IMAGE COMPRESSION BY SUBSAMPLING AND RECONSTRUCTION FROM COMPRESSED IMAGE USING MORPHOLOGICAL FILTERS

Lakshmi Vydehi. J
Lecturer, Department of Computer Science, Bhavans Vivekananda College of Science Humanities and Commerce, Sainikpuri, Secunderabad-500094, Telangana State, India
E-mail: jlvydehi.bhavans@gmail.com, lakshmi.cs@bhavanavc.ac.in

Three Star Paper

Abstract

All image compression algorithms found in standard literature invariably alter various features and statistical parameters when applied to a digital image and change the originality of the image. Alternatively, this paper proposes a novel technique of subsampling a given digital image of size W×H, by forcing alternative columns and rows of the image and compressing it to 25% of the size of the original image by losing 75% of the pixel values. In such a case, the actual image values are kept intact, but the resolution of the image is reduced to W/2×H/2 from W×H. The compressed image of size W/2×H/2 could be further compressed by subsampling it and compressing it to 6.25% of the original image. Thus, the resolution is reduced to W/4×H/4 from W×H. Repeated subsampling could be carried out on a given image. The compressed image of size W/4×H/4 could be further compressed by subsampling it and compressing it to 1.5625 % of the original image, The resolution is now reduced to W/8 ×H/8 from W×H. In this case, 98.4375 % of the pixel information of the original image is lost, but 1.5625 % of the pixel values of the original image are kept intact. Image or video compression is an essential requirement in increasing the throughput rate in a digital communication system. The technique proposed in this paper has been found to be very useful for this purpose. After receiving the compressed image through a communication medium, one would like to uncompress it for regular use. This paper advocates 'Morphological Filters' for reconstructing the original image from its compressed version. It has been found that there is not much of a reduction in the visual quality of the reconstructed image when compared to the original image.

Keywords: Image Compression, Spatial Sparsing, Morphological Filters

1. Introduction

Wired or wireless transmission of uncompressed image data would take considerable amount of time and that is why image data is compressed before transmission and decompressed after reception. Most of the image data compression techniques, both lossless and lossy ones, work on redundant symbol removal. In order to remove redundancy, it is essential to reduce the entropy contained in the given data and this amounts to information loss. Hence a trade-off is made between the compression ratio and information loss. Redundant symbol removal is otherwise also known as 'sparsing'. Image data usually consists of (i) significant information and (ii) insignificant information. In addition to redundant data removal, sometimes insignificant information can also be ignored. For example, foreground information sometimes can be viewed as significant and background information as insignificant. In other words, one can retain foreground entropy in an image and reduce background entropy so that the image is further compressed for transmission and decompressed after reception without causing any loss in the significant information. The basic compression schemes under the lossless compression technique are: Huffman Coding, Arithmetic Coding, Dictionary Techniques, and Predictive Coding. There are five types of basic lossy compression schemes and these are listed here: Scalar and Vector Quantization, Differential Encoding, Transform Coding, Sub-band Coding, and Wavelet-Based Compression.

In this paper, a novel technique of 'Spatial Sparsing' is presented. Given an image, one can do spatial sparsing and subsampling of a given image as shown below. Consider a lattice of cells of size 36 · 14 as shown in Fig. 1. We shall make use of the algorithm given below to carry out subsampling of the lattice of cells.

Algorithm for rectangular sub sampling

$$\text{pixel\_val\_rect}(i, j) = \text{pixel\_val}(2i, 2j)\ ;\ \text{if } i \text{ is even}$$

