

FlashPix Image Post-processing

Abstract

FlashPix images, based on the JPEG compression standard, present the typical artifacts common to the DCT compression schemes. Deblocking and denoising DCT artifacts has attracted the attention of lots of scientific. In this paper I will explain several methods that range from simple spatial filtering, to wavelet transforms.

1. Introduction

The FlashPix format was developed jointly by Kodak, Hewlett-Packard, Microsoft and Live Picture. FlashPix is basically a storage format that fundamentally doesn't break any technical ground.

In FlashPix, images are stored at different resolutions, every resolution is the half of the previous one, until the whole image fits in 64x64 pixels, typically resulting in an image 33 larger ($1 + 1/4 + 1/16 + 1/64$..) that a comparable flat file.

Images are split in tiles; every tile is 64x64 pixels. Each tile can be uncompressed, or compressed using the standard JPEG coding. This mechanism makes easier to the applications to work with high resolution images, even when the image is stored remotely using a low speed connection, because allows to the application to apply image transformations and previews with a low resolution image. When the high-resolution image is needed (ex: for printing), then the same image in high resolution is used.

Images are stored in files using the Microsoft structured documents format, or COM/OLE documents. This is the format used for the Microsoft applications like Word, Excel, etc. This format allows storing the images jointly with other image information, like color spaces, transforms to apply to the image, etc.

FlashPix images present the same artifacts as JPEG coded images, tiling don't add other artifacts because isn't a lossy scheme.

The lossy image compression in the DCT is achieved by quantization of the DCT coefficients. The quantization causes the reconstructed image to be different, block by block. This artifact is known as blocking. Our goal here is to reduce the blockiness without reducing the accuracy of the reconstruction.

Also when quantization completely eliminates the high-frequency coefficients, the resulting image is characterized by blurring of sharp edges.

In this document then I will study a set of methods for removing artifacts from blockwise DCT compressed images. The only constraints that add FlashPix images are related to the processing in a 'per tile' way. (This is not a requirement but is advised). Also, I will work on methods for post processing of the images. Some methods exist that process the image prior to the input in the coder, but I won't talk about those.

This document is composed of the next sections:

- Image distortion metrics.
- Smoothing filters
- Spatially adaptive smoothing filters
- Wavelets and soft-thresholding
- Implementation of filters in FlashPix
- Results

1. Image distortion metrics

Before starting to enhance images, we need a metric to help us in the selection of which method is better.

During the course we used the SNR (signal to noise ratio) between the original image and the

reconstructed one. When dealing to DCT transforms, the most common measure is a variation of the SNR, the peak-signal to noise ratio PNSR. For an image with the range of values between 0-255,

$$PSNR=10 \cdot \log \left[\frac{255^2}{MSE} \right] dB$$

Some images can have better PSNR than other but look worst, this measure is not that good. This is because good values of PSNR can be obtained when the image is blurred or with low accuracy. Also spatial masking and other factors of the visual perception can lead in a better image with less PSNR, than other.

Because we are interested in the block artifacts, is interesting to have a metric to know how much improvement the filters achieve reducing blocks.

In [4] metric for calculating the blokiness of the image is defined. Consider i_1 , and i_2 as the image values of two pixels that are next to each other in the same column, but are in different blocks. The edge variance is defined as the sum of squared differences for each pixels pair.

$$E = \sum (i_1 - i_2)^2$$

Also, I will use the apparent quality metric defined in CCIR 500 as the Double-Stimulus Continuous-Quality-Scale method. In this method the subject is shown an unpaired reference picture and a test picture, and asked to score each test image on a continuous scale. [2]

