**Military Technical College**
**Kobry El-Kobbah,**
**Cairo, Egypt**

**7th International Conference**
**on Electrical Engineering**
**ICEENG 2010**

# Object based video coding algorithm

*By*

A. M. Sallam* M. Shaarawy** O. M. Elmowafy * R. A. Elbordany * A. M. Fahmy *

***Abstract:***

This paper introduces a new trend in video compression that based on content based coding this solution called Object Based Video Coding (OBVC). A new solution for Discrete Cosine Transform (DCT) used in video compression is presented. This solution uses the Artificial Neural Networks (ANNs) and has been implemented and tested with the OBVC. This paper tries to present a solution based on OBVC and ANN to speed-up the encoding and decoding processes for video compression and get more compression ratio.

***Keywords:***

Neural Networks, DCT, JPEG, M-JPEG, H-261, Image and Video Compression, Object Based Video Coding

* Egyptian Armed Forces
** Faculty of Computers & Information, Hallwan University, Egypt

## *1. Introduction:*

All the current standards of video compression such as MPEG for video compression, M-JPEG for moving sequence and H-261 for video conferencing are based on DCT and motion compensation. DCT based compression depends on dividing the frame into small square blocks for processing to be able to adapt their performance. Image compression methods can be divided into 2 categories, lossless and lossy compression methods. The compression method is called lossless when the original image can be exactly reconstructed from the compressed version without any loss in the image information while the compression method is called lossy when an approximated image is reconstructed from the decompressed image file [1].
In this paper, the discussion will be centered on the lossy compression and tries to get a solution for video compression based on OBVC and ANN.

## *2. Discrete Cosine Transform:*

The main characteristics of the DCT of a typical block of an image are transformed the visual information of that block to coefficients (DC coefficient and AC coefficients). The most energy of the block is concentrated in the DC and a few coefficients that follow the DC coefficient as shown in figure 1. For this reason, the DCT is often used in image compression applications. For example, the DCT [2, 3] is at the heart of the currently worldwide international standard lossless and lossy image and video compression algorithms. Such as Joint Photographic Expert Group known as JPEG for image compression [1, 4] and the standard H-261 for video conferencing and videophone [4, 5].

## *3. Image and Video Compression Algorithms:*

### 3.1 JPEG Standard Algorithm for Image Compression:

1-JPEG lossless compression figure 3 [6, 7],
2-JPEG lossy compression figure 4 [6, 7].

The essiential blocks used in the JPEG lossless and lossy compression:

Mapper: The basic idea of the Mapper is to transform the original data (image) into an alternate form where redundancy can be reduced by quantization.

Quantizer: A key feature of the quantized transform coefficients is that many of them are zero, making them suitable for efficient coding.

Symbol encoder: This unit assigns a code word to the symbols that appear at its input, and it generates the bit-stream that is to transmitted or stored. Huffman coding usually employed for variable-length coding of the symbols.

3.2 Video Compression Algorithms:

- Intra picture (I-picture):

I-picture is coded isolated with no reference to any other frame in the sequence and it act as a reference point for random access into the video stream (still image compression).

- Non-Intra picture (Inter):

P-picture is coded with reference to the nearest pervious I- or P-picture, which are collectively referred to as reference pictures. This technique is called forward prediction (it provides more compression and serves as a reference for B-pictures and P-pictures).

The B-pictures is coded relative to the nearest past reference picture, the nearest future picture or both. This technique is called backward prediction.

- Motion compensation (MC):

It is a technique for enhancing the compression of P-pictures and B-pictures by eliminating temporal redundancy. Motion compensation typically improves compression by about a factor of three compared to intra-picture (I-picture) coding. It works at the macroblock level [8].

*3.2.1 M-JPEG Video Compression Algorithm:*

Motion-JPEG (M-JPEG) video image compression algorithm is a compression scheme uses the intra-frame (I-frame) coding technique it is like JPEG standard algorithm for still image compression in a loop while ending the image sequence. The M-JPEG not uses the motion estimation and compensation technique it does not have the same compression ratio as other video image compression standards that uses the motion estimation and compensation techniques.

