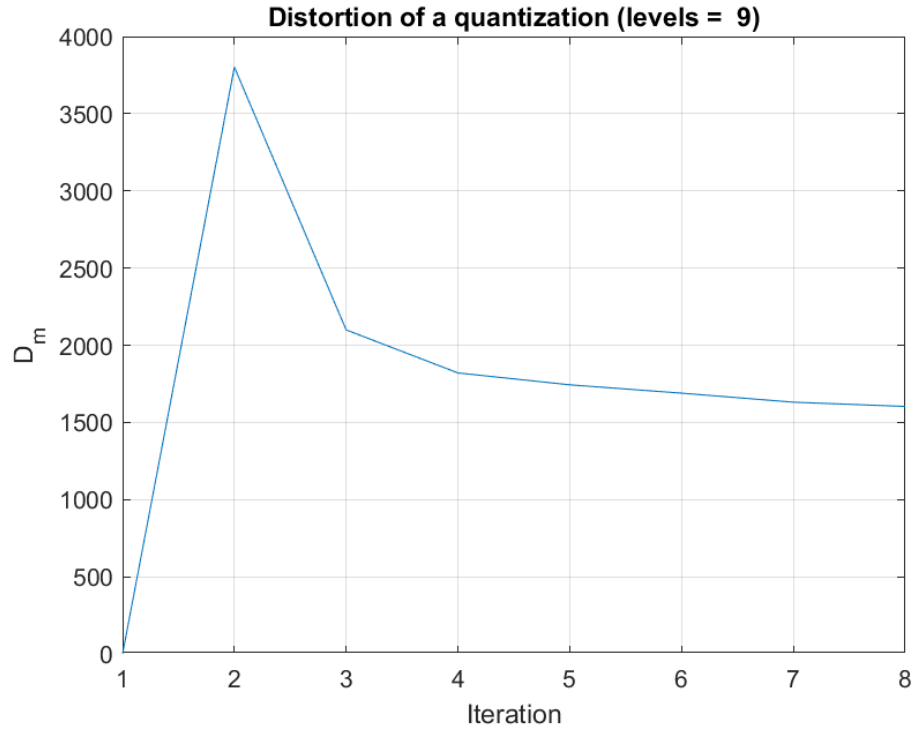
4.

We have extended the algorithm to accept also the parameter of initialization. This time, for the 9 levels, the results are the following:

Original image

Quantized image (levels = 9)

Distortion of a quantization (levels = 9)

Minimal relative improvement (levels = 9)

5.

In this case, the colors were not picked randomly. We have used the prior information, and assumed that the colors will be spanned through the variety of spectrum. Several conclusions may be drawn:

1. By having a fixed initial representation, each new run of an algorithm gives the

same result.

2. We did not observe improvement in iterations number. By logically, yes, the iterations number should be smaller. Perhaps, if we run the algorithm numerous times, we can observe it.

3. Comparing to the 15-levels, random initialization, the distortion for 9-level fixed initialization is larger. Because, still, the levels amout is a major factor in the distorion.

4. When the algorithm was re-run on 9 levels random initialization, the distortion obtained was larger. Which proves that using a prior information on a color spectrum helps to deliver the best quantization initial parameters
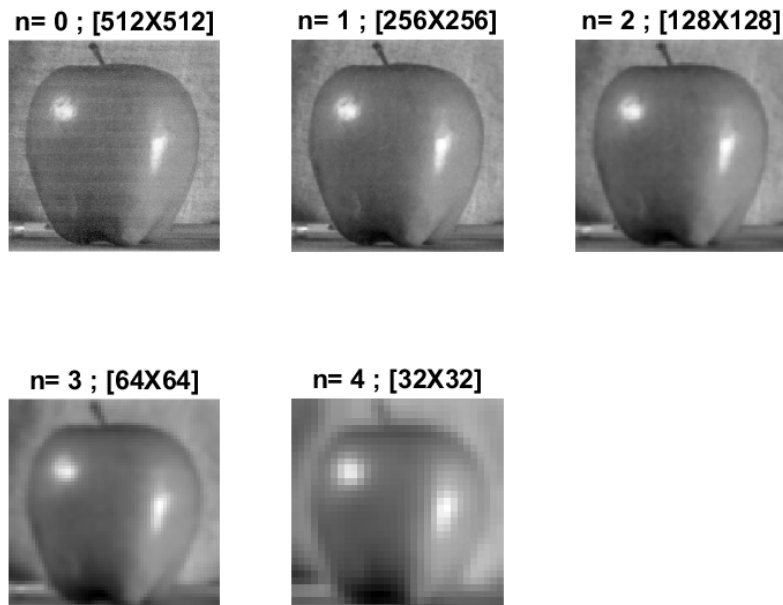
III. Question 5

1.

