Image Statistics Decoding for Convolutional Codes

G. H. Pitt, III, and L. Swanson
Communications Systems Research Section

J. H. Yuen
Telecommunications Science and Engineering Division

It is a fact that adjacent pixels in a Voyager image are very similar in grey level. This fact can be used in conjunction with the Maximum-Likelihood Convolutional Decoder (MCD) to decrease the error rate when decoding a picture from Voyager. Implementing this idea would require no changes in the Voyager spacecraft and could be used as a backup to the current system without too much expenditure, so the feasibility of it and the possible gains for Voyager were investigated. Simulations have shown that the gain could be as much as 2 dB at certain error rates, and experiments with real data have inspired new ideas on ways to get the most information possible out of the received symbol stream.

I. Introduction

After seeing an image decoded by the MCD recently, it was noticed that many errors in a decoded Voyager image could be detected by eye since pixels in error differ markedly from the image (Fig. 1). It was proposed that an expert look at an image, identify errors, and declare them as erasures for the Reed-Solomon decoder after the MCD. In this way, the Signal-to-Noise Ratio (SNR) could be lowered a bit, allowing even more errors to occur without damaging the chances of receiving a perfect image in the end.

The important facts are as follows:

(1) Many errors differed markedly from the image.

(2) The image differs only slightly from itself. This means that pixels that are near each other are usually very similar in value, and ones that are not similar are usually errors.

The Image Statistics Decoder (ISD) that we developed uses fact (2), but inside the convolutional decoder instead of after it, and actually *avoids* many of those errors noticed above. Therefore, in all results so far, the ISD performs better than even the best possible results using the MCD and an expert to declare erasures. In the following, more precise statements of fact (2) will be given in graphs, and the problems faced, the genesis of solutions, and results will be outlined.

II. First Results of Programming the Image Statistics Decoder

A. Feasibility

The first thing to check was how correct fact (2) actually is; how close are adjacent pixels in a Voyager image, and is this closeness fairly constant from image to image? Constant statistics from one image to the next are necessary since we need to estimate a priori the statistics for each picture before decoding. With b_j defined as the grey level of the j th received pixel in a stream of pixels, the statistic that we decided to use at the beginning of this experiment was $D_j = |b_j - b_{j-1}|$, the absolute difference between the grey levels of two adjacent pixels (pixel values on Voyager range from 0 to 255). Preliminary results using this statistic were reported earlier [1], and some will be repeated here. We computed the theoretical distribution of D assuming independence between adjacent pixels (the MCD assumes independence), and plotted it against the distribution observed in several images. The results may be seen in the chart (Fig. 2).

Three different Voyager images were used for this demonstration: one of dark sky with Titan in the foreground, one of Saturn seen fully in the frame, and one of the rings of Saturn with the planet in the background. These three images seem completely different, but the distributions of D are quite similar for them when compared to the independent case, and quite different from the independent case. Therefore, statistics from any one of these images will be a much better model for another image's statistics than the independent case would be.

B. Theory

Decoders work on a simple principle: find the information bytes that were most likely sent, given that you received symbols $\{r_1, r_2, \ldots, r_m\}$. More precisely, if we let B be the event that the information sent was a set of bytes $\{b_1, b_2, \ldots, b_n\}$ and R the event that the received symbols[1] were $\{r_1, r_2, \ldots, r_m\}$, then we want to find the information bytes $\{b_1, b_2, \ldots, b_n\}$ that maximize the probability of B given R. In mathematical notation, we want to maximize $P(B|R)$ over all possible information sets $\{b_1, b_2, \ldots, b_n\}$. From probability theory,

$$P(B|R) = \frac{P(B,R)}{P(R)} = \frac{P(R|B)P(B)}{P(R)}$$

where $P(B,R)$ is the probability that B and R both happened.

