The Effects of Transfer Encoding on Image Quality

Charles J. Turner and Larry L. Peterson*
Department of Computer Science
University of Arizona
Tucson, AZ 85721

Abstract

In order to transfer digital images over computer networks, it is necessary to encode the pixels that make up the image into a set of network packets. Because the network does not reliably deliver all the packets from the source to the destination, this encoding influences the quality of the image that is delivered to the end user. This paper explores the effects of various encoding algorithms on image quality in the presence of unreliable computer networks.

1 Introduction

In the near future, the scientific community will require the ability to transfer large amounts of digital imagery over computer networks. Earth observing platforms will be collecting data continuously at a rate of several hundred gigabytes per day. Researchers around the world will be performing interdisciplinary, multi-sensor experiments using data from distributed archives. At this collection rate, these users could become one of the largest consumers of wide-area network bandwidth. In order to use this volume of data effectively, it will be necessary to give researchers the fastest possible transfer methods. Further, many researchers would benefit from a transfer service with the ability to predictably trade image quality for improved response-time.

Most wide-area networks provide a base service that is fundamentally unreliable: packets may be lost, delayed, duplicated, or delivered out of order. With considerable expense to the user, higher level protocols retransmit missing packets to provide a reliable, ordered transfer service. In order to improve image transfer response time, we are investigating the use of retransmission-free network protocols. By not retransmitting lost packets, the image transfer protocol can be more economical with respect to latency, thereby improving end-to-end response time.

Instead of retransmitting lost packets, we take advantage of the redundancy inherent in digital images to recover missing information. The key to making this work is to chose an encoding algorithm that maximizes the effectiveness of the recovery algorithm. The encoding algorithm controls the placement of pixels from the image into network packets, and as a consequence, determines the relationship among the pixels contained in any packet lost by the network. Intuitively, an encoding that eliminates the temporal to spatial relationship among packets has the best results.

We have already described various encoding and recovery algorithms and quantified their effect on image transfer time in a companion paper [4]. This paper briefly summarizes that work, and then explores on the effect of various encoding algorithms on image quality. Note that to date most of our experience has been with satellite imagery, although we do have preliminary evidence that the results apply to other types of images, as well.

2 Algorithms

Image transfer happens in three phases: the sending host encodes the image into network packets, the network delivers each packet independently from the sending host to the receiving host, and the receiving host reconstructs the image from the arriving packets. Note that the final phase really consists of two parts: the pixels in each packet that arrives must be extracted and placed in the appropriate location in the image (this is called decoding and is the inverse of the encoding function), and a recovery algorithm must be run to fill in the missing pixels (those that were carried by packets lost in the network). We refer to this second part as the recovery algorithm, and discuss decoding in conjunction with encoding.

Intuitively, our approach is to eliminate the temporal to spatial relationship among packets so that information lost during transmission can be recovered more effectively through some other means than retransmission. This is accomplished by encoding the source images in such a way as to insure that information that is local in “image space” is not local in “packet space”. Ideally, the encoding scheme should guarantee that no two adjacent image pixels will ever be in the same network packet. When a packet is lost, the corresponding lost bytes will be scattered over the full image. This provides the optimal conditions under which to run various recovery algorithms around the missing pixels.

2.1 Underlying Network

We have designed our image transfer protocol for networks like the TCP/IP Internet. In this network, the low-level protocol (IP) provides only best-effort delivery—bandwidth is not reserved, and should a packet forwarder (gateway) become congested, it discards packets without regard for the requirements of the competing connections. Because of this design, it is not uncommon to see significant packet loss rates, on the order of 10% or more when the sender is transmitting aggressively. This loss is due to congestion at the the bottleneck gateway, it is generally not because of a faulty transmission media.

---

*This work supported in part by National Science Foundation Grant IRI-15407 and DARPA Contract DABT63-91-C-0030.

Furthermore, when packet loss does occur, it is often bursty in nature—two or more back-to-back packets are lost. For example, in experiments on the TCP/IP Internet, we measured an average burst loss size of 3.3 packets, with 12% of the bursts being four or more packets. This burst size corresponds to an overall loss rate of 2–4%.

TCP is a high-level protocol that uses retransmission to turn IP's unreliable service into a reliable, ordered delivery service. We are not using TCP in our approach.

2.2 Recovery Algorithms

The key to error recovery techniques is the local redundancy found in most types of imagery. Any given pixel of data in an image is frequently correlated with its surrounding neighbors. This is because surface brightness tends to change gradually in most natural scenes, and many of the processes involved in image formation have a smoothing effect. Additionally, images like maps and documents contain large uniformly colored regions. These local similarities can be used to reconstruct or approximate the value of each missing pixel.

