Evaluation of JPEG-LS, the New Lossless and Controlled-Lossy Still Image Compression Standard, for Compression of High-Resolution Elevation Data

Shantanu D. Rane and Guillermo Sapiro, *Member, IEEE*

***Abstract*—The compression of elevation data is studied in this paper. The performance of JPEG-LS, the new international ISO/ITU standard for lossless and near-lossless (controlled-lossy) still-image compression, is investigated both for data from the USGS digital elevation model (DEM) database and the navy-provided digital terrain model (DTM) data. Using JPEG-LS has the advantage of working with a standard algorithm. Moreover, in contrast with algorithms like the popular JPEG-lossy standard, this algorithm permits the completely lossless compression of the data as well as a controlled lossy mode where a sharp upper bound on the elevation error is selected by the user. All these are achieved at a very low computational complexity. In addition to these algorithmic advantages, we show that JPEG-LS achieves significantly better compression results than those obtained with other (nonstandard) algorithms previously investigated for the compression of elevation data. The results here reported suggest that JPEG-LS can immediately be adopted for the compression of elevation data for a number of applications.**

***Index Terms*—Compression, elevation data, JPEG-LS, standard.**

I. Introduction

THE COMPRESSION of elevation data is fundamental for a number of applications including storage, transmission, and real-time visualization in navigation exercises. The storage and transmission of high-resolution elevation information can consume considerable amounts of resources, and with the increased interest in mapping the earth and having maps for real time navigation, the development of compression techniques to help in theses tasks is becoming very important.

In this paper we investigate the use of JPEG-LS, the new ISO/ITU standard for lossless and near-lossless compression of continuous-tone imaging for the compression of high-resolution elevation data. The advantages of using a standard algorithm are numerous, including making it possible for any user to compress and decompress the data and the wide availability of supporting software and hardware. This is what has motivated the development of popular standards such as JPEG (lossy still image compression), MPEG (lossy video compression), the new JPEG-LS (lossless and near-lossless still image compression), and the forthcoming JPEG2000. In particular, for elevation data, JPEG-LS has a number of advantages over other standards (e.g., JPEG). First of all, it is capable of lossless compression. This is not just important but mandatory for databases such as those collected and maintained by the U.S. Geological Survey (USGS). Secondly, JPEG-LS includes a near-lossless mode through which the maximal error in pixel value can be controlled, thereby limiting the maximal elevation error in the reconstructed image. This is fundamental for applications such as landing, where the terrain slope is of primary importance. This makes the standard JPEG-LS a perfect candidate for the compression of elevation maps. While other papers have reported results on the compression of elevation maps, e.g., [1], [2] and references therein, to the best of our knowledge none of them have these three fundamental qualities all together (using a standard algorithm and having both lossless and controlled-lossy modes). Moreover, JPEG-LS has very low computational complexity, and as we will detail later in this paper, it achieves significant improvements on compression ratios over previously reported results.

Digital elevation data for a region normally consists of an array of elevations for ground positions at regularly spaced intervals, see Fig. 1. In this report we investigate data from both the USGS digital elevation model (DEM) and the Navy digital terrain models (DTM). The exact details and resolutions for this data are given in the results section. For about 100 images tested from the USGS, we have obtained an average lossless compression ratio of 14.23 : 1 for 16-bit images, or 1.12 bits per pixel. For similar algorithmic complexity, [2] reports compression ratios of 8–9 : 1, with increased buffer size and without all the JPEG-LS qualities described previously. With a significant addition on algorithmic complexity, [2] reports compressions of 9.85 : 1 for this type of data. As stated earlier, we can also perform controlled lossy compression with JPEG-LS and we also report in this article significant compression ratios on high-resolution data of 24 bits or more with a guaranteed maximal error in height which is insignificant for most applications. We also tested compressing the image in subblocks, thereby allowing for semirandom access to the data. In the rest of this paper, we will detail all these findings.

This paper is divided in two major parts. In the first part a brief description of JPEG-LS is provided, while the second part reports results on applying JPEG-LS to the elevation data. In the concluding remarks section we provide some design suggestions for future systems using JPEG-LS for the compression

Manuscript received September 25, 2000; revised June 25, 2001. This work was supported in part by a Grant from the Office of Naval Research Grant ONR-N00014-97-1-0509, the Office of Naval Research Young Investigator Award, the Presidential Early Career Awards for Scientists and Engineers (PECASE), a National Science Foundation CAREER Award, and by the National Science Foundation Learning and Intelligent Systems Program (LIS).

The authors are with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: guille@ece.umn.edu).

Publisher Item Identifier S 0196-2892(01)08126-8.

