ECE 508

Project 2

Performance Comparison Between Vector and JPEG Quantization of Images

Brankov Jovan

Comparison of two GLA programs

In this experiment the performance comparison of given GLA, written in C, and my GLA code, written in Matlab, was performed.

Experiment set up:

- The speech signals and parameters set up from project 1
  - Threshold for both $\Delta SNR=0.01dB$
  - Initialization :
    - C: Randomly taken training samples
    - Matlab: Randomly generated points through the max/min space of the sample space.
  - Elimination of empty cells:
    - C: by assigning one sample from the most populated cell to empty cell
    - Matlab: by assigning 20% of the population of the most populated cell to empty cell
  - Same training set for both programs

Note: The given GLA code, written in C, has some problem with empty cell elimination. Some times it get stacked in the loop of trying to eliminated empty cells.

Comparison:

Note: Comparison is performed for two sets of parameter values. One set that highly utilizes Matlab features (optimized build in function) and second set where the use of for-next loop degraded the speed performance of Matlab code.

Parameter set: k=1 r=8 Less influence by “for loop”

C:

Training set: SNR=29.75dB
Test set: SNR=23.736dB
Execution Time: 742.28 in 35 iterations ~21sec/iteration

Matlab:

Training set: SNR=39.0168
Test set: SNR=26.5794dB
Execution Time: 939.313 in 9 iterations ~100sec/iteration

Parameter set: k=8 r=1 Influenced by “for loop”

C:

Training set: SNR=10.85dB
Test set: SNR=7,298dB
Execution Time: 488.5 sec in 30 iterations ~16sec/iteration

Matlab:

Training set: SNR=11.39dB
Test set: SNR=7.825dB
Execution Time: 1748.11sec in 12 iterations ~145sec/iteration

Voronoi cells for k=8 r=1, 1-2 view:

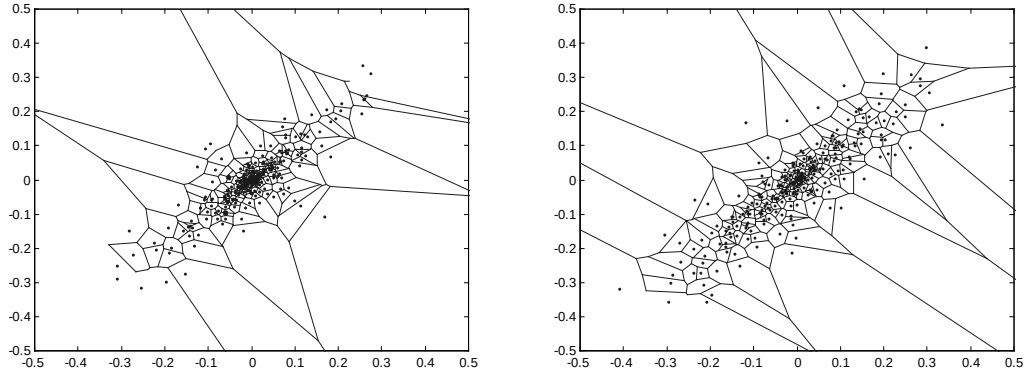

Figure 1. Estimated Voronoi plot of Code book values in pairs x_1, x_2 for r=1,k=8.
Left from C code, right from Matlab code.

From Figure 1 one can notice that the first code book is denser around the zero. That is probably influenced by codebook initialization scheme.

Conclusion:

The C code run faster then Matlab code, but in this set up C code takes more iteration to converge to a minimum.

The difference in the execution time becomes significant for code that can not utilized Matlab build in function.

The performance of the Matlab code is slightly better in terms of the SNR.

VQ for images

Training set contains from 10 images shown on Figure 2.

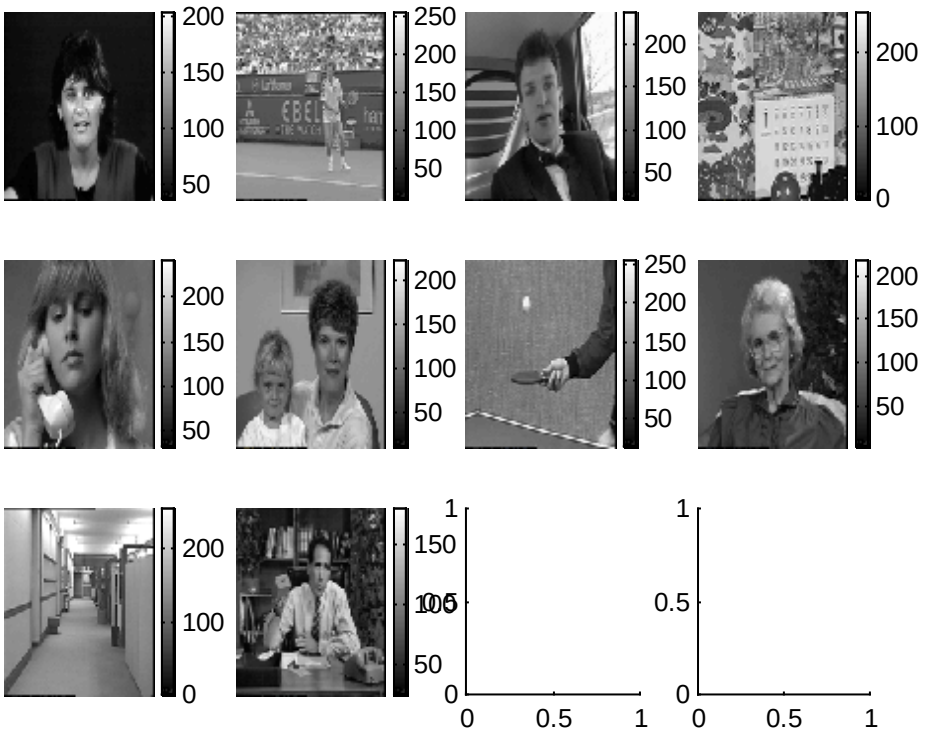

Figure 2. Training set

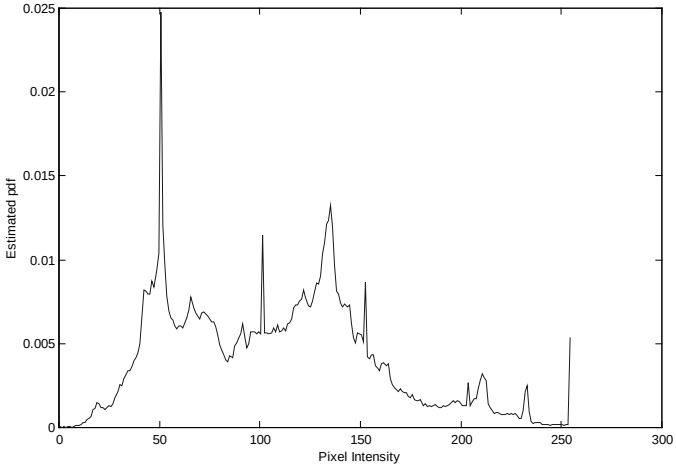

Figure 3. Estimated *pdf* of the training set.