The reconstructed value for the lost pixel can be obtained in several ways from the surrounding valid values. Because this operation must be performed during image transmission, it is desirable to make it as fast as possible. Like many other parts of an end-to-end system, the right recovery method is a function of required image quality and execution speed requirements. The user has a range of recovery options ordered by increasing accuracy and processing requirements [1, 3]. In some cases it is acceptable to do no recovery, or merely replace missing pixels with precomputed global averages. A variety of interpolation options are more expensive but yield better results. Finally, techniques like median or morphological filtering can be used effectively on certain classes of imagery. The reader is referred to [4] for more details on recovery algorithms.

2.3 Encoding Algorithms

The success of most recovery algorithms is determined by the amount of information available for recovery around each lost pixel. The encoding algorithm used prior to image transmission determines the manner in which lost data is distributed throughout the receiver's image. If pixels associated with a given packet can be scattered uniformly over the image in both horizontal and vertical dimensions, then for any single lost packet, no two pixels will be adjacent in the reconstructed image. Consider three possible encodings.

First, a *raster-based* encoding assigns a complete row (column) to each network packet. This is the encoding typically used when retransmission is used to recover missing information. From a computational perspective, this is the ideal encoding—pixels are copied sequentially into and out of network packets. From a quality perspective, however, the loss of a single packet means that an entire row of the image is missing. While correction algorithms exist for this type of loss—using adjacent rows to perform interpolation—in general they have poorer performance than algorithms which have access to a complete surrounding neighborhood. When a full row is lost, each pixel's surrounding information is reduced by 25% of that available to an isolated pixel. More importantly, the loss of multiple adjacent packets—e.g., due to a burst error or network congestion—causes row-based interpolation performance to degrade rapidly.

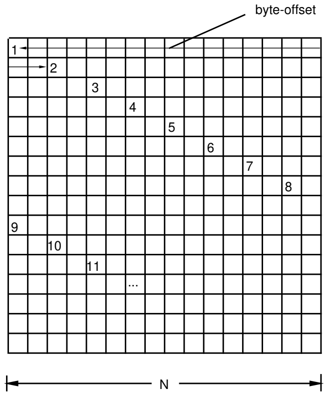


Figure 1: Regular-Scattered Encoding

Second, a *random-scattered* encoding assigns pixels from the image randomly over the network packets. Computationally, a random-scattered encoding is relatively expensive—the location of each pixel must be determined individually. On the other hand, errors that remain after lost pixel recovery manifest themselves as unstructured, high-frequency noise. Since this noise does not interact with image structure, it is considered best by image analysts.

Third, a *regular-scattered* encoding assigns adjacent pixels to non-adjacent packets, but uses a constant offset rather than making the assignment randomly. This approach has the advantage of making it easier to decode the packet at the receiving host, while having the potential to retain the non-locality advantages of the random-scattered encoding.

Specifically, given an image of size RowLength x ColLength pixels, the regular-scattered encoding is expressed using two key parameters: ByteOffset and PacketOffset. ByteOffset controls how bytes within a single packet are distributed throughout the image; it specifies the number of bytes between adjacent pixels in the same packet. It is this parameter that insures that no two bytes within the same packet are ever adjacent in image space. Figure 1 shows a packet constructed using ByteOffset = RowLength + 2, which we have found to be the best with respect to compression.[1] PacketOffset addresses the problem of bursty packet loss; it specifies the number of bytes between starting pixels of adjacent packets. This parameter controls how far apart temporally adjacent packets are in image space.

3 Experimental Results

This section presents the results of three different quality measures for three different image encoding algorithms under a variety of packet loss conditions. We have tested our system on samples

---

[1] The relationship between the encoding algorithms and data compression is discussed in [4].

from four different types of digital imagery, and report the average measurements for five sample images of each type:

MSS: Landsat Multispectral Scanner.

TM: Landsat Thematic Mapper.

SPOT: French SPOT Satellite.

EO: Underwater Electro-Optical Camera.

The experiments assume a burst-oriented packet loss, in which a variable percentage of packets are dropped by the network. Packets are dropped in randomly located groups uniformly distributed with a normally distributed group size, with a mean of 8 and variance of 1 packet. Then, a neighborhood-based linear interpolation is run on each image to recover the values of the pixels in the lost packets.