M-JPEG is used to reduce only the spatial redundancy and is used in the moving objects moving background because if we use the inter-frame coding techniques such as motion compensation it will produce in that case more bits and reduces the compression ratio [9, 10].

*3.2.2 H-261 Video Compression Algorithm:*

H.261 video coding standard, use the hybrid DCT-MC framework. H.261 is a video image compression technique developed by the ITU-T (International Telecommunication Union), meant for videoconferencing, video-telephone application over ISDN telephone lines [11].

There are two types of frames I-frames (Intra-frames) and P-frames (Predicted-frames). The I-frames uses basically JPEG codec algorithm where P-frames uses “pseudo-differences” from previous frame by the technique of Motion Compensation, so frames depend on each other.

The I-frames provide us with an accessing point. I-frame is coded by sub-dividing the frame into 8 pixel by 8 lines blocks which are segmented and transformed by the DCT and quantized. These resulted coefficients, after the zigzag are quantized and applied to the RLE and entropy encoder.

The main characteristics of the DCT of a typical block of an image are transformed the visual information of that block to coefficients (DC coefficient and AC coefficients). The most energy of the block is concentrated in the DC and a few coefficients that follow the DC coefficient as shown in figure 1. For this reason, the DCT is often used in image compression applications. For example, the DCT [2, 3] is at the heart of the currently worldwide international standard lossless and lossy image and video compression algorithms. Such as Joint Photographic Expert Group known as JPEG for image compression [1, 4] and the standard H-261 for video conferencing and videophone [4, 5].

## ***4. The Proposed ANN Approaches for DCT:***

This section uses ANN to implement the DCT and Inverse Discrete Cosine Transform (IDCT). For both networks we use a NN with the following characteristics figure 3:

1- Single-hidden-layers, fully connected ANNs [12, 13] is chosen to implement the DCT and the IDCT NNs (DCTNN & IDCTNN).
2- 64 input nodes, 64 output nodes, and the transfer function for the output nodes is a linear function. A feed-forward back-propagation ANNs will be used.
3- The weights and biases for the DCTNN were saved into a data file, and the weights and biases for the IDCTNN were saved into another file.

Training both DCTNN and IDCTNN:

Using a set of images from the MATLAB Image Processing Toolbox were used for training. The image file "camerman.tif" that shown in figure 4 was one of these images.

The training steps:

1- Subdividing the image into non-overlapping data blocks of size 8-by-8 pixels.
2- Compute the 2-D DCT for each 8-by-8 block.
3- Transform each 8-by-8 block into 64-by-1 vector for the purpose of training.

Training the DCTNN:

The gray level of the 8-by-8 blocks form the columns of the input vector and the DCT of the 8-by-8 blocks form the columns of the target vector. The final MSE of this network = $7.8605 \times 10^{-14}$.

Training the IDCTNN:

The DCT of the 8-by-8 blocks form the columns of the input vector and the gray level of the 8-by-8 blocks form the columns of the target vector. The final MSE of this network = $7.829 \times 10^{-14}$.

## ***5. Object Based Video Coding (OBVC):***

Image and Video Compression Generation:

Image and video compression can be divided into six generations table (1). The fourth

generation is called content based coding (Object Based Coding).

Table 1 Image and Video Coding Generation [14]

| Coding Generation | Approach | Techniques |
|---|---|---|
| Zero Generation | Direct waveform coding | PCM (Pulse Code Modulation) |
| First Generation | Redundancy removal | DPCM (Differential Pulse Code Modulation), DCT (Discrete Cosine Transform), DWT (Discrete Wavelet Transform), VQ (Vector Quantization) |
| Second Generation | Coding structure | Image segmentation |
| Third Generation | Analysis and synthesis | Model-based coding |
| Fourth Generation | Recognition and reconstruction | Content-based coding |
| Fifth Generation | Intelligent coding | Semantic coding |

Benefits and draw-back of the object based coding:

The benefits of using the object-based compression:

1- Increasing the efficiency of the compression (compression ratio).
2- Enable manipulation of compression ratio.
3- Improve robustness in unreliable communication system.
4- Close the gap between technical systems and the human visual system (HVS).