Let $\{s_1, s_2, \ldots, s_m\}$ be the convolutionally encoded symbol stream associated with the information bytes $\{b_1, b_2, \ldots, b_n\}$ and S be the event that $\{s_1, s_2, \ldots, s_m\}$ was sent. Then S and B are equivalent events.

$$P(B|R) = \frac{P(R|S)P(B)}{P(R)} \tag{1}$$

Over a memoryless Gaussian channel[2],

$$P(R|S) = K \exp\left[-\Sigma(r_i - s_i)^2/\sigma^2\right]$$

where K is a constant, r_i and s_i are described above, and σ^2 is the SNR. Therefore,

$$P(B|R) = K \exp\left[-\Sigma(r_i - s_i)^2/\sigma^2\right] \frac{P(B)}{P(R)}$$

R and K are fixed, so $P(R)$ and K don't enter into the maximization. Therefore, we have reduced the problem of maximizing $P(B|R)$ over a Gaussian channel to that of maximizing

$$\exp\left[-\Sigma(r_i - s_i)^2/\sigma^2\right] P(B)$$

We may take the natural log, multiply by -1 and equivalently minimize

$$\sum_i \frac{(r_i - s_i)^2}{\sigma^2} - \ln(P(B)) \tag{2}$$

This is the point where the interdependence of the information bytes makes a difference. If they are completely random and independent, then $P(B)$ is the same for every possible set of information bytes. Thus, $\ln(P(B))$ is constant, and we just have to minimize

$$\sum_i \frac{(r_i - s_i)^2}{\sigma^2} = \frac{1}{\sigma^2} \sum_i (r_i - s_i)^2$$

The SNR, determined by σ^2, remains constant as we vary $\{b_1, b_2, \ldots, b_n\}$, so it may also be removed, leaving

$$\sum_i (r_i - s_i)^2 \tag{3}$$

The MCD minimizes expression (3) over all possible sets $\{s_1, s_2, \ldots, s_m\}$. However, as shown in Fig. 2, the information bytes are not independent, so $P(B)$ is not constant and should be used in decoding; we must minimize expression (2). Using the rules of probability again

$$P(B) = P(b_1) \prod_j P(b_j | b_{j-1}, b_{j-2}, \ldots, b_1)$$

[1]The actual received symbol stream is $\{r_1, r_2, \ldots, r_m\}$, so R is an event that actually happened. This is why we are given that R happened.

[2]The deep space link to Voyager is a memoryless Gaussian channel.

Trying to observe the distribution of $(b_j | b_{j-1}, b_{j-2}, \ldots, b_1)$ would be impossible, but it may be possible to approximate it using an almost sufficient statistic like D described above.[3] Assuming that D will be used as a sufficient statistic, we may write

$$P(B) \approx P(b_1) \prod_j P(b_j | D_j) \qquad (4)$$

where D_j is not hard to calculate, and the distribution of $(b_j | D_j)$ is known (or at least approximated). Therefore we must minimize

$$\sum_i \frac{(r_i - s_i)^2}{\sigma^2} - \ln (P(b_1) \prod_j P(b_j | D_j))$$

$$= \sum_i \frac{(r_i - s_i)^2}{\sigma^2} - \ln (P(b_1)) - \sum_j \ln (P(b_j | D_j))$$

Assuming that $P(b_1)$ is constant[4], we must minimize

$$= \sum_i \frac{(r_i - s_i)^2}{\sigma^2} - \sum_j \ln (P(b_j | D_j)) \qquad (5)$$

This is exactly what the original ISD (Image Statistics Decoder) did in the case where $D_j = |X_{j+1} - X_j|$. The SNR is σ^2, so the SNR must be known or estimated to use the ISD.

C. Programming

The MCD computes the metric associated with the addition of a single bit at a time, requiring each possible state (determined by 6 bits in the Voyager code) to be checked for two possible previous states: those associated with a 1 and a 0 (Fig. 3). The ISD must compute the metric associated with the addition of a whole byte at a time, requiring each state (now determined by 8 bits) to be checked for all 256 possible previous states (Fig. 4). In addition to the regular metric, a special metric, consisting of $\ln (P(b_j | D_j))$, must now be added into each state. The number of computations and comparisons necessary to decode a byte with the ISD has risen by a factor of over 64 over the MCD.

[3]The first statistic used was D. Later D and M (described below) were used.

[4]That is, assuming that without any prior information, all pixels are equally likely.

Within a few weeks of starting to modify an existing software simulated Viterbi decoder (written by Fabrizio Pollara, Communications Systems Research Section), a working model was finished, but with a major drawback: A picture would be decoded with this software in about 8 months on a dedicated VAX 11/750 computer. This compares with about 7 hours for the existing software simulated Viterbi decoder, and about 20 seconds for the MCD. A decoder as slow as the first ISD was useless even for simulation work, since it would take months just to plot enough points on the error rate curve, especially if it improved these error rates.