$$= \text{pixel\_val}(2i, 2j)\ ;\ \text{if } i \text{ is odd}$$

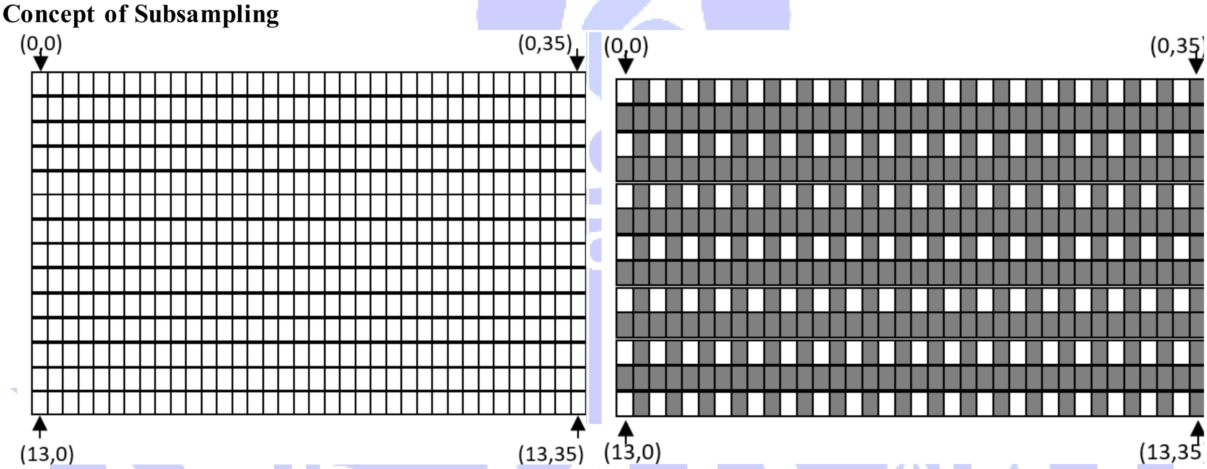
Sub-Sampling eliminates the alternate rows and columns from the given array of pixels of the image.

In general, one can consider a lattice of cells of size $2n \cdot 2n$, where 'n' is any integer. Two cases are considered here (i) the size of the image (lattice of cells) is a square lattice, i.e., both rows and columns are equal, (ii) the size of the image (lattice of cells) is a rectangular lattice, i.e., both rows and columns are not equal. Let us evaluate the compression ratio for both the cases.

Case #1: Assume width = 2n and height = 2n. Then number of cells in the lattice is $(2n)^2 = 4n^2$. After subsampling the image of size $2n\times2n$ turns out to be of size $n\times n$ and the number of cells (pixels) in the subsampled image is n^2. The ratio of the number of pixels in the subsampled image to the number of pixels in the original image is $n^2/4n^2 = 0.25$. The compression ratio is 25% of the original image.

Case #2: Assume width = $2n_1$ and height = $2n_2$. The number of pixels in the original image is $2n_1*2n_2.=4n_1*n_2$. The subsampled image would then be of size $(2n_1/2)*(2n_2/2) = n_1*n_2$. The ratio of the number of pixels in the subsampled image to the number of pixels in the original image is $n_1*n_2 / 4n_1*n_2=0.25$. The compression ratio is 25% of the original image.

Concept of Subsampling

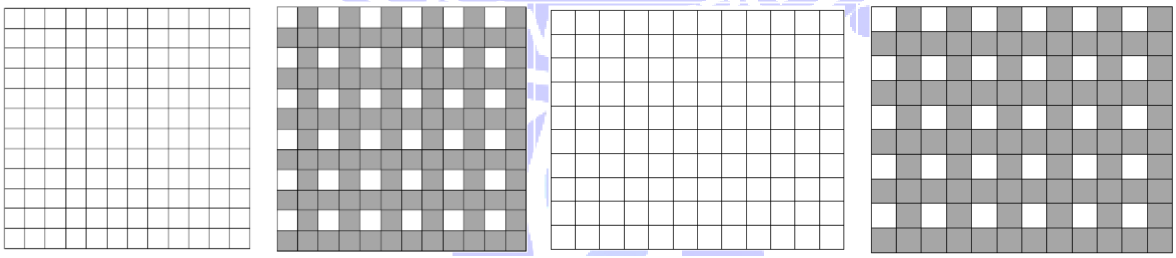


(a): Regular rectangular lattice (b): Subsampled rectangular lattice

Fig. 1: Concept of a lattice of size 14×36 and its subsampled version of size 18×7

Legend: Values of shaded cells are forced to '0'.

Example: Consider an image of size 12×12, which is shown in Fig. 2(a). Fig. 2(b) shows the subsampled version of size 6×6. Now the compression ratio is evaluated as (36/144)*100 = 25%. Fig. 2(c) shows image of size 12×10 and Fig. 2(d) the subsampled version of size 6×5. In this case, the compression ratio is evaluated as (30/120)*100 = 25%.



