# PERFORMANCE ANALYSIS OF A BLOCK COMPLEXITY BASED DATA EMBEDDING (ABCDE) FOR COLOR IMAGES IN SPATIAL DOMAIN

S. K. GUPTA[1], SUNEETA AGRAWAL[2]
[1]CSED, BIET, JHANSI
[2]CSED, MNNIT, ALLAHABAD
E-mail: guptask_biet@rediffmail.com [1], suneeta@mnnit.ac.in [2]

**ABSTRACT**

The watermarking techniques for grayscale images can be easily extended to handle color images, for this purpose, watermark casting is done by generating three different watermark patterns $\mathbf{S_R}$, $\mathbf{S_G}$, $\mathbf{S_B}$, one for each RGB channel and modifying for each channel, the intensity of the pixels that belong to the corresponding sets $\mathbf{A_R}$, $\mathbf{A_G}$, $\mathbf{A_B}$. The watermark casting and detection procedures for color images are exactly the same as for the corresponding procedures for grayscale images. Instead of marking the three R, G, B components we can choose to mark the luminance and the chrominance components of the image. In this paper A Block Complexity based Data Embedding (ABCDE) watermarking technique, for color images in Spatial Domain, is considered for implementation and the performance analysis is done on the basis of different performance measures for various types and formats of images.

**Keywords:** *Watermarking, Color Images, Casting And Detection, Data Embedding, Performance Analysis*

## 1. WATERMARKING [6,7,10]

Watermarking is the process of embedding data called a watermark (also known as Digital Signature, Tag or Label) into a multimedia object such that watermark can be detected or extracted later to make an assertion about the object. It is simply adding invisible information into a picture so as to "sign" the picture for a more secure copyright protection. The object may be an audio, image or video. Ideally it is impossible to remove the embedded information (owner identification, customer information or other information about the multi-media data) without damaging the data. The idea behind watermarking is to embed information data within the image with an insensible form for human visual system but in a way that protects from attacks. The goal is to produce an image that looks exactly the same to a human eye.

## 2. DIGITAL WATERMARKING[7,8,9]

A digital watermark is a digital signal or pattern inserted into a digital image. Since this signal or pattern is present in each unaltered copy of the original image, the digital water-mark may also serve as a digital signature for the copies. A givenwatermark may be unique to each copy or be common to multiple copies. The watermarking of the document involves the transformation of the original into another form.

Digital watermarking is similar to watermarking of physical objects except that the watermarking technique is used for digital content instead of physical objects. In digital watermarking a low energy signal is imperceptibly embedded in another signal. This low energy signal is called ***watermark*** and it depicts some metadata, like security or right information about the main signal. The main signal in which the watermark is embedded is referred to as ***cover signal,*** since it covers the watermark. The cover signal is generally a still image, audio clip, video sequence of a text document in digital format.

## 3. TYPES OF WATERMARKS[7,9]

Watermarking is used to add a key in the multimedia data that authenticates the legal copyright holder and that cannot be manipulated or

removed without impairing the data in a way that removes any commercial value.

Watermarking technique involves visible watermark and invisible watermark. The first one is used to mark, in a clearly detectable way. On the other hand, invisible watermark are used for author identification. Both can also be used in unauthorized images copies detection either to prove ownership or to identify a customer.

### 3.1. Visible Watermarks[9]

A visible watermark is a visible translucent image which is overlaid on the primary image. Perhaps consisting of the logo or seal of the organization which holds the rights to the primary image, it allows the primary image to be viewed, but still marks it clearly as the property of the owning organization.

It is important to overlay the watermark in a way which makes it difficult to remove, if the goal of indicating property rights is to be achieved.

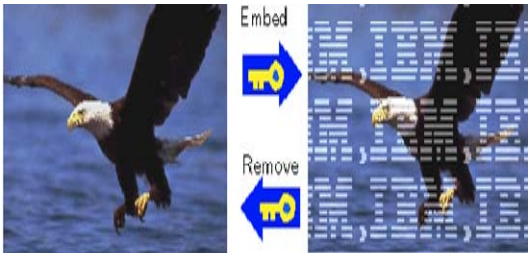


*Figure 1. Visible Watermark*

### 3.2. Invisible Watermarks[9]

An invisible watermark is an overlaid image which *cannot* be seen, but which ***can be detected algorithmically***. Different applications of this technology call for two very different types of invisible watermarks:

A watermark which is destroyed when the image is manipulated digitally in any way, may be useful in proving authenticity of an image. If the watermark is still intact, then the image has not been "doctored." If the watermark has been destroyed, then the image has been tampered with. Such a technology might be important, for example, in admitting digital images as evidence in court.