The draw-back of using the object based compression:

1- Complexity in time to find the object boundaries (contour).
2- Complexity in coding and tools used to separate the object by the segmentation techniques according to (color, motion, …).
3- Complexity to get the whole background that is in the scene or sequence of frames, because the movement of the object will produce a new segment will be grouped to the background.

Finding the object:

We can find the object by one of the two following techniques:

1- Chroma-keying where mask is created from the difference between color of the object to be masked and the color of the background with a specific color such as blue [1].
2- Luma-keying is like Chroma-keying, except that the mask is derived from the difference between luminance of the background and the object to be masked, and is not as easily discernable unless using a black or white mask [1].

## ***6. Proposed System for Object Based Video Coding:***

A proposed video compression algorithm using a new technology for compression called OBVC introduced in this section. OBVC algorithm is belong to the fourth generation of image and video compression that is called recognition and reconstruction generation. This algorithm uses the object-based (content-based) coding technology and the algorithm consists of 4 steps:

1- Finding the object (by the mean of luma-keying).
2- Describe and separate the object (by segmentation).
3- Describe the background in the video sequence and code it with low quality by Intra-frame coding technique like (JPEG algorithm).
4- Used the Inter-frame coding technique like (Hybrid coding) with special tools to encode the movement of the object.

Figure 5 illustrates the operation of the OBVC.

Performance Analysis of OBVC

This section tests the proposed OBVC algorithm and compares the OBVC and the two standard algorithms M-JPEG and H-261 in encoding and decoding time and in the quality (MSE, PSNR, CR). Figures 6, 7, 8, and 9 presents the reconstructed video signal using standard M-JPEG, H-261 and OBVC for the four video sequence ("ball", "M_balls", "Object", "M_object").

Table 2 provides comparison between the standard JPEG, standard H-261 and OBVC in terms of encoding time, decoding time, MSE, PSNR and CR of the reconstructed video. From this table we may note the following:

1- The OBVC for all video sequence gets more speed in the decoding process than the standard M-JPEG and standard H-261.
2- The whole sequence takes more time for encoding process when using the OBVC, because this new coding technique is complex and perform more iterations on every frame in the sequence.
3- The quality of the video (PSNR) of the standard M-JPEG, standard H-261 and OBVC is exactly the same because they use the same algorithm (DCT).
4- The compression ratio of the standard H-261 is higher than the standard M-JPEG, because the latter codes the whole frame in the all sequence, but the standard H-261 codes the difference between the frames of the sequence.
5- The compression ratio of the OBVC is higher than that of both standard H-261 and the standard M-JPEG, because the OBVC algorithm codes the new location of the object in the frame while the H-261 codes the difference between the frame and the M-JPEG codes the whole frame in the sequence.
6- The encoding and decoding times for the M-JPEG is independent on the number of objects in the frame sequence. While the H-261 encoding and decoding times are the

same for the same number of objects in the sequence of the frames such that the relative motion between objects does not change. But OBVC encoding and decoding times are proportional to the number of objects and the number of the frames in the sequence. This property is considered as a draw back for this algorithm and needs more investigation to have for example an algorithm for parallel object segmentation or let these times independent on the number of objects.

7- It is observed that the performance (CR, PSNR, encoding and decoding times) is affected by the variability in the contents of the frames in the sequence. In other words M-JPEG achieves better performance than H-261 and OBVC.

Proposed System for OBVC Video Compression by ANN:

A proposed video compression algorithm using OBVC based on the ANNs introduced in this section, by replacing the JPEG encoder in figure 5 by JPEG encoder that uses DCTNN the proposed algorithm will be OBVCNN.