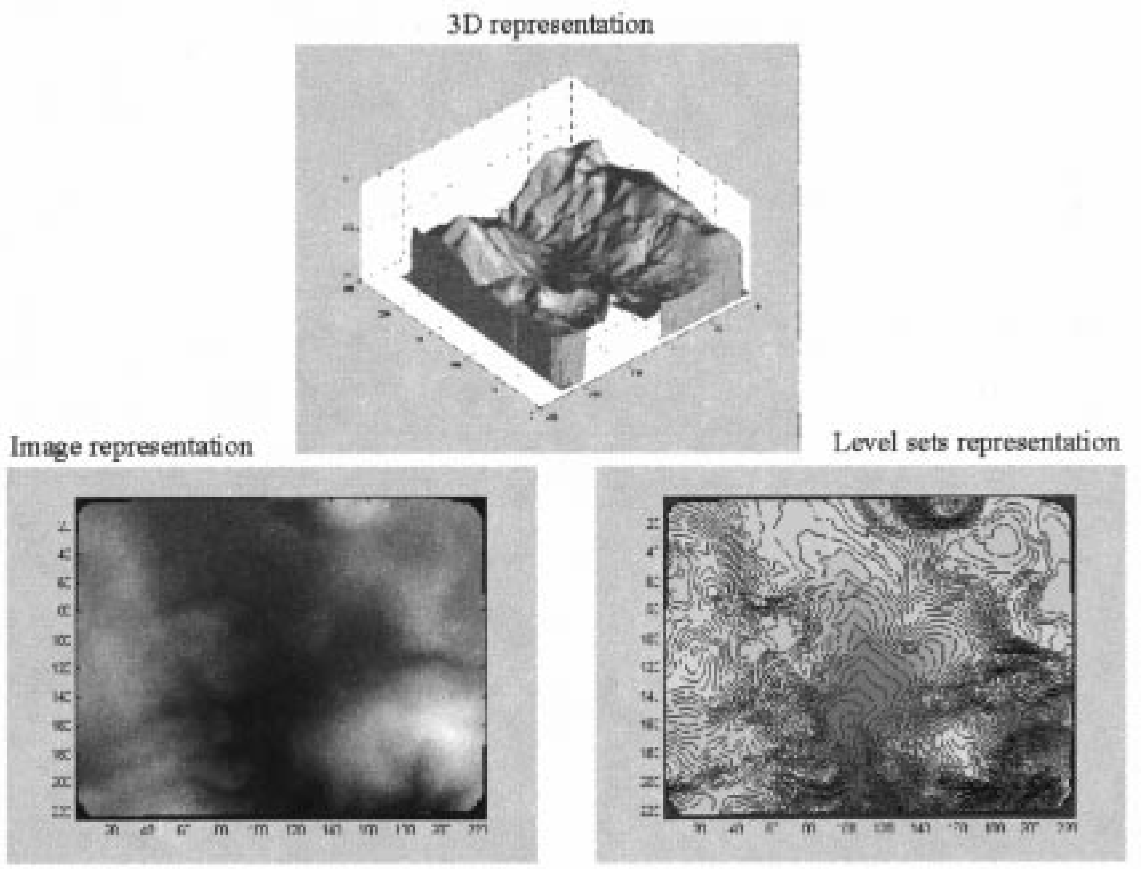

Fig. 1. Rendering of elevation maps.

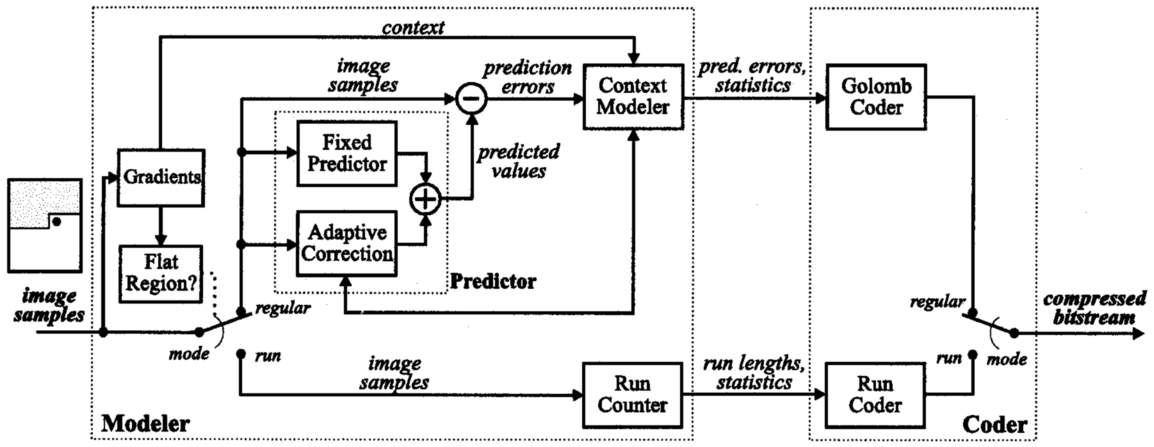

Fig. 2. Basic block diagram of JPEG-LS.

of elevation data and suggest, based on the results here reported, the adoption of JPEG-LS as a possible compression algorithm for this type of information.

II. Brief Description of the JPEG-LS Algorithm

JPEG-LS achieves state-of-the-art compression rates at very low computational complexity and memory requirements. These characteristics are what brought to the selection of JPEG-LS, which is based on the LOCO-I algorithm developed at Hewlett-Packard Laboratories, as the new ISO/ITU standard for lossless and near-lossless still image compression.

The basic block diagram of JPEG-LS if given in Fig. 2. In this section, we briefly describe the main components of JPEG-LS. A detailed description of this algorithm is contained in [3], from where the brief description below has been adapted.

A. Modeling and Prediction

Modeling in lossless image compression can be formulated as an inductive inference problem [4]. In a raster scan, after having scanned past data, one infers the next pixel value by assigning a

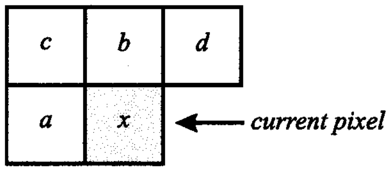

Fig. 3. Causal template for JPEG-LS.

conditional probability distribution to it. In state-of-the-art lossless image compression schemes, this probability assignment is generally broken into the following components.

a) A prediction step, in which a deterministic value $\hat{x}_{t+1}$ is guessed for the next sample x_{t+1} based on a finite subset (a causal template) of the available past sequence x^t.
b) The determination of a context in which x_{t+1} occurs. The context is a function of a (possibly different) causal template.
c) A probabilistic model for the prediction residual (or error signal) $\epsilon_{t+1} \triangleq x_{t+1} - \hat{x}_{t+1}$, conditioned on the context of x_{t+1}. This model determines how the residual is compressed.

The prediction and modeling units in JPEG-LS are based on the causal template depicted in Fig. 3, where x denotes the current sample, and a, b, c and d are neighboring samples in the relative positions shown in the figure. The dependence of a, b, c, d, and x, on the time index t has been deleted from the notation for simplicity. Moreover, by abuse of notation, we will use a, b, c, d, and x to denote both the values of the samples and their locations. The use of the template of Fig. 3 implies a buffering requirement of just one scan line.

Specifically, the fixed predictor in JPEG-LS is based on the following simple formula (which has limited edge detection capability):

$$\hat{x}_{\text{MED}} \triangleq \begin{cases} \min(a, b) & \text{if } c \geq \max(a, b) \\ \max(a, b) & \text{if } c \leq \min(a, b) \\ a+b-c & \text{otherwise.} \end{cases} \tag{1}$$

B. Context Modeling

Reducing the number of parameters is a key objective in a context modeling scheme. The total number of parameters in the model depends on the number of free parameters defining the coding distribution at each context and on the number of contexts.

Addressing the issue of number of parameters per context, it is an accepted observation, adopted by JPEG-LS, that the global statistics of residuals from a fixed predictor in continuous tone images are well modeled by two-sided geometric distributions (TSGD) centered at zero. For context-conditioned predictors, this distribution has an offset, and this is addressed by JPEG-LS as well. For each context then there is a need to estimate the exponential decay value and center of the distribution, just two parameters.