Test set contains two images shown in Figure 4.

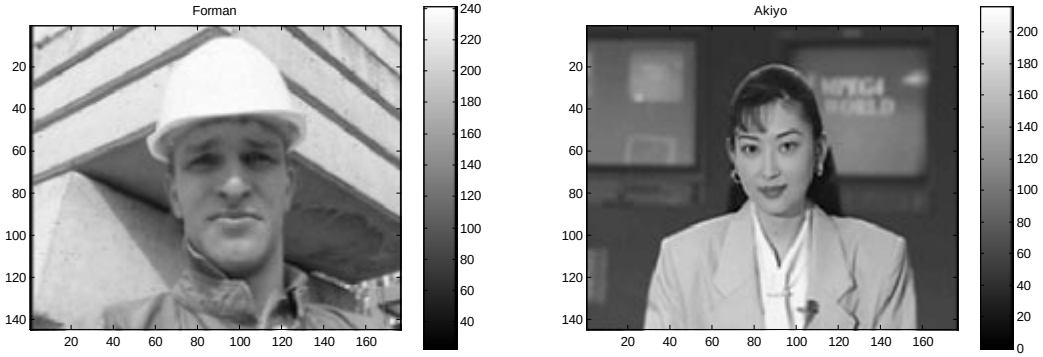

Figure 4. Test set ‘Forman’ and ‘Akiyo’

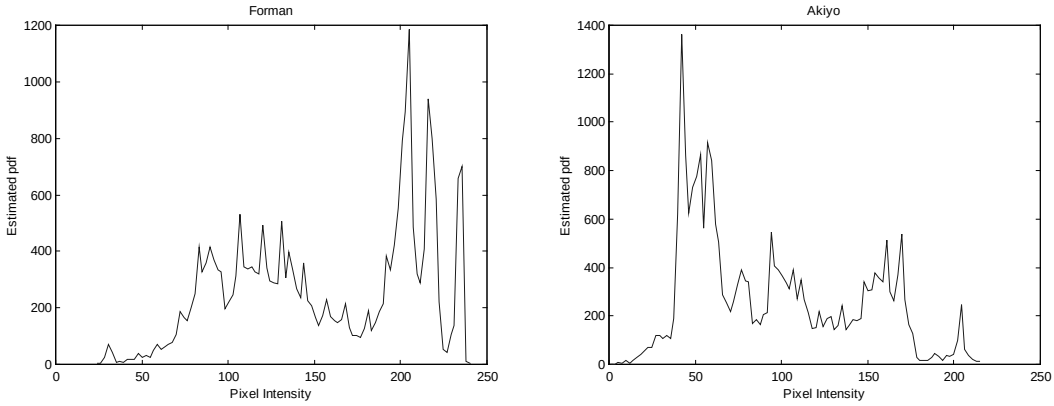

Figure 5. Estimated *pdf* for the ‘Forman’ and ‘Akiyo’

From estimated pdf’s one can notice that the first test image is not represented well in training set. We can expect poorer performance of VQ quantizator for the ‘Forman’ image compare to the ‘Akiyo’ image.

Block scanning

Four scanning strategies were explored: horizontal, vertical, zigzag and random. In fist strategy we scan pixels in the block row by row, in second column by column. In the third the same zigzag strategy as for JPEG was used. And in fourth the pixels are taken in random order.

The difference in PSNR due to different scanning scheme was about 0.05dB. The only observed effect was a slight difference in the edge preservation for low bit rate. For horizontal scanning the horizontal the edges were better preserved. The same observation can be made for vertical scanning. In zigzag scanning both, the horizontal and vertical edges were preserved. So, in the following experiments the zigzag strategy was used.

VQ and JPEG performance measurements: PSNR and coding rate

The measure of the distortion was *peak signal-to-noise ratio* (PSNR), in dB.

$$PSNR = 10 * \log 10 \left(\frac{255^2}{\frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{x}_i)^2} \right)$$

where x_i and $\hat{x}_i$ are the i[th] pixel of lexicography ordered original image $\mathbf{x}$ and quantizied image $\hat{\mathbf{x}}$, respectively.

Codeword length:

The length of i[th] codeword is obtained from the codebook as:

$$l_i = \lceil -ld(p_i) \rceil$$

where p_i is empirical probability measured during codebook training phase.

Coding rate:

$$r = \frac{\sum_{i=1}^{level} q_i l_i}{m * n}$$

where q_i is empirical probability measured for particular image during coding phase, l_i as defined above, and m, n are the block size.

VQ and JPEG performance

The *operational rate-distortion performance* (ORDP) is defined as PSNR vs. the coding rate for given image and coding scheme. The coding scheme is better if the graph of the ORDP of the scheme lies above that of another scheme.