Performance Analysis of OBVCNN:

This section tests the proposed algorithm OBVCNN with the OBVC in encoding and decoding time and in the quality (MSE, PSNR, CR). Figures 10, 11, 12, 13 presents the reconstructed video signal using OBVC, and OBVCNN for the four video sequences ("ball", "M_balls", "Object", "M_object").

From the table 3 it is found that:

1- Applying the DCTNN on the OBVC gets more speed in the encoding and decoding time than the standard DCT.
2- The quality of the video sequence (PSNR) of the OBVCNN are exactely the same as OBVC because the DCTNN has a high performance.
3- The compression ratio of the OBVCNN is the same as OBVC because the DCTNN produce coefficients as the standard DCT after quantization.

## ***7. Analysis of the Obtained Results:***

From tables 2, 3 it is found that most of the computation time required consumed for finding the objects in the frame in the encoding process of the OBVC, but the decoding process have less time than the standard M-JPEG or the standard H-261. OBVCNN spent less time than OBVC and reserving the PSNR high. These times still high for the encoding process. The platform that used to obtained these results is (Pentium 4, 1.7 GHz, with 128 MB RAM) and MATLAB version 6.1.

This paper introduced a new technique of using ANN in DCT for using in video compression that based on Content-based coding.

The main achievements of using ANN in DCT with the Content-based coding:

1- Reducing complexity.
2- Speed-up the time.
3- Number of epochs used in training phase to get the MSE = 5.3515e-14 does not exceed 200 epochs in 0.72 second for the NNDCT.
4- Number of epochs used in training phase to get the MSE = 2.2601e-12 does not exceed 200 epochs in 0.761 second for the NNIDCT.
5- The PSNR for the reconstructed image in the training phase is very high and equal (484.333 db).

The obtained results in this paper showed the robustness of using Object Based Video Coding and ANN in DCT. This research is part of project of video image compression. However, the time needed for separate the objects from the background of the frame sequence is high was not the interest of this research at this stage that is why it was kept very high. It is expected; in the future using an algorithm for parallel object segmentation can improve the encoding time for OBVC.

***References:***

[1] Hand book of Image & Video Processing by ALBovik, academic press, 2000 .
[2] N. Ahmed, T Natrajan, and K. R. Rao, "Discrete Cosine Transform", IEEE Trans. Computer c-23,90-93 (1994).
[3] Chen, W.-H., C. H. Smith, and S. C. Fralick "A Fast Computational Algorithm For the Discrete Cosine Transform ", IEEE Commun. Trans. COM-25: 1004-1009 (1977).
[4] ISO/IEC 13818-2 Information Technology-Generic coding of moving pictures and associated audio information video, 1996.
[5] Digital Image Processing by Rafael C. Gonzalez, Richard E. Woods, 1993.
[6] ISOIEC JTC1/SC29/WG1 congener-Dr. Daniel Lee A study of JPEG 2000 still image coding versus other standards July 2000.
[7] Techniques & standards for Image, Video, and Audio coding by K. R. Rao, J. J. Hwang copyright 1996.
[8] D. T. Hang, P. M. Lang, and J. S. Vitter,"Explicit Bit Minimization for Motion-Compensated Video Coding", in Proceeding 1994 Data Compression Conference, Snowbird, UT, Mar.1994, pp.175-184, IEEE Computer Society Press.
[9] MATLAB Image Processing User , MATHWORK, 2000.
[10] Fundamental of Digital Image Processing, A. K. Jain copyright 1989.
[11] Multimedia Image and Video Processing, Ling Guan, Sun- Yuan Kung, Jan Larsen, CRC Press, copyright 2000.
[12] Neural Networks A comprehensive foundation by Simon Haykin copyright 1999.

[13] MATLAB Neural Network Toolbox User Guide, MATHWORK, 2000.
[14] Luis Torres, Edward J. Delp, "New Trends in Image and Video Compression".