The prediction residual ϵ can take on any value in the range $-\alpha < \epsilon < \alpha$, where α is the size of the image alphabet. Actually, given the predicted value $\hat{x}$ (known to both the encoder and decoder from the causal template), ϵ can take on only α-possible different values. This property is exploited in JPEG-LS by reducing, modulo α, the actual value of the prediction residual to a value between $-\lfloor \alpha/2 \rfloor$ and $\lceil \alpha/2 \rceil - 1$, thus remapping large prediction residuals to small ones. Merging the "tails" of peaked distributions with their central part does not significantly affect the original two-sided geometric behavior. In the common case in which $\alpha = 2^\beta$ (i.e., β bits per sample), the previous remapping consists of just interpreting the β least significant bits of $x - \hat{x}$ in 2's complement representation.

JPEG-LS has then to address the determination of the context, and guaranteeing this is a relatively small number, although large enough to capture the different local statistics of the image. The context that conditions the encoding of the current prediction residual in JPEG-LS is built out of the differences $g_1 = d-b$, $g_2 = b-c$, and $g_3 = c-a$. These differences represent the local gradient, thus capturing the level of activity (smoothness, edginess) surrounding a sample, which governs the statistical behavior of prediction errors. For further model size reduction, each difference g_j, $j = 1, 2, 3$, is quantized into a small (fixed) number of approximately equiprobable, connected regions by a quantizer $\kappa(\cdot)$ independent of j. This aims to maximize the mutual information between the current sample value and its context, an information-theoretic measure of the amount of information provided by the conditioning context on the sample value to be modeled. We refer to [5] and [6] for an in-depth theoretical discussion of these issues.

To preserve symmetry, the regions are indexed $-T, \ldots, -1, 0, 1, \ldots, T$, with $\kappa(g) = -\kappa(-g)$, for a total of $(2T+1)^3$ different contexts. To further reduce the number of contexts, symmetric contexts are merged. The total number of contexts then becomes $((2T+1)^3+1)/2$. For JPEG-LS, $T = 4$ was selected, resulting in 365 contexts.

JPEG-LS provides default thresholds T1, T2, T3 to define the boundaries between quantization regions. These depend on the size of the alphabet, and can also be changed by the user. A suitable choice collapses quantization regions, resulting in a smaller effective number of contexts, with applications to the compression of small images. This will be important for the partition of large elevation maps into small regions for semi-random access.

The systematic context-dependent biases (offsets) in prediction residuals deteriorate the performance of the Golomb–Rice coder used by JPEG-LS (see below), which relies heavily on two-sided geometric-distributions (TSGDs) of prediction residuals centered about zero. To alleviate the effect of systematic biases, JPEG-LS uses an error feedback aimed at "centering" the distributions of prediction residuals. This bias cancellation is based on keeping counters (per context) for the number N of total context occurrences and the accumulated prediction residual (B). See [3] for details on this very low complexity approach.

C. Coding

To encode bias corrected prediction residuals distributed according to the TSGD, JPEG-LS uses a minimal complexity family of optimal prefix codes for TSGDs, sequentially selecting the code among this family.

Golomb codes were first described in [7] as a means for encoding run lengths. Given a positive integer parameter m, the Golomb code G_m encodes an integer $y \geq 0$ in two parts: a unary representation of $\lfloor y/m \rfloor$ and a modified binary representation of $y \bmod m$ (using $\lfloor \log m \rfloor$ bits if $y < 2^{\lceil \log m \rceil} - m$ and $\lceil \log m \rceil$ bits otherwise). Golomb codes are optimal [8] for one-sided geometric distributions of the nonnegative integers, i.e., distributions of the form $(1 - \theta)\theta^y$, where $0 < \theta < 1$. Thus, for every θ there exists a value of m such that G_m yields the shortest average code length over all uniquely decipherable codes for the nonnegative integers.

The special case of Golomb codes with $m = 2^k$ leads to very simple encoding/decoding procedures. The code for y is constructed by appending the k least significant bits of y to the unary representation of the number formed by the remaining higher order bits of y (the simplicity of the case $m = 2^k$ was already noted in [7]). The length of the encoding is $k + 1 + \lfloor y/2^k \rfloor$. We refer to codes G_{2^k} as Golomb-power-of-2 (GPO2) codes.

In order to use these codes, the TSGDs have to be first mapped into one-sided geometric distributions. This is elegantly addressed in JPEG-LS, based on theoretical results reported in [9], [10]; see [3] for details.

In keeping with the low complexity constraints set for JPEG-LS, as in [11], JPEG-LS uses the sub-family of codes for which the Golomb parameter is a power of 2. To conclude the coding then, we need to select the specific code from this sub-family, mainly adaptively select the context-dependent k. The explicit computation method, as opposed to an exhaustive search for the best code, is used in JPEG-LS for determining the optimal code in the subfamily, based on the sufficient statistics which are functions of the number of times a given context was previously used and the total error previously accumulated for the given context. The adaptive selection of the code is based on results proved in [10], and it is detailed in [3]. Just to give the reader an idea of how computationally efficient is to compute k, in the C programming language it is done by the "one-liner"

$$\texttt{for}\ (\texttt{k} = 0;\ (\texttt{N} \ll \texttt{k}) < \texttt{A};\ \texttt{k}++).$$

Here A is an accumulator related to the statistics previously mentioned.

D. Embedded Alphabet Extension: Run Mode

Golomb codes, being subsets of the class of Huffman coding (as opposed to arithmetic coding) have a problem of redundancy (i.e., excess code length over the entropy) for contexts representing smooth regions, which have peaked distributions as a prediction residual of zero is very likely. This is due to its fundamental limitation of producing at least one code bit per encoding. JPEG-LS addresses the problem of redundancy by embedding an alphabet extension into the context conditioning. Specifically, the encoder enters a differently encoded "run" mode when a context with $a = b = c = d$ is detected, as this indicates a flat region. Since the central region of quantization for the gradients g_1, g_2, g_3 is the singleton $\{0\}$, the run condition is easily detected in the process of context quantization by checking for the quantized context $[q_1, q_2, q_3] = [0, 0, 0]$. Details on this run mode are provided in [3].