An invisible watermark which is very resistant to destruction under any image manipulation might be ***useful in verifying ownership*** of an image suspected of misappropriation. Digital detection of the watermark would indicate the source of the image.

Two measurements were taken, embedded capacity and PSNR.

embedded capacity = (# embedded bits) / (image size)

$$MSE = \frac{\sum [f(i,j) - F(i,j)]^2}{N^2}$$

$$PSNR = 20 \log_{10} \left( \frac{255}{RMSE} \right)$$

## 4. LIST OF ABBREVIATIONS

PSNR : Peak-Signal-to-Noise-Ratio
MSE : Mean-Square-Error
NC : Normalized Correlation
BER : Bit-Error-Rate
JPG : Joint Picture Group, extension used for JPEG (Joint Picture Expert Group) Images.
BMP : Bit Map Picture, extension used for BMP images.
GIF : Graphics Image Format, extension used for GIFF (Graphics Image File Format) images.
TIF : Tagged Image Format, extension used for TIFF (Tagged Image File Format) Images.
PNG : Portable Network Graphics, extension used for PNG images.

## 5. BIT PLANE COMPLEXITY SEGMENTATION (BPCS)[11] :

BPCS attempts to address the shortcomings of LSB insertion by searching an image for noise-like regions. These regions are replaced with watermark data, while informative regions are undisturbed. The algorithm works as follows:

1. Convert a $2^m$ x $2^m$ N-bit/pixel image from natural binary into N-bit Gray Code.
2. Decompose the N-bit Gray code into N single-bit planes. Each plane forms a binary image.
3. Divide each bit plane into 8x8 tiles.

4. For each tile, compute the complexity **α**.
5. If **α** is above a threshold, replace the tile with watermark data.
6. The bit-planes are recombined to form an N-bit channel in Gray code.
7. The Gray code is converted back to natural binary.

The complexity metric is defined as:

$$\alpha = k / (2 \times 2^m \times (2^m - 1))$$

where a tile has size $2^m \times 2^m$ and k is the sum of xor-ing adjacent bits in a tile, both in the horizontal and vertical direction. The valid range for **α** is $0 < \alpha < 1$. With this measurement, an all white or all black tile would have an **α** of 0, while a checkerboard pattern would have an **α** of 1.

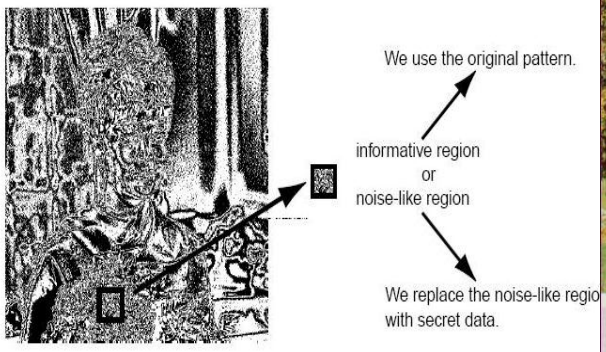


*Figure 2. A bit plane tile being examined[11]*

In the results below, the BPCS algorithm was run with the threshold **α**=0.3. Notice how the sky, car, and grass seem relatively unaltered compared to the original image, while the tree in the foreground contains a relatively large amount of distortion. From a standard viewing distance, however, this distortion does not appear artificial. The tiled nature of the watermark is only evident on close inspection.

In BPCS, an operation called *conjugation* is applied to convert simple blocks into complex ones. In order to extract the resource file embedded in the container image, we should know which blocks of the resource file are conjugated and which are not. This information is stored in a *conjugation map*. It should be embedded along with the resource file as blocks. Note that the blocks of the conjugation map should be also conjugated if they are not complex enough. The above black and white border complexity is a suitable measure for classifying blocks into complex ones and simple ones. This measure is, however, not always applicable. For example, blocks with periodical patterns, like chessboard or stripes are classified as complex ones by this measure. Such blocks are, however, not suitable for embedding data, since replacing them with noisy blocks disorders regular patterns. This measure may also indicate that blocks on the boundary of a noisy region and an informative region are complex. Embedding data in such blocks may bring remarkable changes to a container image.

