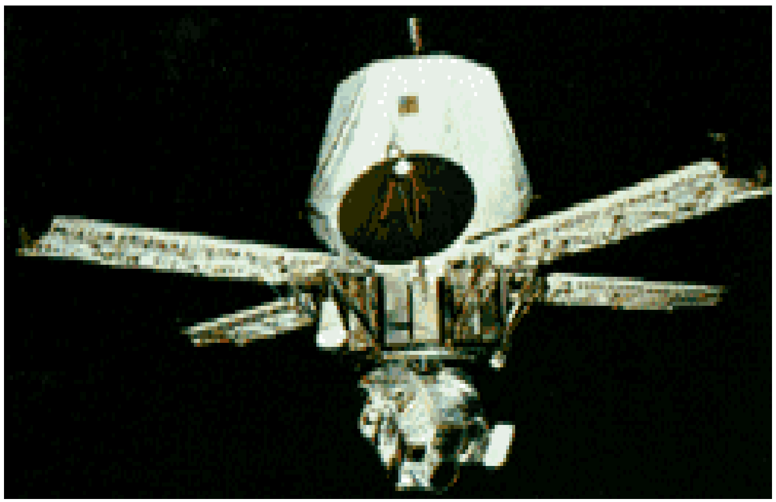
Combinatorics in Space

The Mariner 9 Telemetry System

Mariner 9 Mission



Launched: May 30, 1971
Arrived: Nov. 14, 1971
Turned Off: Oct. 27, 1972

Mission Objectives:
(Mariner 8): Map 70% of Martian surface.
(Mariner 9): Study temporal changes in Martian atmosphere and surface features.

Live TV

A black and white TV camera was used to broadcast "live" pictures of the Martian surface.

Each photo-receptor in the camera measures the brightness of a section of the Martian surface about 4-5 km square, and outputs a grayness value in the range 0-63. This value is represented as a binary 6-tuple.

The TV image is thus digitalized by the photo-receptor bank and is output as a stream of thousands of binary 6-tuples.

Coding Needed

Without coding and a failure probability $p = 0.05$, 26% of the image would be in error ... unacceptably poor quality for the nature of the mission.

Any coding will increase the length of the transmitted message. Due to power constraints on board the probe and equipment constraints at the receiving stations on Earth, the coded message could not be much more than 5 times as long as the data.

Thus, a 6-tuple of data could be coded as a codeword of about 30 bits in length.

Other concerns

A second concern involves the coding procedure. Storage of data requires shielding of the storage media – this is dead weight aboard the probe and economics require that there be little dead weight. Coding should therefore be done “on the fly”, without permanent memory requirements.

Finally, decoding needs to be done rapidly. The Jet Propulsion Laboratory in Pasadena, California will process the signals and reconvert them to picture images for the press which will be gathered at JPL.

Besides this NASA priority, rapid decoding is needed so that feedback to the probe becomes viable – redirecting the camera based on what is seen.

The Code

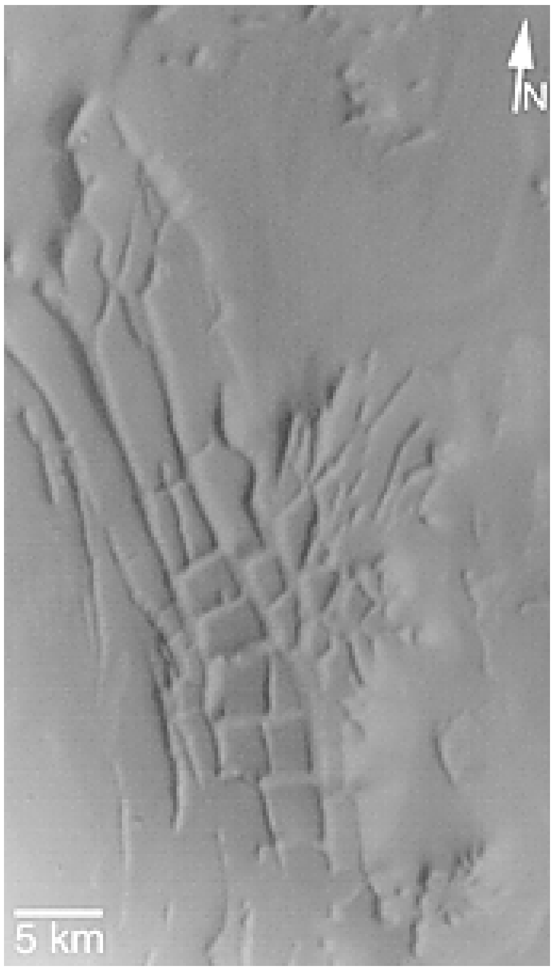
The 5-repeat code would satisfy the mission specs, but it is only 2-error correcting ... leaving 1% of the image in error.