Over the remainder of this project, an attempt has been made to increase the speed of the decoder by using all of the available knowledge about its structure and by using different computers. The decoder speed has been increased considerably, so it is now fast enough to obtain some results (Table 1).

III. Simulated Error Rate Results

Figure 5 shows error rates for a normal MCD and for the first ISD used on simulated data over a range of SNRs of interest. Notice that it has a much better error rate for very low SNR, but this rate does not fall off very quickly as the SNR is raised. Theoretically, a decoder should do no worse with the pixel statistics than without them, so we sought an explanation for the poor performance at high SNRs. Even so, there is a gain of about 2.0 dB for a byte error rate of 0.028, and of about 1.0 dB for 0.016. These two error rates are of interest in deep space communications since they translate to failure rates of 10^{-3} and 10^{-6}, respectively, for the Reed-Solomon decoder.[5]

A. The Simulation

Fabrizio Pollara simulated the current MCD in software, thus generating the MCD error rates shown above. The simulation used data from a random number generator and simulated noise, also from a random number generator. We modified these programs to read actual pixel data from a file for decoding, but used the same random noise generator. The modules of the program that generated the input data and the MCD were the only ones that we changed so that the comparisons would be as close as possible.

Two images were necessary to complete the simulation: one image to generate the statistics to use, and one to decode. Using the same image for both purposes would be unrealistic: It would assume that we knew the image's exact statistics

[5]A Reed-Solomon code is used as an outer code on some Voyager data.

before decoding. Therefore, a single Voyager image has been used throughout the simulation for sample statistics: the one of Saturn fully in the frame[6] (Fig. 6). This image was chosen because Fig. 2 shows that its statistics are closer to random than the other two.

B. A Possible Explanation for the Error Rates

Tools which have been very useful to this project have been a Macintosh and LaserWriter. The final version of the ISD was developed on the Macintosh and images were printed on the LaserWriter. They allowed images to be viewed at high resolution for the first time, making the following problems very obvious (Fig. 7).

A typical Voyager image contains several features which decrease the performance of the ISD. The most prominent of these are limbs and rizzo marks.

1. **Limb streaking.** Most of a Voyager image is dark sky, but the features of interest are usually the planets and moons, which are not dark, so many Voyager images contain a portion that is much more bright than the background. This creates a limb, the transition from black to white within the span of a few pixels, and then back again. Over the whole image, there are only a few transitions this severe, so this is a rare case according to the statistics. Since the ISD relies on the fact that adjacent pixels are *usually* close in value, it tries to suppress large transitions, and thus causes streaks of errors propagating along a row of the image when a true transition is very large.

2. **Rizzo marks.** Rizzo marks are the grid of dots put on a Voyager image by the spacecraft for calibration. Within a few pixels the grey level may change from very bright to completely black and back again, creating even more large transitions in an image.

The ISD propagates pixel values along a row of an image, one error causing more after it, so that streaks of pixels in error may be seen in any edge[7] of the decoded image.

IV. Results With Real Data

With help from Hamil Cooper of JPL's Radio Frequency and Microwave Subsystems Section, we acquired some real Voyager symbol streams[8] from the Uranus encounter and decoded them with both the MCD and the ISD.

A. A Fix for a Problem

A problem with the original ISD was that it assumed that the statistics for an image were constant over the whole image, which is not true. Therefore, some method of determining which part of an image is being decoded (an edge, a planet, or black sky) had to be devised.

Just as grey level transitions from one pixel to the next can be predicted in the *horizontal* direction, they can also be predicted in the *vertical* direction (Fig. 8). Therefore, the rows above a given pixel (which have already been decoded) can be used in conjunction with the pixel to its left to predict its grey level.

Three possible schemes were suggested.

1. **Averaging grey levels.** The grey levels of the pixels above and to the left of the current pixel could be averaged before comparing those levels to the current pixel. In a preliminary experiment, results were disastrous, possibly because errors propagated not only to the right, but down also. The resulting image had massive areas of errors, and not much of the original image left.

2. **Regressing edges.** The location of an edge could be determined for a few rows previously decoded, and then the edge on the next row could be predicted using a statistical regression technique. The problem with regressing edges is that only true edges would be detected. Rizzo marks and small features like craters in the middle of a planet would not be detected since they only cover a few rows of an image.