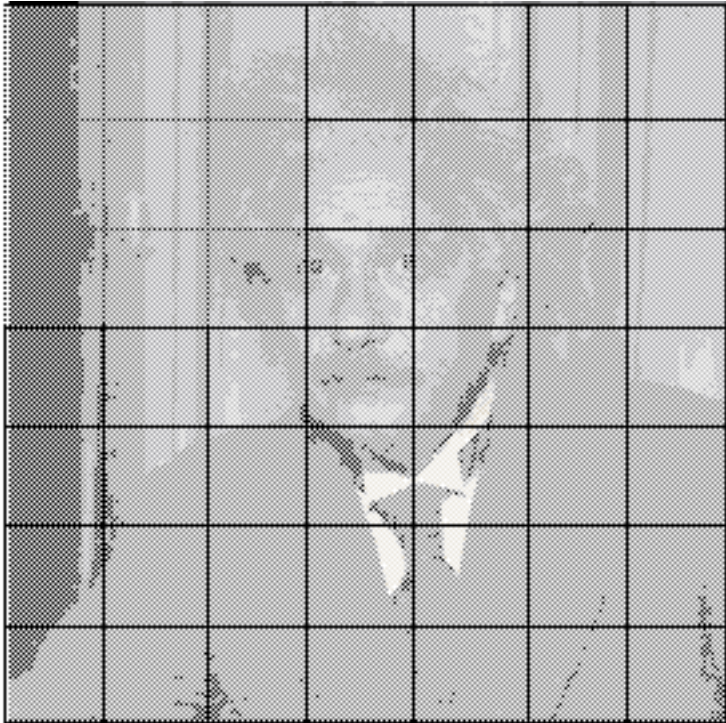
Some images even having less score in the other metrics (PSNR and Blockiness variance) can rank better in the apparent quality test, because are more visually pleasant. (Ex: random noise is more pleasant than a repeated pattern of noise)

Smoothing filters

The easiest way to remove blocking artifacts from a DCT blockwise compressed imaged, is to use a low-pass filter to the image. This way the effect of the sharp edges is reduced but at the same time accuracy of the image is lost, because is blurred.

JPEG comes with his own filter, defined in section K8 of the JPEG standard. This method is based on the interpolation of the image over every 3x3 array of 8x8 blocks of pixels by a quadratic polynomial.

Interpolation leads to the prediction of the low-frequency AC coefficients in the middle of the block based on the DC coefficients.



To follow this method consider the previous image, the colored block represent the 3x3 array of block of 8x8 pixels. Let DC1...DC9 represent the DC coefficients of each block. Then the AC coefficients of the central block are calculated using the next formula:

$$AC01 = 1.13885 \cdot (DC4 - DC6)$$

$$AC10 = 1.13885 \cdot (DC2 - DC8)$$

$$AC20 = 0.27881 \cdot (DC2 + DC8 - 2DC5)$$

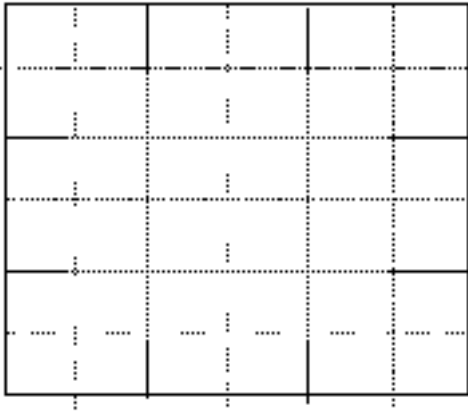
$$AC02 = 0.27881 \cdot (DC4 + DC6 + 2 \cdot DC5)$$

$$AC11 = 0.16213 \cdot (DC1 - DC3) - (DC7 - DC9)$$

This method does not result in good images when the images have sharp edges, this method even adds more large blocking artifacts.

In [6], is proposed a similar approach, to calculate the interpolation of the AC coefficients of a block. In the JPEG method, the AC coefficients of the central block are calculated without having in account the DC5 coefficient, that is the coefficient that represents the central block.

In this method the AC coefficients are predicted using the pixel values of the current block and closer pixels (colored cuadrants). Consider each 8x8 block subdivided in 4 quadrants.



Then AC coefficients in the block are predicted as sums of weighted values of rectangular regions of pixels, using the next prediction equations.

$$DC00 = 0.125(d6 + d7 + d10 + d11)$$

$$AC01 = 0.0016((d5 + d9) - (d8+d12)) + 0.137513((d6+d10) - (d7 + d11))$$

$$AC10 = 0.001614((d2 + d3) - (d14 + d15)) + 0.137513((d6 + d7) - (d10 + d11))$$

$$AC02 = 0.017425((d5 + d9 + d8 + d12) - (d6 + d10 + d7 + d11))$$

$$AC20 = 0.017425((d2 + d3 + d14 + d15) - (d6 + d7 + d10 + d11))$$

$$AC11 = 0.001776((d2 + d5 + d12 + d15) - (d3 + d8 + d9 + d14)) + 0.151278((d6 + d11) + (d7 + d10))$$

In [2] an algorithm is proposed to include the prediction equations, in the DCT inverse transforming, allowing filtering of the image on the fly, as in the JPEG smoothing method.