The actual code selected is 7-error correcting and this reduced the probability of error in the image to only 0.01%.

The decision on which code to use was based primarily on the decoding algorithm. The algorithm was carried out by a fairly simple piece of specialized circuitry called "The Green Machine."

The code selected was a particular Reed-Muller Code. We will examine this code later.

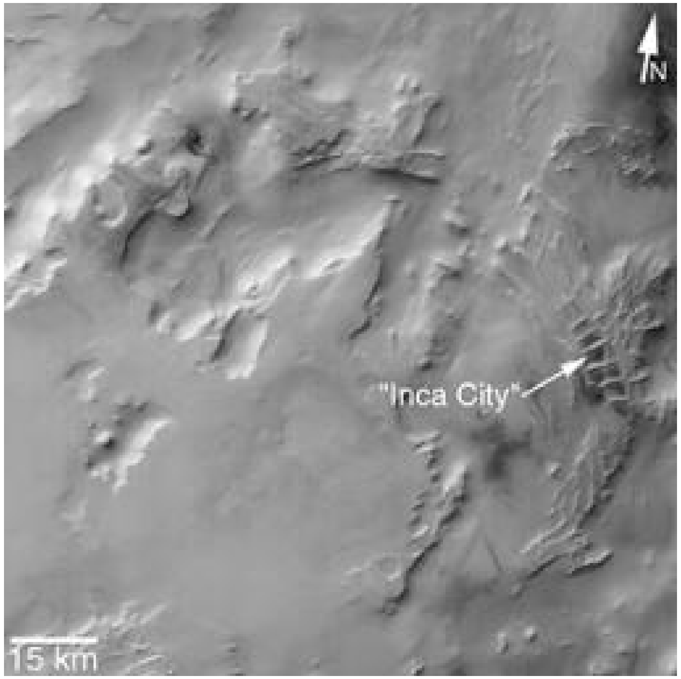
Results



Inca City:
-80 Lat., 64 Long.

"Inca City" is the informal name given by Mariner 9 scientists in 1972 to a set of intersecting, rectilinear ridges that are located among the layered materials of the south polar region of Mars. Their origin has never been understood; most investigators thought they might be sand dunes, either modern dunes or, more likely, dunes that were buried, hardened, then exhumed. Others considered them to be dikes formed by injection of molten rock (magma) or soft sediment into subsurface cracks that subsequently hardened and then were exposed at the surface by wind erosion.