3. **Dynamically changing statistics.** A few pixels directly above the current pixel could be observed to determine whether this is a high transition area, and the statistics could be changed accordingly. The proposed method was to use the difference between the maximum and the minimum of the five pixels directly above the current pixel as an added statistic called M (Fig. 9). The value of M would be large around an edge, medium sized in the middle of a planet, and very small in black sky. The image statistics could then be changed accordingly by observing similar statistics in the Saturn image used above.

After examining the distributions of D when controlled for M (Fig. 10), we decided that this last scheme would be the best

[6]Actually, this image was used as a base for the statistics; the actual statistics used were the result of smoothing the distributions of D observed in this image.

[7]An edge may be caused by a limb, rizzo mark, or even a crater.

[8]That is, Voyager data not yet decoded by the MCD.

to implement, and wrote a 2-way decoder that uses both M and D as sufficient statistics. The decoded images with the 2-way statistics had about one third of the errors that the original ISD had. Areas in planets were still problems for the decoder, but at a greatly reduced error rate; the errors at edges and rizzo marks completely disappeared except on the first decoded row (since there was no previous row to base statistics upon). Perhaps an adjustment of the statistics would be helpful in reducing the rate even more when decoding high pixel-transition areas like planet surfaces.

The actual implementation of the 2-way ISD should be explained briefly. Using the notation of expression (5) above, the 2-way ISD minimizes

$$\sum_i \frac{(r_i - s_i)^2}{\sigma^2} - \sum_j \ln (P(B_j | D_j, M_j)) \qquad (6)$$

Changing image statistics for each of the possible values of M described above would require a 256 by 256 array of transition probability cells, creating a problem in determining what should go in each cell of the array.[9] Therefore, we condensed the values of M into 8 groups, requiring only a 256 by 8 array, but perhaps decreasing the performance of the decoder.

B. Improved Error Rate

As noted before, the ISD is very slow, and now requires much programmer and computer time to decode an image. Only a part of one image, totaling 147 rows of 800 pixels each, has so far been decoded and compared to the image obtained from Glenn Garneau of JPL's Image Processing Applications and Development Section. The part of the image decoded has Miranda covering about 1/4 of the total area, and black sky over the rest. Figure 11 is a fraction of the image decoded by the ISD, zoomed to contain only Miranda.

The software MCD made 176 byte errors over the $147 \times 800 = 117{,}600$ pixels, an observed error rate of $176/117{,}600 = 0.0015$. The 2-way ISD made only 68 byte errors over the same area for an observed error rate of $68/117{,}600 = 0.0006$ at an estimated E_b/N_0 of about 3.0 dB (Fig. 12).

The results are interesting because the original ISD performed worse than the MCD at SNRs this high in simulations. The improvement at low SNR for the 2-way ISD with real data may be even better than that demonstrated in Fig. 6.

[9]Significant results can not be determined for an array of this size with only the 640,000 pixels in a single picture.

Two things should now be mentioned about this particular implementation of the ISD. First the symbols on the tapes acquired during the Uranus flyby are 4-bit quantized instead of real numbers, not necessarily the correct levels for a decoder, so there is some loss in performance for the ISD due to this. The MCD in the DSN uses optimum 3-bit quantization, which represents a very small loss; we don't yet know what the ISD loses, but we assume it is also small, based on past experience. Second, the ISD must know the SNR in order to run, so a module to estimate it was added. This module was written originally in FORTRAN by Gene Rodemich and Vic Vilnrotter of the Communications Systems Research Section, and translated to C in this project.

All of the data obtained is in PB8 format, with IM2 format imbedded in it[10]. The ISD as implemented assumes that this is the format of the input data; it will need to be modified to accommodate any other format.

V. Usefulness

A. Low SNR During Neptune Encounter

During Neptune encounter, the SNR could drop to a level where the loss of a fraction of a dB could make an image undecodable to the Reed-Solomon decoder; such a loss may be caused by a cloud passing by the receiver at Goldstone. To help boost the SNR, the VLA in New Mexico will be arrayed with Goldstone to approximately double the SNR. A problem with this strategy is that the VLA drops out for 1.6 milliseconds every 52 milliseconds to find out what time it is [2]. During this dropout, only Goldstone will be receiving, so the SNR will drop to about half its value when both are receiving. Approximately 4 bytes could be lost during this dropout, which could again mean the loss of whole images to the Reed-Solomon decoder depending on the SNR.