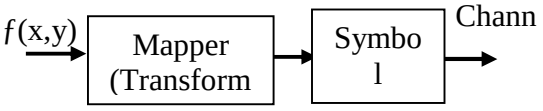



Figure 1 JPEG lossless Encoder

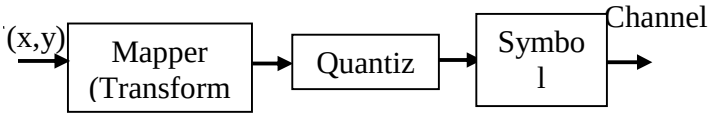



Figure 2 JPEG lossy Encoder

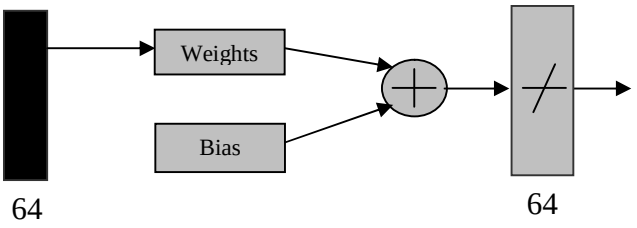



Figure 3 The DCTNN & IDCTNN

Figure 4 The "cameraman" image

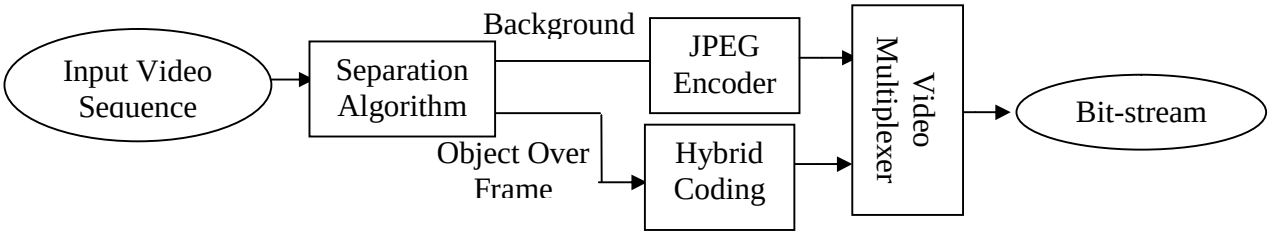



Figure 5 The OBVC Encoder

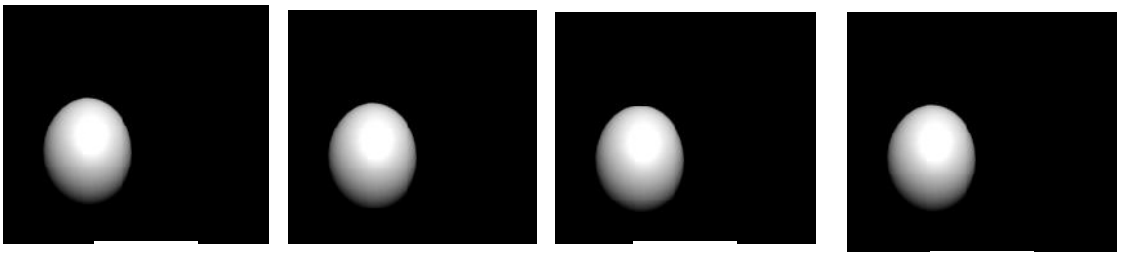
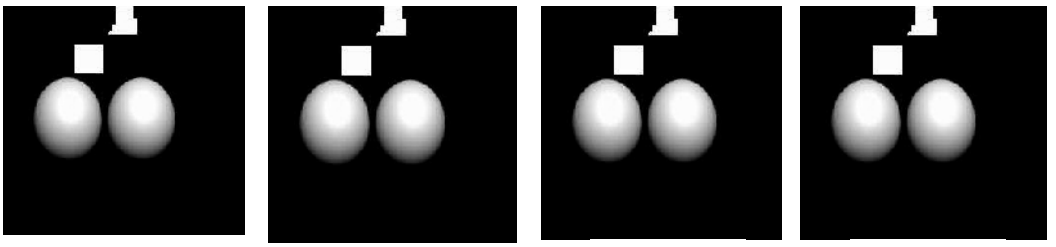