Inca City

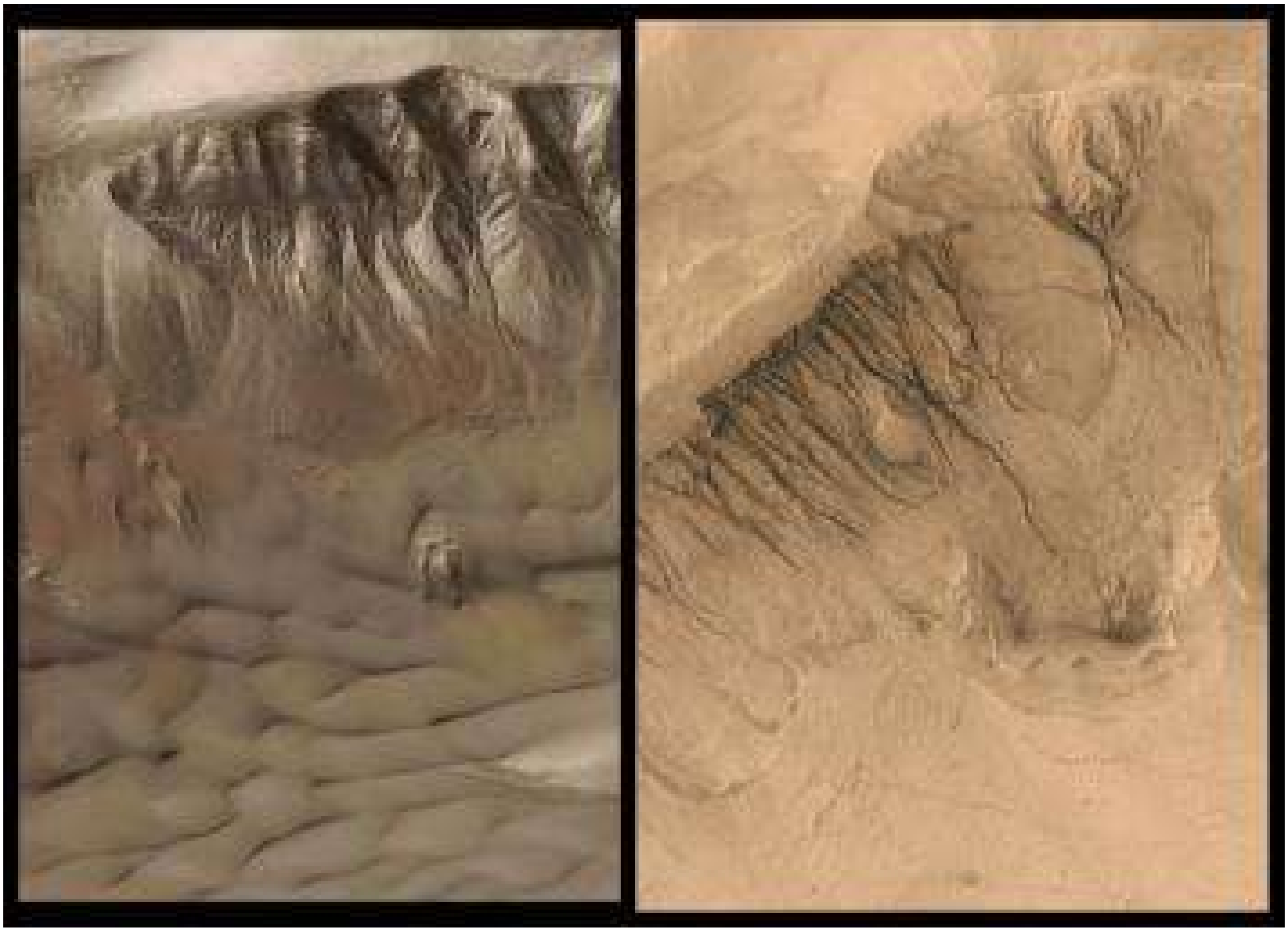


The Mars Global Surveyor (MGS) Mars Orbiter Camera (MOC) has provided new information about the "Inca City" ridges, though the camera's images still do not solve the mystery. The new information comes in the form of a MOC red wide angle context frame taken in mid-southern spring. The MOC image shows that the "Inca City" ridges, located at 82°S, 67°W, are part of a larger circular structure that is about 86 km (53 mi) across.

Inca City

It is possible that this pattern reflects an origin related to an ancient, eroded meteor impact crater that was filled-in, buried, then partially exhumed. In this case, the ridges might be the remains of filled-in fractures in the bedrock into which the crater formed, or filled-in cracks within the material that filled the crater. Or both explanations could be wrong. While the new MOC image shows that "Inca City" has a larger context as part of a circular form, it does not reveal the exact origin of these striking and unusual martian landforms.

Inca City



Recursive Definition of Reed-Muller Codes

Reed-Muller codes are among the oldest known codes and have found widespread applications. They were discovered by Muller and provided with a decoding algorithm by Reed in 1954.

Definition: The (first order) ***Reed-Muller codes*** $\mathcal{R}$***(1,m)*** are binary codes defined for all integers $m \geq 1$, recursively by:

(i) $\mathcal{R}(1,1) = \{00,01,10,11\} = \mathbb{Z}_2^{\,2}$.

(ii) for $m > 1$,

$\mathcal{R}(1,m) = \{(\mathbf{u},\mathbf{u}), (\mathbf{u},\mathbf{u+1}) : \mathbf{u} \in \mathcal{R}(1,m-1)$ and $\mathbf{1}$ = all 1 vector$\}$.

Examples

Thus,

$\mathcal{R}(1,2) = \{0000, 0101, 1010, 1111, 0011, 0110, 1001, 1100\}$ and

$\mathcal{R}(1,3) =$

{ 00000000, 00001111,
01010101, 01011010,
10101010, 10100101,
11111111, 11110000,
00110011, 00111100,
01100110, 01101001,
10011001, 10010110,
11001100, 11000011 }

Linear Codes

V[n,q] denotes a vector space of dimension **n** defined over a field with **q** elements (q is a prime or prime power). Any subset of V[n,q] is a code.