The ISD is ideally suited to solving the problem encountered by arraying with the VLA. It can decode even when the SNR is varying over time without significant loss in performance, as shown in Fig. 5. As long as some signal gets through, the ISD can do a better job than the MCD for these very low SNRs, perhaps up to 2 dB better. The speed problem may even be overcome somewhat by using the ISD only in the area around the gap in the SNR, and the MCD over the rest. The ISD would only be on over about 4% of the image, so an image

[10]J. Morecroft, *Voyager Flight Data Subsystem Flight Software Description*, 618–236, Rev. A (internal document), Jet Propulsion Laboratory, Pasadena, Calif., August 29, 1980.

could be decoded in about 3 hours on a Sun workstation. Future work planned for the ISD includes development of a decoder that can handle a varying SNR, and tests on data simulated to behave as the Goldstone–VLA array will.

B. Estimated Speed on Various Machines

The ISD is written in C, so it is very portable with only minor modifications. It was mostly developed on a Macintosh with a Hyperdrive 2000, but was ported to several UNIX-based computers with only very minor changes. Therefore the speed on the different computers in Table 1 are very comparable.

VI. The Future

A simulation of the Goldstone–VLA array and generation of more complete error rate curves for the 2-way ISD are the first priorities. Experiments are needed on the effect of adjusting the statistics used, with the goal of reducing the number of errors in high pixel-transition areas like planet surfaces.

References

[1] G. H. Pitt and L. Swanson, "Decoding convolutionally encoding images," *The Telecommunications and Data Acquisition Progress Report 42-83*, vol. July–Sept. 1985, pp. 34–38, Jet Propulsion Laboratory, Pasadena, Calif., November 15, 1985.

[2] L. J. Deutsch, "An update on the use of the VLA for telemetry reception," *The Telecommunications and Data Acquisition Progress Report 42-72*, vol. Oct.–Dec. 1982, pp. 51–60, Jet Propulsion Laboratory, Pasadena, Calif., February 15, 1983.

Table 1. Estimated time to decode one image on several different computers

Computer	Cost, $	Estimated Time to Decode One Image (640,000 pixels)
Macintosh	2,000	6 weeks
Dedicated VAX 11/750	100,000	3 weeks
Macintosh with a Hyperdrive 2000	4,000	2 weeks
Sun 3/260	30,000	3 days

