Seafloor Image Compression and Large Tilesize Vector Quantization

Chris Murphy

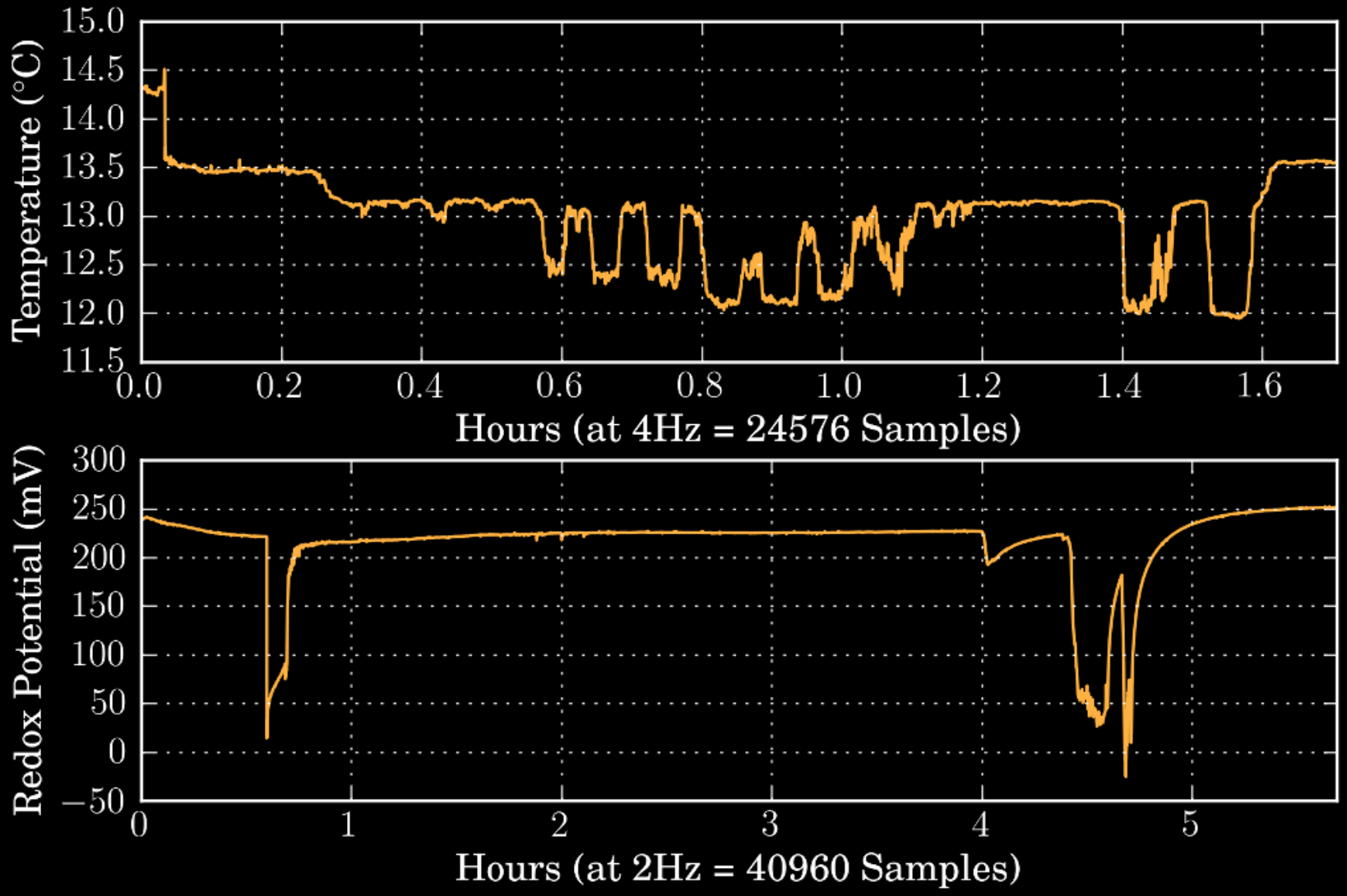
Robert Y. Wang

Dr. Hanumant Singh

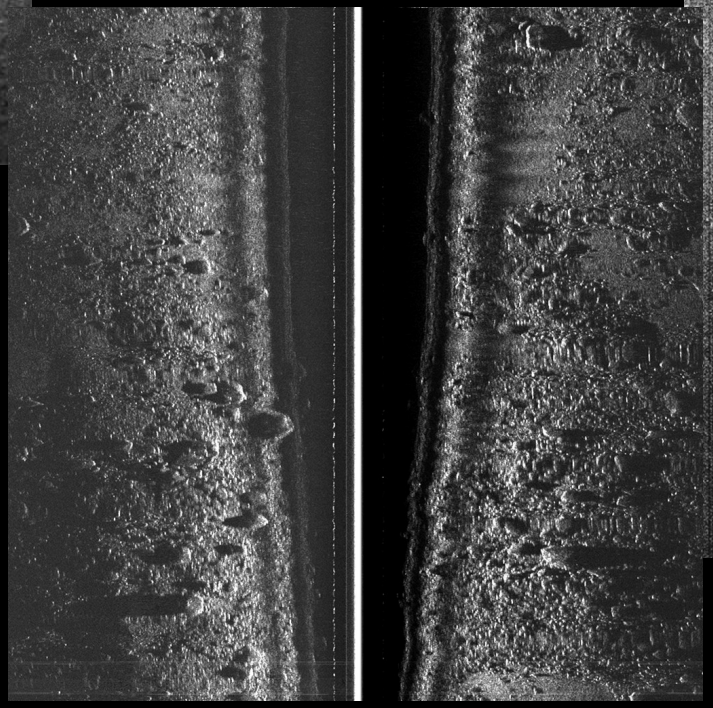
WOODS HOLE OCEANOGRAPHIC INSTITUTION
1930

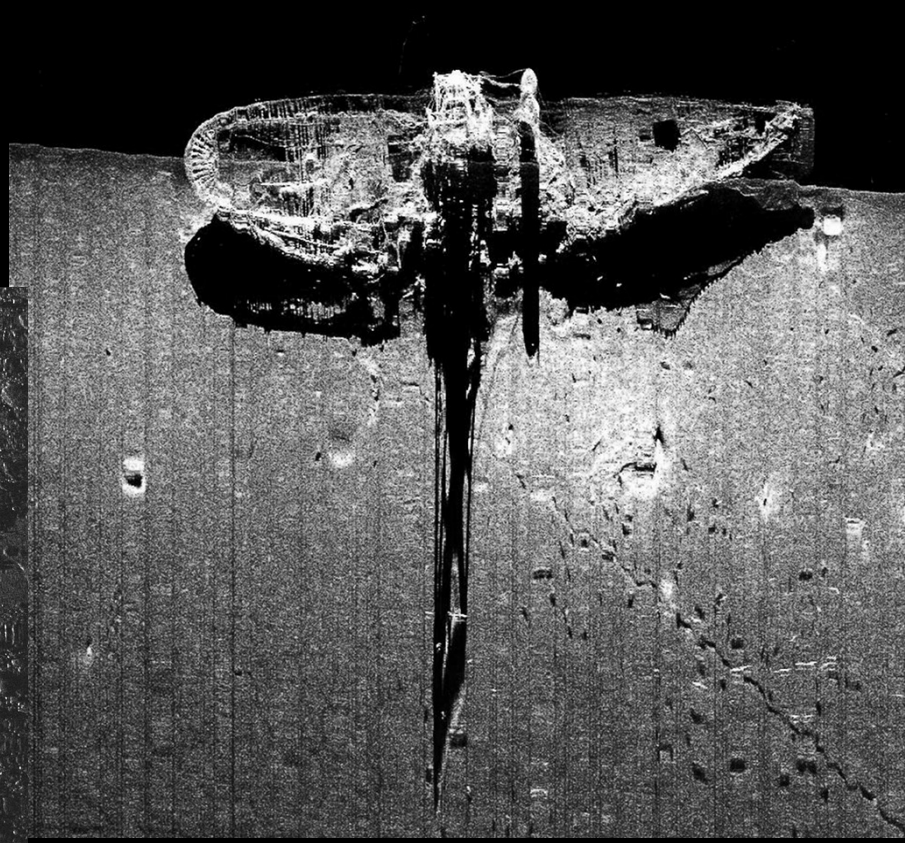
AUV Conference, 2 September 2010

Environmental Sensors



Sidescan Sonar Imagery





Photographs

The Problem

Data is typically unavailable until after recovery.

Effective collaboration requires information sharing, whether from sub-sea to surface or sub-sea to sub-sea.

In real-world conditions, effective throughput of modern acoustic modems can be 10-100 bits per second.

How can we effectively share multimodal information at these rates?

Two Complimentary Approaches

Large Tilesize Vector Quantization

SPIHT

Concept: LTVQ

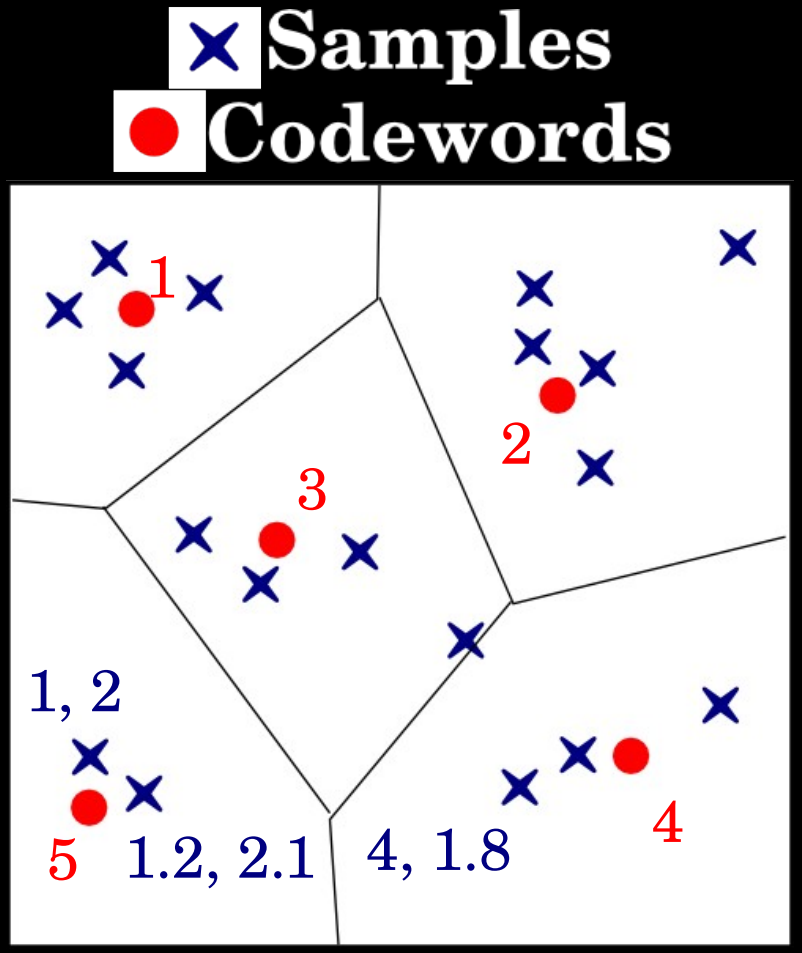
Frequently, we have imagery available from previous dives that is similar in content; possibly from the same geographic location.

How can these previous photos be used to Improve image compression?