Spatially adaptive deblocking

The previous filters were spatial invariant filters, developed under the assumption that the characteristics of the signal not change over different regions of the image. But in a typical image characteristics differ considerably from one region to another, because the DCT characteristics and the visual perception. For example, noise in smooth regions is more visible than that in texture edge areas. More complex filters exploit the image characteristics, like do [4][5], and [7].

In [7], a spatially adaptive image recovery algorithm is proposed based on the theory of projections onto convex sets (POCS), using a constraint to enforce between-block smoothness.

In this algorithm the adjacent pixels that are in different blocks are transformed in to accomplish the constraint that enforces the smoothness. Consider i_1 and i_2 as the values of the two adjacent pixels in the same row, then

$$\tilde{i}_1 = i_2 + \alpha(i_1 - i_2),$$

$$\tilde{i}_2 = i_1 - \alpha(i_1 - i_2),$$

$$\alpha = \frac{1}{2} \left[\frac{E}{\|F\|} + 1 \right]$$

Then the resulting image, is the projection of the smoothness constraint set defined by

$$C_s = \{I : \|QI\| \leq E\}$$

Where E is a scalar upper bound that defines the size of this set. The $Q \cdot I$ gives the difference between adjacent columns at the blocks boundaries of the image I . The norm of QI is a measure of the total intensity variation between columns of adjacent of block. They define it as the square root of the edge variance.

$$\|QI\| = \sqrt{\sum (i_1 - i_2)^2}$$

Applying this would result in a spatially invariant method (POCS invariant), but in order to adapt that the image properties, and spatially approach is required. Consider W as a $(256 \cdot 63) \times (256 \cdot 63)$ diagonal matrix where the values are the weights that capture local properties of the image.

$$\begin{bmatrix} w_1 & 0 & 0 & \dots & 0 \\ 0 & w_2 & 0 & \dots & \vdots \\ 0 & 0 & \ddots & 0 & \vdots \\ \vdots & \vdots & 0 & \ddots & 0 \\ 0 & \dots & \dots & 0 & w_{256 \cdot 63} \end{bmatrix}$$

Then a new constraint set is defined where $W \cdot Q \cdot I$ are the weighted version of the differences between pixels of adjacent blocks.

$$C_s = \{I : \|WQI\| \leq E\}$$

The weight values have to be bigger in the places where the blocking artifacts are more visible. In [7] the authors come with an equation, that increases the weight of bright areas, with low detail. For a pixel at location (i, j) , we can consider it as a variable of mean μ , that represents a measure of the brightness and variance σ (a measure of the detail level of the pixel), then the next equation gives the weight.

$$w_{i,j} = \ln \left(1 + \frac{\sqrt{\mu_{i,j}}}{1 + \sigma_{i,j}} \right)$$

But other equations can be used for better description of the visibility of the block artifacts.

Using some simplifications and approximations, the authors come with a simple algorithm, to calculate the projection of the constraint convex set.

1. Calculate the mean and variance values as a constant for every 8x8 region surrounding a vertical or horizontal block boundary. Let DCr and DCI the dc coefficients of the left a right blocks to the boundary, respectively. Then the estimated mean and variance are:

$$\mu = \frac{DCI + DCr}{28^2}$$

$$\sigma^2 = \frac{VACI + VACr}{28^2} b$$

Where VACI and VACr are the sums of the squared coefficients in the blocks left and right to the boundary.

1. Quantize the w values we get the set (q1, q2,...qM). Se set of 3 levels should be enough
2. Partition the pixels of the image in sets I_i , using the rules:
 - Assign all pixels in block boundary with weighting factor q_i to the set I_i .
 - Assign all pixels off the block boundaries to the same set to which its closest boundary pixel belongs.

1. The I_i , contain segments of the image with the same properties. Calculate the spatially invariant algorithm for each set.

Then the image pixels within a set are treated uniformly since they have nearly the same local properties. Therefore the spatially adaptive nature of the algorithm is preserved, with not very increased complexity.