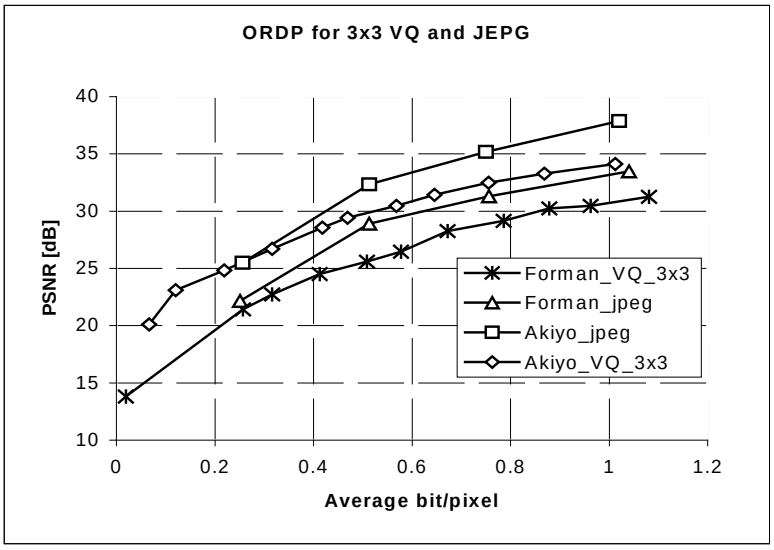

Figure 6. Performance Comparison 3x3 VQ and JPEG

From Figure 6 on can notice that the performance of the 8x8 block JPEG over 3x3 block VQ is better for both test images. If we compare the PSNR for 4x4 VQ whit the 8x8 JPEG, see Figure 7, it can be seen that the performance of the VQ becomes better then JPEG.

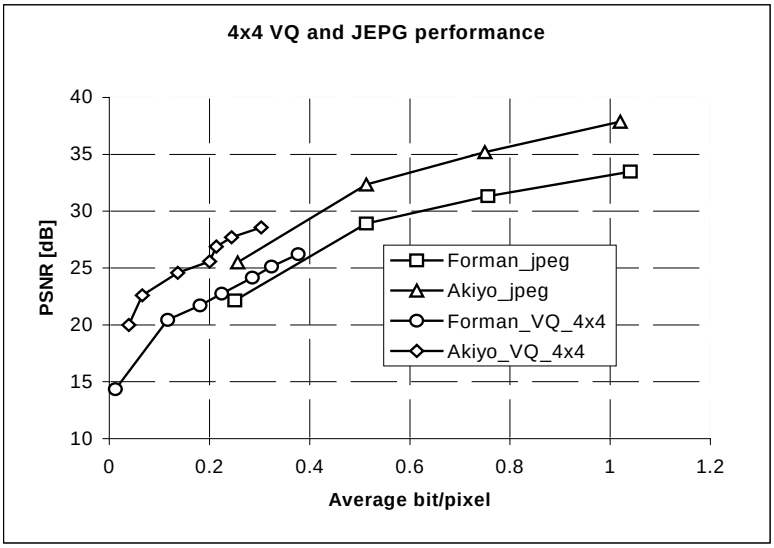

Figure 7. Performance Comparison 4x4 VQ and JPEG

In Figure 8 and Figure 9 the performance of VQ for different block size is given. It can be seen that, as the block size increases the performance becomes better.

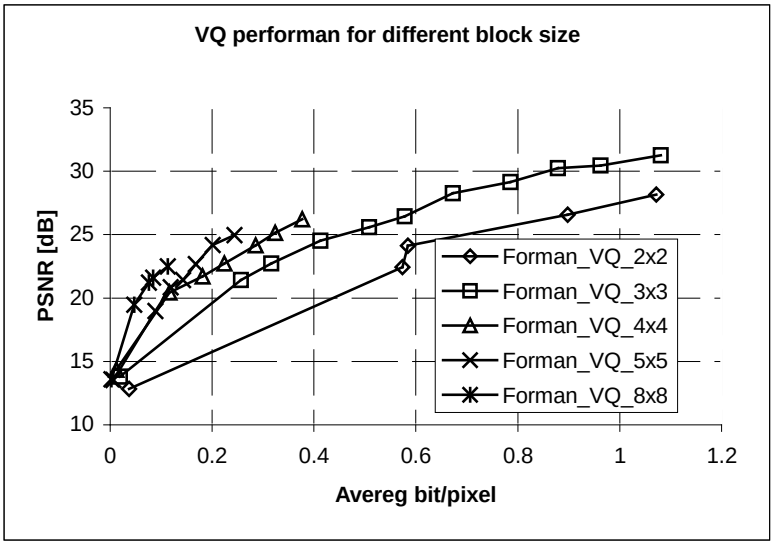

Figure 8. Performance Comparison for different Block Size of the VQ applied on the 'Forman' image

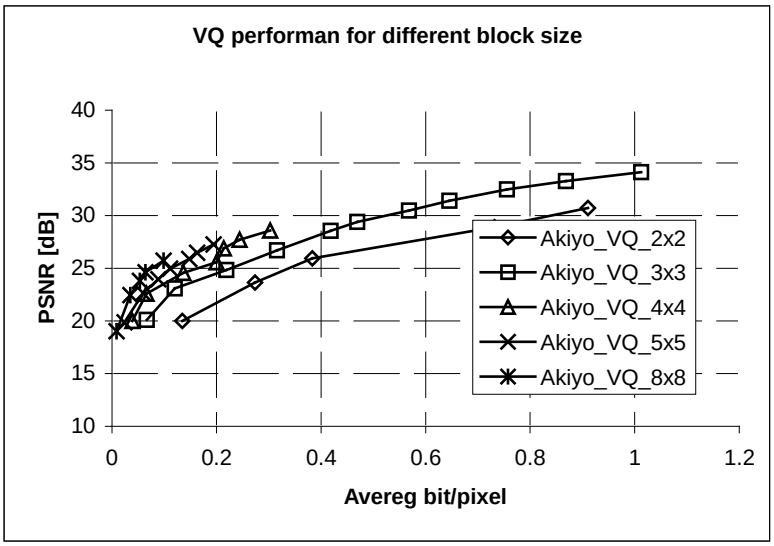

Figure 9. Performance Comparison for different block size of the VQ applied on the 'Akiyo' image

VQ quantized images:

This section contains the 3x3 VQ qunatized images.