Vector Quantization

Basic Idea:

- **Encode samples by referencing indices of previously established "codewords". The appropriate codeword is selected as the "closest" using some metric, such as L2 (Euclidean) distance.**

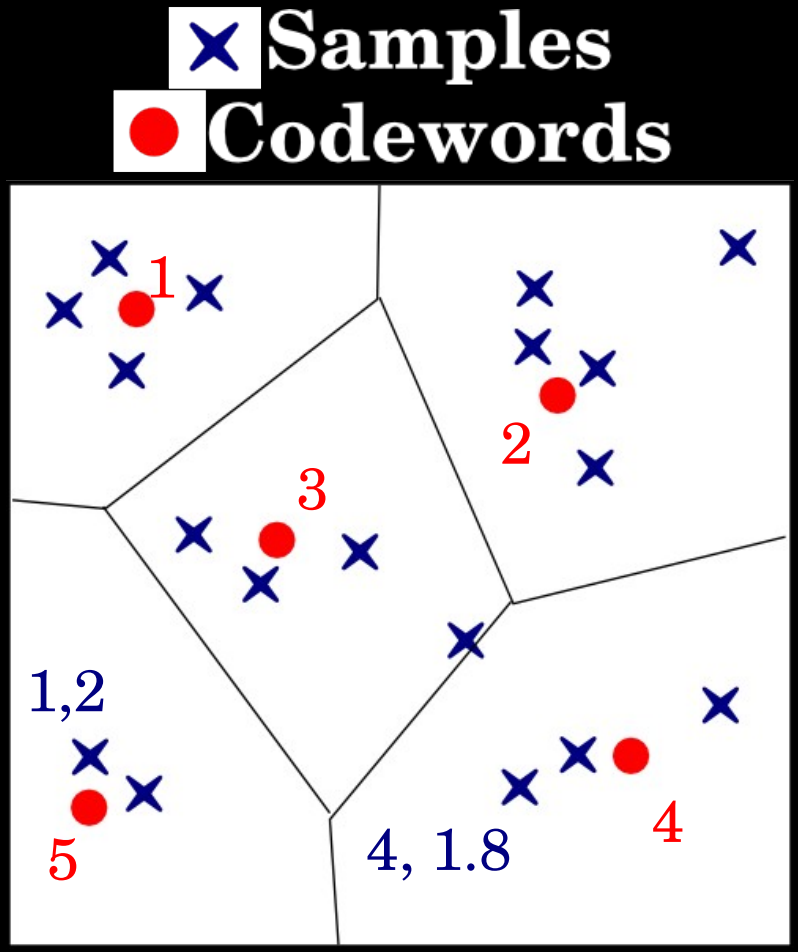


Vector Quantization

Basic Idea:

- **Encode a large number of samples simply by referencing the index of a previously established "codeword" which is closest using some metric, such as L2 distance.**

Early image compression algorithms used this technique to encode small (eg: 4x4) squares of pixels.



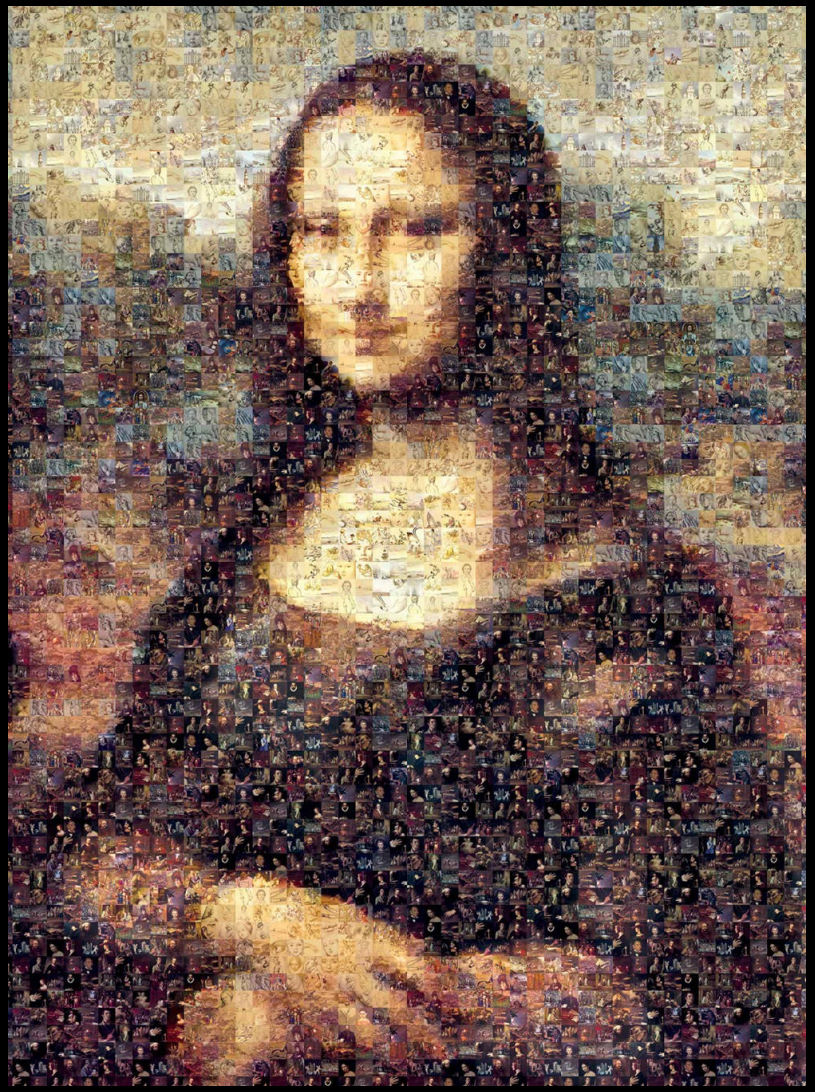
Compressed Image Size

Compressing 4x4 blocks of pixels can only go so far.

$$\frac{\text{bits}}{\text{pixel}} = \overbrace{\frac{1}{w_t \times h_t}}^{\text{Tiles per pixel}} \times \overbrace{\left\lceil \log_2 \underbrace{\left(N \times \left\lfloor \frac{w_i}{w_t} \right\rfloor \times \left\lfloor \frac{h_i}{h_t} \right\rfloor \right)}_{\text{Tiles in library}} \right\rceil}^{\text{Bits per tile}}$$

How can we use VQ to obtain very high compression ratios?

Inspiration



Compressed Image Size

So – how can we use VQ to obtain very high compression ratios?

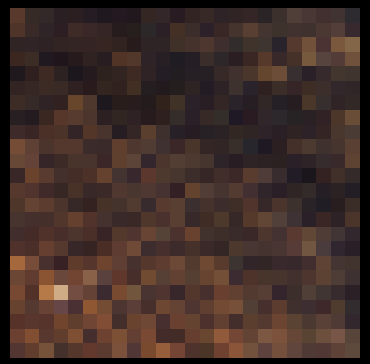
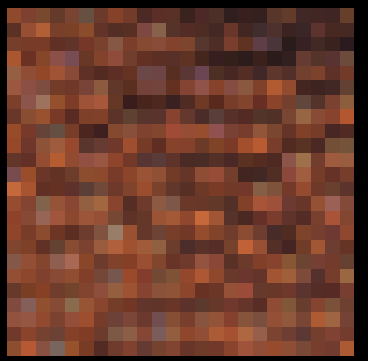
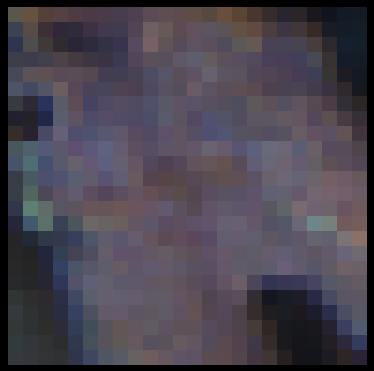
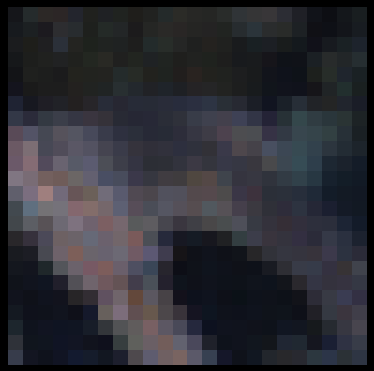
$$\frac{\text{bits}}{\text{pixel}} = \overbrace{\frac{1}{w_t \times h_t}}^{\text{Tiles per pixel}} \times \overbrace{\left\lceil \log_2 \underbrace{\left(N \times \left\lfloor \frac{w_i}{w_t} \right\rfloor \times \left\lfloor \frac{h_i}{h_t} \right\rfloor \right)}_{\text{Tiles in library}} \right\rceil}^{\text{Bits per tile}}$$

Simple - increase the size of the tiles!

Large Tilesize Vector Quantization

Even relatively large seafloor image tiles (here, 24 pixels square) exhibit homogeneity. Previous dive imagery can be sliced into tiles to obtain codewords for future dives.

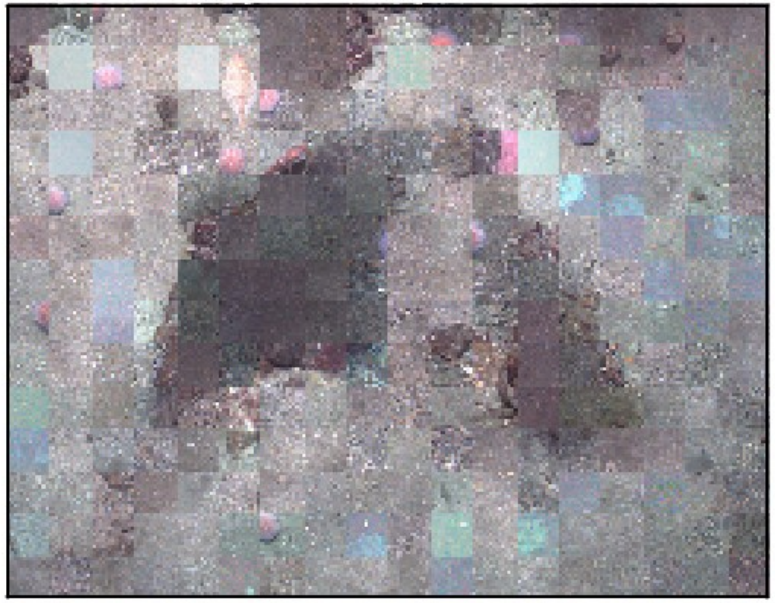


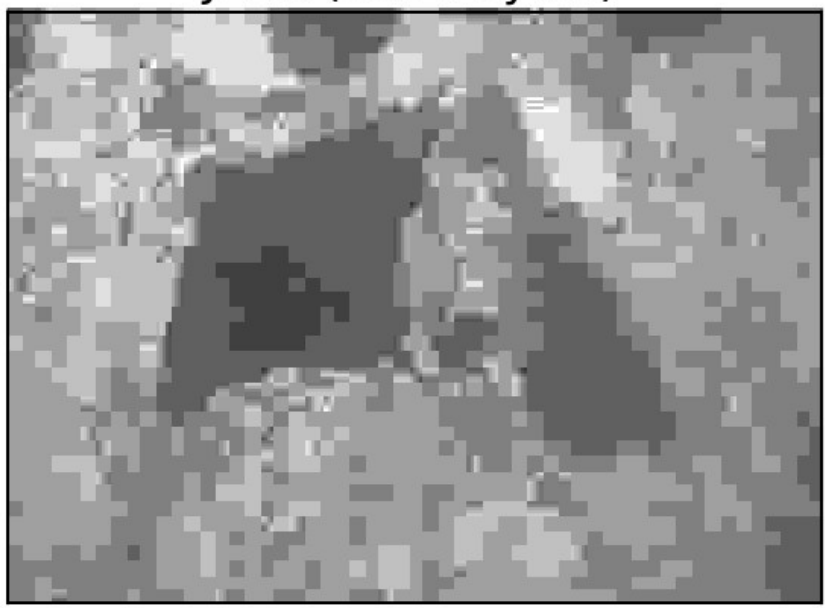
All tiles taken from 150 randomly selected images, collected on a previous dive.

Original (451584 Bytes)

Our Approach (567 Bytes)



JPEG (1524 Bytes)



SPIHT (567 Bytes)

Codeword Matching

Problem: The number of possible codewords (or "tiles") generated from even a single dive can be huge.