A similar spatially adaptive algorithm is used in [5], where the edge variance between blocks and the intra-block roughness is used as a measure for adjusting the DCT coefficient amplitude. Their algorithm works like that:

1. Measure the blockiness of the image and estimate how blocky it should be.
2. Lower the edge variance to the estimate.
3. Ensure that all DCT coefficients quantize to those of the compressed image.

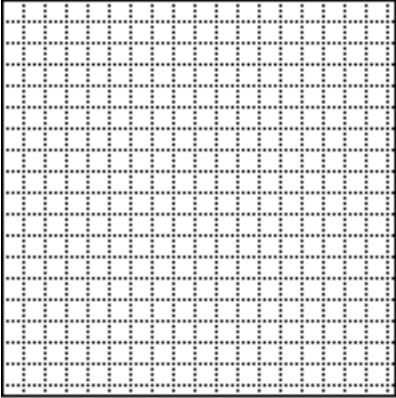
Also proposes after these steps to check the intra-block roughness, and if it increases undo the change of the edge variance. The filter is spatially adaptive if we apply this algorithm for every 8x image block.

The blockiness metrics, is the defined before, consider i_1 and i_2 as the values of two pixels that are next to each other in the same row or column, but in different blocks. Then the next equation defines the edge variance as.

$$E = \sum (i_1 - i_2)^2$$

Computing the same measure for the pixels just inside the block on either side and taking the average

does the estimate of the desired value of edge variance.



In order to lower the edge variance, the pixels i_1 and i_2 are modified as in [7], averaging the edge pixels to obtain the desired amount of edge variance. Adjacent pixels i_1 , and i_2 are replaced by i_1' and i_2' , where.

$$i_1' = \alpha i_1 + (1 - \alpha) i_2$$

$$i_2' = \alpha i_2 + (1 - \alpha) i_1$$

$$\alpha = \frac{1}{2} + \frac{1}{2} \sqrt{\frac{E_d}{E_c}}$$

Where E_d is the desired edge variance and E_c is the current block edge variance. Smoothing is performed only when E_d is less than E_c .

Previous to the filtering, they also propose a method for reducing the MSE of the image, by modifying

the DCT coefficient amplitude, using an exponential distribution.

Considering I as the resulting image after the DCT transform, I is quantized by the operation.

$$S_{u,v} = \text{Round} \left(\frac{I_{u,v}}{Q_{u,v}} \right)$$

And then the reconstruction is done, using:

$$\hat{I}_{x,y} = S_{u,v} Q_{u,v}$$

Horn and Ahumada, replace S modeling it as exponential distribution of mean μ , that is estimated as the mean of $|S|$.

$$S_{u,v} - \frac{1}{2} + \mu - \frac{e^{-1/\mu}}{1 - e^{-1/\mu}}, (S_{u,v} > 0)$$

$$S_{u,v}, (S_{u,v} = 0)$$

$$S_{u,v} + \frac{1}{2} - \mu + \frac{e^{-1/\mu}}{1 - e^{-1/\mu}}, (S_{u,v} < 0)$$

This step is known to reduce the MSE of the blocks. They report their method to perform better in MSE reduction (up to 2dB), and also give a good apparent de-blocking.

Wavelets and soft-thresholding

Given the image x , after DCT reconstruction we obtain y , $y=x+e$, where e is the reconstruction error incorporated by the quantization.

In [8] Donoho theoretically showed that the wavelet soft-thresholding is a nearly minimax MSE procedure to apply to y , in order to minimize the MSE with x . Also is showed that there is not better estimator satisfying the smoothness condition.

The procedure proposed by Donoho and Johnstone, has three steps:

1. Apply the interval-adapted pyramidal filtering algorithm, obtaining the empirical wavelet coefficients.
2. Apply the soft-thresholding operator T , defined as:

$$T_{\lambda}(s) = \text{sgn}(s)(|s| - \lambda) = \begin{cases} s - \lambda, & s > \lambda \\ 0, & -\lambda \leq s \leq \lambda \\ s + \lambda, & s < -\lambda \end{cases}$$

3. Invert the pyramid filtering recovering.

The whole process for the original image y , to obtain the reconstructed image, can be summarized as:

$$\hat{x}(y) = W^{-1}T_{\lambda}(Wy)$$

The threshold is obtained by an estimation procedure from the observed image data y .