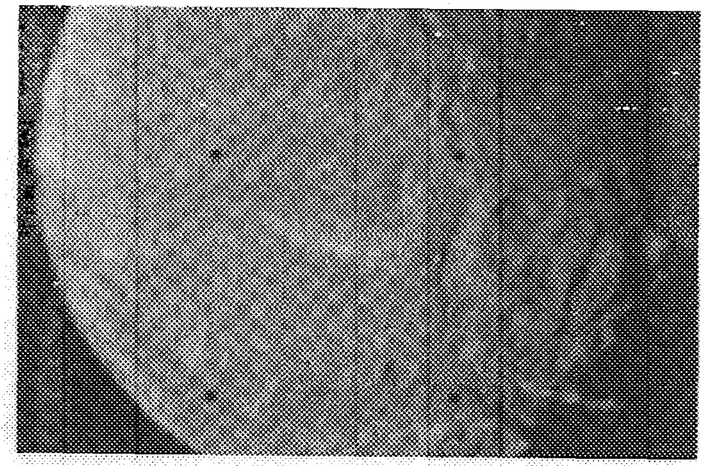

Fig. 1. Part of an image decoded by the MCD

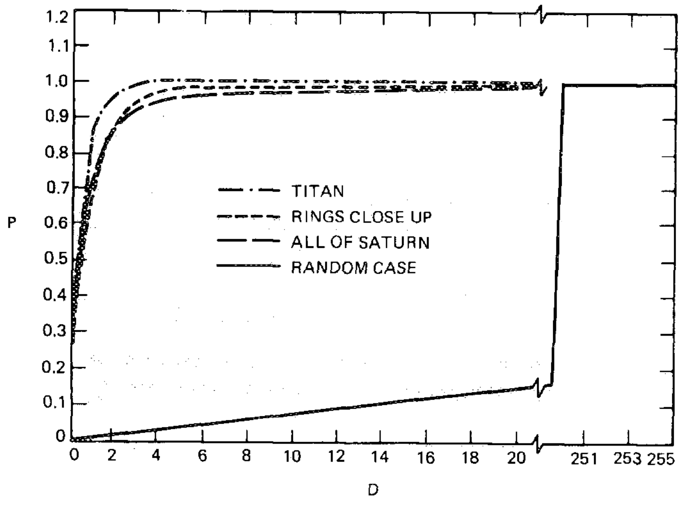

Fig. 2. Distributions of *D* for 3 different Voyager images and the random case

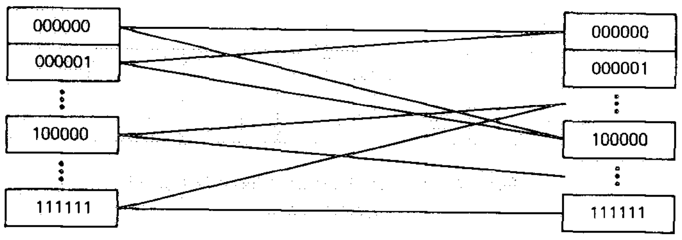

Fig. 3. The state transitions the MCD must check

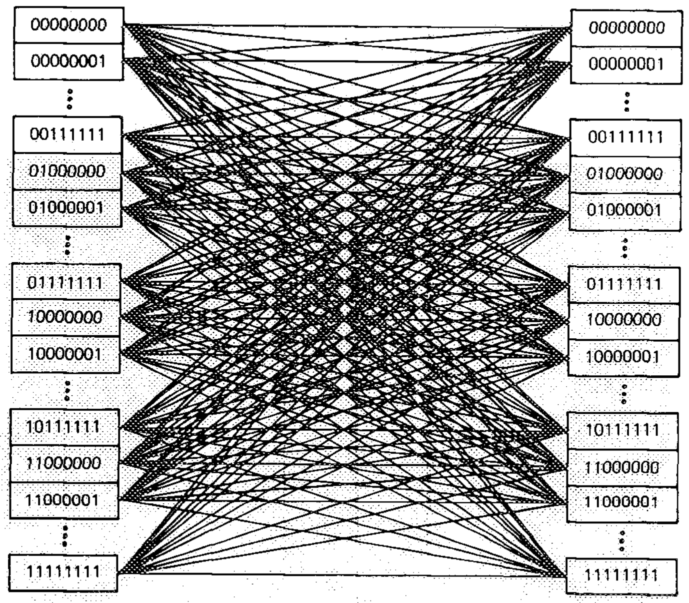

Fig. 4. The state transitions the ISD must check

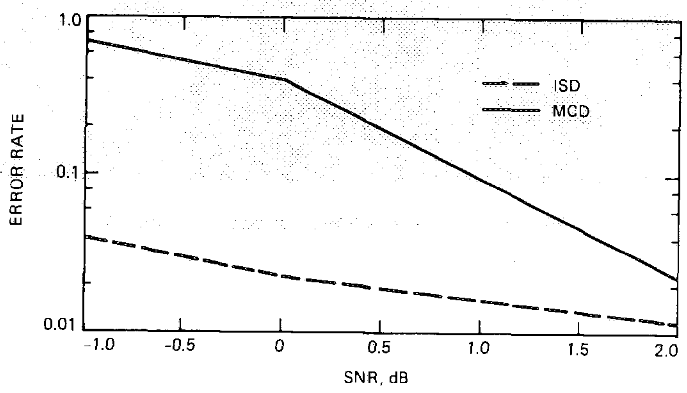

Fig. 5. Byte error rates for the MCD and the ISD

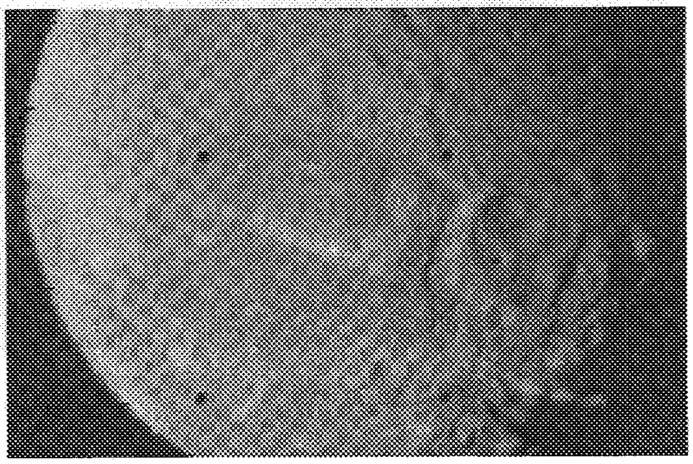

Fig. 7. Same as Fig. 1, decoded by the first ISD

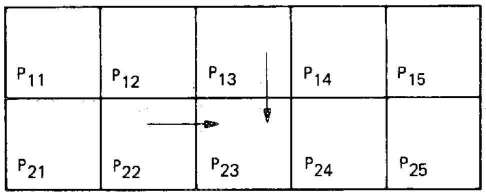

Fig. 8. Two rows of pixels, with arrows indicating which pixels may be used to help decode P_{23}