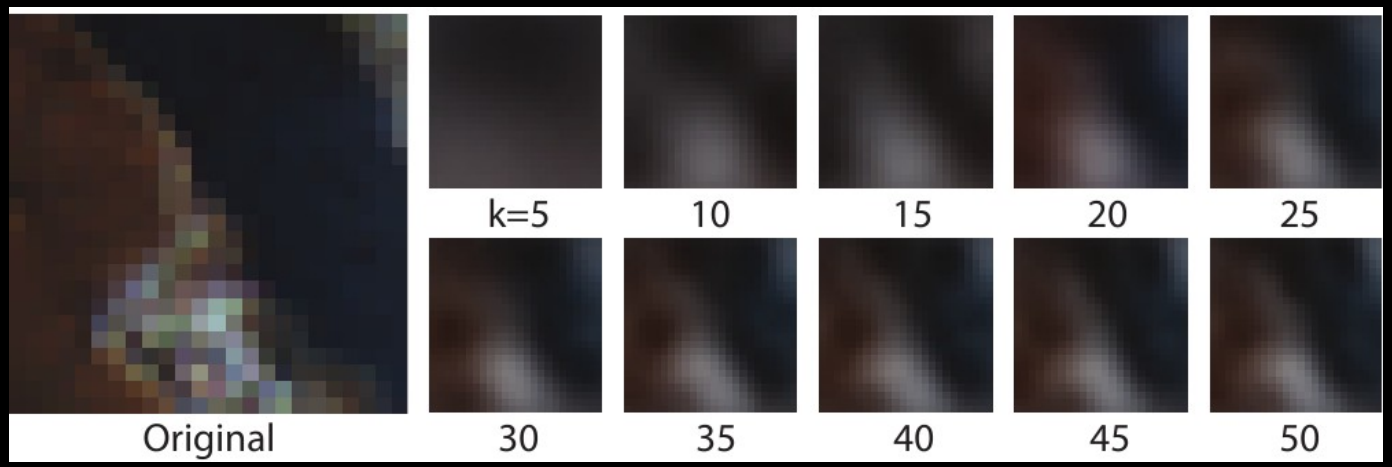
$$\frac{\text{bits}}{\text{pixel}} = \overbrace{\frac{1}{w_t \times h_t}}^{\text{Tiles per pixel}} \times \overbrace{\left\lceil \log_2 \underbrace{\left(N \times \left\lfloor \frac{w_i}{w_t} \right\rfloor \times \left\lfloor \frac{h_i}{h_t} \right\rfloor \right)}_{\text{Tiles in library}} \right\rceil}^{\text{Bits per tile}}$$

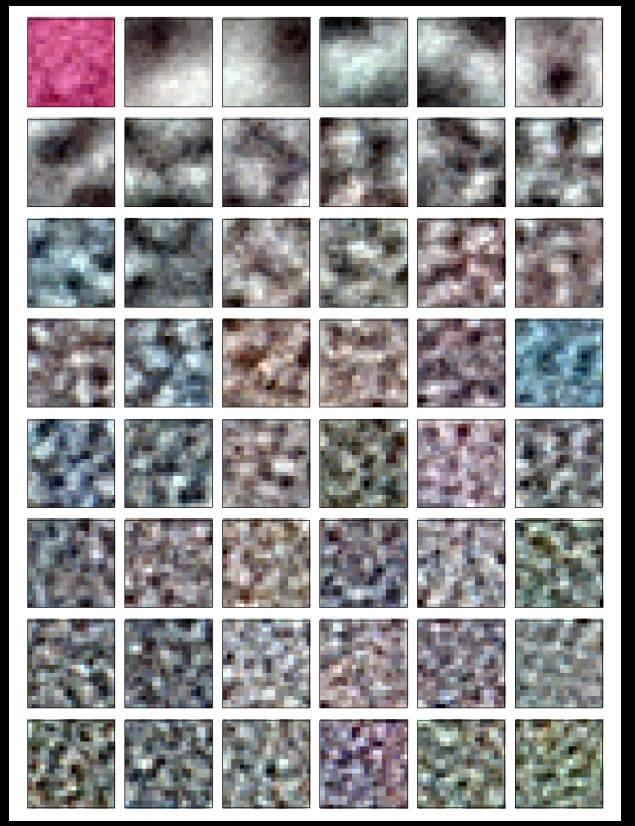
Pixel-by-pixel comparison against each of them is computationally impractical, especially for an AUV.

24 x 24 x 3 = 1728 comparisons per tile.

Codeword Matching

Solution: Calculate principle components for the tiles using PCA. Match principle component weights to reduce dimensionality, rather than tiles directly.

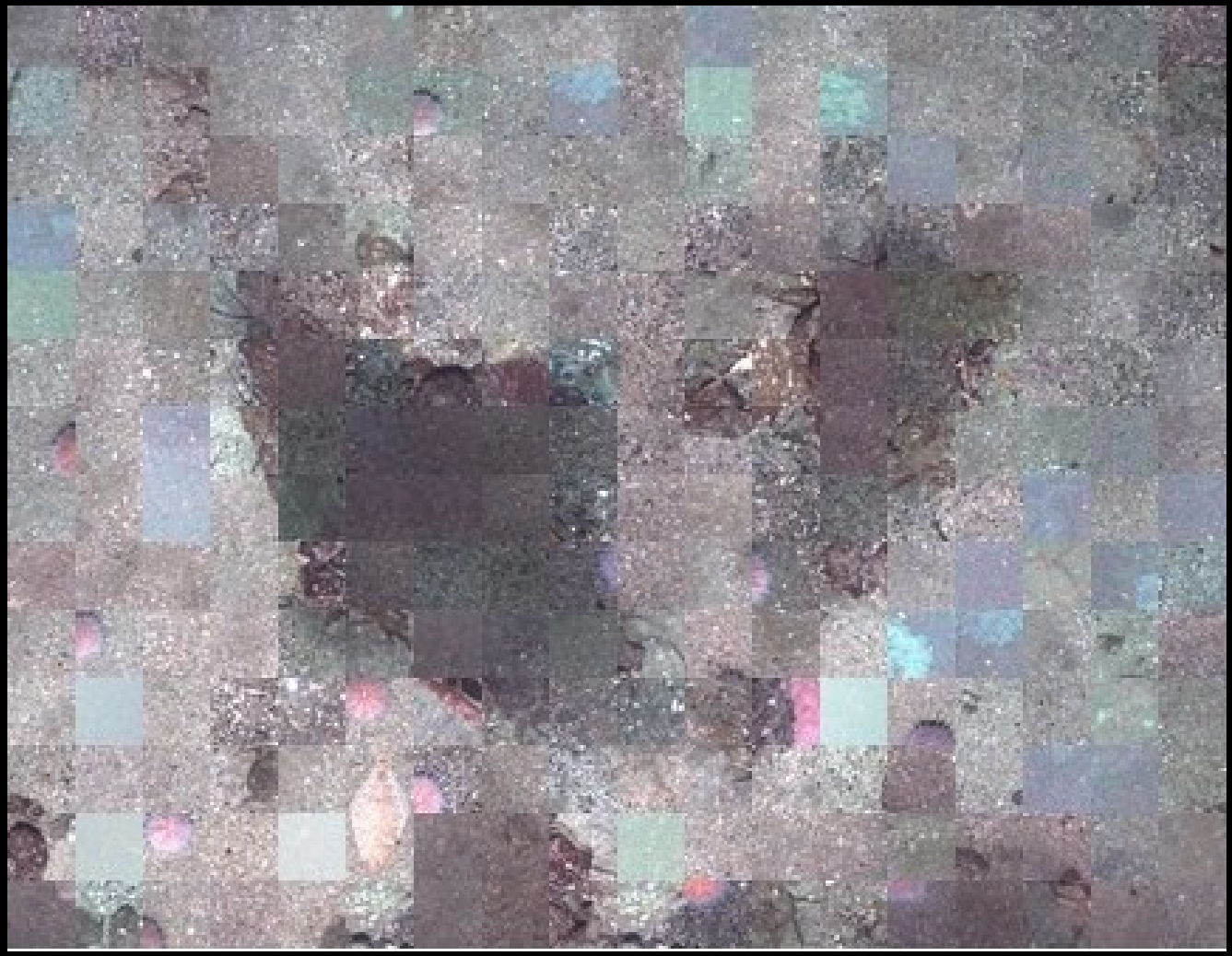




De-Blocking

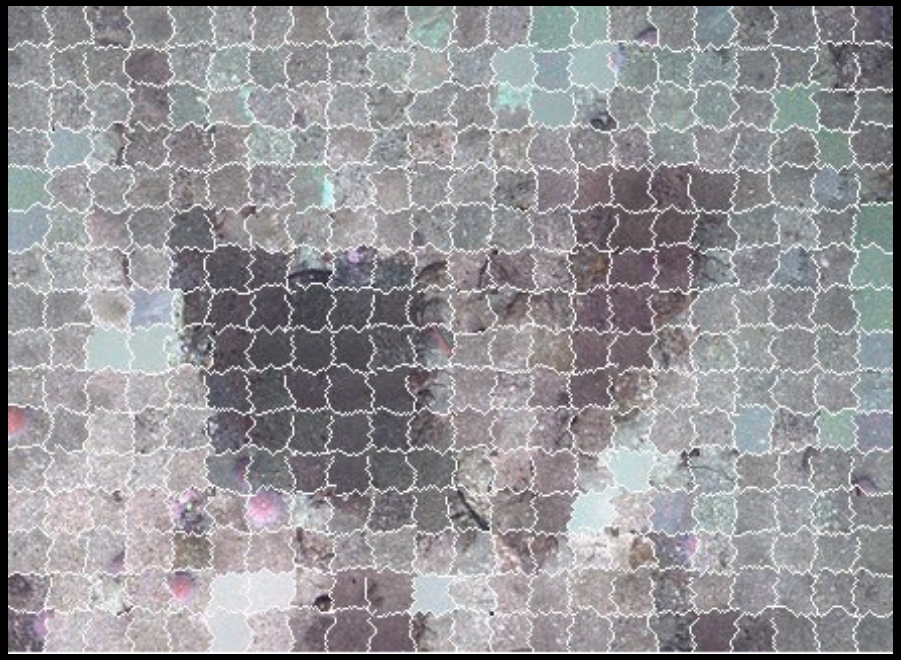
Reconstructed images exhibit heavy blocking artifacts at tile boundaries.

How can those artifacts be reduced or eliminated?



“Image Quilting” / MinCuts

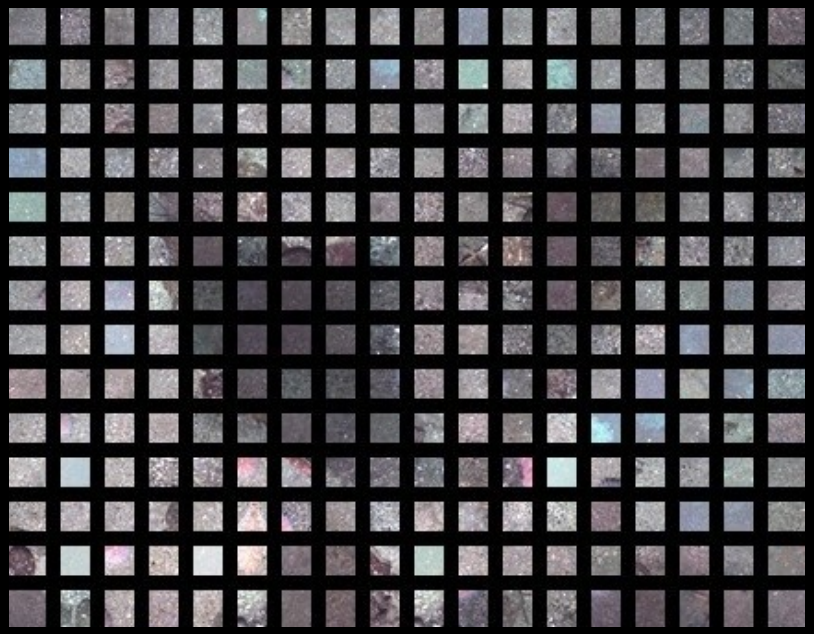
Minimize error at tile borders by carefully selecting tile boundary.