E. Near-Lossless Compression

JPEG-LS offers a lossy mode of operation, termed "near-lossless," in which every sample value in a reconstructed image component is guaranteed to differ from the corresponding value in the original image by up to a preset (small) amount, δ. This is of significant importance for the compression of elevation data as we will see below. Moreover, JPEG-LS is the only standard currently supporting this mode of operation.

The basic technique employed for achieving this near-lossless or controlled-lossy in JPEG-LS is the traditional DPCM loop [12], where the prediction residual (after correction and possible sign reversion, but before modulo reduction) is quantized into quantization bins of size $2\delta + 1$, with reproduction at the center of the interval (thereby giving a maximal error of δ).

Context modeling and prediction are based on reconstructed values, so that the decoder can mimic the operation of the encoder. The condition for entering the run mode is relaxed to require that the gradients $g_i, i = 1, 2, 3$ satisfy $|g_i| \leq \delta$. This relaxed condition reflects the fact that reconstructed sample differences up to δ can be the result of quantization errors. Moreover, once in run mode, the encoder checks for runs within a tolerance of δ while reproducing the value of the reconstructed sample at a. Consequently, the run interruption contexts are determined according to whether $|a - b| \leq \delta$ or not. The relaxed condition for the run mode also determines the central region for quantized gradients, which is $|g_i| \leq \delta, i = 1, 2, 3$. Thus, the size of the central region is increased by 2δ. Consequently, the default thresholds for gradient quantization are scaled accordingly.

Note that this mode allows to "ignore" small errors in the elevation data for general terrains. When combined with the run mode, it provides an additional very efficient form to compress slowly sloped terrains.

The quantized error is scaled and transmitted to the decoder. This scales it back as part of the decoding procedure. The statistics collecting counters to determine the value of k in the context dependent Golomb-code uses the quantized (and scaled) predictions as well; see [3] for details.

For a discussion on Multicomponent images, palettes and sample mapping, and the JPEG-LS bitstream structure, please refer to [3].

III. Compression of Elevation Data

This section describes results on compression of high resolution digital elevation data. We report the different techniques used to compress elevation data based on spatial resolution, bits per pixel, and range of pixel values. Using the number of bits per pixel of the original image as a criteria for classification we have three main types of images, as discussed below. For each of these classes we describe three compression approaches as follows.

a) Compressing the whole image as is, both in lossless and near-lossless mode.

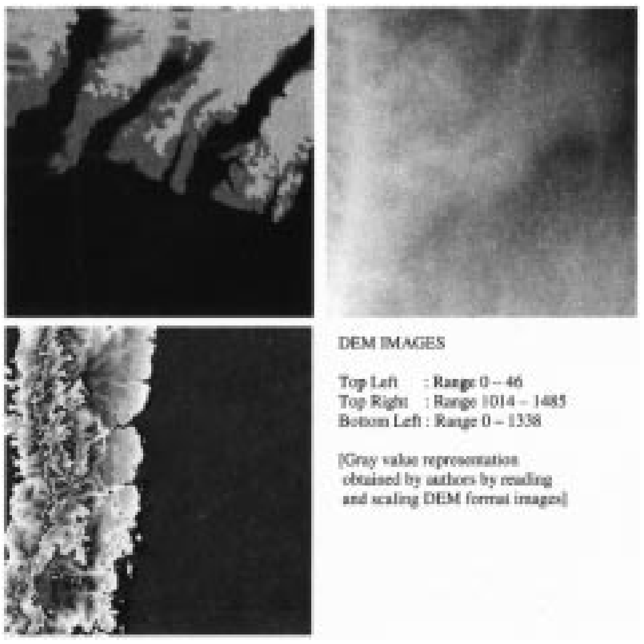
DEM IMAGES
Top Left : Range 0 – 46
Top Right : Range 1014 – 1485
Bottom Left : Range 0 – 1338
[Gray value representation obtained by authors by reading and scaling DEM format images]

Fig. 4. Sample DEM images with elevation range.

b) Partitioning the image into smaller blocks (128 × 128 pixels). This allows semi-random access to different regions of the image as well as local adaptation of the (few) JPEG-LS parameters.

c) Compressing a "slope" image. This is for applications such as helicopter landing, where the terrain slope is a critical factor.

In describing each of these approaches, we provide the results obtained and give methods to optimize the compression ratio. The advantages of a particular approach, and the tradeoff which it entails, will also be discussed.

A. Compressing the Whole Image As Is

1) Images With 16 Bits per Pixel or Less: The images used for testing in this section are in DEM (Digital Elevation Model) format and were obtained from the USGS site http://edcftp.cr.usgs.gov/pub/data/DEM/250. An example of this is provided in Fig. 4. Another available format is the spatial data transfer standard (SDTS). Within DEM, there are images with scales of 1 : 24 000, 1 : 100 000 and 1 : 250 000. The latter were used for the tests described below. Specifications of the DEM format are available at http://edcwww.cr.usgs.gov/glis/hyper/guide/usgs_dem.supplement#typea. When DEM data is read into a raster format, the elevations are merely numbers in a 1201 × 1201 array of integers (specifically short integers). As described earlier, JPEG-LS has default quantization regions for context determination. These are obtained by assigning T1 = 3, T2 = 7 and T3 = 21 as the default quantization level thresholds. (Results with these default parameters are first reported in the table.) Equalizing any two or all of the values collapses the quantization regions and reduces the effective number of contexts available. This might result in a significant improvement in compression ratio for small images such as 1 : 250 000 DEMs (as well as for 128 × 128 blocks). Typically, these images are of size 9.84 MB in their DEM data format. On converting them to a 2-D array of short integers, this size reduces to 2.88 MB. This file is then compressed using JPEG-LS. The "effective compression ratio" entered in Table I, and henceforth, is calculated from the size of the original DEM file. The actual compression ratios and bits per pixel are given in the tables as well.

As observed in Table I, the compression ratios depend on the pixel values range. The first image has a small range of pixel values and compression is improved by reducing the number of contexts. The second and third images also have a reasonably small range and compression improves marginally when T1, T2, and T3 are equalized. Images 4 and 5 however, have a larger range and benefit from having a larger number of contexts. In the table we report the results both for default and for optimized thresholds T1, T2, and T3. As noted, a slightly improvement can be obtained with tailored thresholds, although the results with the default values are already, to the best of our knowledge, state-of-the-art for this type of algorithmic complexity. The specific optimal thresholds could be, for example, learned off-line. This is in particular possible for DEM due to the availability of abundant sample data.