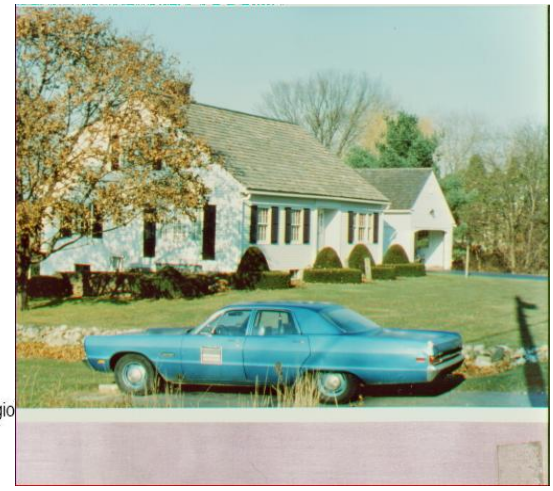

*Figure 3. Original image[from "image tool" of MATLAB*

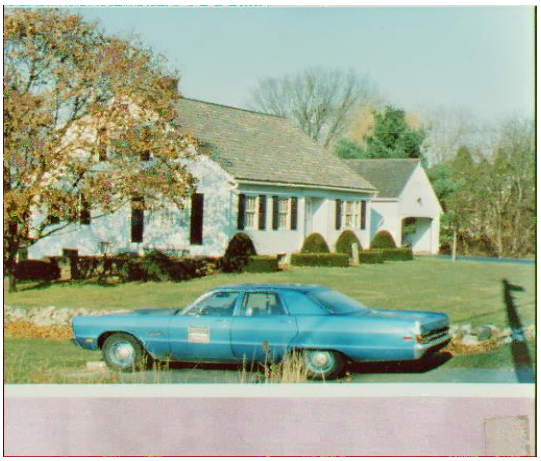

*Figure 4. The Image after applying the BPCS Algorithm[11]*

BPCS with α=0.3, Capacity: 45.0%, PSNR: 27.11

## 6. A BLOCK COMPLEXITY BASED DATA EMBEDDING (ABCDE)[11] :

ABCDE works in a very similar method as BPCS, but employs a more sophisticated complexity metric. BPCS merely counts the number of bit-flips in each row and column of a given tile, with the assumption that a large number of bit flips indicate a lack of structure. ABCDE instead uses a pair of metrics that look for *irregularity* in a tile rather than high spatial frequencies. The justification is that certain patterns, while high frequency, would produce unacceptable distortion if a watermark were embedded in them.

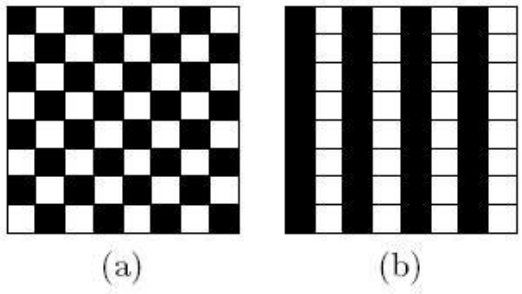


*Figure 5. The Patterns that are high frequency but Regular[11]*

ABCDE replaces **α**-complexity of BPCS with two metrics: "run length irregularity" and "border noisiness". The run length irregularity searches for patterns amongst a tile's rows and columns. The checkerboard pattern would be rejected by this metric. Border noisiness attempts to find edges in an image between two relatively flat regions of color. Rejecting tiles that lie on boundaries avoids the apparent blurring of edges that can occur with BPCS. The new data embedding method ABCDE uses two new complexity measures to properly discriminate complex blocks from simple ones. One is computed from the distributions of black and white pixels in rows and columns of a block. It is called the *run-length irregularity*. The other is computed from the distributions of black-and-white borders between adjacent rows and columns. It is called the *border noisiness*. These measures are used at the same time.

The run length irregularity is introduced to evaluate the non-uniformity of black-and-white patterns within a block. This measure can prevent us from selecting blocks of periodical patterns for embedding. The border noisiness is additionally introduced to check if many black-and-white borders can be found within a block and if they are well-distributed over the block at the same time.