Width×Height =12×12 (a) | Width×Height = 6×6 (b) | Width×Height = 12×10 (c) | Width×Height = 6×5 (d)

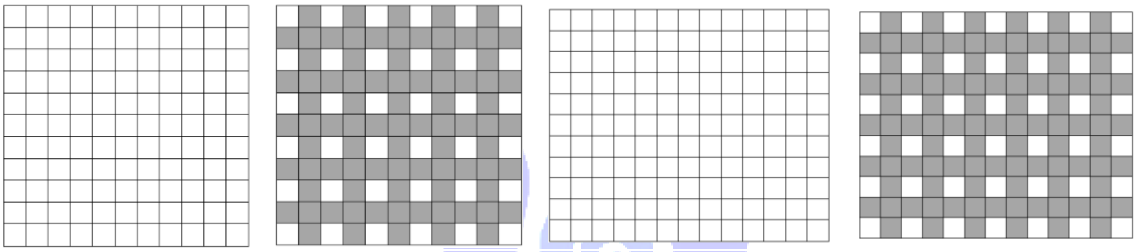
Fig. 2: Lattices of sizes 12×12 and 12×10 and their subsampled lattices of sizes 6×6 and 6×5

Case #3: Assume width = 2n+1 and height = 2n+1. Then number of cells in the lattice is $(2n+1)^2$. After subsampling the image of size $(2n+1)\times(2n+1)$ turns out to be of size $[(2n+1)/2]\times[(2n+1)/2]$ and the number of cells (pixels) in the subsampled image is $[(2n+1)/2]*[(2n+1)/2]$. The ratio of the number of pixels in the subsampled image to the number of pixels in the original image is $[(2n+1)/2]*[(2n+1)/2] / (2n+1)^2 = (2n+1)*(2n+1) / 4(2n+1)^2 = 1/4 = 0.25$. This amounts to saying that the compression ratio is 25% of the original image.

Case #4: Assume width = $2n_1+1$ and height = $2n_2+1$. Then number of cells in the lattice is $(2n_1+1)*(2n_2+1)$. After subsampling the image of size $[(2n_1+1)]\times[(2n_2+1)]$ turns out to be of size $[(2n_1+1)/2]\times[(2n_2+1)/2]$ and the number of cells (pixels) in the subsampled image is $[(2n_1+1)/2] * [(2n_2+1)/2]$. The ratio of the number of pixels in the subsampled image to the number of pixels in the original image is $[(2n_1+1)/2] * [(2n_2+1)/2] / (2n_1+1)*(2n_2+1) =$

$(2n_1+1)] * [(2n_2+1) / 4*(2n_1+1)*(2n_2+1) = 1/4 = 0.25$. This amounts to saying that the compression ratio is 25% of the original image.

Example: Consider an image of size 11×11, which is shown in Fig. 3(a). Fig. 3(b) shows the subsampled version of size 6×6. Now the compression ratio is evaluated as (36/121)*100 = 0.297 * 100 = 29.7%. Fig. 3(c) shows image of size 13×11 and Fig. 3(d) the subsampled version of size 7×6. Now, the compression ratio is evaluated as = (42/143) * 100 = 0.293 * 100 = 29.3%.



Width×Height =11×11 (a) | Width×Height = 6×6 (b) | Width×Height = 13×11 (c) | Width×Height = 7×6 (d)

Fig. 3: Lattices of sizes 12×12 and 12×10 and their subsampled lattices of sizes 6×6 and 6×5

