CubeSat based communication system : Part II-Data compression for IR images

Alexander Gu, Phani Chevali, Ed Richter, and Arye Nehorai
Department of Electrical and Systems Engineering
Washington University in St.Louis

Abstract

The goal of this project is to examine the performance of different lossless compression algorithms and determine which one would be the most suitable for a cubeSat based communication system. As the cubeSat based communication system has both high data collection potential and relatively low data transmission throughput, optimal compression is a key performance bottleneck for the overall system performance. We select an optimal subset of the images that are generated using the IR camera mounted on a cubeSat and compress these subset of images. The compression algorithms we investigated and tested are run-length encoding, difference encoding, and Huffman coding. Initially, we implement these algorithms in MATLAB. Later, we will use the datasets developed in the first part of the cubeSat project.

Overview

Goal

Develop an optimal lossless compression strategy for use on the cubeSat platform

Approach

• mathematical optimization of different compression algorithms using modeled image data.

Applications

mobile data collecting platforms that are constrained by the communication bandwidth.

Background

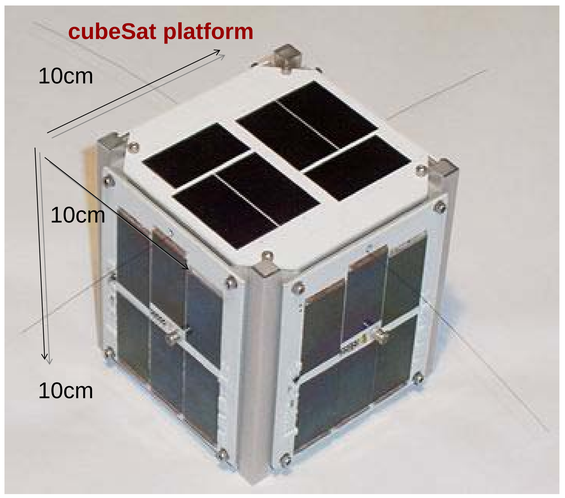


Figure: Photograph of a cubeSat micro-satellite (Source: Stensat Group, LLC)

* This result is calculated using the ascii representations of W and R

Compression algorithms

Fixed byte-length

Run-length encoding (RLE)

ORIGINAL DATA: ***WWWWRRRWWWWRW***
COMPRESSED DATA: W4R3W4R1W1

Stores one data element followed by one run-length element, representing the number of immediate repetitions.

Variable byte length

Variable run-length encoding (VRLE)

ORIGINAL DATA: ***WWWWRRRWWWWRW***
COMPRESSED DATA: W14R13W14R0W0

Similar method as RLE. Except the first bit is reserved for indicating whether a run is present or not.

Difference Encoding

ORIGINAL DATA: ***WWWWRRRWWWWRW***
COMPRESSED DATA: W000-5005000-55 *

Stores one data element followed by a difference value representing the difference between the two data elements

Huffman coding

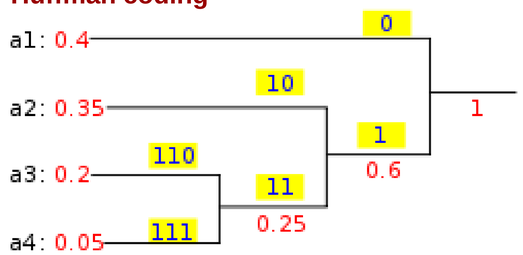
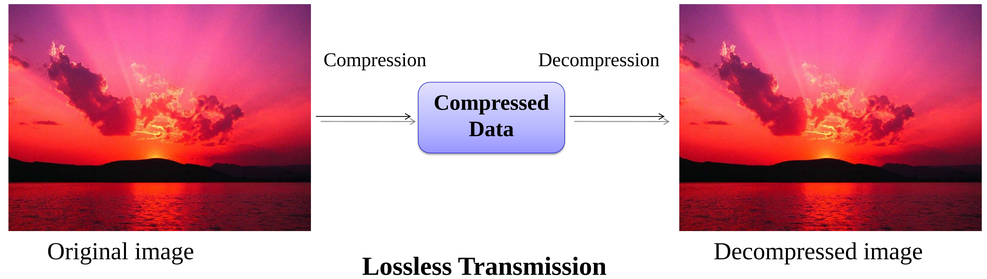


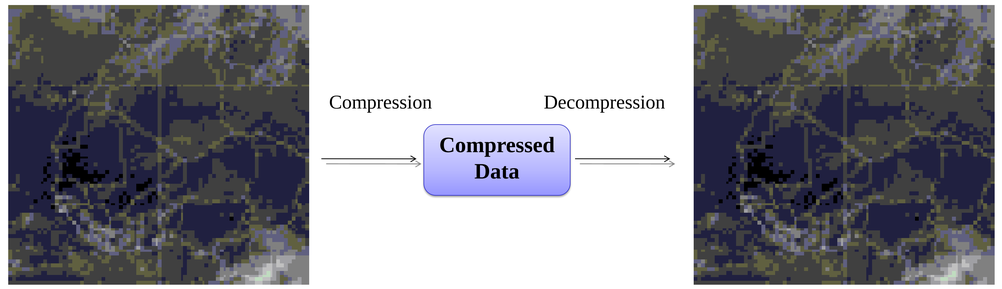
Figure: Example of Huffman coding. Alessio Damato 18 May 2007(2007-05-18)

Huffman coding assigns varying bit-length codes to each symbol based on the symbol frequency. In the above example the four symbols are a1,a2,a3,a4. based on the sorting algorithm their assigned byte-codes are 0, 10, 110, 111 respectively.

Examples of images processed

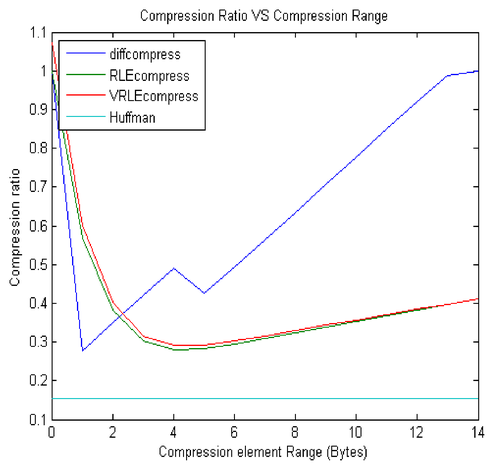


Original image — **Lossless Transmission** — Decompressed image

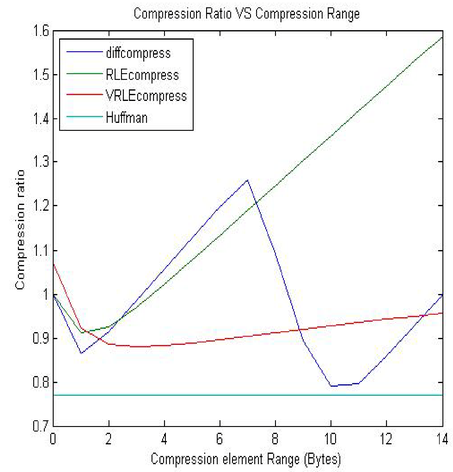


Experimental results

IR image results



Color image results



References

•Rosettacode.org

• Mordecai J. Golin, Claire Kenyon, Neal E. Young "Huffman coding with unequal letter costs" (PDF), STOC 2002: 785-791

•Wikipedia.com

• Korn, D.G.; Vo, K.P. (1995), B. Krishnamurthy, ed., Vdelta: Differencing and Compression, Practical Reusable Unix Software, John Wiley & Sons