Wavelet Homework

Arya Ebrahimi

Abstract

The purpose of this homework is to understand pyramids and wavelet transformation. First, a function has been implemented that calculates Gaussian and Laplacian pyramids associated with the given image. This function is used to solve the starting questions. Also, an inverse function has been implemented to reconstruct the original image based on its Laplacian and Gaussian pyramid.

Furthermore, for the latest questions, a function is implemented that calculates the wavelet transform of an image using Haar analysis (built-in functions of the OpenCV library are used). Like the first part of the homework, an inverse function is also implemented. However, another function is applied to wavelet coefficients to illustrate the changes and applications before using this inverse function.

1. Questions

1.1- For the "Mona Lisa" image, build a 5 level Gaussian pyramid and display it. Also, implement and display a Laplacian pyramid.

1.2- Describe how separability and cascading can help to speed up Gaussian smoothing and design a fast algorithm for computing a 3-step Gaussian pyramid (filtered with σ, √2σ, 2σ) of a 2D image using pseudo-code.

1.3- Given an image of size $N \times N$ where $N = 2^j$ what is the maximum number of levels you can have in an approximation pyramid representation? What is the total number of pixels in the pyramid? How does this number compare with the original number of pixels in the image? Since this number is larger than the original pixel number, what are some of the benefits of using the approximation pyramid? Repeat the step for the prediction residual pyramid. Display and discuss the results.

1.4- For the gray-scale Lena image, manually compute a 3-level approximation pyramid and corresponding prediction residual pyramid. Use 2x2 averaging for the approximation and use pixel replication for the interpolation filters.

1.5- For the gray-scale Lena Image, compute the wavelet transform (with 3-level) using the Haar analysis filters. Comment on the differences between the pyramids generated in Prob 1.4 with the ones generated here.

1.6- Quantize all the wavelet coefficients created in Prob. 1.5 by a step size of $\gamma = 2$. Then reconstruct the image from the quantized wavelet coefficients using Haar synthesis filter. Report PSNR values and discuss the results.

2. Technical Analysis

2.1- *A generalized pyramid function is implemented to solve this problem and those needing a pyramid. The inputs of this function are an image, desired level of the pyramid, a kernel, an up-sample function, and a down-sample function.*

At iteration of the main loop, First, a padding function is applied to the image; The padding function will change the image's shape into even numbers. This is done to prevent errors when the shape changes during up-sampling and down-sampling.

After that, the function will convolve the kernel on the image. This kernel is a Gaussian kernel by default, but for problem 4, it uses an averaging filter, and in general, it could be any kernel.

Next, the function will down-sample the result using the down-sampling function given in the inputs. By default, it uses the pixel replication method, but it can also use the averaging down-sample method. After down-sampling, the function also calculates the up-sample image using up-sample method given in the inputs. This step is because of the Laplacian pyramid. Subtracting the up-sampled image from the padded image in each iteration results in the Laplacian at the corresponding pyramid level of that iteration. The Gaussian pyramid result, at each level, is the down-sampled image from the iteration corresponding to that level.

Note that for each iteration, the function will also save the Gaussian and Laplacian at a list; this list will help us reconstruct the image from Gaussian and Laplacian pyramids.

2.2- *Suppose we have a Gaussian kernel with size $M \times M$ and an image with size $N \times N$, Then the complexity of convolution would be $O(N^2M^2)$.*

A 2D kernel is called separable if there exists two vectors exist such that their matrix product is equal to the kernel. For example, as shown below, matrix G is separable because $G = g_1 . g_2$.

$$G = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$g_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}, \quad g_2 = \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

When the filter kernel is separable, the convolution between the image and kernel can be performed in 2 steps.

First, we perform convolution between the image and the first separated vector. The complexity of this step is $O(N^2M)$. Then the second separated vector will be convolved into the image. The complexity of this step is the same as the first part.

In this way, the total complexity would be $O(N^2M)$. This is a speed up than the direct convolution of the kernel into the image.

The concept of cascading filters refers to applying several small filters sequentially.

For example, convolving this into itself will result in a bigger Gaussian filter vector.

$$\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Since the time complexity of convolution is $O(N^2M^2)$, using the cascading concept will decrease the filter's size, making the convolution faster.