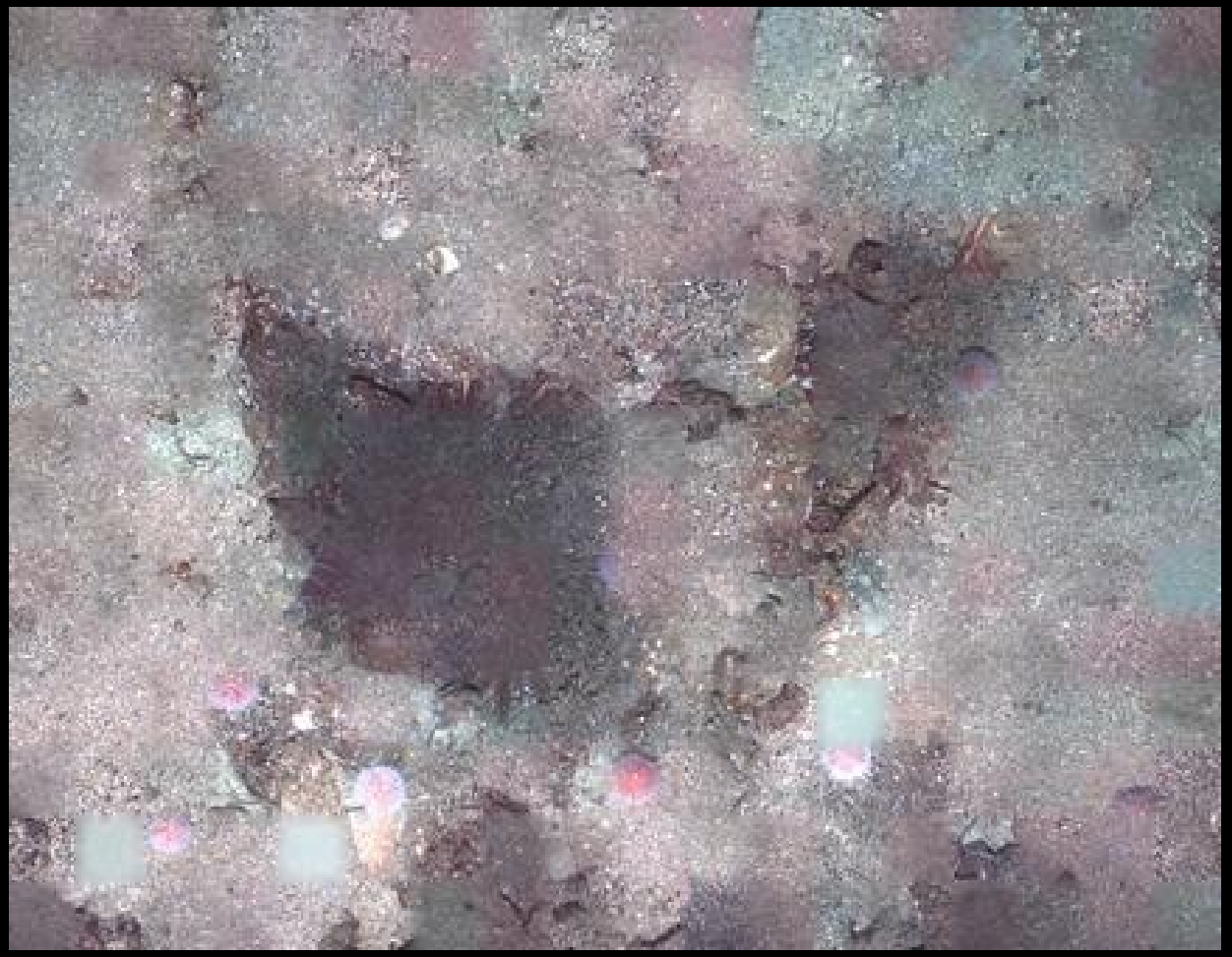


Efros + Freeman, “Image Quilting for Texture Synthesis and Transfer,” SIGGRAPH 2001.

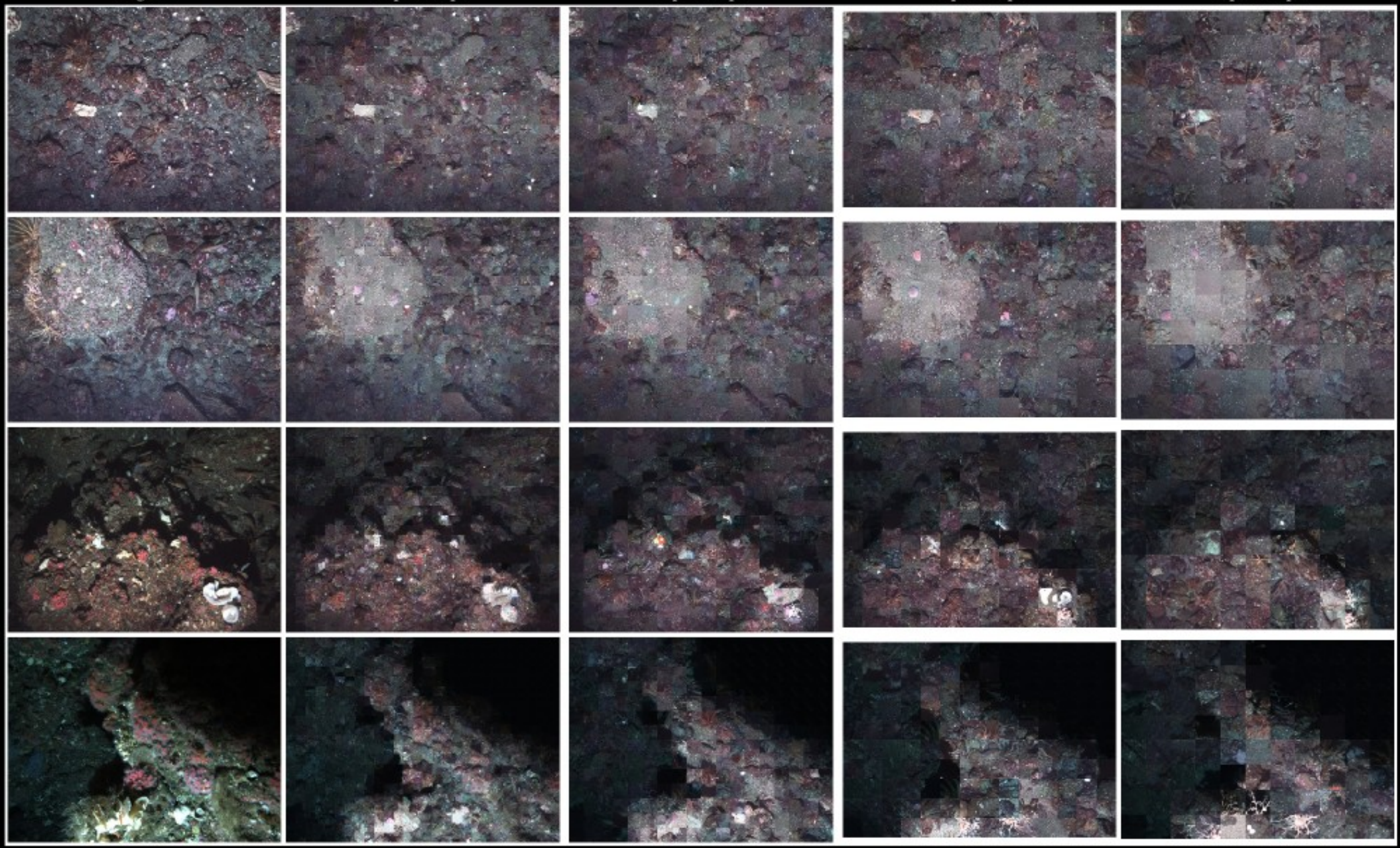
Poisson Image Editing

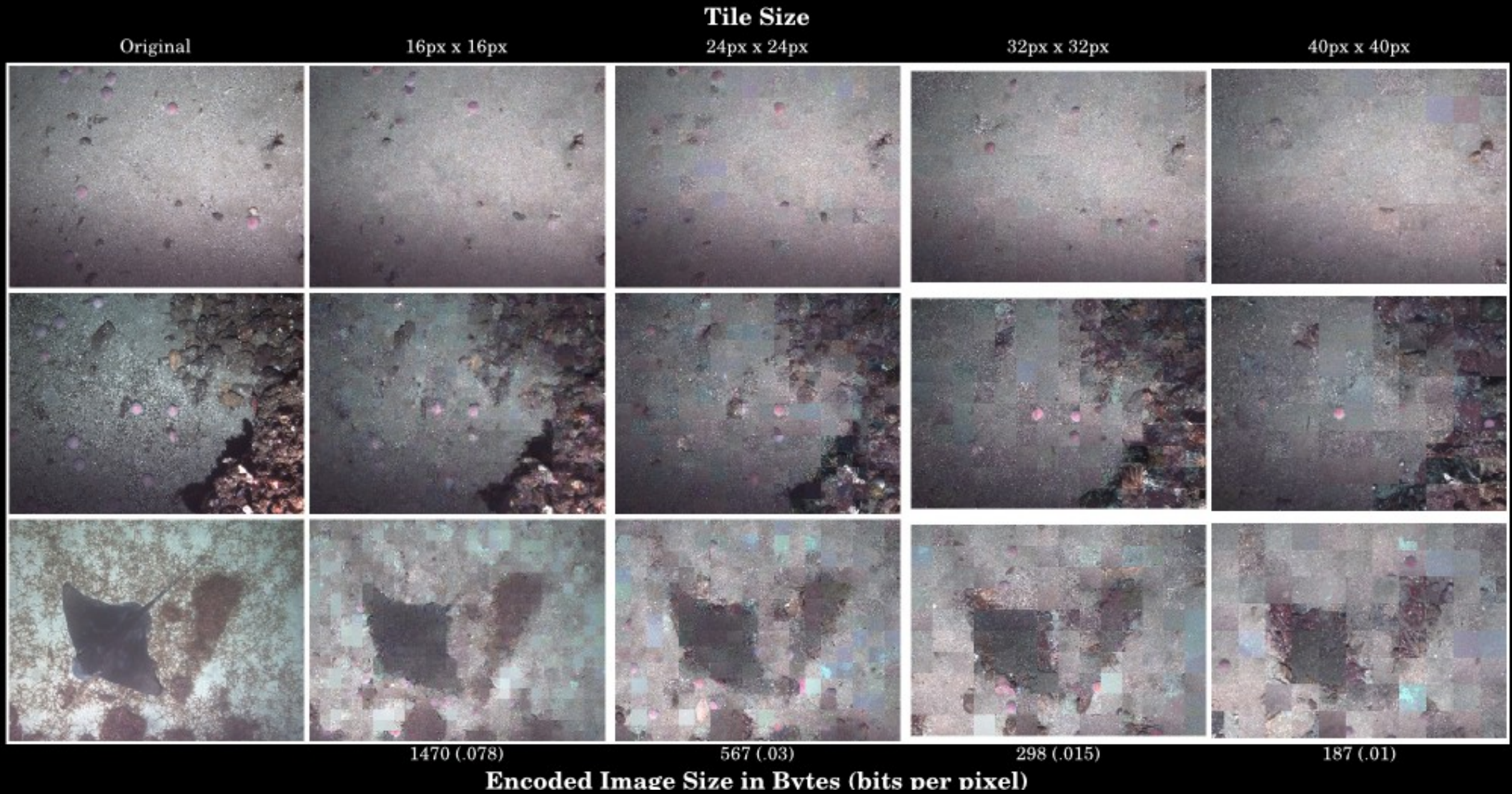
Interpolate within boundaries, while preserving gradients.





Perez et al., “Poisson image editing,” ACM Trans. Graph., 2003.


Tile Size
Original
16px x 16px
24px x 24px
32px x 32px
40px x 40px
1470 (.078)
567 (.03)
298 (.015)
187 (.01)
Encoded Image Size in Bytes (bits per pixel)


Tile Size
Original
16px x 16px
24px x 24px
32px x 32px
40px x 40px
1470 (.078)
567 (.03)
298 (.015)
187 (.01)
Encoded Image Size in Bytes (bits per pixel)

SPIHT

Set Partitioning In Hierarchical Trees *

Works in Wavelet Domain

- Wavelets have been shown to provide an efficient, sparse representation for a variety of real-world signals.