We have tested JPEG-LS on over 100 16-bit DEM images from the USGS data set, and obtained an average compression ratio of 14.23 : 1. JPEG-LS applied to images of very flat terrain produces extremely high compression ratios (1000 and more). These high values would produce a distorted average compression ratio (mean is much larger than the median), which will not be indicative of the true performance of JPEG-LS algorithm. Hence they have been omitted from the data set when computing the average compression. Fig. 5 shows compression ratios for a these DEM images (the horizontal axis is just the image number.) To improve the visualization, we have removed from the graph as well all images that compressed more than 60 : 1. The average compression ratio for the set shown, which is then a subset from those tested, is 11.75 : 1. Average effective compression ratio is 40.08. JPEG-LS default values of the thresholds T1, T2, and T3 were used for this large data set.

Comparing with other results reported in the literature, e.g., [2] (which includes among others tests with popular packages like GZIP), JPEG-LS achieves significantly better compression at a similar or lower computational cost. To this we add all the advantages of working with JPEG-LS that were mentioned in the introduction.

2) Images With 16 to 32 Bits per Pixel: These are very high resolution images in DTM (Digital Terrain Model) format with 3 m postspacing. The 3 m resolution DTM data consists of points described by three coordinates: x-coordinate (easting), y-coordinate (northing) and z-coordinate (elevation). The elevation values are stored in double float format (64-bit per pixel) in all DTM files. However, for this class of images, the actual number of bits required to completely represent the z-coordinate (elevation) is far smaller than 64. For example, all elevations in the example "cosogeo3.asc" could be represented using 24 bits to an accuracy of 10^{-11} m. JPEG-LS can directly compress only (at most) 16-bit images. So each 24-bit elevation in cosogeo3.asc was split into a 16 bits value and a eight bits value and each was compressed separately. The results appear in Table II. A large saving is realized even prior to compression,

TABLE I
COMPRESSION RESULTS ON 1 : 250 000 DEM DATA

Image	JPEG-LS CR (default T1,T2,T3)	Effective CR (default T1,T2,T3)	Optimized JPEG-LS CR	Optimized Effective CR	Optimum T1,T2,T3	Optimum Bits Per Pixel	Pixel value range
richmond-e	45.15	154.01	46.87	159.89	17,17,17	**0.34**	5-46
grand-rapids-e	13.76	46.99	13.82	47.13	4,4,4	**1.16**	192-381
dallas-e	11.69	39.86	11.79	40.22	3,3,3	**1.36**	73-262
puerto-rico-c	9.75	33.27	9.79	33.40	5,3,25	**1.63**	5-680
denver-w	4.12	14.07	4.36	14.89	7,13,33	**3.67**	1557-4350

[CR = Compression ratio, Original images are 16-bit per pixel]

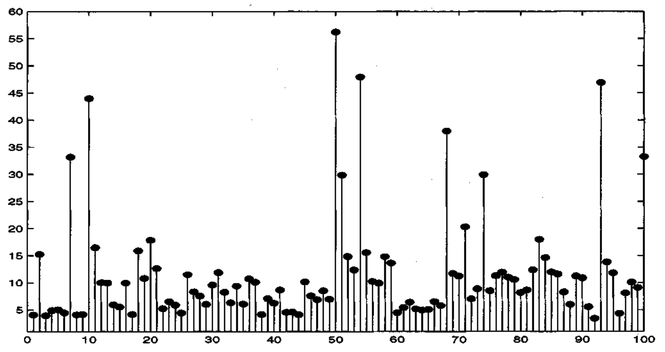
Fig. 5. Compression ratios for 100 DEM images. The vertical axis indicates the compression ratio while the horizontal one stands for the DEM image number. The average compression ratio is 11.75.

TABLE III
COMPRESSION RESULTS ON swath3.asc DTM DATA (ORIGINALLY 25-BIT PER PIXEL IMAGE)

Part of Elevation Data	Entirely Lossless	NL=1 for MID1 Neglect LO	NL=3 for MID1 Neglect LO
HI	27.46	27.46	27.46
MID2	3.29	3.29	3.29
MID1	1.10	1.38	1.71
LO	5.47	—	—
MAX ERROR (meters)	0	0.0003	0.0008
Total Compression	2.45	2.94	3.38
Bits Per Pixel	**10.18**	**8.50**	**7.39**
Effective Compression	7.37	8.85	10.17

[CR = Compression Ratio, NL = Near-lossless parameter]
[Original data size = 61.94 MB, DTM file size = 226 MB]
[Best Compressed File-size (Lossless) = 9.43 MB]

TABLE II
COMPRESSION RESULTS ON cosogeo3.asc DTM DATA (24-BIT PER PIXEL)

FOR 16-8 SPLIT						
Method	CR for Upper 16 bits	CR for lower 8 bits	Total CR	Effective CR	Bits Per Pixel	Maximum Error(meters)
Totally Lossless	4.63	1.06	2.18	7.07	**11.00**	0
Upper 16 Lossless Lower 8 NL = 3	4.63	1.65	2.89	9.35	**8.30**	0.0003
Upper 16 Lossless Lower 8 NL = 5	4.63	1.88	3.11	10.07	**7.72**	0.0005
Upper 16 NL = 1 Lower 8 NL = 3	7.02	1.65	3.37	10.90	**7.12**	0.0515
Upper 16 Lossless Lower 8 neglect	4.63	—	6.94	22.46	**3.46**	0.0255
FOR 12-12 SPLIT						
Method	CR for Upper 12 bits	CR for lower 12 bits	Total CR	Effective CR	Bits Per Pixel	Maximum Error(meters)
Totally Lossless	11.03	1.36	2.42	5.87	**9.92**	0
Upper 12 Lossless Lower 12 NL = 3	11.03	1.77	3.05	7.40	**7.87**	0.0003

[CR = Compression Ratio, NL = Near-lossless parameter]
[Original data size = 5.37 MB, DTM file size = 19.35 MB]
[Best Compressed File-size (Lossless) = 0.76 MB]

when the double data is converted to the aforementioned 16–8 split.

Table II also shows the results when the elevation data was split into two groups of 12 bits. The compression ratio is superior (as expected, because the upper 12 bits show very little variation for adjacent pixels, and hence can be compressed better than the upper 16 bits). However, the effective compression ratio is less, because we now have to use (16 bits) integers for both parts.