To create a fast 3-Step Gaussian Pyramid , separability and cascading can be used.

1. *the first step is to apply a Gaussian filter with standard deviation of σ to the given image. This filter is separable, meaning that it can be broken down into two 1D convolutions which are applied to the rows and columns of an image. The result is an*

image with a lower resolution and reduced noise.

2. *The second step is to apply a Gaussian filter with standard deviation of $\sqrt{2}\sigma$ to the lower resolution image, again using separability. This will reduce the resolution further.*
3. *The third step is to apply a Gaussian filter with standard deviation of 2σ to the reduced resolution image one last time. This will give us the final result which is a 3-Step Gaussian Pyramid. This algorithm is fast because it uses the concept of separable filters.*

2.3- *The image is down-sampled at each level of the Gaussian pyramid; the image size will be divided by two at each level. Since N is equal to 2^j , dividing by two is like shifting the binary number to the right; after j shifts to the right, N would be equivalent to 1, so the maximum number of levels is j.*

The formula given below can compute the total number of pixels in the pyramid. The summation can be approximated to $\frac{4}{3}N^2$, where N is the size of the original image.

$$p = N^2 + \frac{N^2}{4} + \frac{N^2}{16} + ... + \frac{N^2}{2^j} \approx \frac{4}{3}N^2$$

As shown above, the total number of pixels needed in the Gaussian pyramid is bigger than the Original pixels of the image; thus, it needs more space. However, due to their applications, they are used vastly.

- *Coarse to Fine strategies: With pyramids, the image can be represented in different scales. It can be used to search for an object at different levels of the pyramid; since deeper levels of the pyramid are smaller, the time complexity of searching is also cheaper. So it can enhance searching for objects in images.*
- *Image Blending: Laplacian pyramids can be used to blend two or more images. For the two pictures, first, their Laplacian pyramids must be constructed, and then the third Laplacian pyramid can be made using those. It results in better output than the original method of changing the pixels.*
- *Image Compression: Laplacian pyramids can also be used to compress images. The downsampled approximation of the image is in the final level of the Laplacian pyramid, and in previous layers, only the picture's edges exist. Since the edges are similar to a sparse matrix, the needed space can decrease using sparse representation methods.*

2.4- *The first problem's general pyramid function is used for this problem. But instead of the Gaussian kernel, a two-by-two box filter is used to downsample, and for up-sampling, the pixel replication method is used.*

2.5- *To solve this problem, a generalized wavelet function is implemented that gives in an image, the wavelet's levels and a mode set to Haar as the default. The outputs of this function are Wrep, which is the wavelet representation, LLrep, which is all the LL approximations during the wavelet transformation. coeffs, which are the coefficients; and finally, the last LL image.*

There is a for loop in this function that iterates over the number of levels, and for each level, first, it calculates the four parts of the wavelet transform using the dwt2 function and then puts each one in the Wrep matrix. Since this problem has to

compare wavelet approximations with the Gaussian pyramid, all the LL approximations will be saved in LLrep matrix with the same theme as the Gaussian pyramid. In each level of the wavelet, the LL values are multiplied by two in comparison with their pervious step, so a dividing step is performed to map LL values between 0 and 255.

2.6- *Two functions are implemented to solve this problem. First, a function that applies the given formula to the coefficients and returns the result coefficients. And a function to reconstruct the original image using the last LL and the coefficients.*

This function has a for loop that iterates over the levels of the wavelet using a bottom-up approach.

At each iteration, the inverse of the wavelet is calculated using LL and coefficients of that level. This results in the LL of the next layer. This process continues for each level, and the original image will be constructed at the end.

At the end of this problem, it is asked to calculate PSNR. It is an engineering term for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Typical values for the PSNR in the lossy image and video compression are between 30 and 50 dB, provided the bit depth is 8 bits, where higher is better.

$$PSNR = 20 \cdot log_{10}(MAX_I) - 10 \cdot log_{10}(MSE)$$

where MAX_I is the maximum possible pixel value of the image.