Dimensionality-Independent

- Same encoding works for CTD data, Imagery, Volumetric data

Embedded Coding

- Low-fidelity versions are *identical* to the beginning of high-fidelity
- Each additional (in-order) byte improves the estimate
- Sending up a higher quality version doesn't require 'starting over'

* *Said and Pearlman, 1996*

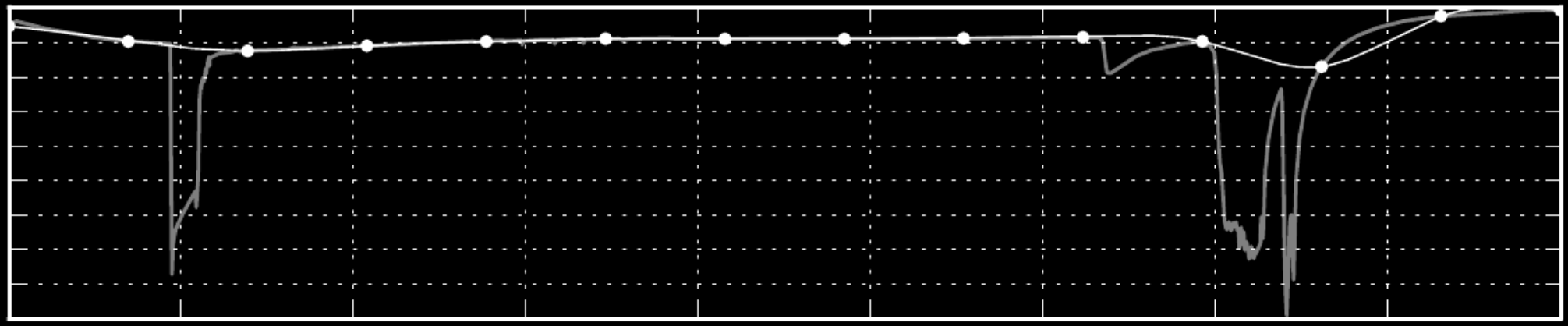
Fully Embedded Coding

Fully embedded coding – A high quality version of encoded data shares first N bits *identically* with a poor quality encoding of length N.

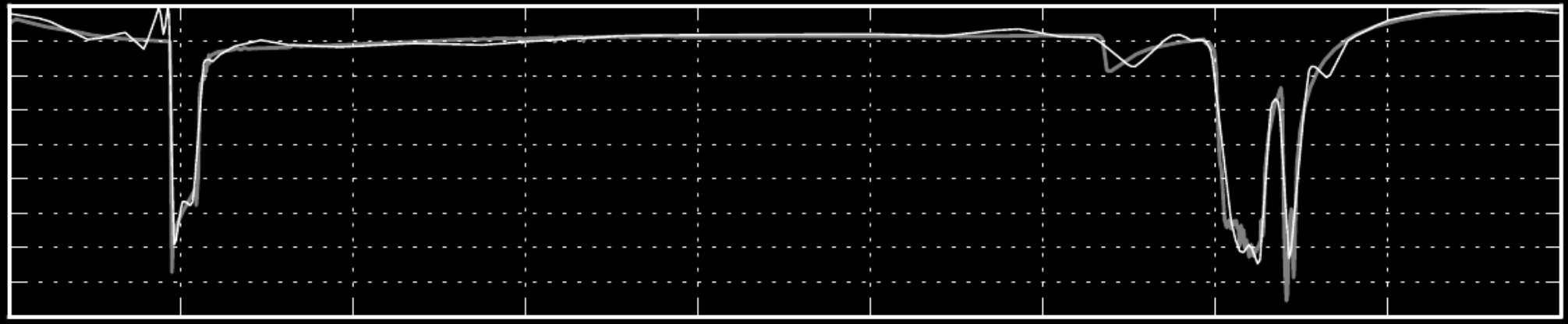
If preview is interesting, recipients can request additional packets to obtain high quality sections of signals or images.