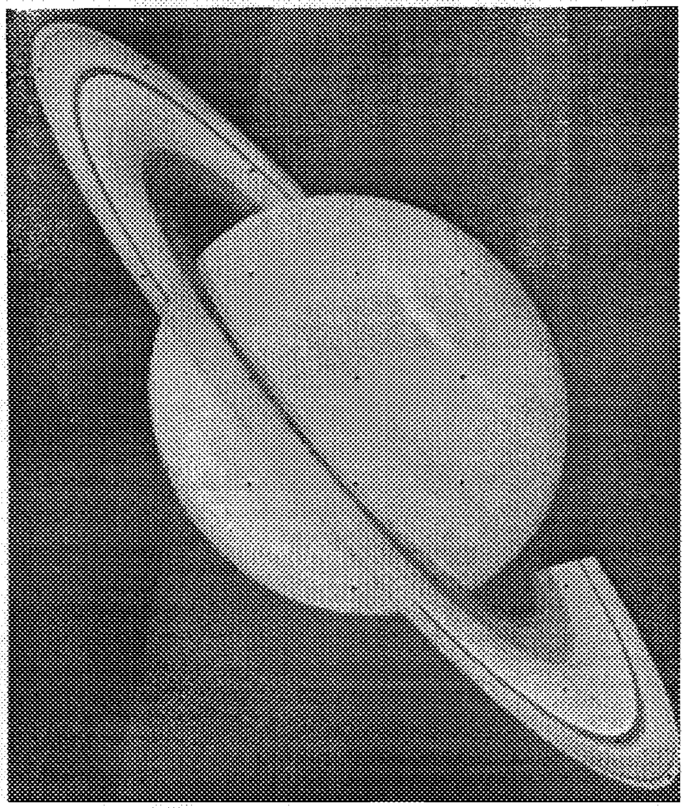

Fig. 6. Most of the image of Saturn close up

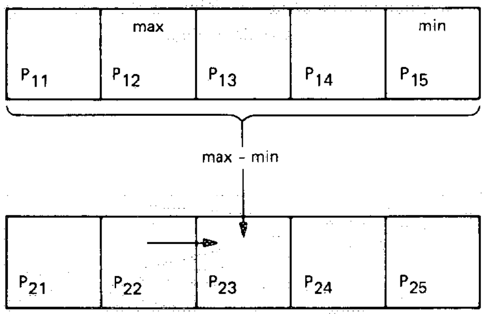

Fig. 9. Demonstrating where D and M come from

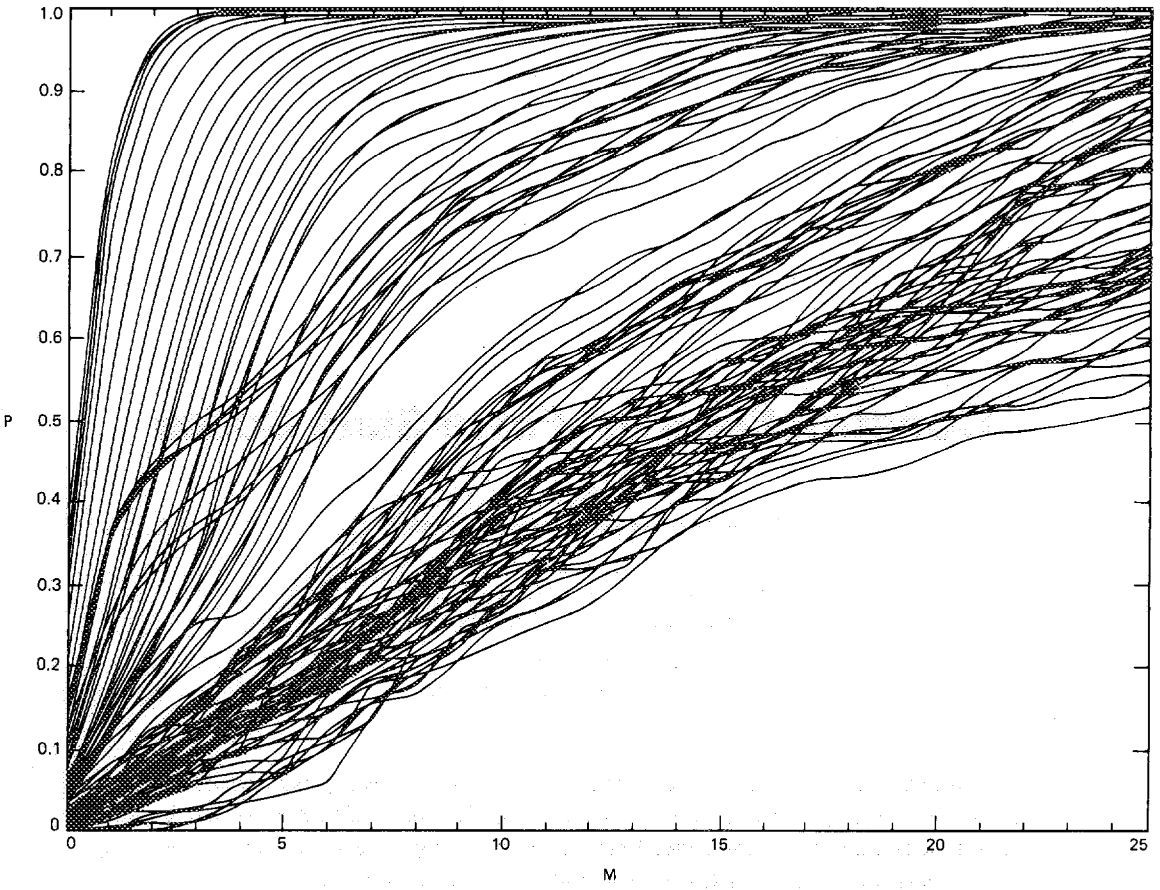