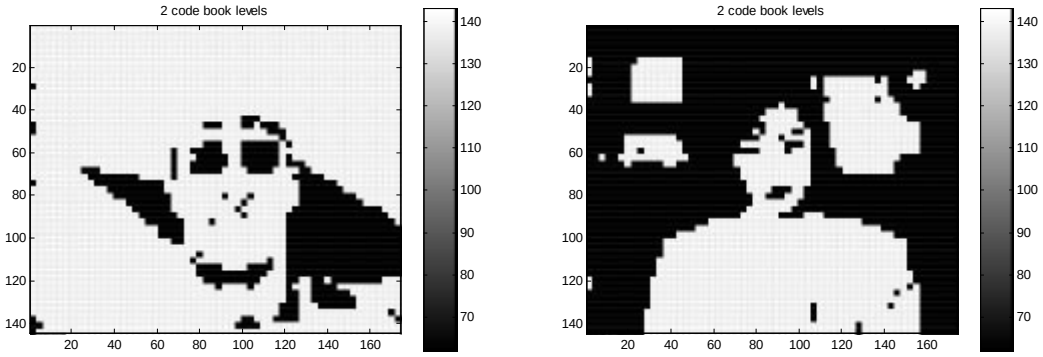

Figure 10. Images obtained by VQ of 2 code point codebook.
left image r=0.019 , right image r=0.066

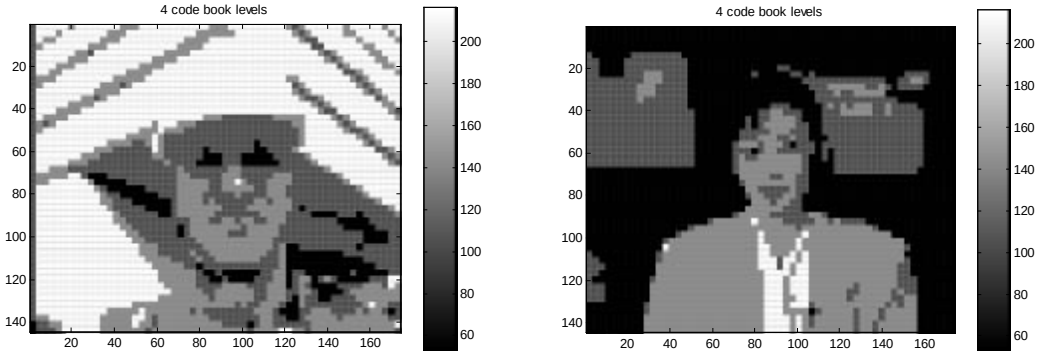

Figure 11. Images obtained by VQ of 4 code point codebook.
left image r=0.25 , right image r=0.12

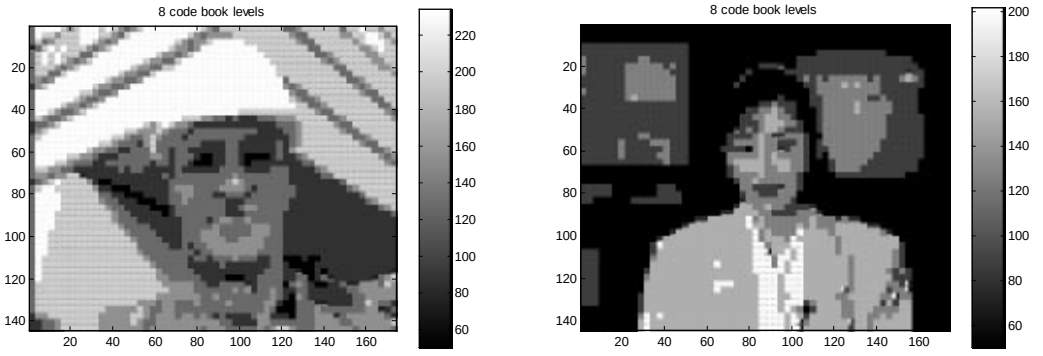

Figure 12. Images obtained by VQ of 8 code point codebook.
left image r=0.31 , right image r=0.21

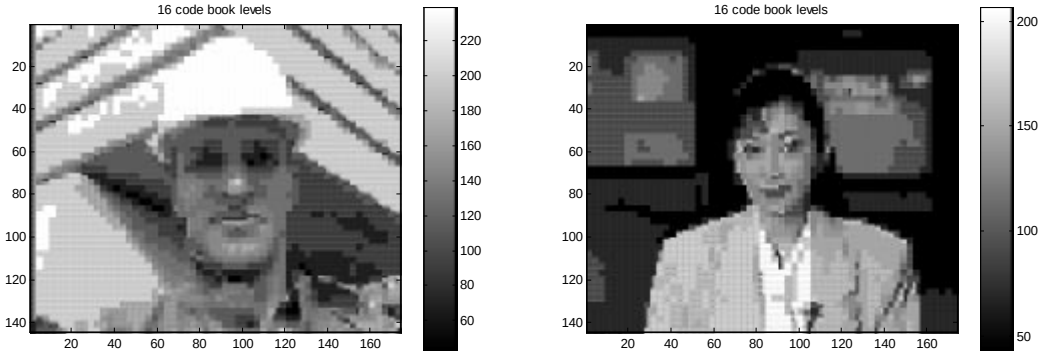

Figure 13. Images obtained by VQ of 16 code point codebook.
left image r=0.41 , right image r=0.31

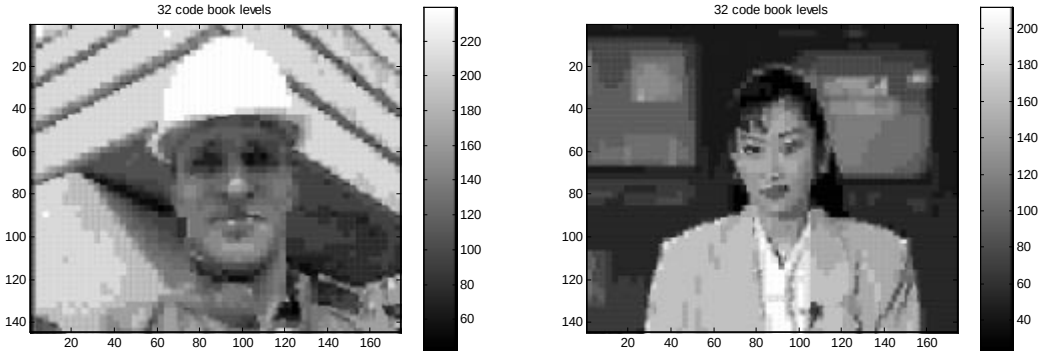

Figure 14. Images obtained by VQ of 32 code point codebook.
left image r=0.50 , right image r=0.41