The difficulty in quantifying image quality is that the quality of an image depends on how the image is being used. For example, someone doing rough analysis or simple counting of obvious features has significantly lower quality requirements than someone else who is doing mensuration or very detailed analysis of structures within the image. Thus, to demonstrate the effects of transfer encoding on reconstructed image quality, we measured quality in three very different ways: (1) global RMS error, (2) power spectra of difference images, and (3) application performance.

3.1 Global Error

A commonly used technique for measuring image quality is the root mean squared error (RMS), which measures the average absolute error between the original image and the corrected image. An RMS error of 0 indicates perfect recovery. Table 1 shows the average RMS error for five 20 meter SPOT images with packet loss rates of 1, 5, 10 and 20 percent. The rows of the table are labeled with the three different encoding algorithms examined. This test was run on all image types, with results similar to those presented for SPOT.

	1%	5%	10%	20%
Raster	1.3	3.3	4.8	9.6
Regular	0.7	1.6	2.4	3.6
Random	0.5	1.5	2.3	3.6

Table 1: RMS Error

As one would expect, the global error increases as the percentage of lost packets increases. Comparing the algorithms to each other, it is clear that global reconstruction success improves as the amount of local information available for reconstruction increases. When using a burst-oriented packet loss model, raster-based encoded images suffer the most locally concentrated loss. With an average burst size of 8 packets, there is a significant decrease in recovery success towards the center of the burst. The interpolation algorithms used are forced to propagate values in from the closest available valid pixels, which are on the edges of the burst. The greater this distance, the large the resulting error. The other two algorithms, regular-scatter and random-scatter, both attempt to minimize the locality of pixel loss by distributing the loss over the full image. Consequently, there is more information available surrounding each lost pixel for use in recovery, leading to a lower global error.

3.2 Spatial Error

Another important consideration in the selection of an encoding algorithm is the effect on structure or features within the image. Clarity of contour lines and boundaries, pattern and shading reproduction, elimination or degradation of important details, and the introduction of invalid texture or granularity are all important considerations in the analysis of reconstruction quality. In general, image analysts find that uncorrelated, random errors are less harmful than regular correlated patterns which interact strongly with existing image structure or introduce new structure.

All three encoding schemes use some form of interpolation from neighboring valid pixels. This has the effect of decreasing some of the high frequency information in the final image. The three graphs given in Figure 2 illustrate some of the differences the three encodings have on the recovered image. Each graph depicts the power spectra of difference images at two different packet loss levels (1% and 20%). Each line represents a smoothed vertical slice through a two-dimensional power spectrum in a region just outside the central peak of the spectrum. The patterns reflected in the graphs are representative of other regions of the spectra.

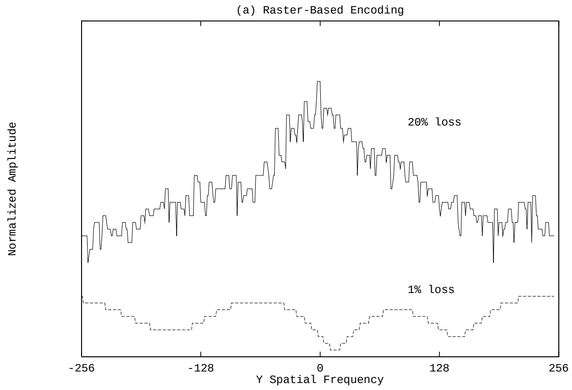
Several characteristics of the encoding algorithms can be seen in the graphs. First, consider the performance of the raster-based encoding given in (a). A one percent packet loss frequently manifests itself as a single burst. Consequently, the loss comes from a single block of data where the error increases smoothly towards the center for the block resulting in a smooth, fairly uniform distribution of energy. As the percentage of lost packets increases, there tends to be an increase in the amount of energy lost in the lower frequencies because the blocks become more frequent and begin to fall in adjacent areas, creating even larger blocks.

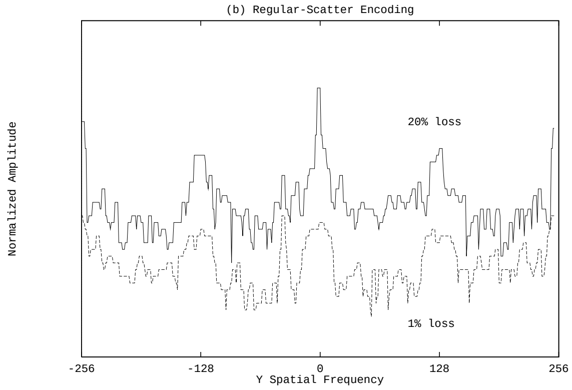
Second, graph (b) shows the same slice of the power spectra for regular-scatter encoded images at two different loss rates. Notice the several broad peaks in both loss curves. These are caused by correlated patterns of error introduced by regular-scatter encoding. While it is true that each pixel in a regular-scatter encoded packet will not be spatially adjacent, there is a definite pattern in the traversal across the image. Each packet that is lost introduces some error in the pixels along one or more lines across the image. The angle the packets form within the image is a constant determined by the ByteOffset parameter. As the number of lost packets increases, the same linear error pattern is introduced at random locations. The regularity of this introduced error can be seen in the spikes.