Figure 6 Applying M-JPEG, H-261, OBVC on the "ball" sequence

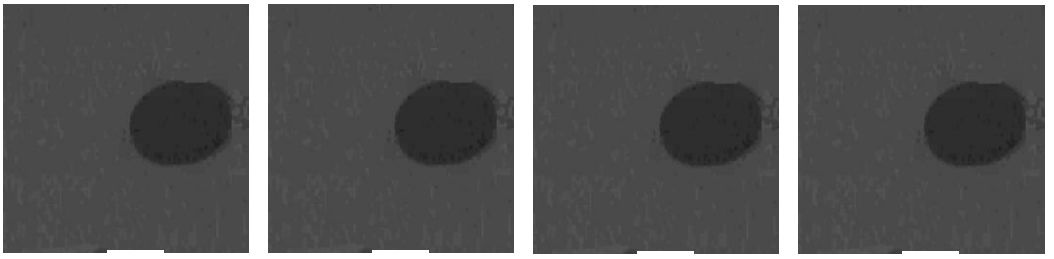
(a) Original "ball" sequence
(b) Reconstructed image after applying the M-JPEG algorithm with CR = 44.44
(c) Reconstructed image after applying the H-261 algorithm with CR = 66.67
(d) Reconstructed image after applying the OBVC algorithm with CR = 240



a b c d

Figure 7 Applying M-JPEG, H-261, OBVC on the "M_ball" sequence

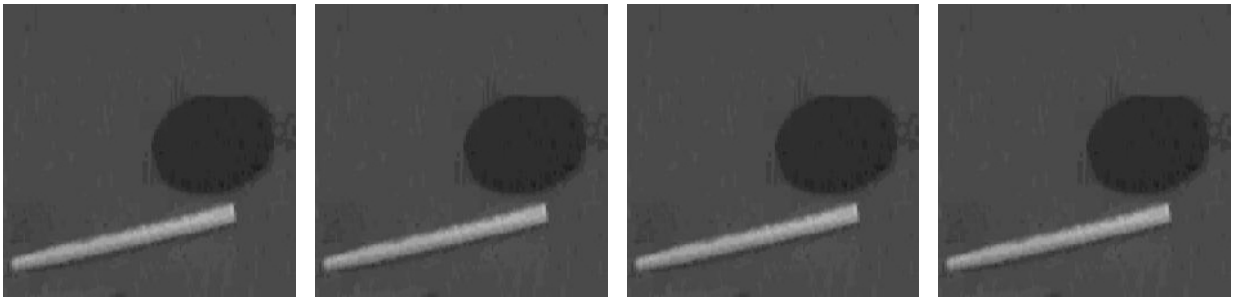
(a) Original "M_ball" sequence
(b) Reconstructed image after applying the M-JPEG algorithm with CR = 25.2
(c) Reconstructed image after applying the H-261 algorithm with CR = 27.82
(d) Reconstructed image after applying the OBVC algorithm with CR = 162.1



a b c d

Figure 8 Applying M-JPEG, H-261, OBVC on the "object" sequence

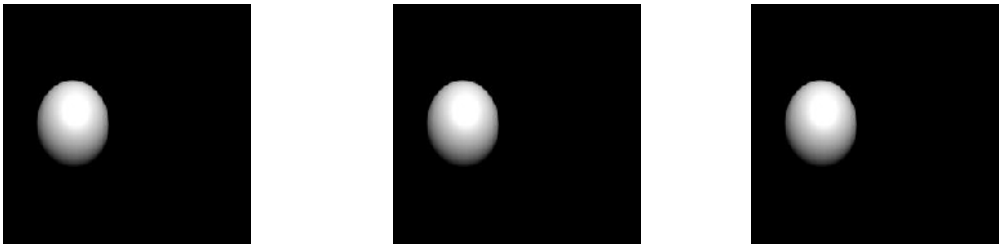
(a) Original "object" sequence
(b) Reconstructed image after applying the M-JPEG algorithm with CR = 35.7
(c) Reconstructed image after applying the H-261 algorithm with CR = 53.72
(d) Reconstructed image after applying the OBVC algorithm with CR = 218.4