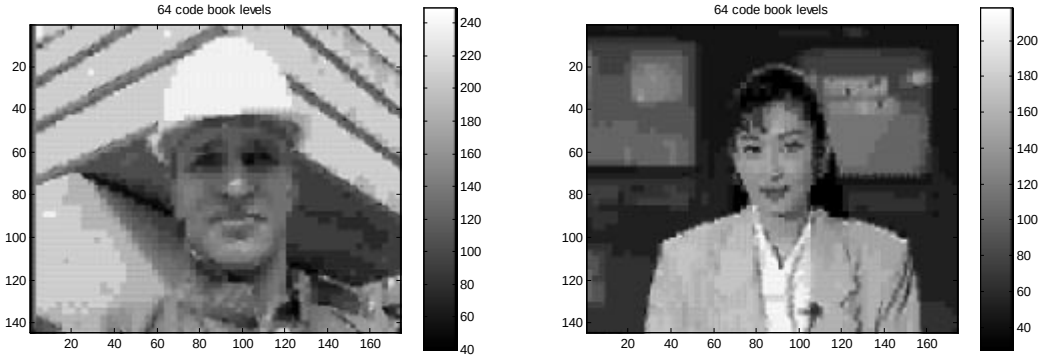

Figure 15. Images obtained by VQ of 64 code point codebook.
left image r=0.57 , right image r=0.469

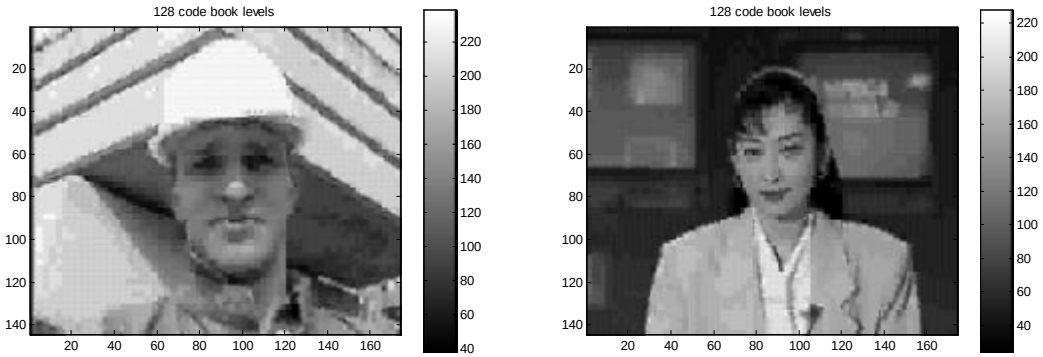

Figure 16. Images obtained by VQ of 128 code point codebook.
left image r=0.67 , right image r=0.56

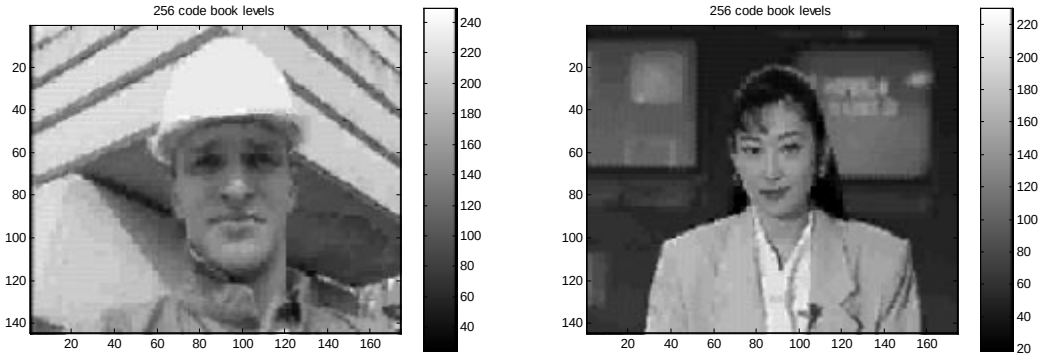

Figure 17. Images obtained by VQ of 256 code point codebook.
left image r=0.78 , right image r=0.64

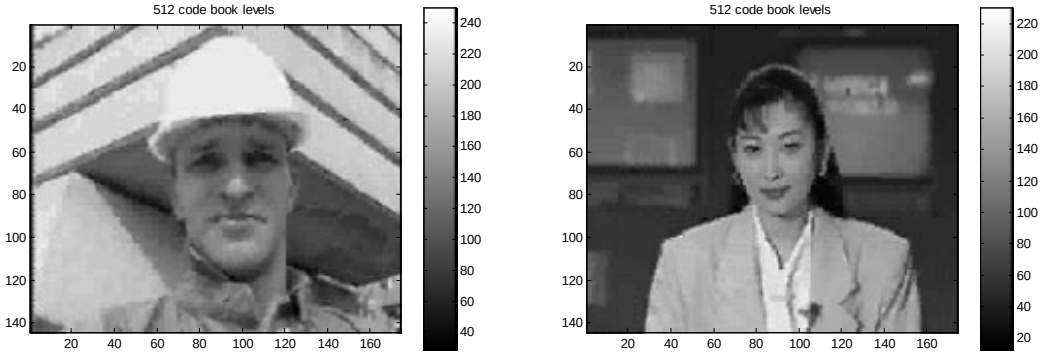

Figure 18. Images obtained by VQ of 512 code point codebook.
left image r=0.87 , right image r=0.75

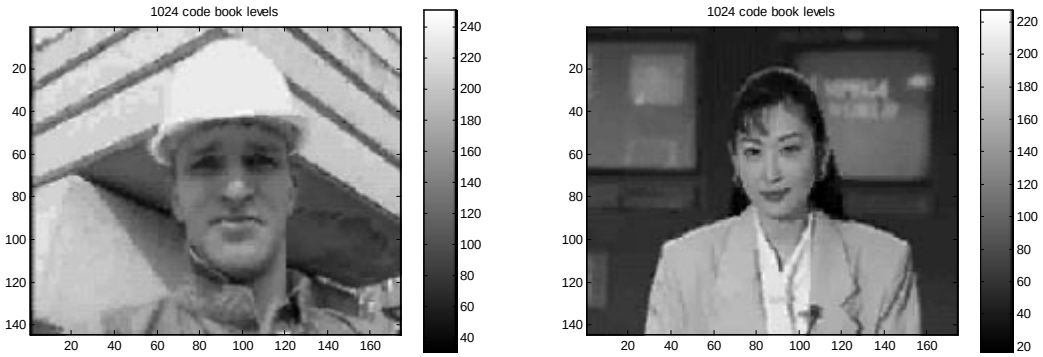

Figure 19. Images obtained by VQ of 1024 code point codebook.
left image r=0.96 , right image r=0.86