Third, graph (c) shows the same slice of the power spectra for two random-scatter encoded images. In this case, there is a relatively uniform error across all frequencies. One feature of both spectra that is not clearly demonstrated due to the smoothing is the amount of low energy frequencies intermixed with the frequencies illustrated. The relatively uniform distribution of errors accurately captures uncorrelated nature of the random encoding.

3.3 Application Error

One potential use of image transfer protocols is to deliver images to large application programs. An interesting question, then, is if these applications will produce the same results with an image that has been degraded by transfer, as it would with a perfect


(a) Raster-Based Encoding


(b) Regular-Scatter Encoding

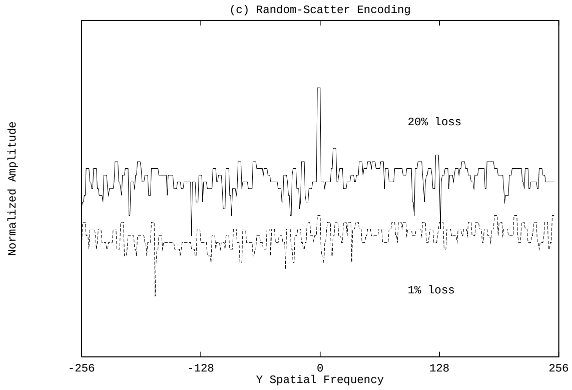

(c) Random-Scatter Encoding

Figure 2: Power Spectra of Difference Images

image. We would expect the answer to be yes, since applications of this type already operate under a variety of conditions and are designed to handle systematic error in their inputs, which is a natural part of most remote sensing systems. To verify this intuition, this section reports on the performance of three image processing applications.

First, consider a multispectral *k*-means classification algorithm, which is based on the standard isodata clustering algorithm [2]. In measuring the performance of this application, we ran the system on samples of 20m multispectral SPOT imagery of a complex urban area. Such areas are believed to provide a more difficult test for our encoding algorithms due to the highly variable nature of such scenes.

To measure this application's performance, we first classified the original images into ten classes and saved the class maps. We then ran the same test on the twelve different combinations of encoding and packet loss rate. Performance was measured by measuring the difference between the resulting class maps and those generated from the original images. The results are presented in Table 2.

	1%	5%	10%	20%
Raster	99.0	96.8	95.7	94.0
Regular	99.2	98.0	97.1	96.0
Random	99.2	98.0	97.3	96.1

Table 2: *k*-Means Classification Performance

The results are quite similar across the three encoding schemes, with raster-based encoding performing slightly worse than the other two. In all three cases, there is a gradual degradation in performance as the number of lost packets increases. Overall, performance is good because the averaging process that occurs when clusters of pixels are given a single class label compensates for minor changes in individual members of the cluster. Those errors that do occur are caused by boundary pixels which change enough to move from one class to another.

Second, consider a road finding application, which relies heavily on edge detection. This application was developed as part of an urban morphology project that is studying the movement of different cultural influences from Europe to North and South America. One of the indicators of cultural influence is town and city organization and street layout. This application attempts to automate the measurement of key properties of city road networks in order to determine the influences present during early town planning. To accomplish this, a linear road finding algorithm was developed. It was run on individual bands of 20 meter SPOT multispectral images of a cultural area (Tucson, AZ).

The first step in the processing sequence is to apply a number of directional gradient operators to the image. Candidate edge pixels are then run through a Hough transform to locate potential segment areas. The output of the Hough transform is the top N sets of pixels that could all be part of actual edges in the image. As a final step, an edge continuity algorithm is applied to all pixels within each candidate set. Chains of pixels that are all within a minimum distance of each other, fall on the same line, and have an overall length greater than some minimum segment length are classified as valid road segments.

We ran this application on the twelve combinations of encoding and packet loss rates. The results presented in Table 3 indicate the number of correct road segments found by the application after encoding, loss, and recovery relative to those found when using the original imagery. All parameters and thresholds within the system were held constant during the tests.