a b c d

Figure 9 Applying M-JPEG, H-261, OBVC on the "M_object" sequence

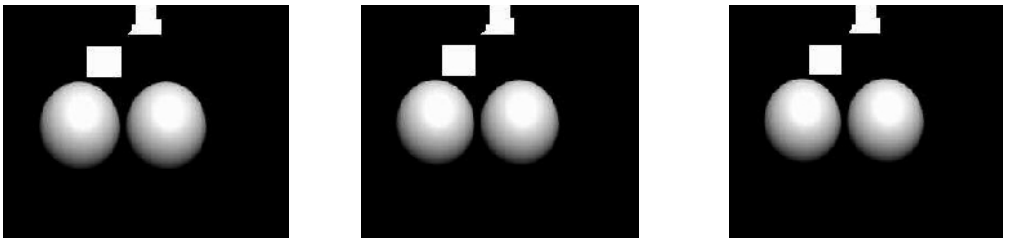
(a) Original "M_object" sequence
(b) Reconstructed image after applying the M-JPEG algorithm with CR = 29.72
(c) Reconstructed image after applying the H-261 algorithm with CR = 31.7
(d) Reconstructed image after applying the OBVC algorithm with CR = 187.2



a b c

Figure 10 Applying OBVC, OBVCNN on the "ball" sequence

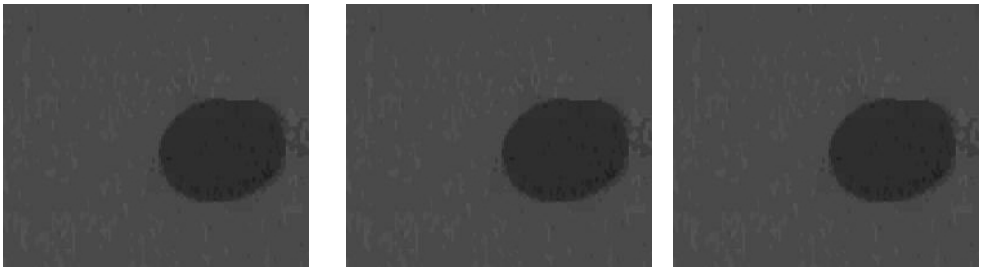
(a) Original "ball" sequence
(b) Reconstructed image after applying the OBVC algorithm with CR = 240
(c) Reconstructed image after applying the OBVCNN algorithm with CR= 240



a b c

Figure 11 Applying OBVC, OBVCNN on the "M_ball" sequence

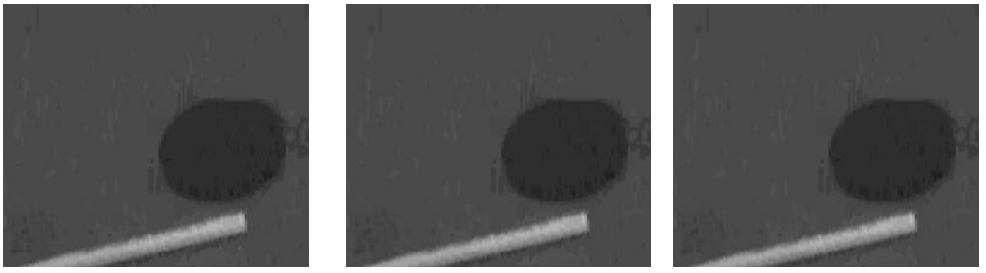
(a) Original "M_ball" sequence
(b) Reconstructed image after applying the OBVC algorithm with CR = 162.1
(c) Reconstructed image after applying the OBVCNN algorithm with CR = 162.1



a b c

Figure 12 Applying OBVC, OBVCNN on the "object" sequence

(a) Original "object" sequence
(b) Reconstructed image after applying the OBVC algorithm with CR = 218.4
(c) Reconstructed image after applying the OBVCNN algorithm with CR = 218.4



a b c

Figure 13 Applying OBVC, OBVCNN on the "M_object" sequence

(a) Original "M_object" sequence
(b) Reconstructed image after applying the OBVC algorithm with CR = 187.4
(c) Reconstructed image after applying the OBVCNN algorithm with CR = 187.4