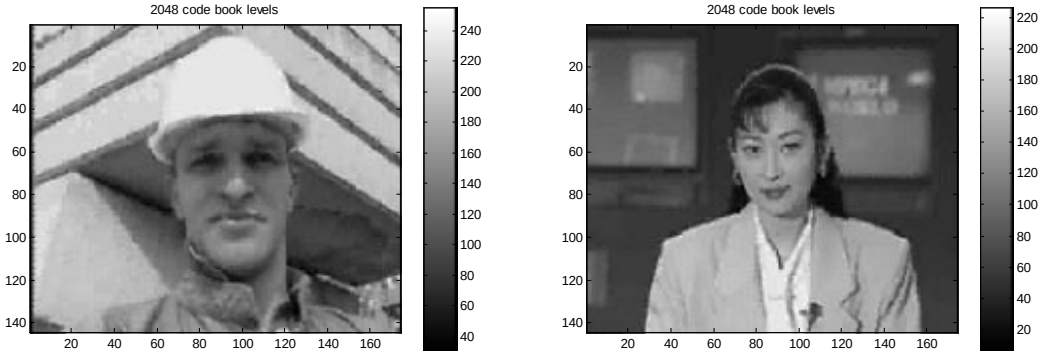

Figure 20. Images obtained by VQ of 2048 code point codebook.
left image r=1.08 , right image r=1.012

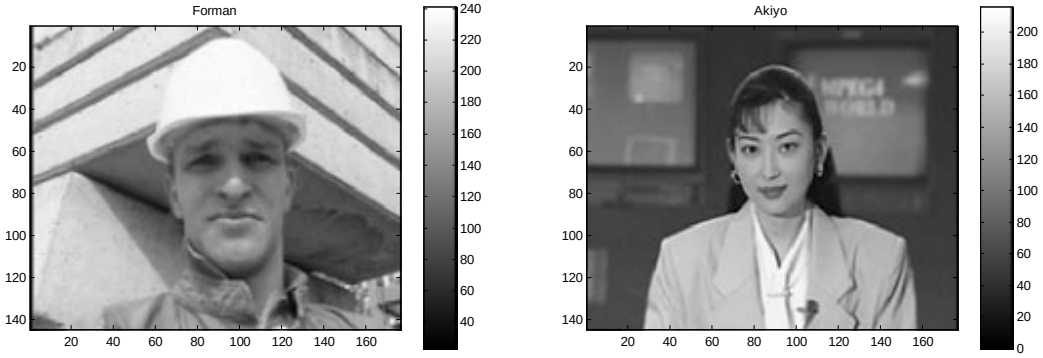

Figure 21. Test set ‘Forman’ and ‘Akiyo’

VQ and JPEG images for the same bit reat

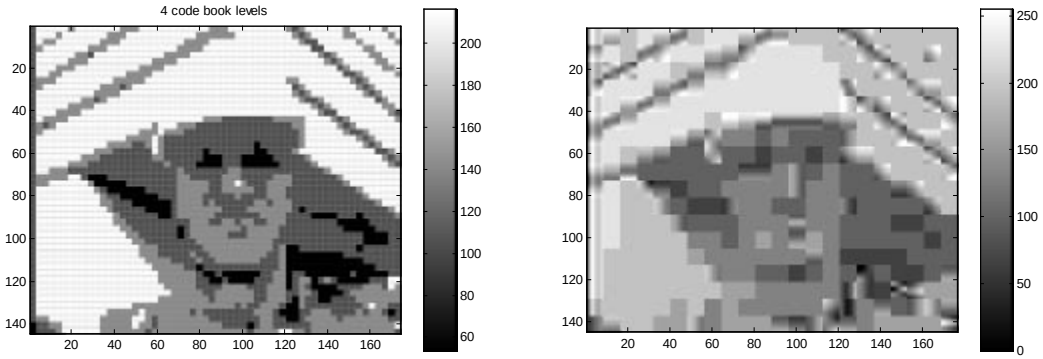

Figure 22. Images obtained by VQ (left) and JPEG (right)
left image r=0.25 , right image r=0.251

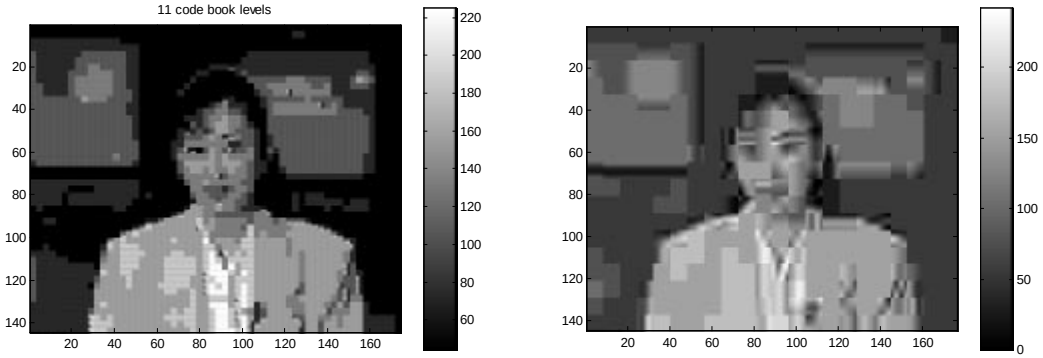

Figure 23. Images obtained by VQ (left) and JPEG (right)
left image r=0.261 , right image r=0.256