This approach not also reduces the mean-squared error alone, it also provides better visual quality than procedures based an on the reduction of the mean-squared error like the POCS filter. Donoho and Johnstone called this method *VisuShrink* in reference to the good visual quality of the wavelet shrinkage.

In an attempt to implement this algorithm there are several parameters to be determined, that will optimize the filter:

- The type of transform W , type of wavelet filter. I will use a two dimensional wavelet transform using the Daubechies transform of length 4.
- The number of levels to use. I will limit the test to 3 level wavelet expansions.
- The value of the threshold parameter. I will use a simple uniform threshold, this gives good result for gaussian noise. More complex approaches can be used to reduce blocking noise that is highly correlated. These approaches use a non uniform/adaptive soft-thresholding to compensate data correlation.

The parameters I selected for the test are based in [9]. Other parameters selection criteria exist that try to minimize the entropy.

For the implementation of this filter I used Wabelab from the Stanford University developed by Donoho and others. This Matlab toolbox is downloadable from the Stanford University Web.

Wavelet soft-thresholding results in the best results both for MSE and visually, with a relatively low processing cost.

FlashPix implementation of filters

Because the FlashPix use of tiles, the filters I tested were developed for working on an on tile based fashion. Working on a tile basis has some effects in the way the filters work.

The smoothing filters based on the DCT coefficients prediction, are easily implemented in a tile fashion. The problem with these filters is that are based on the modification of the JPEG algorithm. The filters we need, must post-process the image, not change the JPEGdecoder. (FlashPix is based in high-performance, low level algorithms that are hard to change)

The POCS filter can be simplified because the use of tiles, especially when high resolution images are used. When the spatially invariant POCS algorithm is used on a tile basis, because the measures of the current E_c edge variance, and the desired edge variance E_d , are for a single tile of 64x64 pixels, the results are comparable to the spatially adaptive POCS algorithm.

Also the Horn and Ahmada algorithm, can be also easily adapted to work on a tile basis.

For tests of the wavelet filters I used the Wavelab package (downloadable from the Stanford University), I didn't develop code specific for working with 64x64 tiles. But the selection of different threshold values for each of the tiles depending of the variance of the pixels, can yield in an effective filter even when an uniform threshold is used, when denoising every tile alone

This is the easier way, but that will leave the noise between the tiles. The second approach is simply modify the algorithms to work from tile to tile, with out the need of keeping the whole image in RAM.