A different approach was used for the example "swath3.asc" which had a maximum of 25 bits per pixel. Here the split used was 8–8–8–1, the reasons being the following.

a) The LSB could be neglected entirely, if tolerable, or suited to the application.
b) The most significant byte varies little from pixel to pixel. The effective compression for the upper 16 bits was more if bits 18–25 were compressed separately from bits 10–17 as opposed to compressing bits 10–25 at once.

The larger size of this data set allowed the JPEG-LS adaptive predictor to train itself. This, along with point b) from earlier, is responsible for improving the compression ratio (2.79 for swath3.asc as compared with 2.18 for cosogeo3.asc; both files have the same spatial resolution). Results appear in Table III.

As expected, when increasing the number of bits used to represent the elevation data, the compression ratio is reduced. On the other hand, when neglecting a few of the lower significant bits or using JPEG-LS in the controlled lossy mode (see the following), significant improvements are achieved. The error in elevation in this case is insignificant for most applications. Therefore, JPEG-LS achieves significant compression ratios for this type of data as well. The same conclusion holds for the very high resolution data reported below.

3) Images With 32 or More Bits per Pixel: These images are also in DTM format but with 10 m postspacing. Thus they have lower resolution, but the pixels now take up more bits (up to 48). Note that the numbers do not increase in magnitude, but only in precision (i.e., the number of digits after the decimal point is increased). These were split as 16–16–16 and

TABLE IV
COMPRESSION RESULTS ON swath10.asc DTM DATA (48-BIT PER PIXEL)
[CR = COMPRESSION RATIO] [NL = NEAR LOSSLESS PARAMETER]

FOR 8-8-8-8-8-8 SPLIT				
Part of Elevation Data	Entirely Lossless	NL=1 for LO	NL=3 for LO	Neglect LO and MID1
HI (8 bits)	29.32	29.32	29.32	29.32
MID4 (8 bits)	3.47	3.47	3.47	3.47
MID3 (8 bits)	1.07	1.07	1.07	1.07
MID2 (8 bits)	1.13	1.13	1.13	1.13
MID1 (8 bits)	1.05	1.05	1.05	—
LO (8 bits)	1.05	1.30	1.55	—
MAX ERROR (meters)	0	10^{-11}	3×10^{-11}	6.5536×10^{-6}
Total Compression	1.48	1.55	1.60	2.80
Bits Per Pixel	**32.43**	**30.97**	**30.00**	**17.14**
Effective Compression	3.73	3.91	4.03	7.05
FOR 16-16-16 SPLIT				
Part of Elevation Data	Entirely Lossless	NL=1 for LO	NL=3 for LO	Neglect LO
HI (16 bits)	6.01	6.01	6.01	6.01
MID (16 bits)	1.10	1.10	1.10	1.10
LO (16 bits)	1.08	1.20	1.30	—
MAX ERROR (meters)	0	10^{-11}	3×10^{-11}	6.5536×10^{-6}
Total Compression	1.49	1.57	1.62	2.77
Bits Per Pixel	**32.21**	**30.57**	**29.63**	**17.33**
Effective Compression	3.76	3.94	4.08	6.98

[CR = Compression Ratio, NL = Near-lossless parameter]
[Original data size = 9.54 MB, DTM file size = 28.7 MB]
[Best Compressed File-size (Lossless) = 2.54 MB]

TABLE V
LOSSLESS COMPRESSION RESULTS ON mariposa-w

Method	(T1,T2,T3)	Avg. CR
As a whole	3,7,21	3.58
81 segments	3,7,21	2.84
81 segments	7,7,7	3.43

[Each block is 128 × 128 pixels]

TABLE VI
COMPRESSION ON SEGMENTS OF cosogeo3.asc DTM

2.65	2.10	2.05	2.11	2.14	2.54
2.36	2.00	1.93	1.97	1.94	2.43
2.25	2.07	2.08	2.09	1.97	2.31
2.02	2.15	2.15	2.12	2.07	2.43
2.02	2.12	2.13	1.90	2.01	2.38
3.21	2.55	2.53	2.49	2.51	3.32
(T1,T2,T3)=(3,7,21)					

2.64	2.10	**2.08**	2.14	2.11	2.48
2.27	1.95	1.90	1.94	1.93	**2.45**
2.18	2.02	2.08	2.07	1.97	2.30
2.02	2.13	2.15	2.01	2.03	**2.46**
1.98	**2.15**	2.13	1.83	2.01	**2.40**
3.08	2.49	2.53	**2.54**	2.50	**3.33**
(T1,T2,T3)=(4,4,4)					

8–8–8–8–8–8. Due to the larger post spacing, the least significant bits in adjacent pixels are uncorrelated. The compression performance is good only for the upper 16 bits and there is little or no compression for the other packets; see Table IV. Thus, totally lossless JPEG-LS, which preserves the elevation (accurate to 10^{-11} ft) does not provide appreciable compression for these images. A considerable saving is realized if, for applications which don't require this accuracy, the lower 16 bits are neglected totally. Even this drastic omission in swath10.asc gives a maximum pixel error of 0.000 006 553 6 m. Lossless compression results for the 16–16–16 split are slightly better than those for 8–8–8–8–8–8 split. This indicates that, if there are 16 bits of almost uncorrelated data, then we are better off compressing them together instead of splitting them into groups of eight bits. But when the 16 lower bits are sacrificed, the 8–8–8–8–8–8 method wins, for the reasons discussed in Section III-A2. (Note the very high compression ratio for the upper eight bits.) From Table IV, we surmise that the best possible split is 8–8–16–16. This yields a total lossless CR of 1.50 and an effective lossless CR of 3.77. Once again, keeping this high precision is not necessary for most applications.

B. Partition of Image Followed by Compression

Partition of the image into tiles of 64 × 64 or 128 × 128 pixels permits a semi-random access to the data. In addition, the near-lossless parameter can be independently selected per tile, and while some tiles can be compressed in lossless mode, others can be compressed in the near-lossless mode and with different error tolerance.

We now investigate the effects of tiling in the compression of elevation data. On one hand, tiling might improve the compression ratio when the JPEG-LS parameters can be adapted to each particular tile. On the other hand, the compression ratio for the total image will be negatively affected if the adaptive predictor has significantly less data to learn and adapt. As we will see in the following, the overall changes in compression ratio for tiled data are not very significant, and we gain the semi-random accessibility to the elevation map.