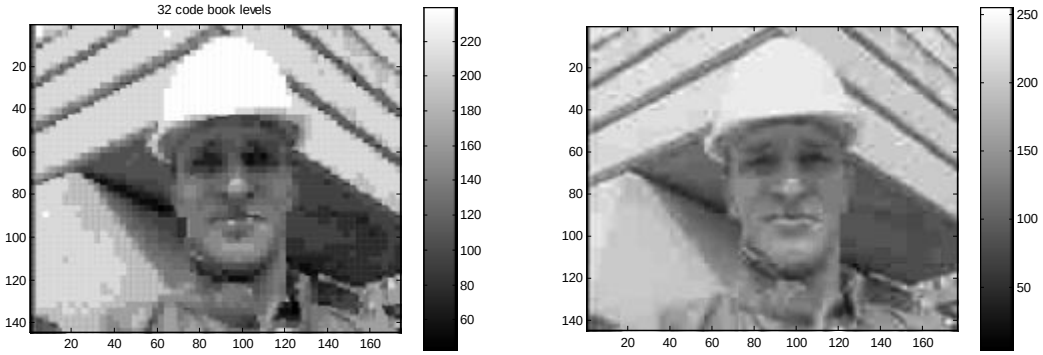

Figure 24. Images obtained by VQ (left) and JPEG (right)
left image r=0.50 , right image r=0.513

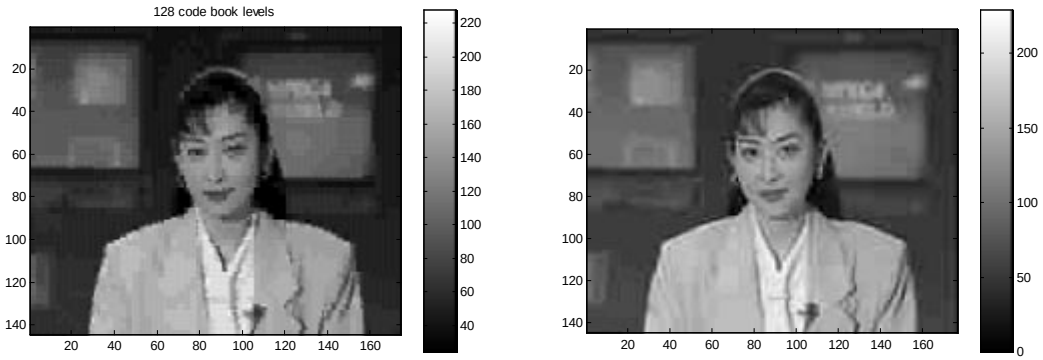

Figure 25. Images obtained by VQ (left) and JPEG (right)
left image r=0.56 , right image r=0.513

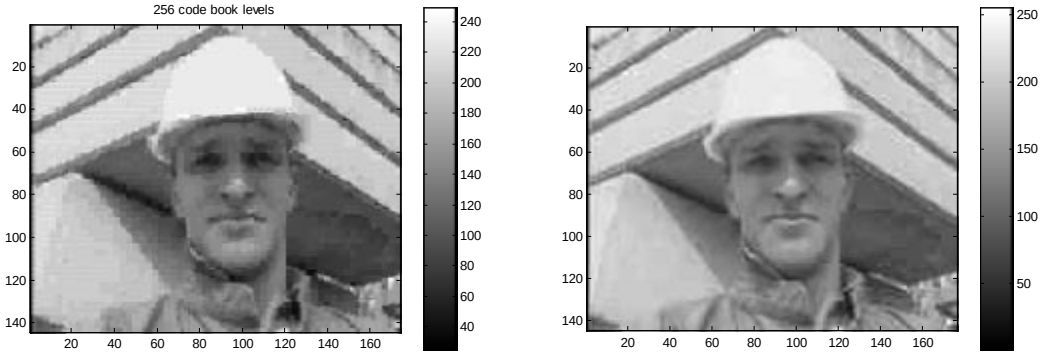

Figure 26. Images obtained by VQ (left) and JPEG (right).
left image r=0.78 , right image r=0.756

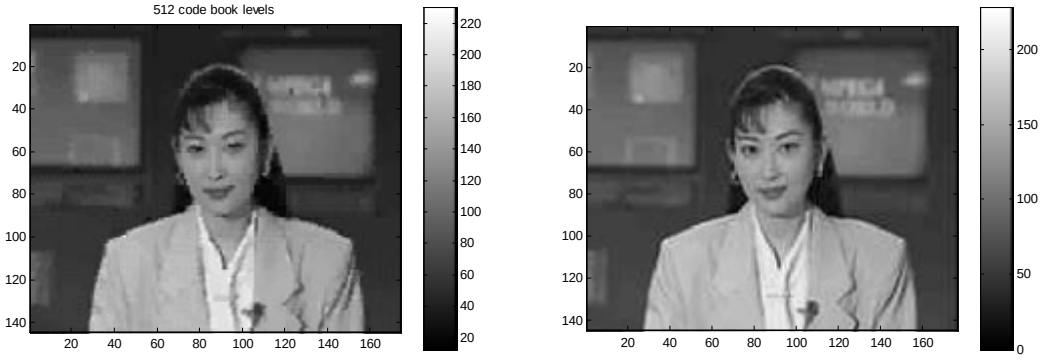

Figure 27. Images obtained by VQ (left) and JPEG (right).
left image r=0.75 , right image r=0.75

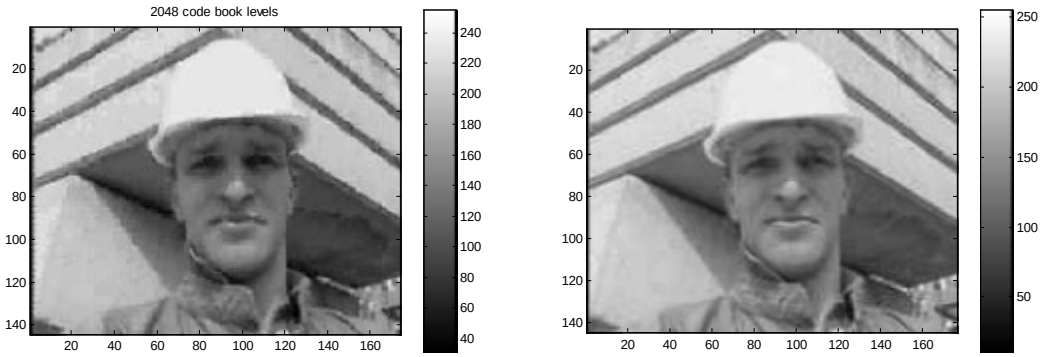

Figure 28. Images obtained by VQ (left) and JPEG (right)
left image r=1.08 , right image r=1.04