We can avoid using blocks not entirely in noisy region with this measure. A block is regarded as complex if its run-length irregularity and border noisiness are larger than threshold values given for these complexity measures. The threshold values can be specified separately for each bit-plane. This enables us to make the embedding capacity large while keep image quality high. The defaults for these threshold values are provided for convenience. In ABCDE, data embedding is performed in the same way as in BPCS. A resource file is regarded as a stream of binary blocks. All the binary blocks are converted into complex ones. They are then replaced with complex blocks in bit-planes of a container image. An M- sequence is employed for converting the blocks of the resource file into complex ones. Although, the block stream of the resource file may contain both complex blocks and simple ones, all of them are uniformly converted. As a result, we are not required to embed information like the conjugation map. This makes the embedding algorithm simple. At the same time, overheads required for embedding in ABCDE are smaller than those required in BPCS.

We can specify an M-Sequence as a parameter. We can either embed the M-sequence parameter and the threshold value parameters for complexity measures with a resource file or keep them outside. When these parameters are not embedded, we should specify them correctly on extraction to retrieve the resource file embedded in a container image. It is almost impossible to extract the embedded resource file without knowing them. This *secure embedding mode* is useful for steganographic applications. On the other hand, when these parameters are embedded, we do not have to know any parameters except the block size.

If we use a default block size in common, anyone can extract the embedded resource file. This *self-contained embedding mode* is useful for data management applications ABCDE can be successfully applied to various images. We can expect that near 50% pixel data of a container image can be used for embedding without degrading its quality.

The results shown in the images below are particularly impressive. Despite the watermarked image having more than 60% of its image data overwritten, the differences between it and the original image are nearly imperceptible.

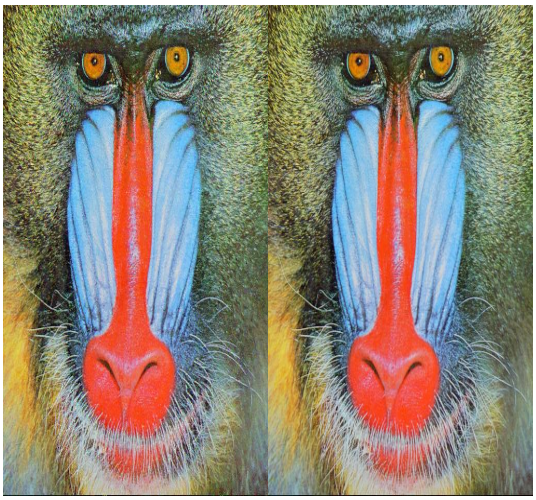

*Figure 6. Original image[8,10] Figure 7. The Image after applying the ABCDE* Algorithm[11]

## 7. EXPERIMENTAL RESULTS

*Figure 8. For lena image[from "image tool" of MATLAB 7.0]*

*Table 1. For lena image*

| Image name | MSE | PSNR |
|---|---|---|
| Lena_0.jpg | 0.0781 | 25.9289 |
| Lena_0.bmp | 0.0781 | 25.9421 |
| Lena_0.png | 0.0786 | 25.9046 |

*Figure 9. For finger image[from "image tool" of MATLAB 7.0]*

*Table 2. For finger image*

| Image name | MSE | PSNR |
|---|---|---|
| Finger_jpg.jpg | 0.0814 | 25.7511 |
| Finger_bmp.bmp | 0.0792 | 25.8693 |
| Finger_png.png | 0.0780 | 25.9322 |

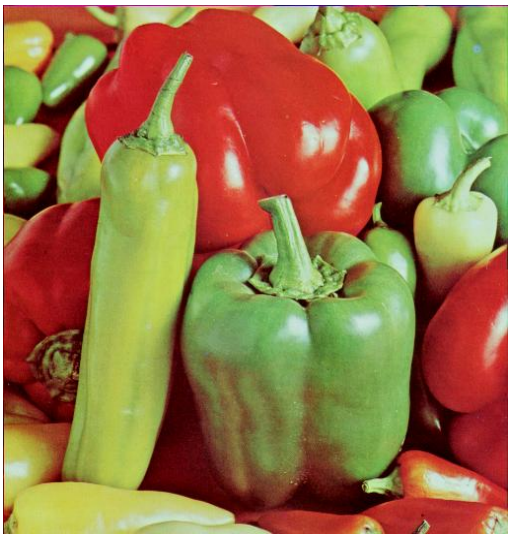

*Figure 10. For fruits image[from "image tool" of MATLAB 7.0]*

*Table: 3. For fruits image*

| Image name | MSE | PSNR |
|---|---|---|
| Fruits.jpg | 0.0701 | 26.3985 |
| Fruits.bmp | 0.0695 | 26.4375 |
| Fruits.png | 0.0706 | 26.3663 |