1) Images With Less Than 16 Bits per Pixel: Table V shows the performance of JPEG-LS for mariposa-w, when the image is partitioned into blocks of 128 × 128 pixels, for two different triads of T1, T2, and T3. As expected, the compression ratio suffers due to the tiling operation. However, after tiling, it is possible to equalize T1, T2, T3 as shown, to obtain a significantly higher average compression ratio. It is thus clear that reducing the number of contexts results in better compression of an individual tile.

2) Images With 16 to 32 Bits per Pixel: Table VI (left) shows compression ratios for the different 128 × 128 pixel tiles of the cosogeo3.asc (DTM) image. The compression ratio for cosogeo3.asc was 2.18 and the average ratio for the tiled version is 2.096 for the default triad (T1,T2,T3) = (3,7,21). The reduction is due to the tiling operation as discussed previously. Table VI (right) contains the compression ratios for (T1,T2,T3) = (4,4,4). As expected, compression ratio increases for some segments and decreases for others. This suggests that different segments have different optimum triads, and the segment of interest can be compressed more if its optimum triad is found. Equalizing T1, T2, and T3 is profitable while compressing the upper 16 (or 12) bits. These do not vary appreciably from pixel to pixel and a smaller number of contexts suffices. The lower eight (or 12) bits vary significantly across pixels and hence require a different optimum triad, wherein T1, T2, and T3 are not necessarily equal. A better average overall compression ratio than that obtained in the above two tables will thus result if (4,4,4) is used to compress the upper 16 bits and (3,7,21) is used for the lower eight bits.

C. Compression of a "Slope" Image

Preserving the terrain slope, not its absolute elevation, it is of importance for operations such as helicopter landing. As a variation, JPEG-LS was tested on a "slope" image. This image was

TABLE VII
COMPRESSION RESULTS ON cosogeo3.asc SLOPE DATA

FOR 16-8 SPLIT						
Method	CR for Upper 16 bits	CR for lower 8 bits	Total CR	Bits Per Pixel	Effective CR	Maximum Error (meters/meter)
Totally Lossless	1.48	0.98	1.26	**19.05**	4.09	0
Upper 16 Lossless Lower 8 NL = 3	1.48	1.47	1.47	**16.33**	4.76	10^{-7}
Upper 16 Lossless Lower 8 NL = 5	1.48	1.66	1.54	**15.58**	4.96	1.667×10^{-7}
Upper 16 Lossless Lower 8 neglect	1.48	—	2.22	**10.81**	7.17	8.5×10^{-6}
0.1234 FORMAT RETAINING SLOPE TO 4 DECIMAL PLACES						
Method	Total CR			Bits Per Pixel	Effective CR	Maximum Error (meters/meter)
Totally Lossless	1.88			**12.77**	9.11	0
NL = 3	2.75			**8.73**	13.34	10^{-4}

[CR = Compression Ratio, NL = Near-lossless parameter, Originally 24 bpp]
[Original data size = 5.33 MB, DTM file size = 19.35 MB]
[Best Compressed File-size (Lossless) = 1.30 MB]
[Best Compressed File-size in 0.1234 format = 0.585 MB]

constructed out of the original cosogeo3.asc using the following relations:

$$I_x(i, j) = (I(i+1, j) - I(i-1, j))/2$$
$$I_y(i, j) = (I(i, j+1) - I(i, j-1))/2$$
$$Gradient(I) = \sqrt{I_x^2 + I_y^2}.$$

I_x and I_y are the derivatives in the horizontal and vertical directions. To avoid boundary problems, we neglect the first and last rows and columns [from an $N \times M$ image we compute an $(N-2) \times (M-2)$ size image]. The square root operation will result in very fine precision, but we persist with the same number of digits after the decimal points as that in the original image. The compression ratio then is superior to that obtained by compressing the image as is. Table VII shows the results.

Once again, the motivation for this approach comes from landing applications wherein slope of the terrain is more important than the absolute elevation. We have the following options.

a) Transmit a compressed image as in Section III-A2 and perform slope calculations after reception. This is once again where the advantages of JPEG-LS come into play. Since JPEG-LS bounds the elevation error (in the controlled lossy mode), we obtain a bound in the slope error of the lossy reconstructed image as well, see the following for an example.
b) Computing the slope image and compressing it prior to transmission, as earlier.

D. Near Lossless Mode

Section II-E describes the near lossless mode in JPEG-LS. If we can tolerate a fixed error in some or all the pixels of an image, then the near lossless mode can be use to advantage, especially for images with high elevation resolution (adjacent pixel values are correlated and their difference is small). Tables II and III show how a higher compression ratio is obtainable if near-lossless JPEG-LS is used to compress the least significant byte portion. For example, when the lossy parameter $\delta = 1$, the pixels of cosogeo3.asc will be offset by at most 0.0001 m. For landing-related applications, this translates into a maximum slope error of 6.667×10^{-5} m/m for 3 m DTM data. Even with these insignificant errors, compression ratios are considerably improved.

Note that controlled-lossy results are also obtained when entire low significant bytes of data are neglected for large dynamic range images. This is also reported in the tables, where compression per byte (or word) is given. Also in this case, the elevation error is considerably small and the compression ratio is significantly improved.

To recapp, JPEG-LS near-lossless mode permits a significant improvement in the compressibility of the elevation data at a cost which is irrelevant for most real applications.

IV. CONCLUSIONS

We have studied the application of JPEG-LS for the compression of elevation data. Using JPEG-LS has three main advantages. First, it is a low complexity, high-compression ratio standard. Second, it permits lossless compression, which is fundamental for applications such as storage. And third, it provides a controlled lossy mode that permits the user to dictate a maximal error in the elevation (slope), thereby improving the compression ratio while guaranteeing performance.