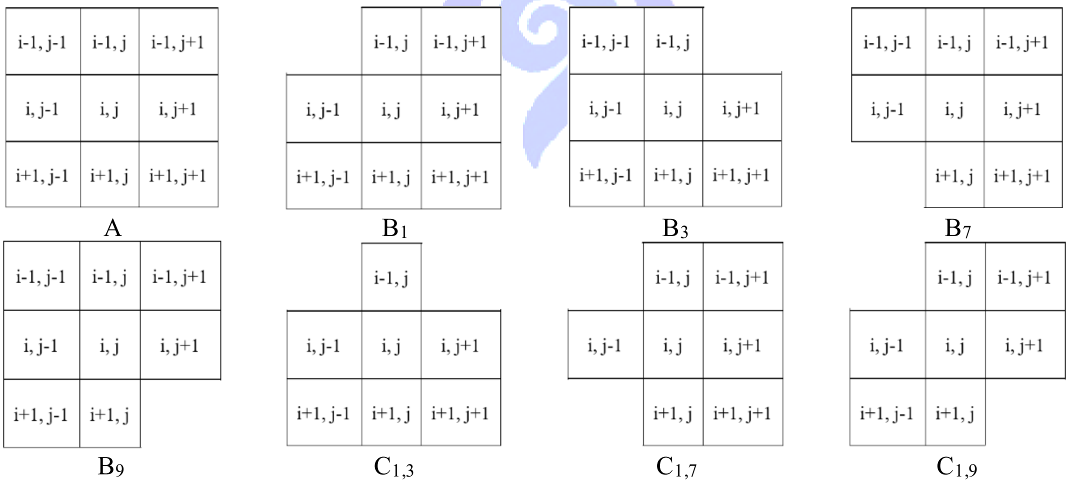
Theoretically, the compression ratio of any given digital image, irrespective of its number of columns and rows, turns out to be 25% of the original image. One gets 25% compression ratio when the width and height of the image are even. On the other hand, one gets 29.7% compression ratio when both width and height of a given image are odd, and 29.3% compression ratio when the width or height is even and the other is odd. This variation from the theoretical calculations is due to the fact that the algorithm for subsampling rounds off the fraction to its next integer. For example, in the case of a lattice of size 11×11, the value of 11/2 = 5.5 is rounded off to 6 and hence the subsampled version is seen of size 6×6 instead of 5.5×5.5. In any case, subsampling of a digital image could be assumed to yield 25% compression ratio with a 75% loss of pixels in a given digital image.

Once a given image is subsampled, one needs to reconstruct the original image from its subsampled and compressed version by predicting the 75% pixel loss in the original image. The first step to do this is to introduce 0's at every adjacent column and row of the compressed image so that the size of the compressed image gets doubled in size. This process of inserting 0's at every column and row of the compressed image is called 'Zero-Dilution'. The zero-diluted version is actually the subsampled version of the original image before compression. The question that arises here is how to estimate or predict the pixel values of original image. The process of predicting these lost pixel values is called 'Interpolant Prediction'. There are two types of interpolant prediction techniques, (i) interpolant prediction using neighborhood pixel averaging and (ii) Interpolant prediction using extended morphological filtering. These techniques are described briefly in the next section.

2. Interpolants Prediction

2.1 Interpolants prediction using neighborhood pixel value averaging

This is a computationally intensive process for interpolant prediction. Fig. 4 shows sixteen basic scanning windows of size 3×3.



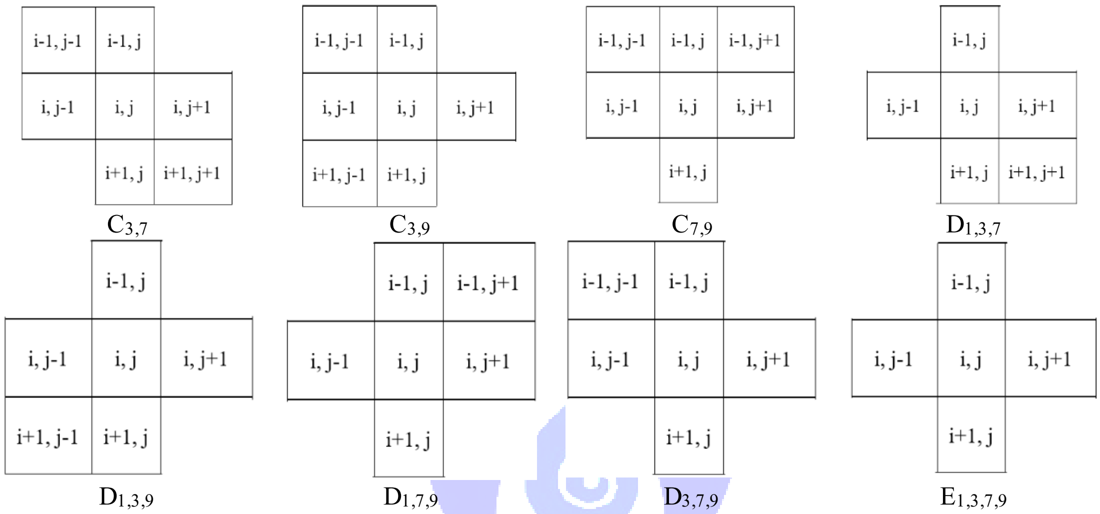
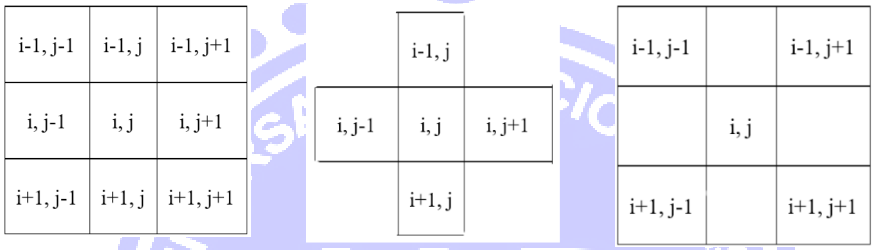