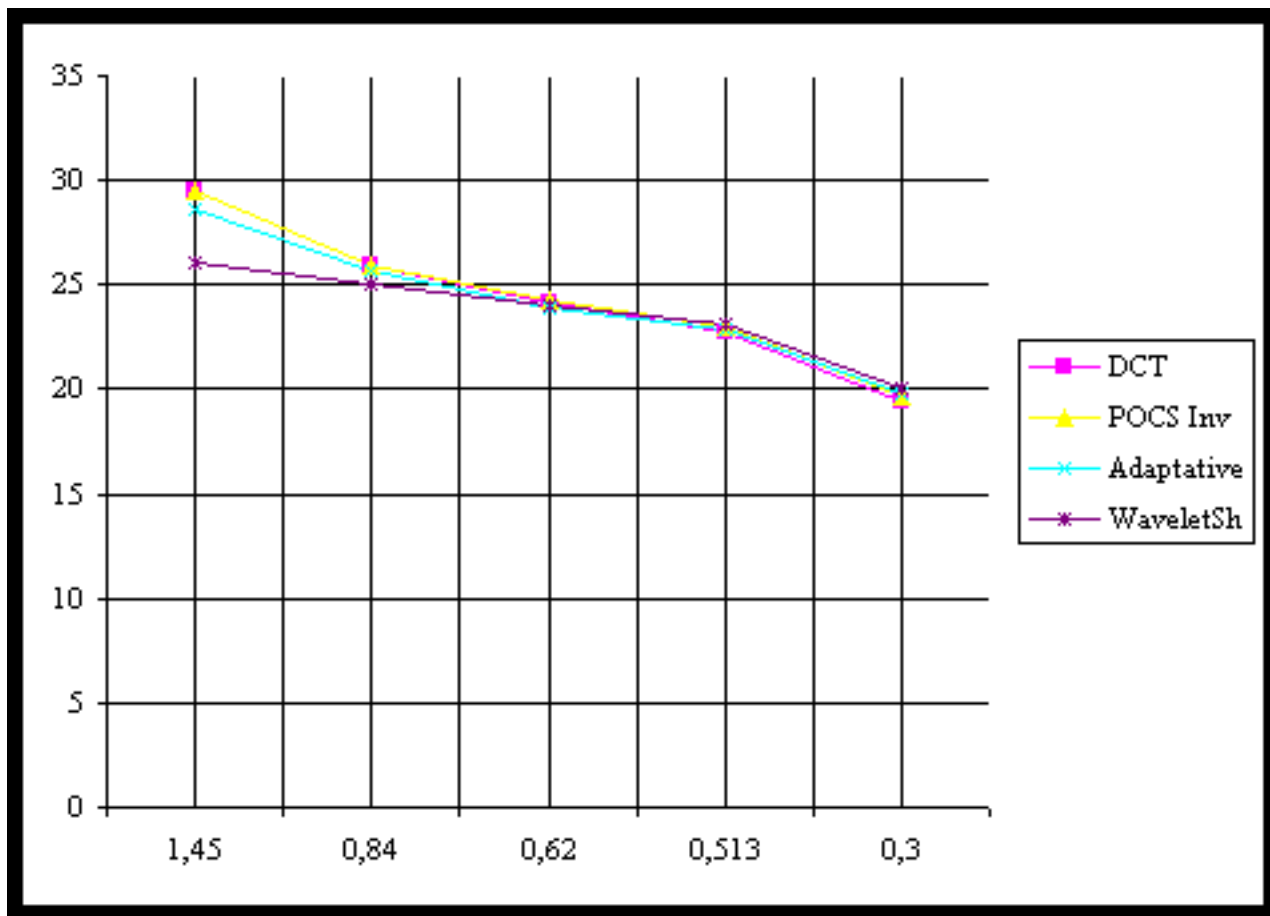
Results

In this section I will show the results of applying this filters to a set of images. The entire test where executed with the Einstein picture, at different compression ratios. Table 1 shows the results obtained by applying the different algorithms to a compressed image of Einstein with different bit rates.