Fig. 10. Distributions of *D* when controlled for *M*

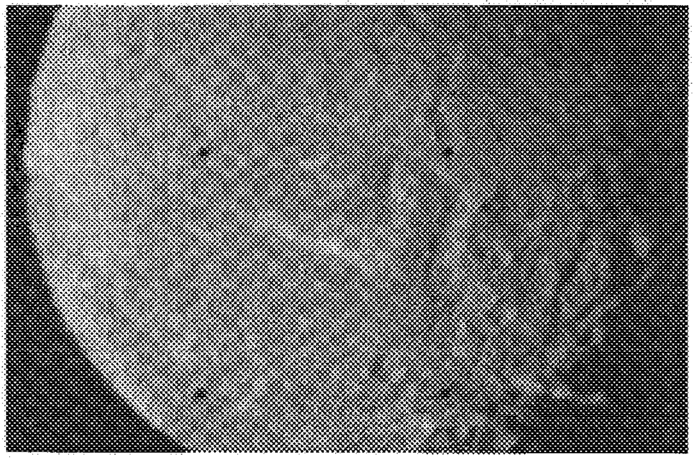

Fig. 11. Same as Fig. 1, decoded by the 2-way ISD

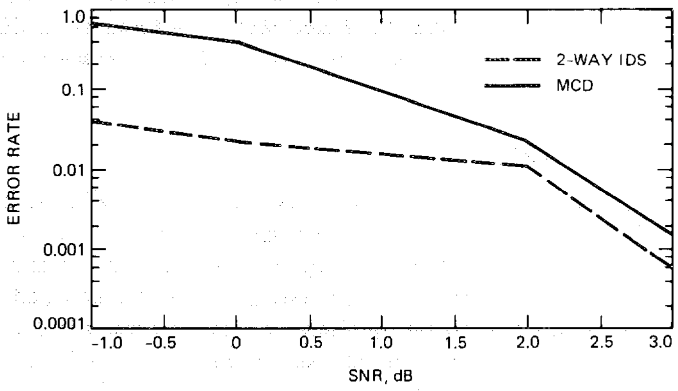

Fig. 12. Same as Fig. 5 with the two new points added at 3 dB