Fig. 4: Sixteen scanning windows of size 3×3

Fig. 5 shows three windows that are normally used for scanning images.



9-neighborhood window | 5-neighborhood window 1 | 5-neighborhood window 2

Fig. 5: Certain scanning windows used in image processing

Algorithm for Pixel Averaging

Scan the subsampled image with the 9-neighborhood window shown in Fig. 5. At every position, the interpolant is evaluated as the average of all available non-zero values using the prediction formula:

$$\frac{x(i-1,\ j-1)+x(i-1,\ j)+x(i-1,\ j+1)+x(i,\ j-1)+x(i,\ j+1)+ x(i+1,\ j-1)+ x(i+1,\ j)+ x(i+1,\ j+1)}{n}$$

where n is number of non-zero pixel values. The algorithm is applied to whole image. The basic idea behind this algorithm is that an unknown pixel value is evaluated as an average of surrounding non-zero pixels. Fig. 6 shows a child's image, its subsampled and pixel averaged versions. There is hardly any change between original child image and its reconstructed version.

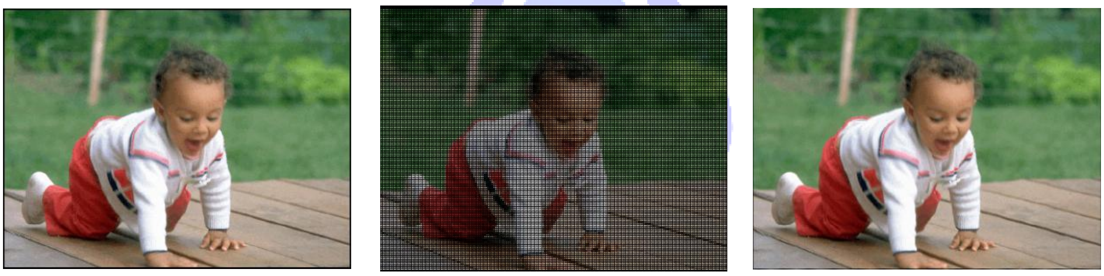


Image of a child (Size = 384×288) | Subsampled version (Size = 384×288) | Reconstructed image using pixel averaging (Size = 384×288)

Fig. 6: Sample image, its subsampled and pixel averaged versions

Fig. 7 shows the sample image, its subsampled, pixel averaged version and error image along with respective histograms.