	1%	5%	10%	20%
Raster	80.9	66.7	52.4	47.6
Regular	100.0	95.2	90.5	85.7
Random	95.2	95.2	90.5	66.7

Table 3: Road-Finder Performance

As with the previous test, there is a decrease in performance as the percentage of lost packets increases. In this test, the raster-encoded imagery resulted in noticeably lower performance. This can be explained by the spatially located nature of errors when a block of raster-encoded packets are lost. A single block of lost packets cuts a narrow swath across the image, interacting with all the edges which cross this area. Thus, if this burst is large enough or happens to fall near another burst, it could cause certain edges to break apart and consequently be removed by continuity or length thresholding operations. The two scatter-based encodings have about the same level of performance for this application.

Third, consider a starfish detection and classification application that was developed to improve the timeliness and accuracy of environmental impact studies. The imagery used in this test was obtained from an underwater camera mounted on a autonomous submarine traveling at a constant depth from the bottom and attempting to maintain a uniform lighting while pictures were taken. The complete process of detecting and classifying starfish with a high degree of accuracy under a variety of image conditions is quite computationally intensive. Consequently, the overall approach is a coarse-to-fine approach in which a series of algorithms of increasing complexity is applied to successively smaller regions of the image.

As a first step, key statistics of the image are computed (mean, variance) to set certain parameters in the initial screening phase. Next, a locally adaptive bright pixel detector is run over the whole image, identifying all pixels that are significantly brighter than their surrounding neighbors. This is followed by a dilation/erosion step used to fill in holes and remove isolated pixels. This step is followed by a pass in which candidate starfish are identified by their center disks using standard correlation techniques. Those regions of the image that are selected as candidate starfish disks become the starting locations for the more expensive identification of legs. The leg finding algorithm is a model-based graph algorithm that attempts to traverse the gradient image starting from the candidate disk and working out radially, relaxing its directional constraints the farther away from the disk it gets. The final step in the process is a classification step which considers the number and lengths of legs and disk size to determine whether a match has been found.

Table 4 contains the results of the twelve combinations of encoding and packet loss rates run through the starfish identification program with all parameters and thresholds left unchanged. Overall, this application behaves better under degraded image conditions than the previous two applications. This is probably because the original image quality was such that the algorithms needed to process the imagery accurately have always been required to be more robust in order to work successfully. To the extent that the results degrade at all with increased packet loss, the results are consistent with the previous two tests: performance deteriorates with increased packet loss, and the two scatter encoding algorithms perform better than the raster-encoding.

	1%	5%	10%	20%
Raster	100.0	100.0	91.7	83.3
Regular	100.0	100.0	100.0	100.0
Random	100.0	100.0	100.0	91.7

Table 4: Starfish Classification Performance

In summary, some general statements can be made about the relative performance of three encoding algorithms themselves. In all three cases, the two techniques that distribute loss in some way performed better than the raster-based encoding scheme. Comparing the two scattering algorithms is more difficult. With a few exceptions, they have almost identical performance behavior when measured at this high level. This is ultimately promising because it indicates a potential robustness to changes in encoding. This allows much more flexibility in the design of encoding and transmission algorithms.

4 Conclusions

We demonstrate in a companion paper [4] that the end-to-end performance of an image transfer protocol can be improved by not retransmitting packets dropped by the network. To compensate for lost packets, we run various recovery algorithms on the transmitted images. The effectiveness of these algorithms depends on how the image was encoded into the network packets. This paper quantifies the impact these encodings have on the quality of the reconstructed images. The data supports the conclusions that (1) encodings that scatter adjacent pixels across non-adjacent packets results in higher quality images, (2) quality degrades gracefully with increased packet loss, (3) the results of the (inexpensive) global RMS error are consistent with the results of the other more detailed (expensive) measures, and (4) applications can be designed to tolerate significant packet loss.

Given that there will be an increased use of networks in image distribution in the near future, it is important that the image processing community become aware of the quality/response-time tradeoff available to them.

References

[1] R. H. T. Bates and M. J. McDonnell. *Image Restoration and Reconstruction*. Clarendon Press, Oxford England, 1986.

[2] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York, 1973.

[3] W. K. Pratt. *Digital Image Processing*. John Wiley and Sons, New York, 1978.

[4] C. J. Turner and L. L. Peterson. Image transfer: An end-to-end design. In *Proceedings of the SIGCOMM '92 Symposium*, Baltimore, Maryland, Aug. 1992.