Table 1

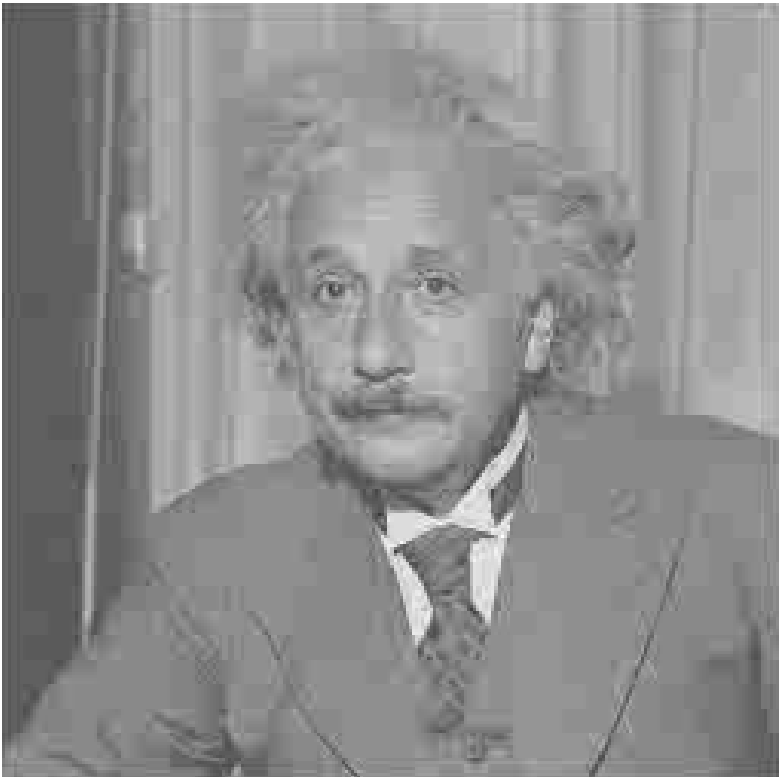
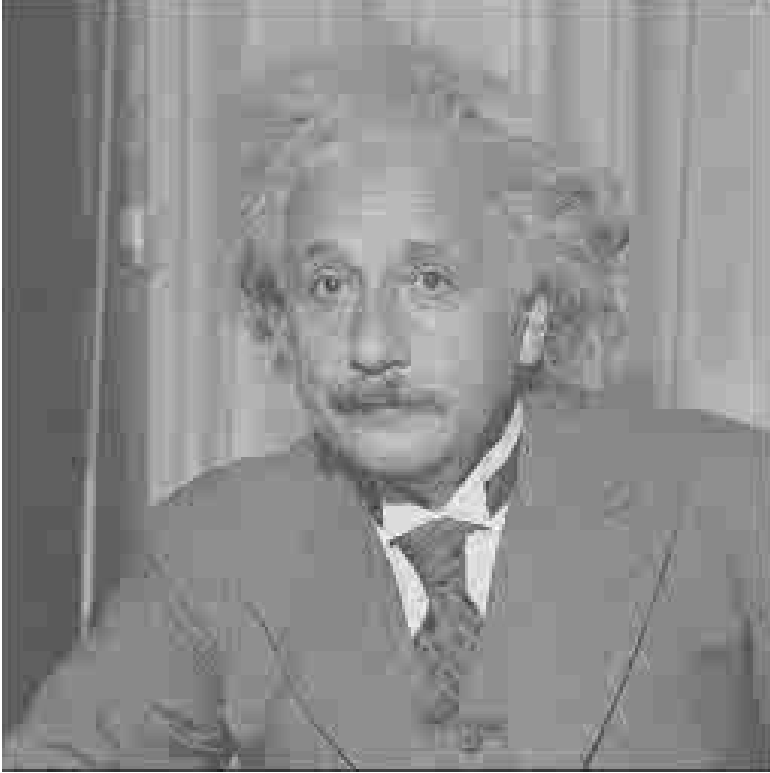
Step	rate	DCT	POCS Inv	Adaptative	WaveletS
16	1,45	29,52	29,52	28,62	26
32	0,84	25,88	25,96	25,64	25,06
48	0,62	24,1	24,25	23,85	24,05
64	0,513	22,71	22,94	22,86	23,11
128	0,3	19,38	19,72	19,84	20

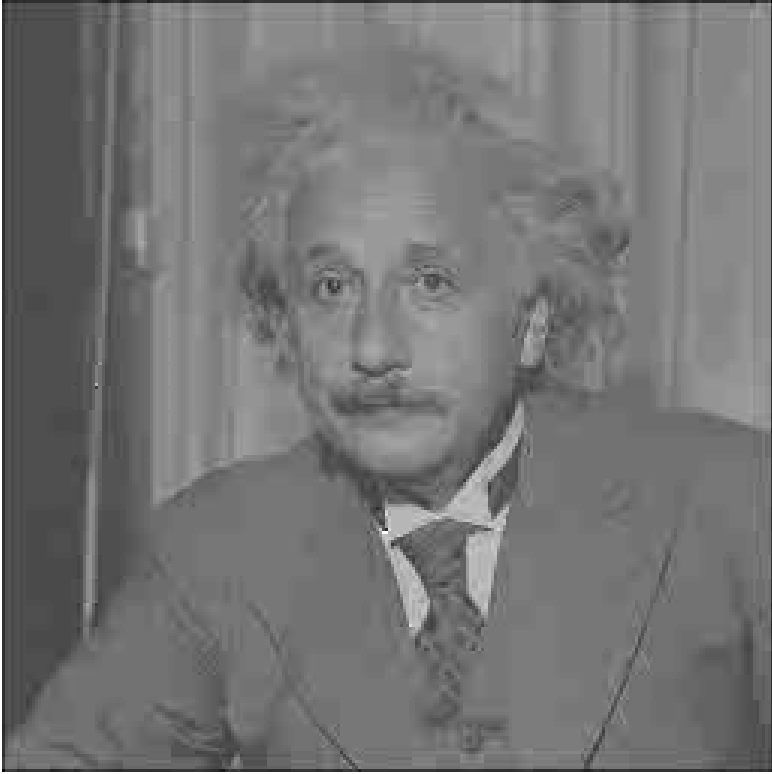
Wavelet soft-thresholding results in the best approach of those I tested in this project, both in the MSE and visually, when low bit rates are used. Good results are obtained even when no adaptive thresholding is applied. It is possible to obtain better results by finding a best threshold factor.