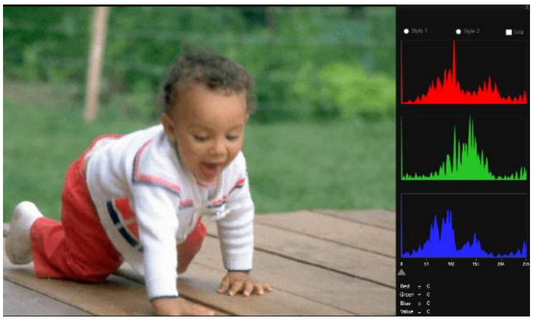
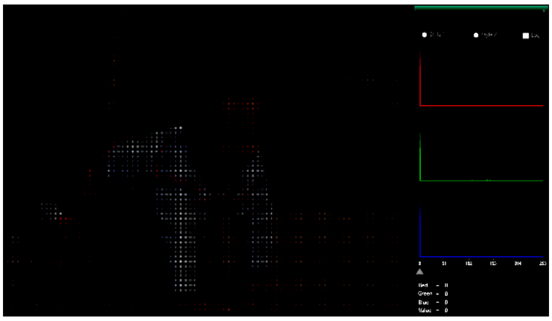
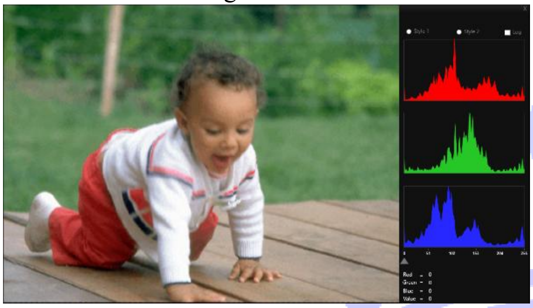


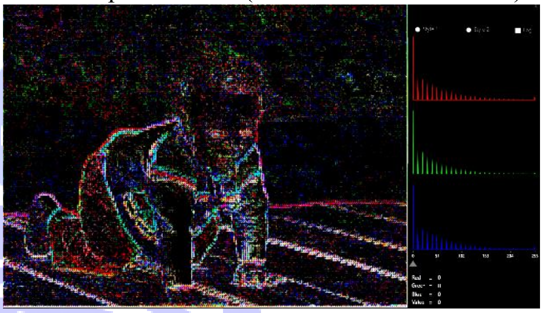
Image of a child



Subsampled version (75% loss of information)



Reconstructed using pixel averaging



Error between original and reconstructed image (Pixel Averaging)

Fig. 7: Sample image, its subsampled, pixel averaged version and error image

The original image of size 384×288 having 110,592 pixels is subsampled due to which, there is a loss of 82,944 [i.e., (384×288-(192×144)] original pixels. This amounts to saying that 75% of original pixels are lost by subsampling. These original pixels have been recovered from the subsampled image using pixel averaging formula.

2.2 Interpolants prediction using extended morphological filtering

Morphological filtering makes use of two fundamental operations of 'dilation' and 'erosion', which are defined as follows.

Dilation: Let image to be dilated be A and the structuring element that dilates A be B. Then the dilation of A by B is defined as the Minkowski addition $D(A,B) = A\oplus B = EXTSUP_{(x,y)\in D_B}[A_{x,y} + B(x,y)]$, where D_B is the domain of the image B, and EXTSUP is an operation of supremum over the union of the domains.

Erosion: Let image to be eroded be A and the structuring element that erodes A be B. Then the erosion of A by B is defined as the Minkowski subtraction $A\theta B = INF_{(x,y)\in D_B}[A_{x,y} + B(x,y)]$ as $E(A,B) = INF_{(x,y)\in D_B}[A_{-x,-y} - B(x,y)]$, where D_B is the domain of the image B, and INF is an operation of infimum over the intersection of the domains.

Fig. 8 shows a sample image and its dilated and eroded versions.