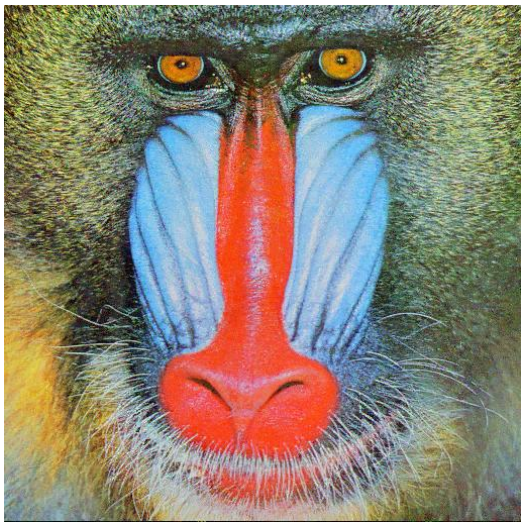

***Figure 11. For mandrill image****[from "image tool" of MATLAB 7.0]*

*Table 4. For mandrill images*

| Image name | MSE | PSNR |
|---|---|---|
| Mandrill.jpg | 0.1144 | 24.2714 |
| Mandrill.bmp | 0.1153 | 24.2385 |
| Mandrill.png | 0.1149 | 24.2534 |

*Figure 12. For cameraman image[from "image tool" of MATLAB 7.0]*

*Table 5. For cameraman image*

| Image name | MSE | PSNR |
|---|---|---|
| Cameraman.jpg | 0.0717 | 26.3025 |
| Cameraman.bmp | 0.0729 | 26.2303 |
| Cameraman.png | 0.0733 | 26.2294 |

## 8. CONCLUSIONS

By analyzing the data of tables (Table: 1 to Table: 5), we can see that the **A Block Complexity based Data Embedding (ABCDE)** provides the best results for **Face Image (Lena_0.bmp)** for image format **"bmp"**, the best results for **fingerprint image (finger_png.png)** for image format **"png"**, the best results for **fruits image (fruits.bmp)** for image format **"bmp"**, the best results for **lion image (mandrill.jpg)** for image format **"jpg",** the best results for **cameraman image (cameraman.jpg)** for image format **"jpg"**, as the PSNR value is largest.

## REFERENCES :


[1]. Rafael C. Gonzalez and Richard E. Woods, "**Digital Image Processing**", Third Edition, Pearson, Prentice Hall Publication.

[2]. C. Rey and J. L. Dugelay, "**A survey of watermarking algorithms for image authentication**", EURASIP Journal on Applied Signal Processing 6, 2002, PP. 613–621.

[3]. W. Bender, D. Gruhl, N. Morimoto and A. Lu, "**Techniques for Data Hiding**", IBM System Jurnal, Vol. 35, Nos 3 & 4, 1996, pp. 313-336.

[4]. Onur Mutlu, "**An Overview of Image Watermarking algorithms**", Project Report EE 371R Digital image Processing, December 6, 2001.

[5]. Jonathan Cummins, Patrick Diskin, Samuel Lau and Robert Parlett, School of Computer Science, "**Steganography and Digital watermarking**", The University of Birmingham, 2004.

[6]. Prof Rudko, "**Hidden Bits: A Survey of Techniques for Digital Watermarking**", Chris Shoemaker Independent Study EER-290, Spring 2002.

[7]. A White Paper on, "**Digital Watermarking: A Technology Overview**", by Wipro Technology, 2001.

[8]. Allan M. Bruce, "**A Review of Digital Watermarking**", Department of Engineering, University of Aberdeen, 2001.

[9]. Alok K. Agarwal, University at Buffalo, "**Digital Watermarking**", ACM, 2001.

[10]. George Voyatzis and Ioannis Pitas, "**Protecting Digital-Image Copyrights: A Framework**", IEEE Computer Graphics and Applications 1999, PP. 18-24.



[11]. Itai Katz, "PSYCH221 : "**A survey of digital watermarking techniques**", PSYCH221 Projects, 2006.

[12]. J. F. Delaigle, C. De Vleeschouwer and B. Macq, "**Watermarking Algorithm Based on a Human Visual Model**", Signal Processing, Vol. 66, No. 3, 1998, PP. 337-355.