Using this metric, the difference between the original image and the quantized version is calculated and discussed in the next part.

The formula that is asked to apply has a parameter called gamma. When the formula with the gamma of two is applied to coefficients, it quantizes them with a step of two. With smaller gamma, this step would be smaller, and with a larger gamma, the step would be larger.

3. Results Analysis

3.1- *Figure 1 and Figure 2 shows the result of the pyramid function for the Mona-Lisa picture.*

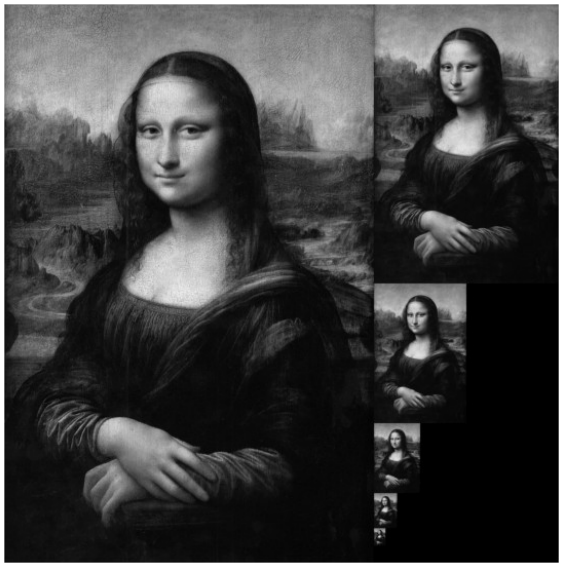

Figure 1: Gaussian pyramid of the Mona-Lisa image. The left part is the original image, and the right is the Gaussian pyramid's levels.

Figure 2: Laplacian pyramid of the Mona-Lisa image. The left part is the original image, and the right is the Laplacian pyramid's levels. Note that the last level of the Gaussian and Laplacian is the same.

3.2- *This problem has no results, and it is discussed in the Technical Analysis part (2.2).*

3.3- *Camera-Man picture is used to solve this problem. Since the Camera-Man is* 512×512*, so* $N = 2^j = 512$ *so* $j = 9$*. A 9-level pyramid is calculated for this problem.*

Figure 3: 9-level Gaussian pyramid of the camera man picture.

Figure 4: 9-level Laplacian pyramid of the camera man picture.

3.4- *In this problem, a pyramid is made for the Lena image using the* 2×2 *averaging filter. Also, the corresponding Laplacian is constructed. The results are illustrated in Figure 5 and Figure 6.*

Figure 5: 3-level pyramid using the averaging filter for Lena image.

Figure 6: 3-level Laplacian pyramid for Lena image.

3.5- *The wavelet transform result for the Lena image is calculated using the wavelet function. Since it has three levels, there are 10 parts in the resulting image. Three LH, HL, and HH for each level and an LL for the last level. The LL of the previous levels can be constructed using the last LL. The results of the wavelet function are shown below. Figure 7 shows the LL of the last level with coefficients of the all pervious levels. Figure 8 shows the LLrep which is the all LLs constructed in the creation of the wavelet. Using Figure 8, the comparison with the previous problem can be made.*

Figure 7: 3-level wavelet transform output.

Figure 8: All the LLs in the wavelet transformation process.

Figure 9 and Figure 10 shows that the quality of the results in the wavelet is better than the pyramid with averaging filter.

Figure 9: Wavelet transform. LL of the final levels.

Figure 10: Pyramid with averaging filter final levels result.

3.6- *In the last problem, it is asked to quantize wavelet coefficients using the given function and reconstruct the image using quantized coefficients.*

As can be seen in Figure 11, quantizing coefficients can reduce image quality, but it also can reduce the size of the image. The PSNR value of the quantized image with the original image is 46.771. for a well-reconstructed image without much noise compared to the original, this should be between 30 and 50, which a higher value represents a better quality. In this case, the reconstructed image is a good approximation without much noise. With a gamma of 8, this value would be 37.236 and with larger gammas, this value would be smaller that means the reconstructed image has more noise compared to the original image.

Figure 11: The reconstructed Lena image using quantized coefficients with $\gamma = 2$.