Table 2 Comparison between standard M-JPEG, standard H-261 and OBVC

| Video file | Frame size | Algorithms Parameter | Standard M-JPEG | Standard H-261 | OBVC |
|---|---|---|---|---|---|
| "ball" sequence | 240x256x20 | Encoding time (sec.) | 36.219 | 8.828 | 49.28 |
| | | Decoding time (sec.) | 66.39 | 10.42 | 3.875 |
| | | MSE | $1.5x10^{-5}$ | $1.5x10^{-5}$ | $1.5x10^{-5}$ |
| | | PSNR | 96.4 | 96.4 | 96.4 |
| | | Compression ratio | 44.44 | 66.67 | 240 |
| "M_ball" sequence | 240x320x20 | Encoding time (sec.) | 37.48 | 31.81 | 78.45 |
| | | Decoding time (sec.) | 87.06 | 28.92 | 4.58 |
| | | MSE | $4.61x10^{-5}$ | $4.61x10^{-5}$ | $4.61x10^{-5}$ |
| | | PSNR | 91.49 | 91.49 | 91.49 |
| | | Compression ratio | 25.2 | 27.82 | 162.1 |
| "object" sequence | 256x256x20 | Encoding time (sec.) | 37.22 | 31.81 | 74.34 |
| | | Decoding time (sec.) | 71.45 | 41.31 | 4.59 |
| | | MSE | $4.1x10^{-5}$ | $4.1x10^{-5}$ | $4.1x10^{-5}$ |
| | | PSNR | 92 | 92 | 92 |
| | | Compression ratio | 35.7 | 53.72 | 218.4 |
| "M_object" sequence | 256x256x20 | Encoding time (sec.) | 37.45 | 101.84 | 108.61 |
| | | Decoding time (sec.) | 71.02 | 59.22 | 5.2 |
| | | MSE | $4.6x10^{-5}$ | $4.6x10^{-5}$ | $4.6x10^{-5}$ |
| | | PSNR | 91.5 | 91.5 | 91.5 |
| | | Compression ratio | 29.72 | 31.7 | 187.2 |

**Table 3 Comparison between OBVC, and OBVCNN**

| Video file | Frame size | Algorithms / Parameter | OBVC | OBVCNN |
|---|---|---|---|---|
| "ball" sequence | 240x256x20 | Encoding time (sec.) | 49.28 | 48.406 |
| | | Decoding time (sec.) | 3.875 | 2.906 |
| | | MSE | $1.5x10^{-5}$ | $1.5x10^{-5}$ |
| | | PSNR | 96.4 | 96.4 |
| | | Compression ratio | 240 | 240 |
| "M_ball" sequence | 240x320x20 | Encoding time (sec.) | 78.45 | 77.438 |
| | | Decoding time (sec.) | 4.58 | 3.75 |
| | | MSE | $4.61x10^{-5}$ | $4.61x10^{-5}$ |
| | | PSNR | 91.49 | 91.49 |
| | | Compression ratio | 162.1 | 162.1 |
| "object" sequence | 256x256x20 | Encoding time (sec.) | 74.34 | 73.078 |
| | | Decoding time (sec.) | 4.59 | 3.593 |
| | | MSE | $4.1x10^{-5}$ | $4.1x10^{-5}$ |
| | | PSNR | 92 | 92 |
| | | Compression ratio | 218.4 | 218.4 |
| "M_object" sequence | 256x256x20 | Encoding time (sec.) | 108.61 | 117.531 |
| | | Decoding time (sec.) | 5.2 | 4.047 |
| | | MSE | $4.6x10^{-5}$ | $4.6x10^{-5}$ |
| | | PSNR | 91.5 | 91.5 |
| | | Compression ratio | 187.2 | 187.2 |