Reduction Potential - 28 Bytes

14 Spline-Interpolated 16-bit Fixed Point Samples

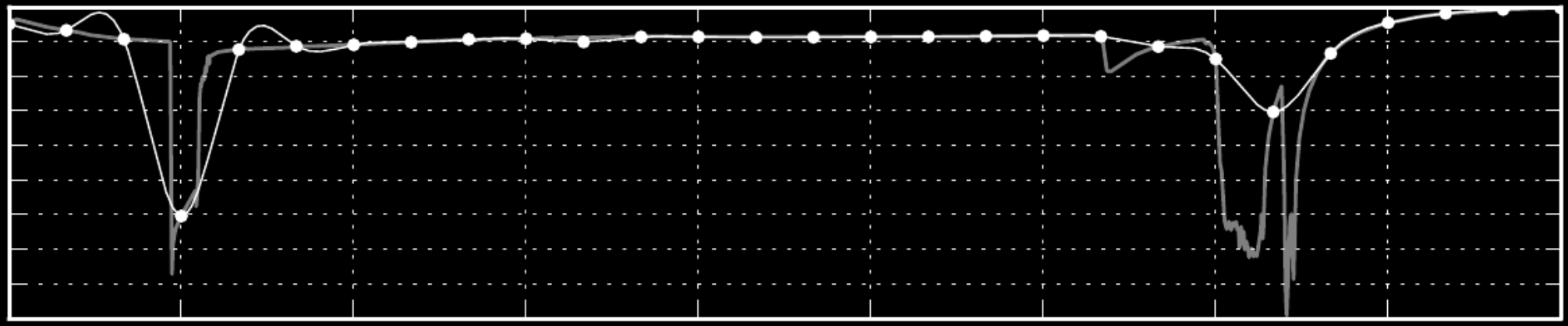


SPIHT Encoded with 28 Bytes

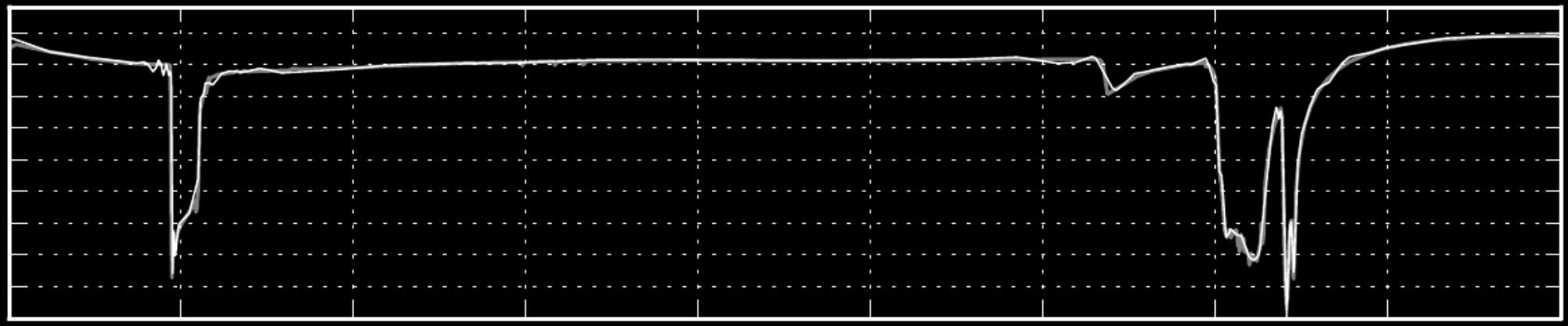


Reduction Potential - 56 Bytes

28 Spline-Interpolated 16-bit Fixed Point Samples

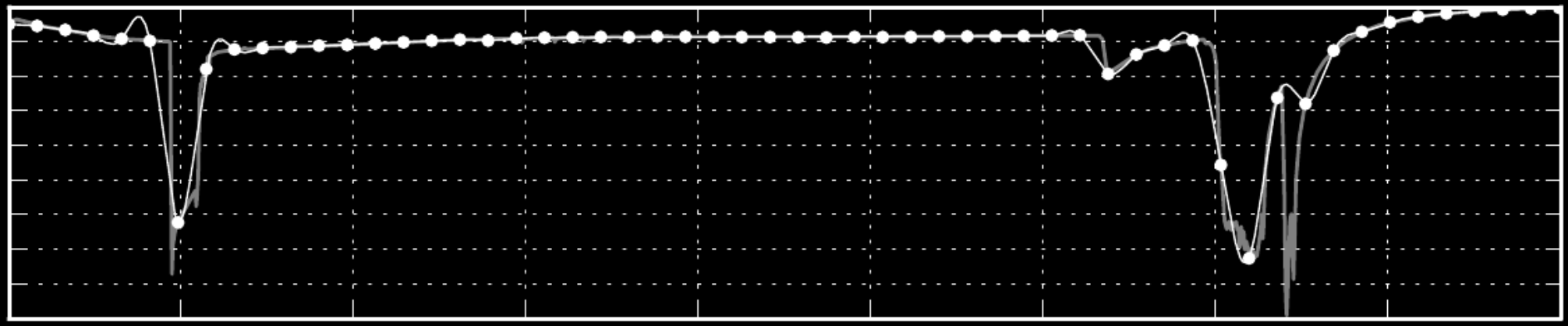


SPIHT Encoded with 56 Bytes

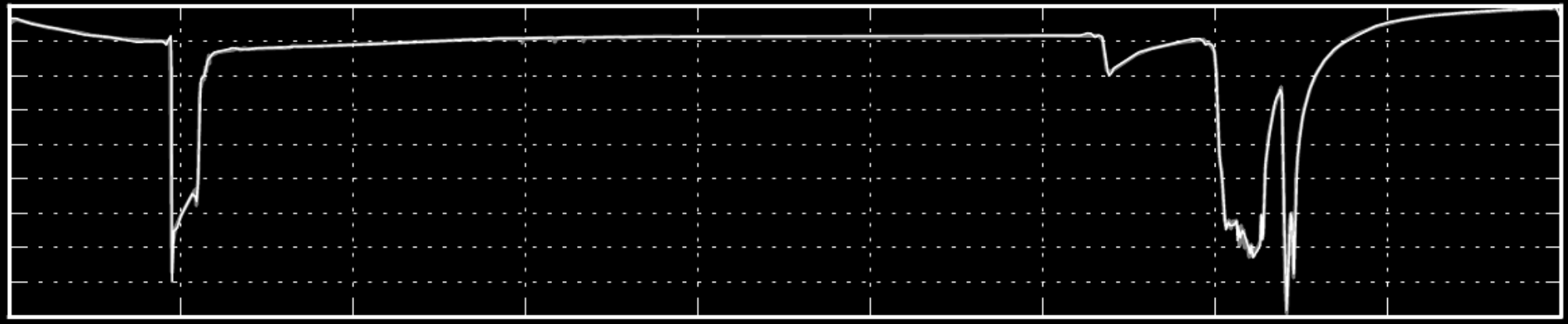


Reduction Potential - 112 Bytes

56 Spline-Interpolated 16-bit Fixed Point Samples

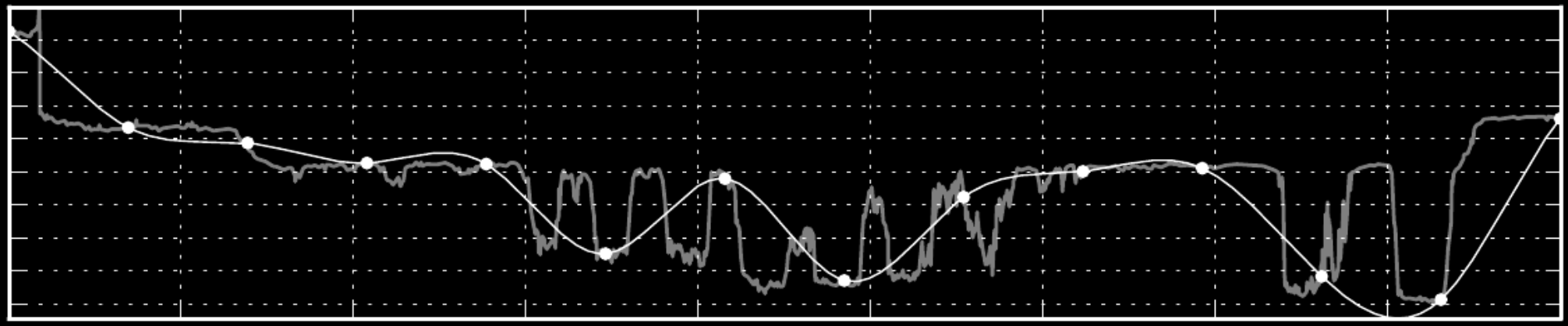


SPIHT Encoded with 112 Bytes

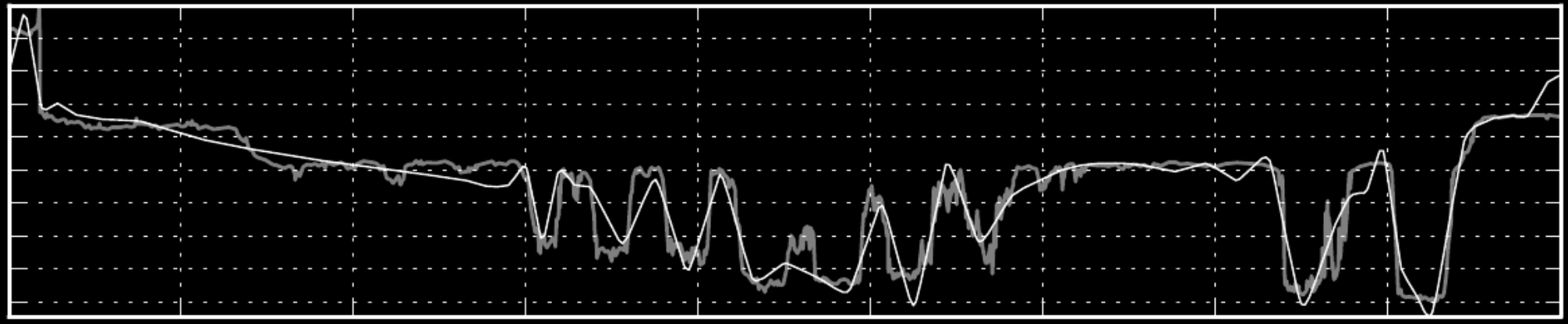


Temperature - 28 Bytes

14 Spline-Interpolated 16-bit Fixed Point Samples

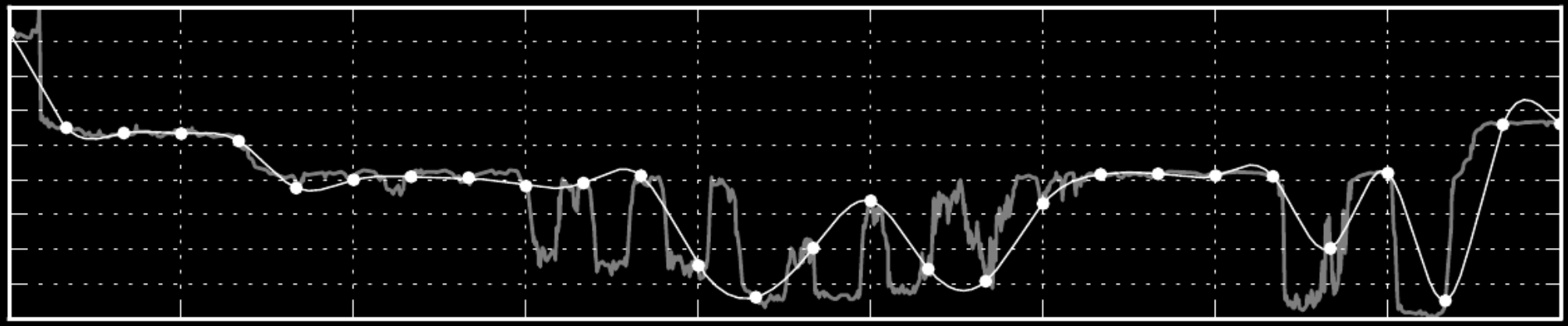