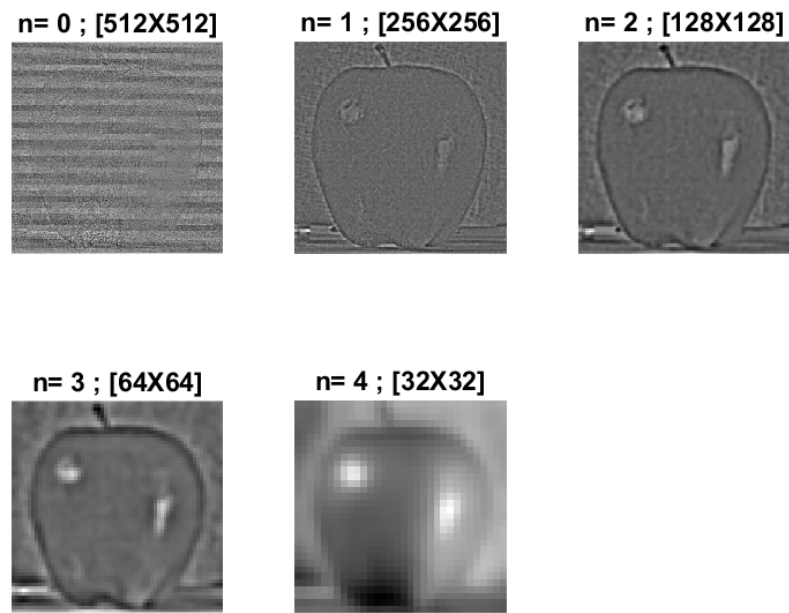The function was created. See MATLAB code.

2.

The Gaussian and Laplasian pyramids for both of the images were created and the results are the following:
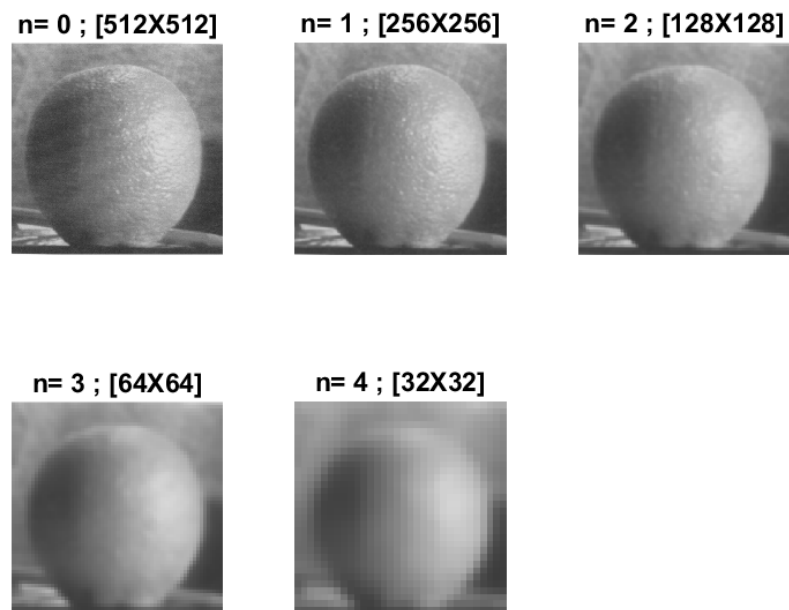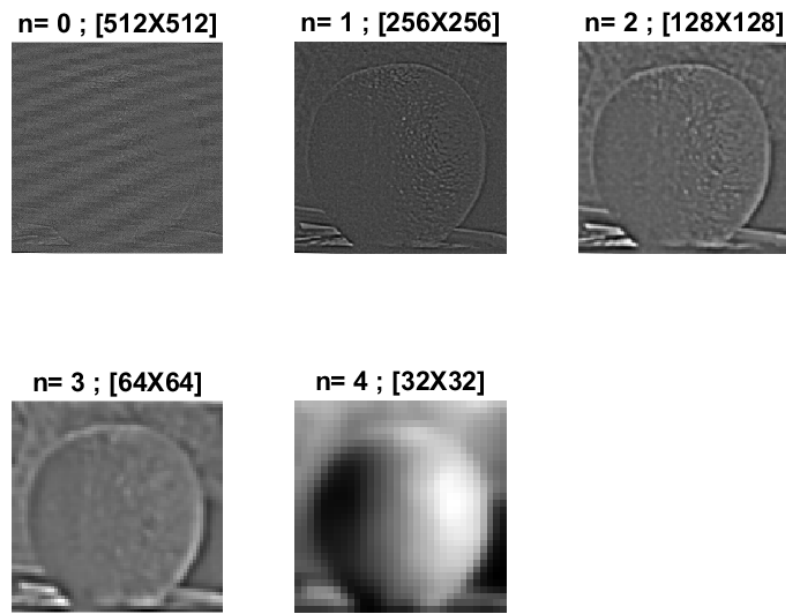
The Gaussian pyramid for apple:

The Laplasian pyramid for apple:

n= 0 ; [512X512]   **n= 1 ; [256X256]**   **n= 2 ; [128X128]**

n= 3 ; [64X64]   **n= 4 ; [32X32]**

The Gaussian pyramid for orange:

n= 0 ; [512X512]   **n= 1 ; [256X256]**   **n= 2 ; [128X128]**

n= 3 ; [64X64]   **n= 4 ; [32X32]**

The Laplasian pyramid for orange:

n= 0 ; [512X512]   n= 1 ; [256X256]   n= 2 ; [128X128]
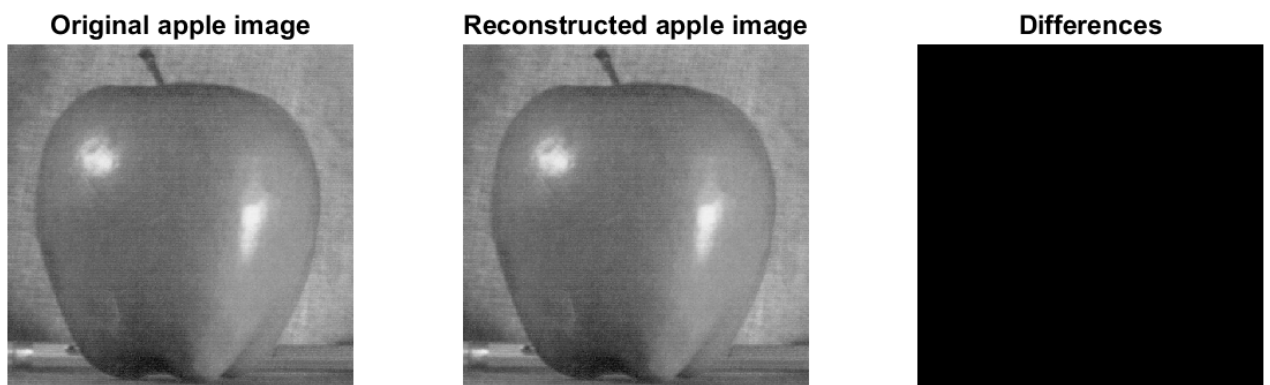
n= 3 ; [64X64]   n= 4 ; [32X32]

3.

The function was created. See MATLAB code.

4.

The apple and orange images were reconstructed. We also show the difference between
the original image and the reconstructed image to verify there are no differences:

Apple reconstructed:

Original apple image   Reconstructed apple image   Differences

Orange reconstructed:

Original orange image   **Reconstructed orange image**   **Differences**
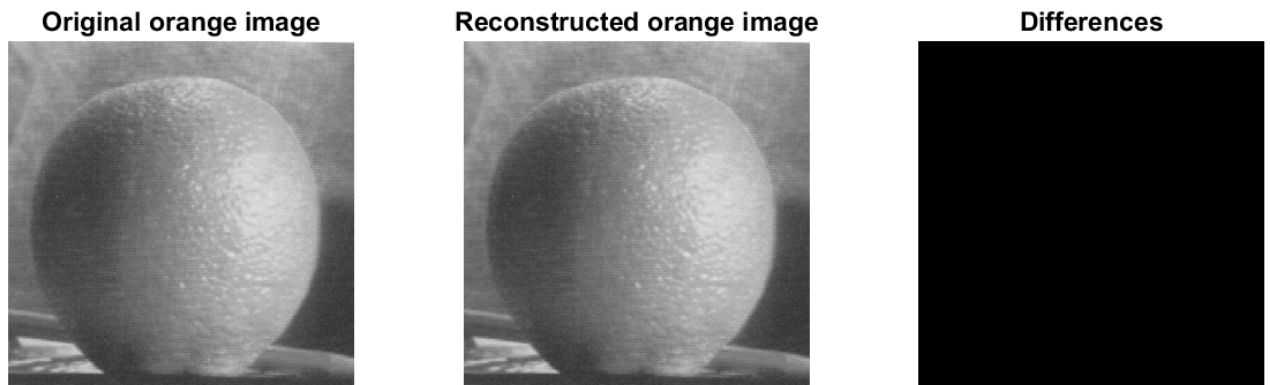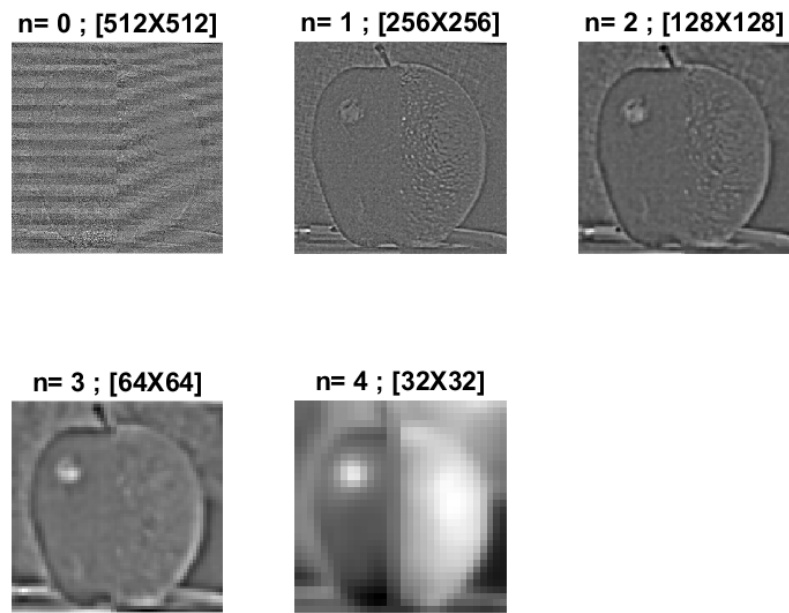
5.

Image mosaic implementation. The function was created, the number of common pixels is the parameter which user can input. Following the instructions, we set this parameter to be '2'. The results are the following:
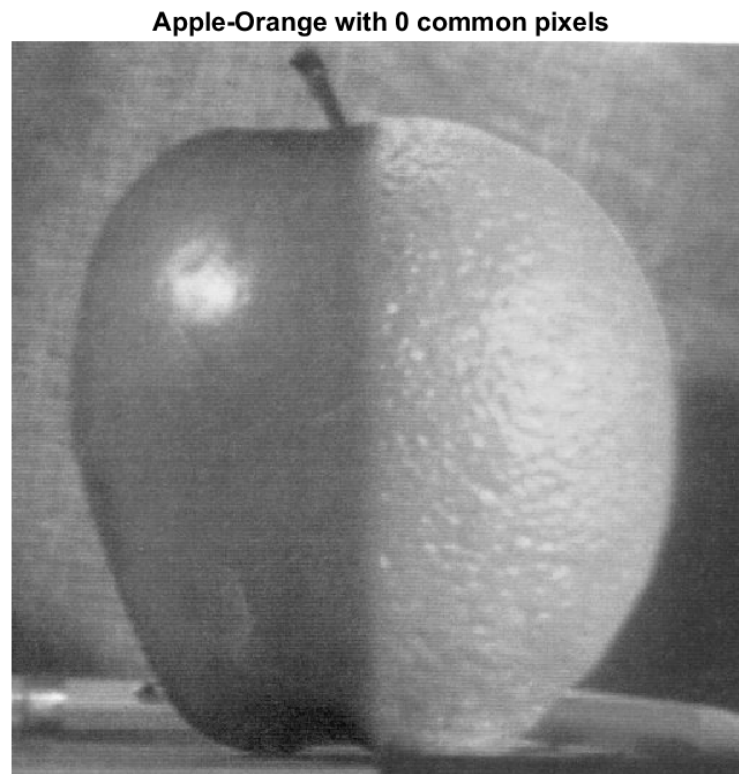