Sample image

Image dilated by A

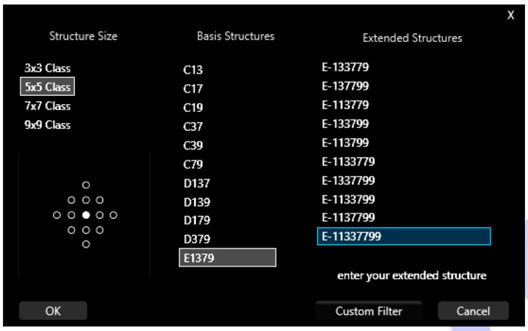
Image eroded by A

Fig. 8: Sample test image and its dilated and eroded versions

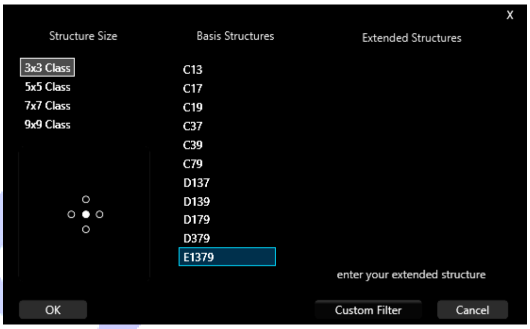
Nine neighborhood window shown in Fig. 6 is used to dilate and erode the image. Following these definitions, morphological filtering operations of ***Closing*** and ***Opening*** are defined in the following manner. Closing of A by B is represented as AoB and defined as $AoB = (A\oplus B)\theta B$. Opening of A by B is represented as A·B and defined as $A{\cdot}B = (A\theta B)\oplus B$. Same structuring element should be used for dilation and erosion in any morphological filtering. Closing and opening are idempotent operators. That is, $(AoB)o(AoB) = AoB$ and $(A{\cdot}B){\cdot}(A{\cdot}B) = A{\cdot}B$

Extended morphological filters

These are essentially morphological filters except that the structuring element need not remain the same for dilation and erosion in any morphological filtering. With reference to Fig. 9, for example, one can use the structuring element $E_{11337799}$ for dilation and subsequently E_{1379} for erosion, so that extended closing operation is carried out on a given image. The structuring elements of $E_{11337799}$ and E_{1379} are shown in the respective dialog boxes.



Neighborhood structure $E_{11337799}$



Neighborhood structure E_{1379}

Fig. 9: Structuring elements used for extended morphological filtering

Extended morphological filtering is a computationally less intensive process for interpolant prediction. At every position, the interpolant is evaluated using the procedure given below. Scan the given subsampled image and dilate it with $E_{11337799}$ shown in Fig. 9. Subsequently, erode with E_{1379} shown in Fig. 9. The resulting image is extended morphological filtered version, which is also the interpolant predicted version.

Fig. 10 shows the sample image, its subsampled, morphological filtered version and error image along with respective histograms.

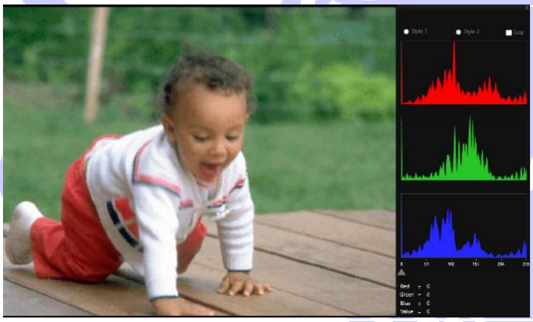
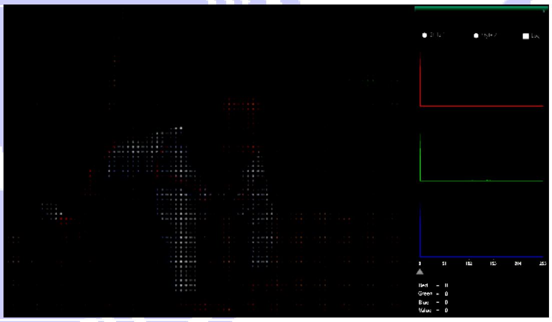
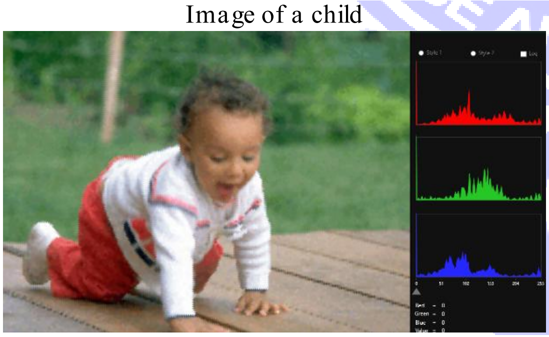


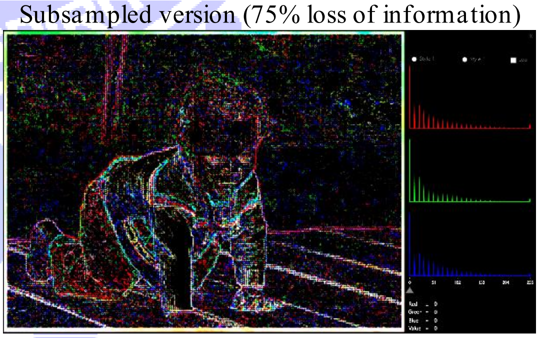
Image of a child



Subsampled version (75% loss of information)



Reconstructed using morphological filtering



Error between original and reconstructed image (Morphological Filtering)

Fig. 10: Sample image, subsampled, morphological filtered, and error image

Having thus seen the effects of prediction algorithms on subsampled version of a sample image, a real time test case image is analyzed in what follows.