In the V[n,q] setting, an important class of codes are the ***linear codes***, these codes are the ones whose code words form a sub-vector space of V[n,q]. If the subspace of V[n,q] is k dimensional then we talk about the subspace as an **[n,k]-*code***. (Note that the square brackets indicate a linear code).

In the V[n,q] setting, the terms "word" and "vector" are interchangeable.

Linear codes, because of their algebraic properties, are the most studied codes from a mathematical point of view.

Linear Codes

There are several consequences of a code being linear.

1) The sum or difference of two codewords is another codeword.
2) The zero vector is always a codeword.
3) The number of codewords in an [n,k]-code C of V[n,q] is q^k.

There are k vectors in a basis of C. Every codeword is expressible as a unique linear combination of basis vectors. Thus, to count the number of codewords, we just have to count the number of linear combinations. There are q choices for a scalar multiple of each basis vector and therefore q^k linear combinations in total.

Since the number of codewords of a linear code is determined by the dimension of the subspace, the (n, M, d) notation for general codes is generally replaced by **[n, k, d]** for linear codes.

Properties

Proposition: For $m > 0$, the Reed-Muller code $\mathcal{R}(1,m)$ is a binary $[2^m, m+1, 2^{m-1}]$ linear code, in which every codeword except **0** and **1** has weight 2^{m-1}.

Which means:

The code words have length 2^m.

There are 2^{m+1} code words.

The code has minimum distance 2^{m-1}, which means that it can correct $2^{m-2}-1$ errors.

The sum of any two codewords is another codeword (linear).

The zero vector is in the code and the all 1 vector (of weight 2^m) is in the code.

All other codewords have half of their positions 0 and half 1.

Back to Mariner 9

Recall that in the Mariner 9 mission, the data consisted of binary 6-tuples (64 grayness levels) and transmission restrictions permitted coding that would lengthen the transmitted words to about 30 bits.

The 5-repeat code would satisfy this condition, but it is only 2 error correcting.

The code that was chosen however is $\mathcal{R}(1,5)$ which is a [32, 6, 16] binary code. Code words are 32 bits long and there are 64 of them. Moreover, as the minimum distance is 16, this code is 7-error correcting.

Back to Mariner 9

Encoding:

As there are 64 code words and 64 data types, any assignment of code word to data type will work, but the requirement that the encoding should require no memory meant that an arbitrary assignment would not do.

Since $\mathcal{R}(1,5)$ is a 6 dimensional code, there is a basis with 6 elements (any linear combination of which gives a code word). The data type 6-tuple is used to provide the coefficients for the linear combination of the basis vectors ... thus associating a unique code word to each data type.

This simple computation can be hard wired and requires no memory.

Back to Mariner 9

Decoding:

As we have previously mentioned, the real reason for selecting this code was that it had a very fast decoding algorithm which we now describe.

First, convert all the code words (and the received vector) to ± 1 vectors by turning the 0's into -1's. Take the dot product of the received vector with each of the code words in turn. As soon as the result is 16 or greater, decode as that code word.

Suppose no errors have been made in transmission. Then the dot product of the received vector with itself will be 32 and with any other codeword will be 0 or -32.

Back to Mariner 9

Decoding:

This follows since the distance between two code words is the weight of their difference (which is another code word) and so is either 0, 16 or 32. If 0, the code words are the same. If 32, the code words have no common component and the dot product of the ± 1 form will be -32. In all remaining cases, 16 places are the same and 16 places are different, giving a dot product of $16 - 16 = 0$.

For each error that occurs, the dot product will decrease by 2 (or increase by 2 from an incorrect codeword). If no more than 7 errors occur, the dot product with the correct code word decreases to at least 18 and the dot product with incorrect code words increases to at most 14 ... so correct decoding will occur. If 8 or more errors occur, there will be dot products of at least 16 but correct decoding is not possible.

Back to Mariner 9

Decoding:

Even though this is a rapid decoding algorithm, the computations involved can be speeded up by a factor of 3 by using a Fast Fourier Transform for Abelian groups. This is what was actually done by the "green machine".

Other Missions

The Voyager 1 & 2 spacecraft transmitted color pictures of Jupiter and Saturn in 1979 and 1980. Color transmission requires 3 times the amount of data, so a different code (the Golay (24,12,8) code) was used. It is only 3-error correcting, but its transmission rate is much higher. Voyager 2 went on to Uranus and Neptune and the code was switched to a Reed-Solomon code for its higher error correcting capabilities.