Apple-Orange with 2 common pixels
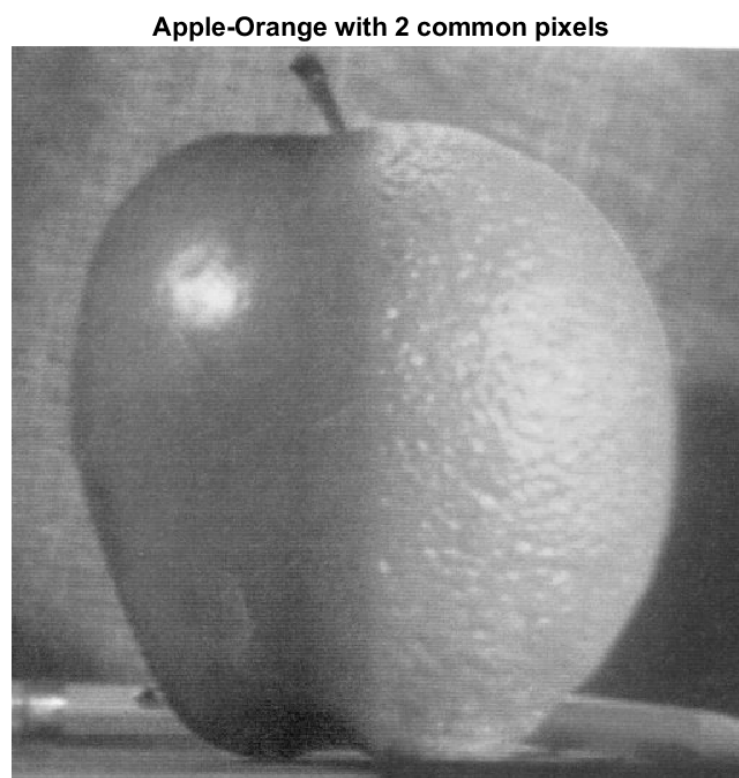
The Laplasian pyramid of the mosaic image:

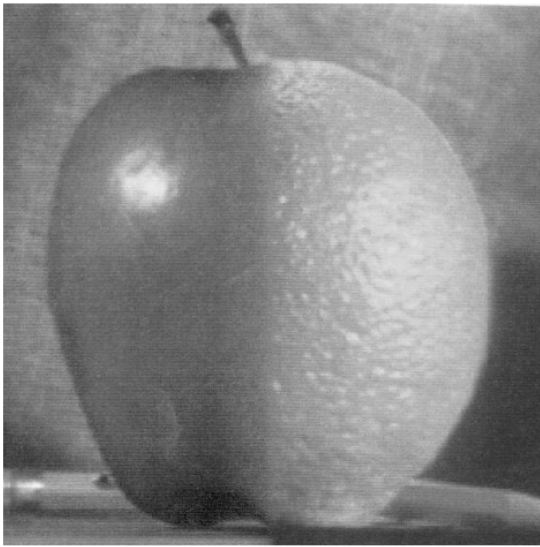n= 0 ; [512X512] n= 1 ; [256X256] n= 2 ; [128X128]

n= 3 ; [64X64] n= 4 ; [32X32]

6.

By doing a simple stitch, by simply putting images together on different Laplasian pyramid levels, we obtain:
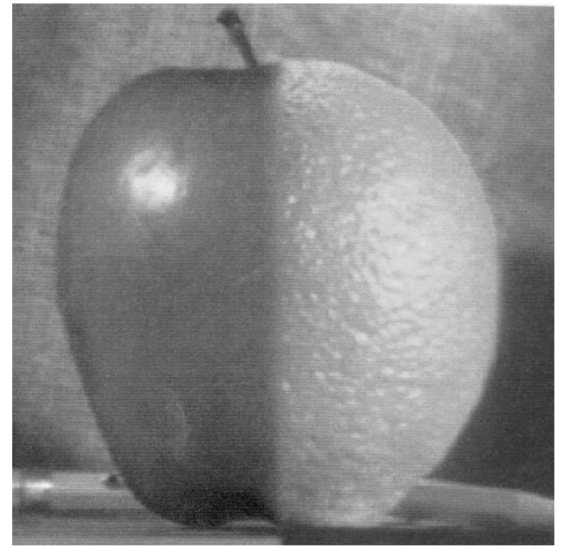

Apple-Orange with 0 common pixels

Comparing this to the first stitching, we can see the differences:

Stitching with 2 common pixels

Stitching with 0 common pixels

We can indeed observe that the "transition" stage between the images becomes more "smooth" as the number of "common pixels" increases. This can be logically explained, since we share the pixels information further "deeper" into the images. But also with a simple method of putting images side-by-side does achieve transition. This happens because the 'expand' function, which is used in the reconstruction phase from the Laplasian pyramid, uses the information of the neighboring pixels in its process.