SPIHT Encoded with 28 Bytes

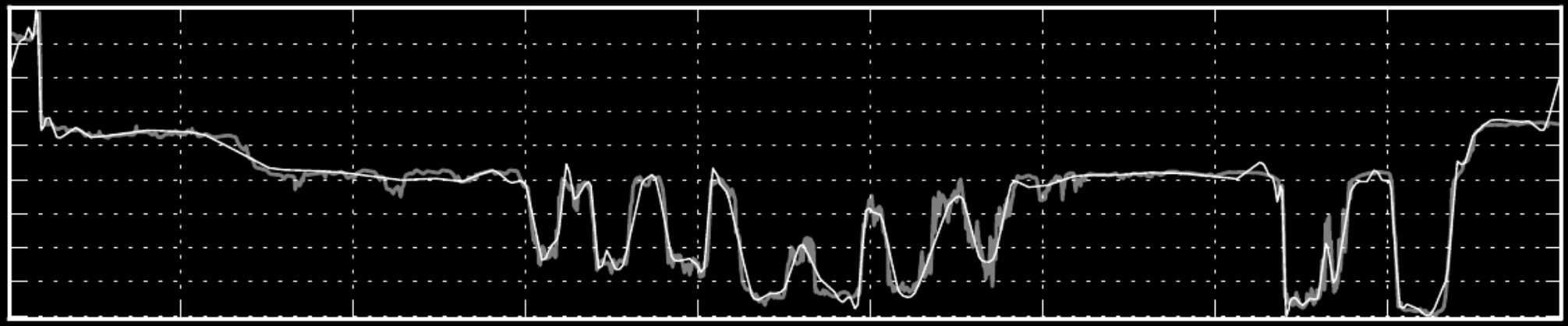


Temperature - 56 Bytes

28 Spline-Interpolated 16-bit Fixed Point Samples

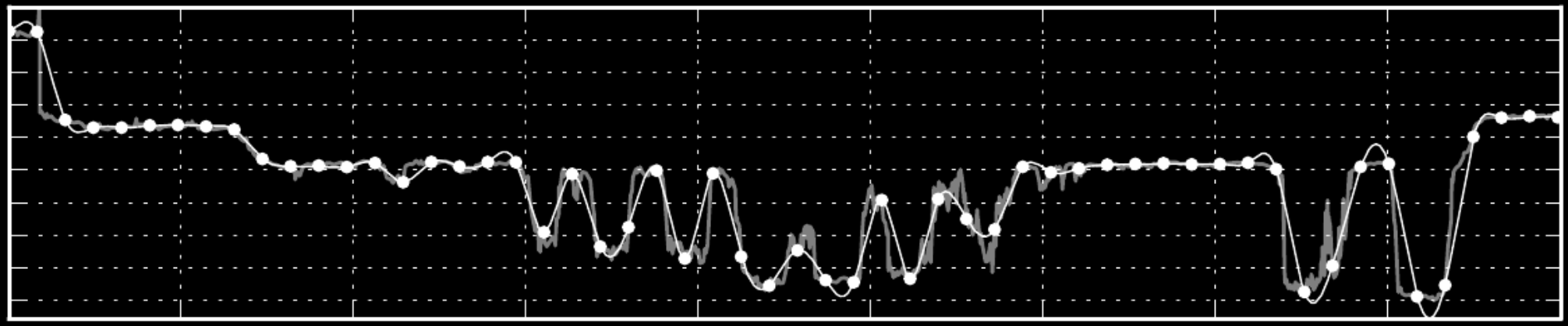


SPIHT Encoded with 56 Bytes

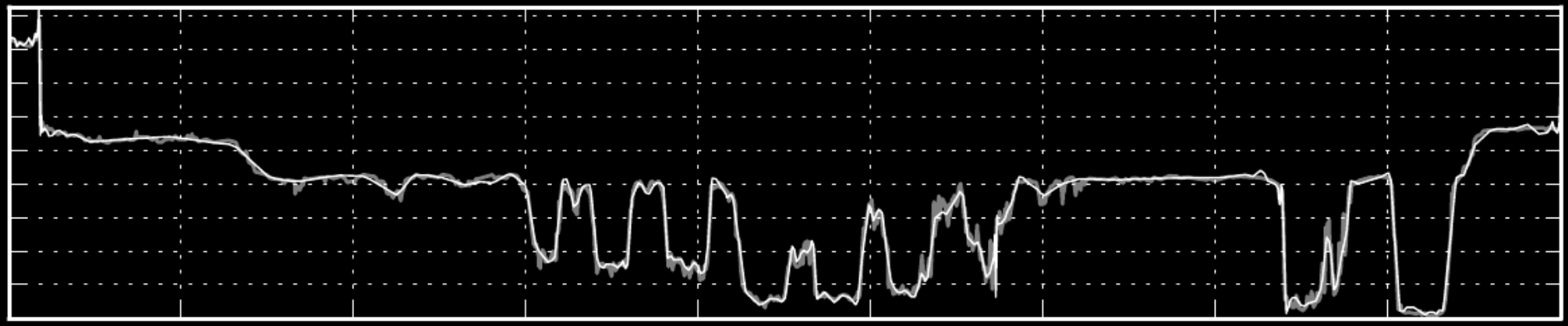


Temperature - 112 Bytes

56 Spline-Interpolated 16-bit Fixed Point Samples

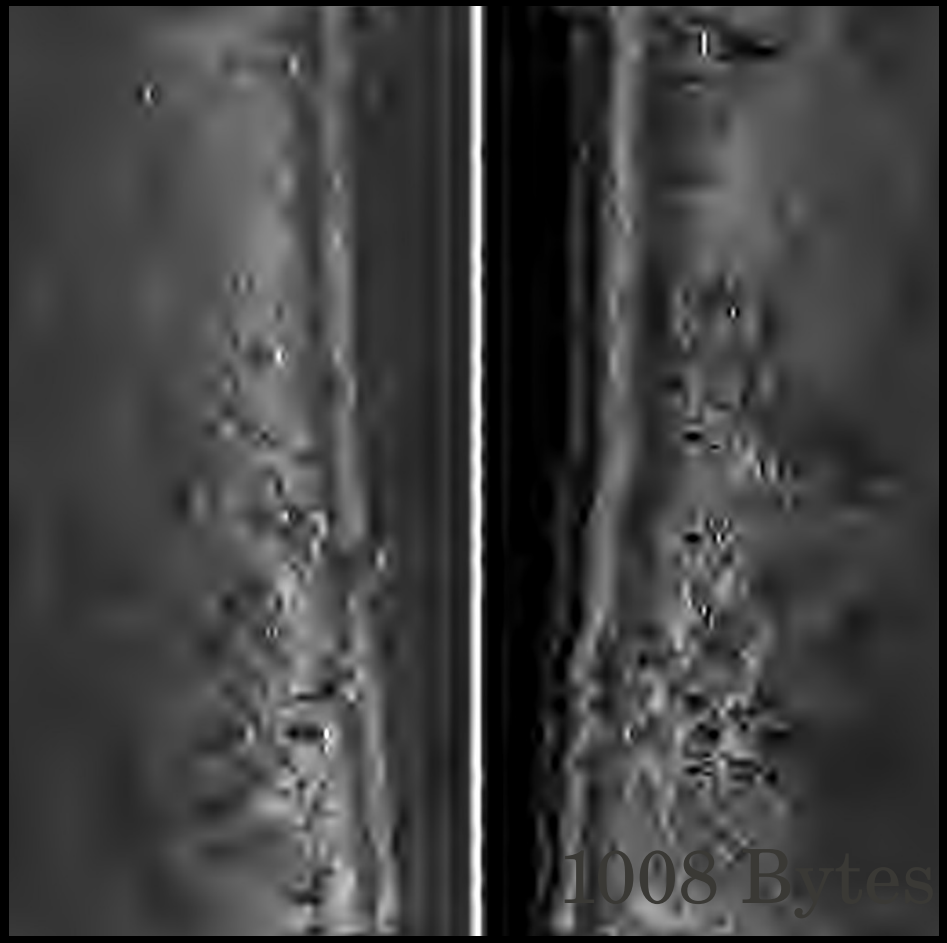


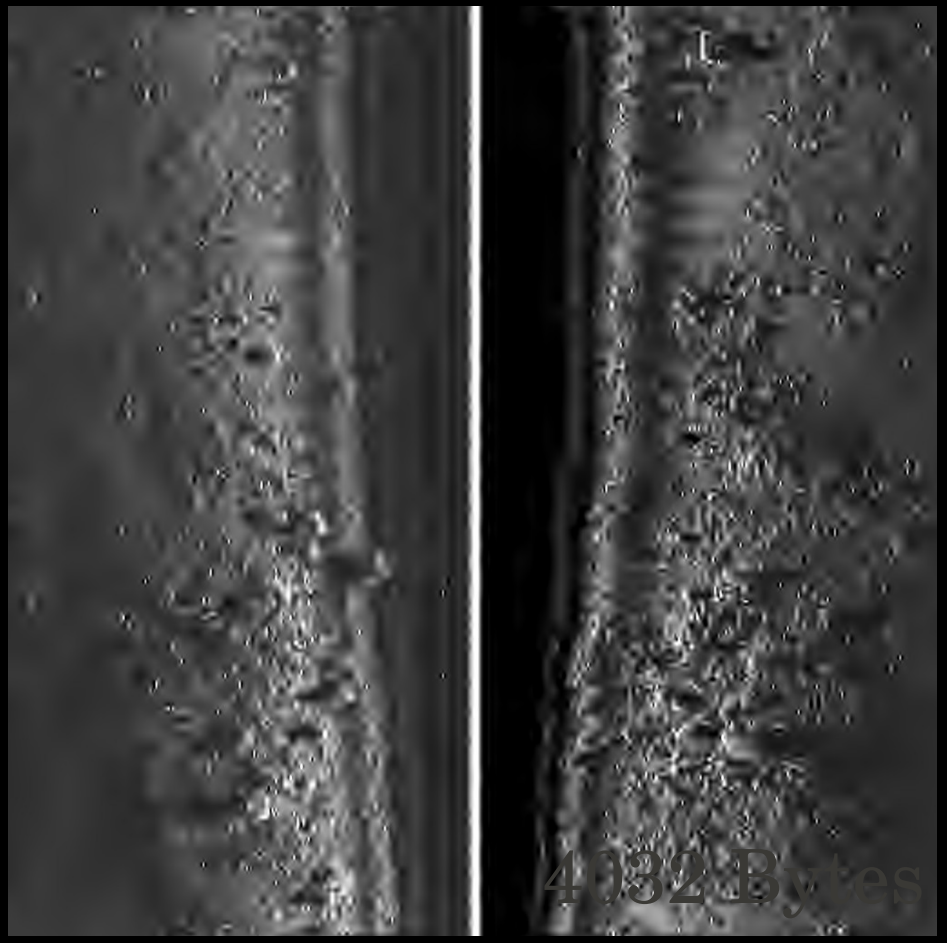
SPIHT Encoded with 112 Bytes



Sidescan Imagery

Sidescan Imagery

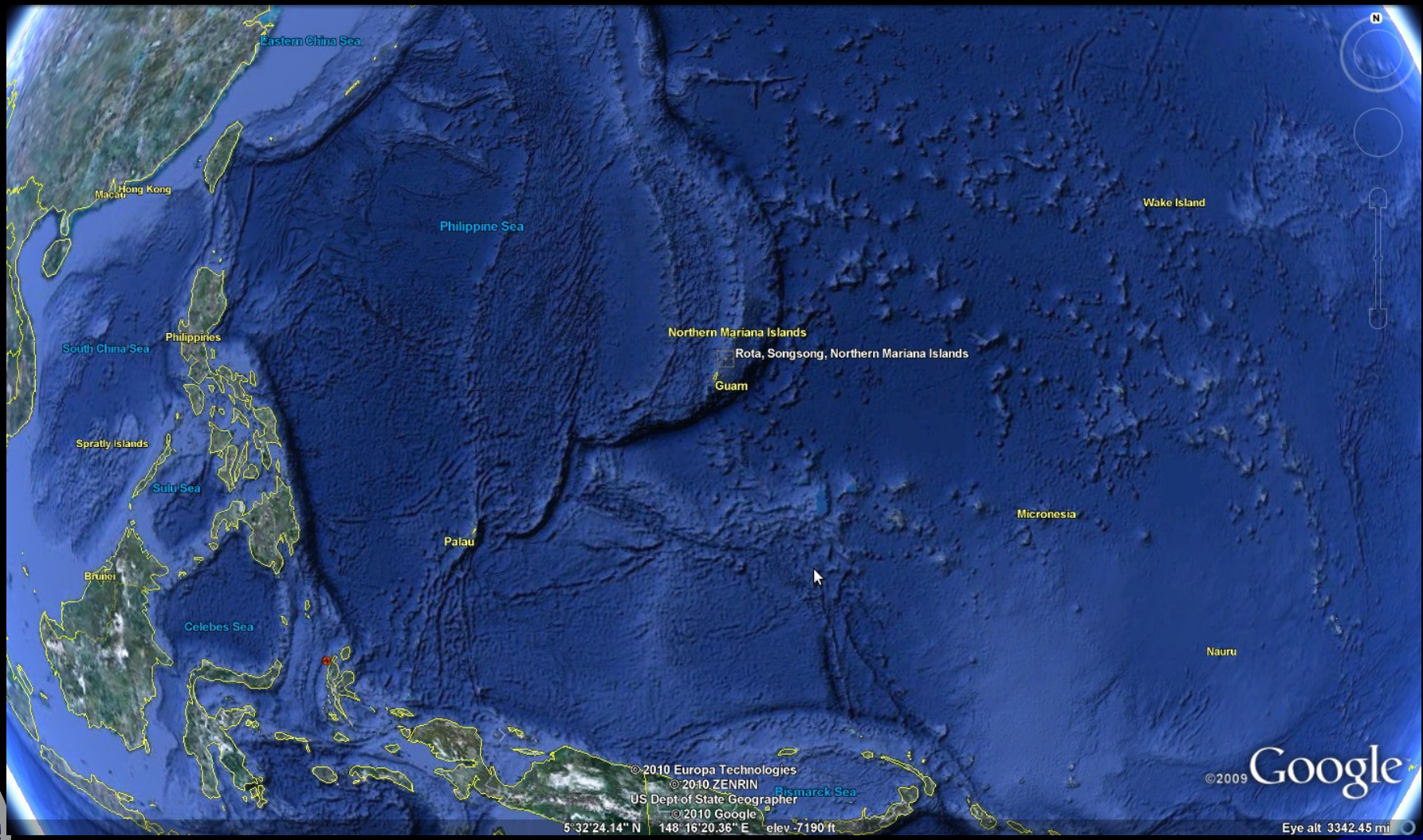




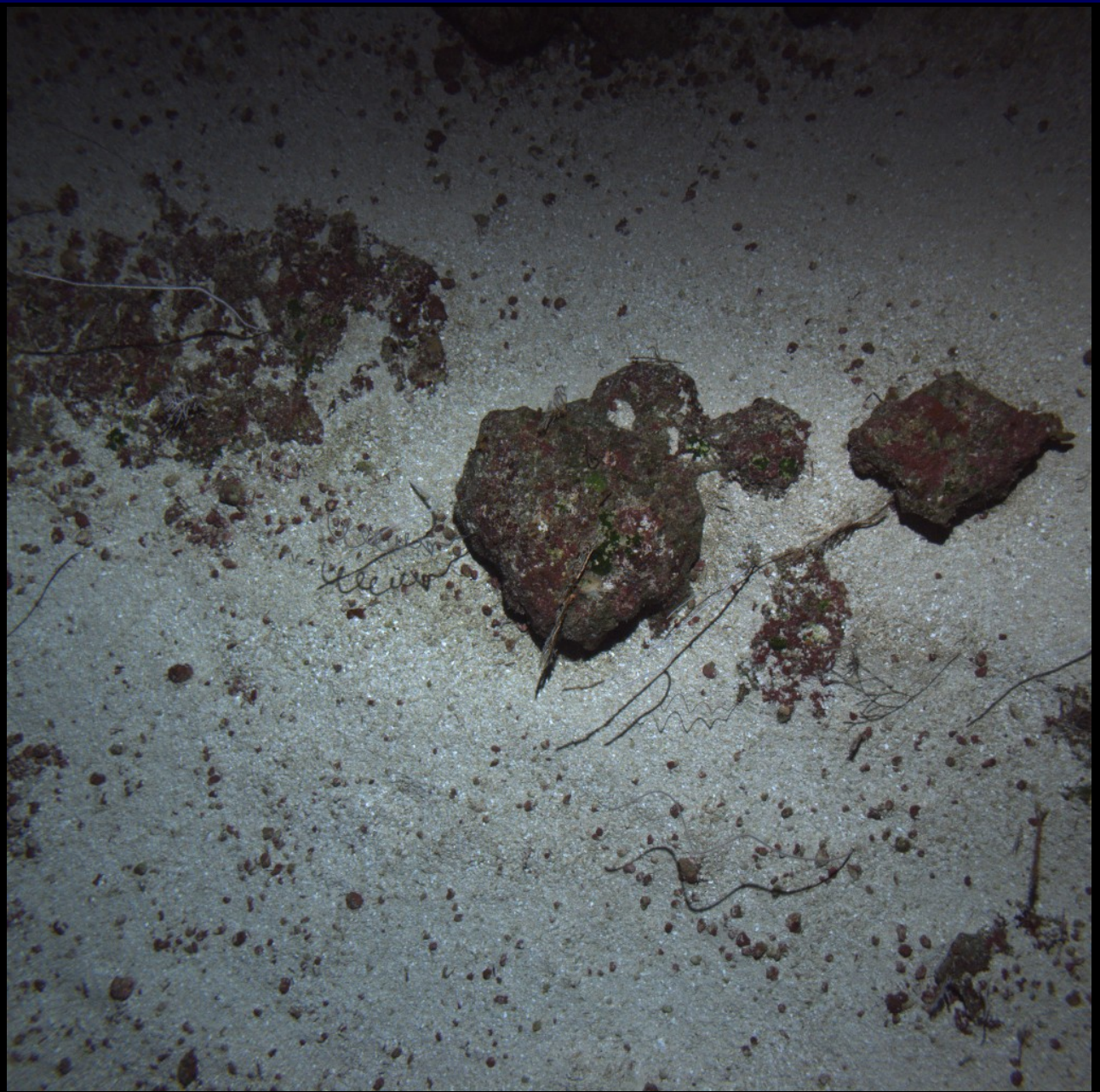
Sidescan Imagery

Live Trials near Rota

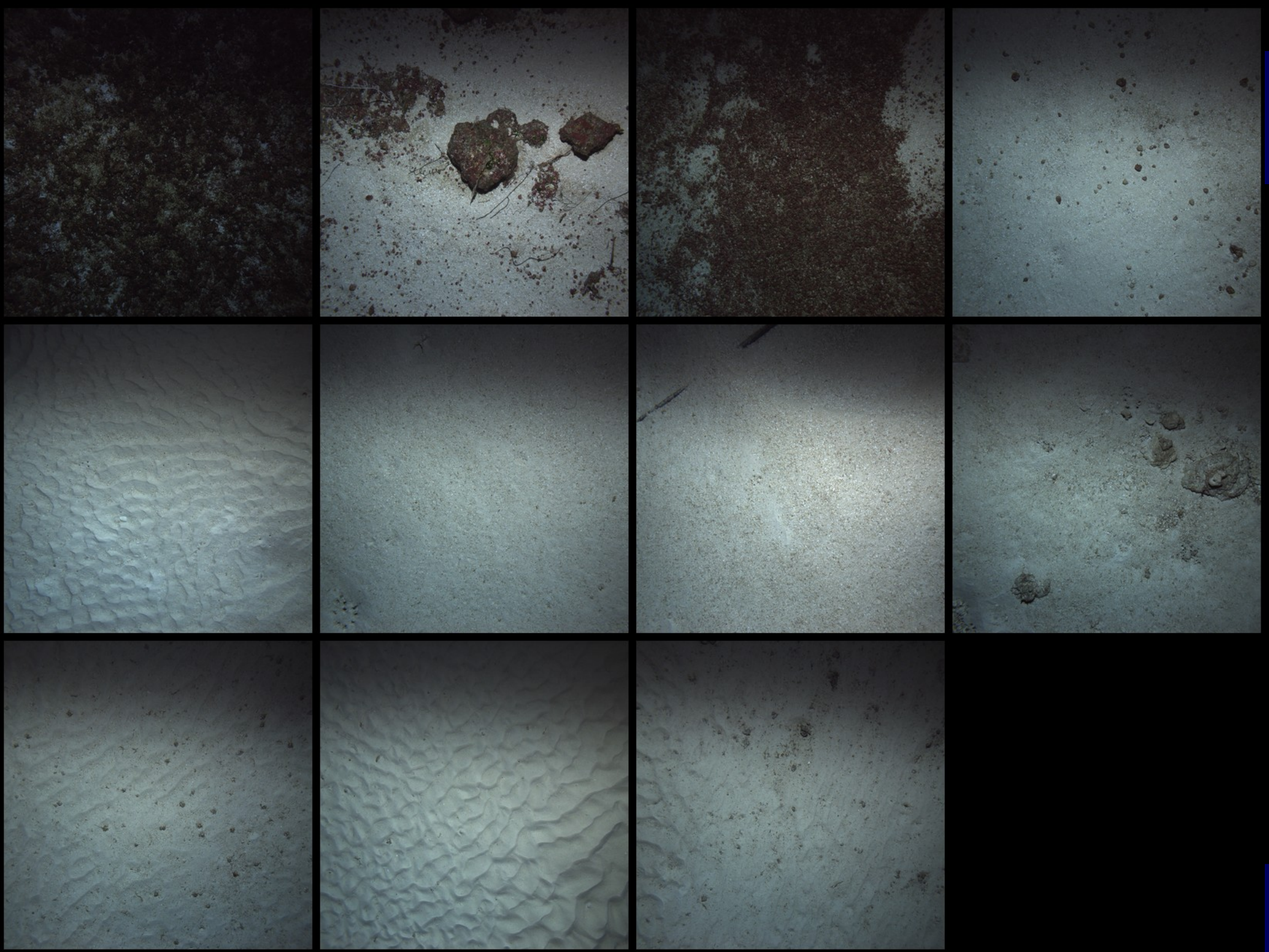
North of Guam, 1600mi east of the Philippines