3. Case Study

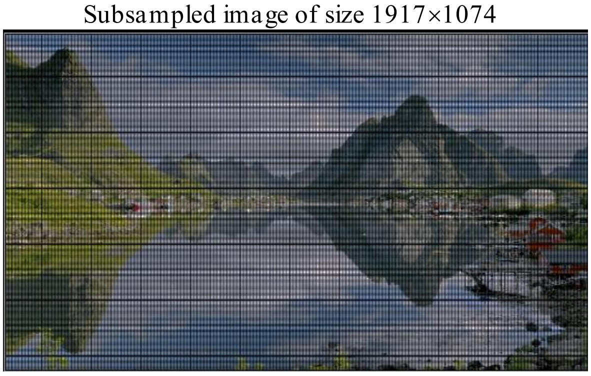
A test image of size 1917×1074 is shown here in Fig. 11 to test the algorithms for subsampling and image reconstruction using pixel averaging and morphological filtering operations.

Fig. 11: A test image of size 1917×1074

Original image of size 1917×1074

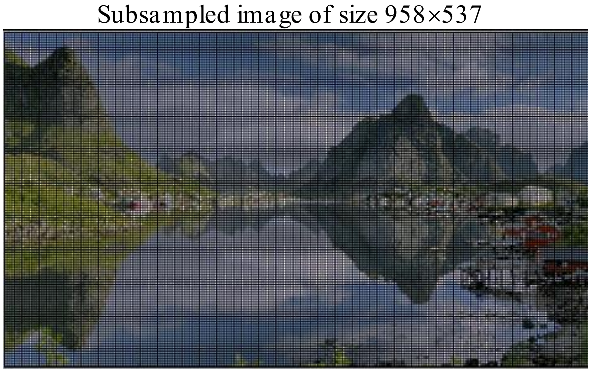
Subsampled image of size 1917×1074

25% Compressed image of size 958×537



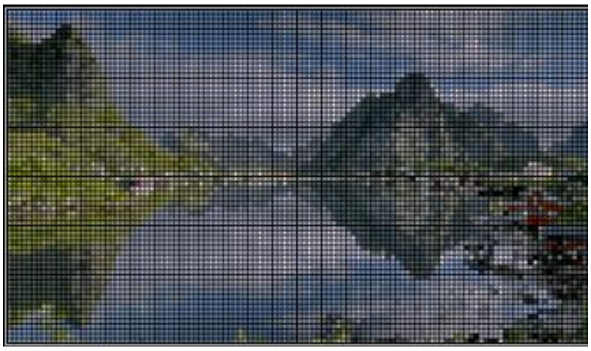
Subsampled image of size 958×537

6.25% Compressed image of size 479×269



Subsampled image of size 479×269

1.5625% Compressed image of size 240×135



Subsampled image of size 240×135

Fig. 12: A test image of size 1917×1074 and down subsampled images

Original image of size 1917×1074

Reconstructed image from
25% Compressed image of size 958×537

Reconstructed image from
6.25% Compressed image of size 479×269

Reconstructed image from
1.5625% Compressed image of size 240×135

Fig. 13: Test image of size 1917×1074 and its reconstructed versions from down subsampled images

With reference to Fig. 12 and Fig. 13, one may observe that visual quality of the reconstructed images deteriorates when more and more compressions are carried out. In any case, the information content is perceivable to a large extent.

3. Conclusions

Results of a systematic study on image compression using spatial sparsing are presented in this paper. As future perspective, one can try out the possibility of improving visual quality even after compressing a digital image up to 1.5625% and reconstructing the original image using Artificial Intelligence based algorithms.

One can also explore the possibility of using the spatial sparsing technique for compressing digital video frames for increasing the throughput rate during digital communication.

Acknowledgement

The author expresses her thanks to the members of the Institute of Ancient Philosophy, Hyderabad, India for encouraging me to prepare this paper and present during the International Conference on Science and Spirituality for Global Peace and Harmony.