From the investigation, we have also concluded the following design decisions.

a) The compression is marginally better if the upper 16 bits are first split into 2 bytes each and then separately compressed, as opposed to direct compression of the upper 16 bits.
b) When lower and middle bits of images are minimally correlated or uncorrelated, it is better to compress them in groups of 16 bits rather than groups of eight bits. This is particularly helpful for 10 m resolution DTM data.
c) It is beneficial if the least significant bits are in as small a group as possible (e.g., 25 bit data can be split as 8–8–8–1). If permissible, this segment can be totally

neglected resulting in higher effective compression. This applies especially to 3 m resolution DTM data.

d) For images where a region of concentration can be defined, it is possible to partition the image into tiles of 128 $\times$ 128 pixels. The region of interest can be compressed with lossless JPEG-LS and the reduced pixel value range makes the compression ratio responsive to the quantization region thresholds T1, T2 and T3. For one optimal choice of the triad, which collapses the quantization regions and reduces the number of contexts, better compression is obtained. The remaining segments can either be neglected or be compressed using near-lossless JPEG-LS, so that, in return for a controlled loss, improved compression is obtained.

e) The compression capability of JPEG-LS improves with the resolution of the DEM data. Higher resolution provides a better correlation between adjacent pixel values and therefore a better performance for the JPEG-LS predictor.

f) The larger the image, the better the predictor in JPEG-LS is trained. Hence, the compression is marginally better for a larger image than for a smaller image with the same spatial resolution.

g) Segmentation allows better compression of an individual segment but the overall compression ratio suffers due to point f) earlier. (JPEG-LS would now compress small blocks of 128 $\times$ 128 pixels instead of one large image).

To conclude, having in mind the immediate availability of JPEG-LS, the results here reported strongly support its adoption for the compression of elevation data for a number of applications, e.g., storage. This does not mean that JPEG-LS provides a complete solution to the problem and indeed, the development of compression algorithms tailored to elevation data is still an open problem.

ACKNOWLEDGMENT

The authors would like to thank Dr. G. Hewer and Dr. L. Peterson from China Lake (NAWCWD) for providing high-resolution DTM data and Dr. L. Peterson for providing DTM-reading software.They would also like to thank Dr. W. Masters and Dr. C. Schwartz from ONR for introducing them to the problem and motivating this investigation, A. Solé for producing Fig. 1, and Dr. G. Seroussi and Dr. M. Weinberger from Hewlett-Packard Laboratories for their transmitted knowledge of image compression.

REFERENCES

[1] W. R. Frankin and A. Said, "Lossy compression of elevation data," in *Proc. 7th Int. Symp. Spatial Data Handling*, Delft, The Netherlands, 1996.

[2] D. B. Kidner and D. H. Smith, "Data compression for digital elevation models," in *3rd Eur. Conf. Exhibition on Geographical Information*, 1997, pp. 96–105.

[3] G. S. Marcelo, J. Weinberger, and G. Sapiro, "The loco-i lossless image compression algorithm: Principles and standardization into jpeg-ls," *IEEE Trans. Image Processing*, vol. 9, pp. 1309–1324, Aug. 2000.

[4] M. J. Weinberger, G. Seroussi, and G. Sapiro, "LOCO-I: A low complexity, context-based, lossless image compression algorithm," in *Proc. 1996 Data Compression Conf.*, Snowbird, UT, Mar. 1996, pp. 140–149.

[5] M. J. Weinberger, J. Rissanen, and R. Arps, "Applications of universal context modeling to lossless compression of gray-scale images," *IEEE Trans. Image Processing*, vol. 5, pp. 575–586, Apr. 1996.

[6] M. J. Weinberger and G. Seroussi, "Sequential prediction and ranking in universal context modeling and data compression," *IEEE Trans. Inform. Theory*, vol. 43, pp. 1697–1706, Sept. 1997.

[7] S. W. Golomb, "Run-length encodings," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 399–401, 1966.

[8] R. Gallager and D. V. Voorhis, "Optimal source codes for geometrically distributed integer alphabets," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 228–230, Mar. 1975.

[9] N. Merhav, G. Seroussi, and M. J. Weinberger, "Modeling and low-complexity adaptive coding for image prediction residuals," in *Proc. 1996 Int. Conf. Image Processing*, Lausanne, Switzerland, Sept. 1996, pp. 353–356.

[10] N. Merhav, G. Seroussi, and M. J. Weinberger, "Optimal prefix codes for sources with two-sided geometric distributions," *IEEE Trans. Inform. Theory*, vol. 46, pp. 121–135, Jan. 2000.

[11] R. F. Rice, "Some practical universal noiseless coding techniques—Parts I–III," Tech. Rep. JPL-79-22, Jet Propul. Lab., Pasadena, CA, Mar. 1979.

[12] A. Netravali and J. O. Limb, "Picture coding: A review," *Proc. IEEE*, vol. 68, pp. 366–406, 1980.

Shantanu D. Rane received the B.E degree in instrumentation engineering from the Government College of Engineering, Pune University, India, in 1999, and the M.S degree in electrical engineering from the University of Minnesota, Minneapolis, in 2001. He is currently pursuing the Ph.D. degree at Stanford University, Stanford, CA.

He was a Project Student with the National Center for Radio Astrophysics, Tata Institute of Fundamental Research, Pune, India, from 1998 to 1999 and worked on the development of a feed-positioning system for antennas of the Giant Meterwave Radio Telescope.

Mr. Rane received the gold medal for the best performance in undergraduate studies in instrumentation engineering from Pune University in 1999. His research interests include compression and reconstruction schemes for images and video.

Guillermo Sapiro (M'95) was born in Montevideo, Uruguay, on April 3, 1966. He received the B.Sc. (summa cum laude), M.Sc., and Ph.D. degrees from the Department of Electrical Engineering, Technion, Israel Institute of Technology, Haifa, Israel, in 1989, 1991, and 1993, respectively.

He was a Postdoctoral Researcher with the Massachusetts Institute of Technology, Cambridge, and then became Member of Technical Staff at the research facilities of HP Labs, Palo Alto, CA. He is currently with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis. He works on differential geometry and geometric partial differential equations, both in theory and applications in computer vision, computer graphics, medical imaging, and image analysis, and recently co-edited an issue of the *Journal of Visual Communication and Image Representation* on this topic.

Dr. Sapiro was awarded the Gutwirth Scholarship for Special Excellence in Graduate Studies in 1991, the Ollendorff Fellowship for Excellence in Vision and Image Understanding Work in 1992, the Rothschild Fellowship for Postdoctoral Studies in 1993, the Office of Naval Research Young Investigator Award in 1998, the Presidential Early Career Awards for Scientist and Engineers (PECASE) in 1988, and the National Science Foundation Career Award in 1999. He recently co-edited a special issue of IEEE IMAGE PROCESSING.