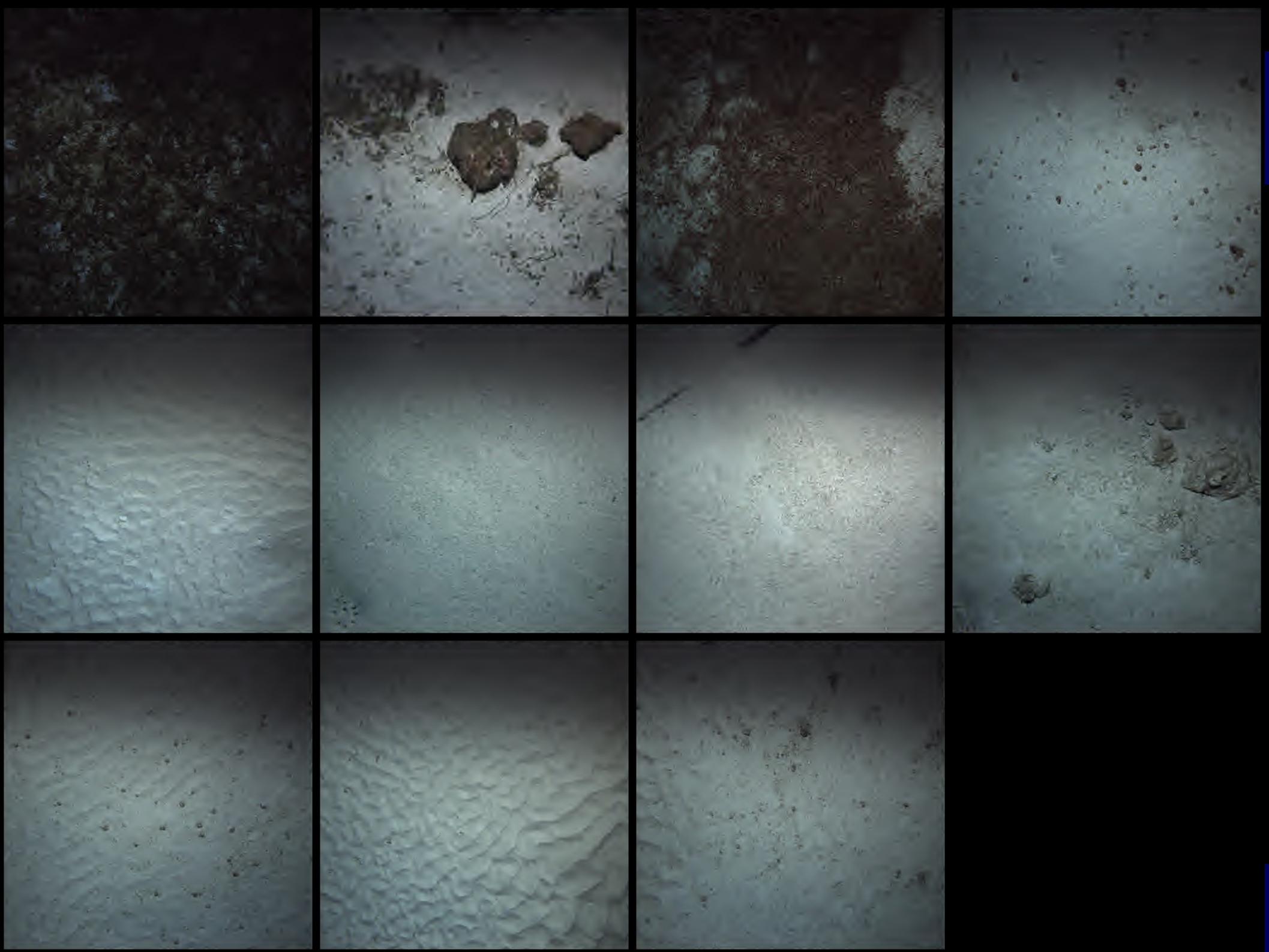


Original



4032 Bytes





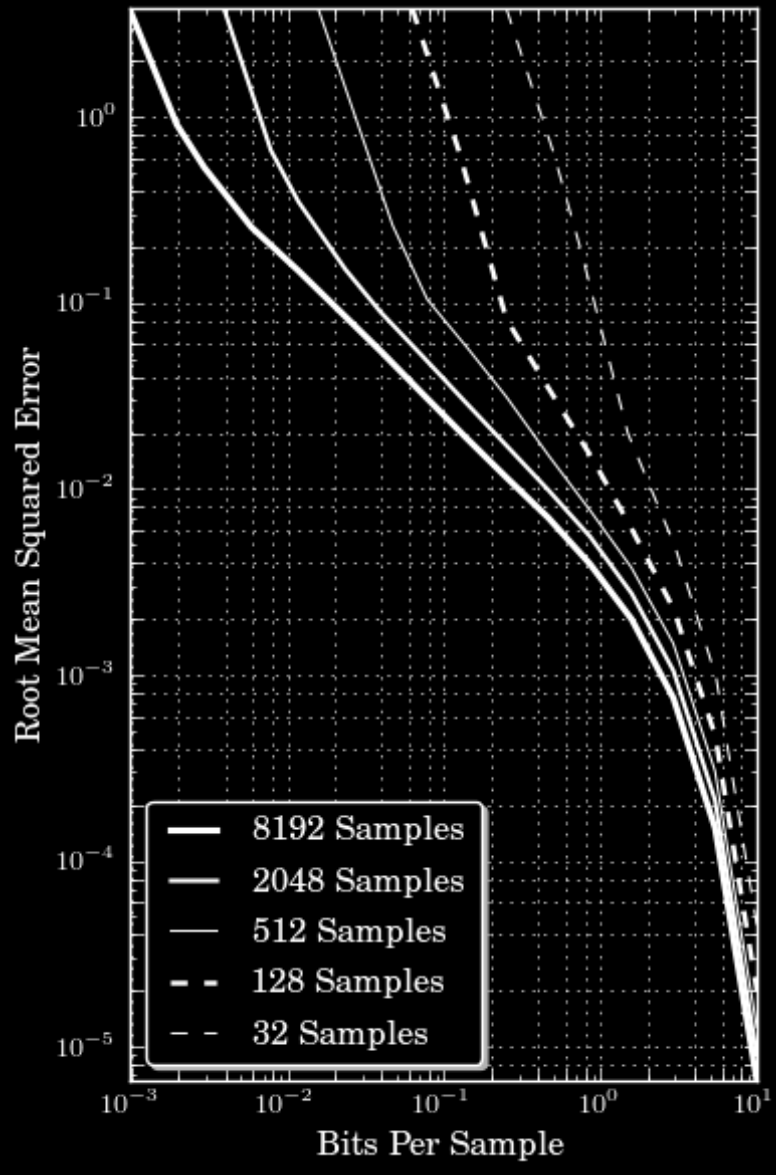

1008 Bytes
2016 Bytes
3024 Bytes
4032 Bytes
5040 Bytes
6048 Bytes
7056 Bytes
8064 Bytes
9072 Bytes
10080 Bytes
11088 Bytes
12096 Bytes

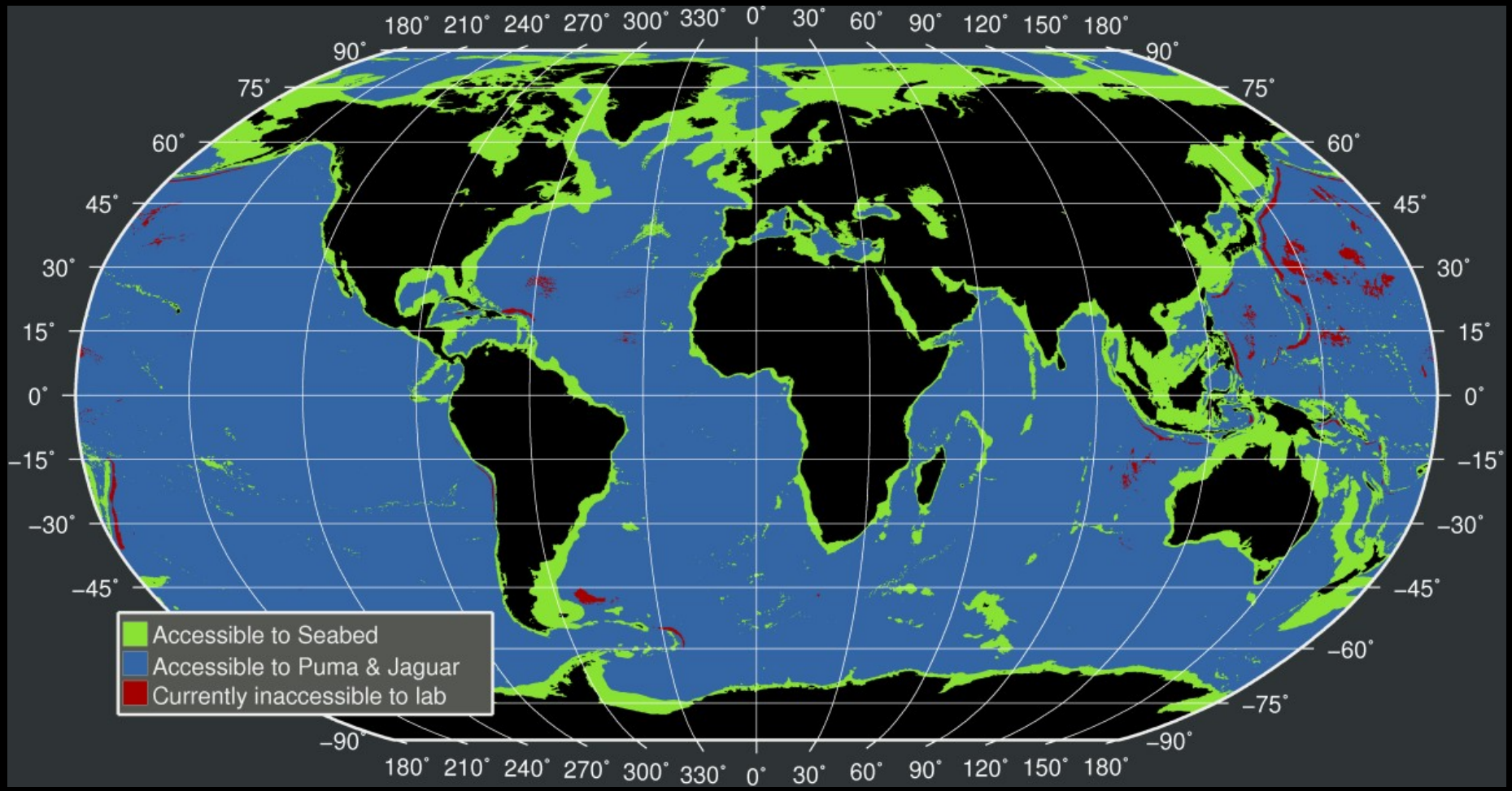
A Side Note About Compression

It is easier to compress *more* (correlated) data than less.

Thus, the less often you share information the more efficiently you can share it.

How does this affect our communication strategy?





Thank You.

chrismurf @ whoi.edu

Brief Introduction to SPIHT

Sorting Bits Indicate

- Is a coefficient "significant"?
- Are any descendants?
- Any grand-descendant?

Refinement Bits Indicate

- The sign of a coefficient
- Coefficient magnitude