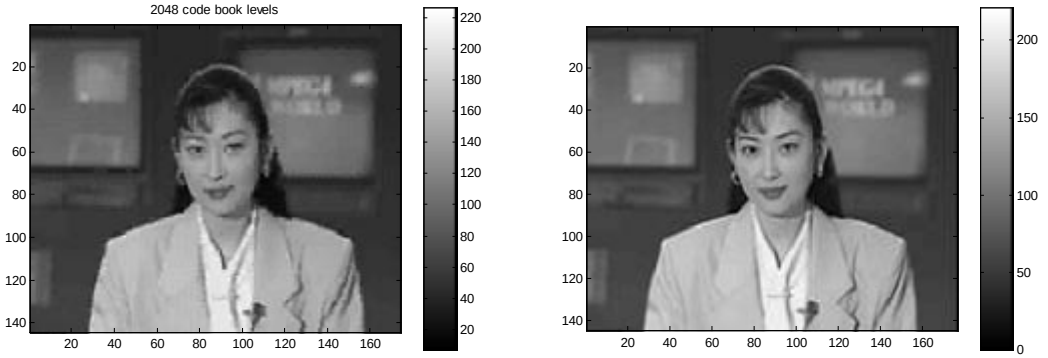

Figure 29. Images obtained by VQ (left) and JPEG (right)
left image r=1.04 , right image r=1.02

By visual comparison I can say that for very low bit rate e.g. r=0.25, the 3x3 VQ is better then 8x8 JPEG, but for higher bit rate the JPEG is better. JPEG gives us smoother images.

VQ codebook as image

During image VQ coding we are basically matching the target block from the original image with one block from the “VQ code book image”, see Figure 30. The one block from VQ “codeblocks” that matches the target block the best, e.g. has minimum distance is output as a code point for target block.

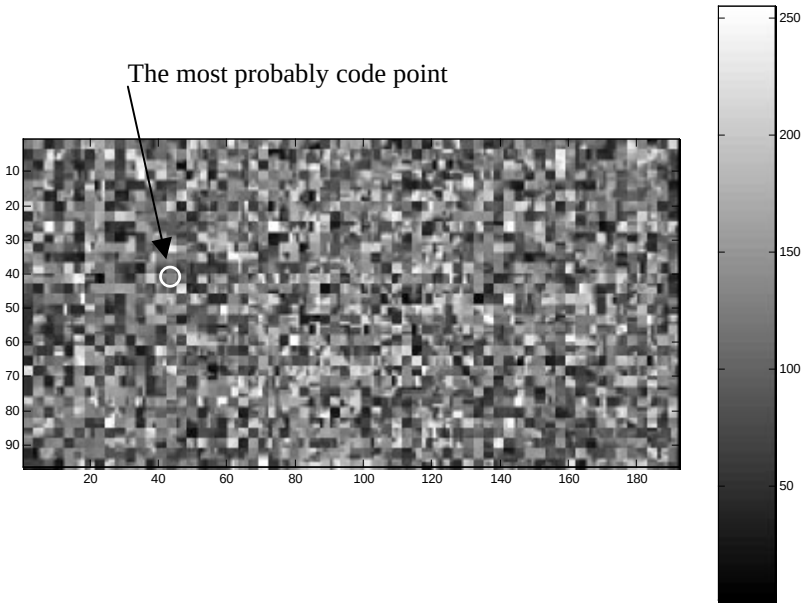

Figure 30. VQ codebook for 3x3 block and 2048 code point presented as an image

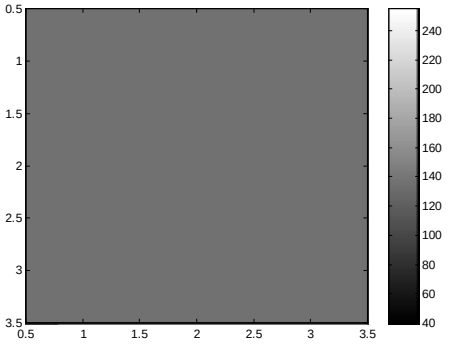

Figure 31. The most probably code point during the training phase.
The pixels values are around the 135.

Visual comparison of VQ and JPEG for the same block size.

The first image, in Figure 32, is obtained by 64 level VQ for 8x8 block size and the second by 8x8 JPEG. One can say that the fist image, obtained by 8x8 VQ, has more visual information then second, obtained by JPEG. The PSNR for both images are close: PSNR_VQ=21.6dB and PSNR_JPEG=22.15dB. The code rates are quite different, r_VQ=0.0845 and r_JPEG=0.251.

So, to summarize observation above: by applying 8x8 VQ instead 8x8 JPEG, we obtained the better image, which can be represented by fewer bits.

The price for applying the VQ is the execution time that is much bigger then for JPEG. Also the execution time increases rapidly as the number of code points increases. This prevents us from using the VQ for large codebooks.

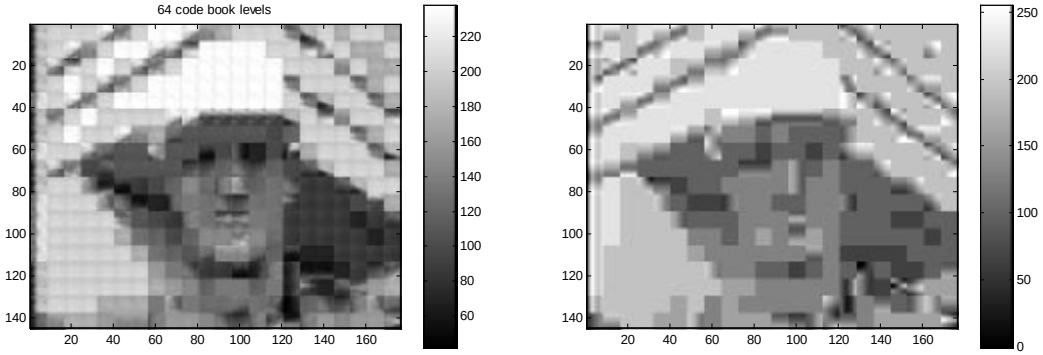

Figure 32. Images for visual comparison between VQ and JPEG.
The left image is obtained by 8x8 VQ and the right by 8x8 JPEG
PSNR_VQ 21.6dB and PSNR_JPEG 22.15dB,
r_VQ=0.0845 and r_JPEG=0.251