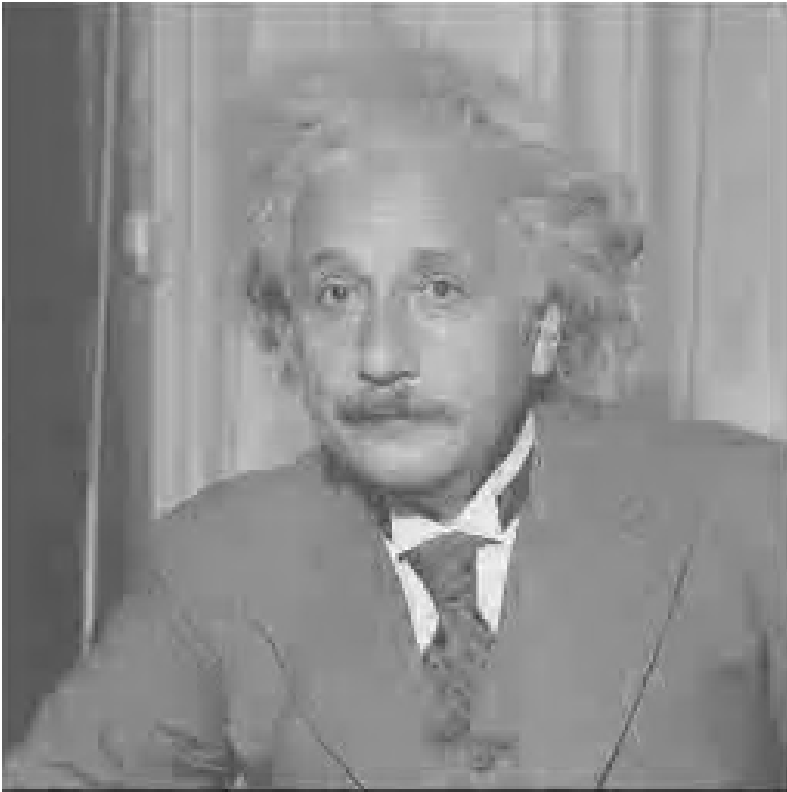


The next images are the result of applying the filters studied in this paper to the Einstein image compressed with a bit rate of 0.514 bpp. The first one is the image as results after the DCT transform. The second results after applying the invariant POCS filter. The third is the resulting of applying the spatially adaptive filter described in [5]. The last image results after applying the wavelet soft-thresholding method.

The best results are obtained by applying the wavelet soft-thresholding. Eve when the algorithm is not maximized.







References

1. **W. B. Pennebaker and J. L. Mitchell, *JPEG still Image Data Compression Standard*, New York ; Van Nostrand 1993**
2. Arun N. Netravali and Barry G. Haskell. *Digital Pictures*, Secon Edition New York; Plenum 1995
3. Gregory K. Wallace, *The JPEG Still Picture Compression Standard*; IEEE Transactions on Consumer Electronics, 1991

4. A .J. Ahumada and Rensheng Horng, *Smoothing DCT Compression Artifacts*, SID Digest, pp 708-711, 1995

5. A .J. Ahumada and Rensheng Horng, *De-blocking compressed images* SPIE Proc Vol 2179 pp. 109-116 , 1994

6. Gopal Lakhani, *Improved Equations for JPEG's Blocking Artifacts Reduction Approach*, Texas Tech University, 1996

7. Youngyi Yang, Nicolas P Galatsanos, Angelos Katsaggelos. *Projection-Based Spatially Adaptive Reconstruction of Block Transform Compressed images*.IEEE Transactions on Image Processing, Vol-4, NO 7, 1995

8. D. L Donoho. *Denoising by soft thresholding*. IEEE Tras Inform Theory, 1994

9. R. A. Gopinath. *Wavelet-Based Post-Processing of Low Bit Rate Tranform Coded Images*. IBM T. J. Watson Research, NY