Gigapixel Images

Douglas Peterson

Thomas O’Brien

Daniel Garcia

Matthew Ostovarpour

Version 2.1

Table of Contents

Introduction

This requirements document provides an overview of our stochastic image reading project. For an overview of the project, please see the Product Description section. A stochastic image reader is a program that randomly samples an image as opposed to using other common methods such as averaging and decimation. This project will result in a functional image reader inside of the existing Geospatial Abstraction Library (GDAL). There will also be a simple lightweight image viewer that utilizes this reader. The reader will be able to accept multiple image types to sample.Ultimately this product is an attempt to improve productivity by speeding up the time it takes to view images, thus allowing analysts to spend less time waiting for images to be preprocessed; as well as ensure that image is of an acceptable quality to be viewed. The most important requirements for this project are as follows:

1. The image reader must be faster than current alternatives
2. The image reader must be implemented inside the existing GDAL library.
3. The image viewer must have basic image viewer functions (pan, zoom, etc..)

There are risks that are involved with this as all projects such as sampling speed being equal or slower in developing than current models and algorithms used to read the images. One of the reasons for the risk could be the sampling rate of which we take for the reader. Another way for this risk to happen is the color variation leading to slow render times. There is another risk of the quality of the images being low causing the image to not be viewable or the the image taking to long to match a higher quality. All the risks depend on time development or quality of the image being fed into the reader.

Problem Statement

Our Sponsor

The sponsor for this project is the United States Geological Survey (USGS). USGS is an organization that works almost exclusively for the National Aeronautics Space Administration (NASA). They are the organization that receives the images sent to us from satellites and rovers in space. After receiving the images, USGS will analyze them to gain information on the climate of a planet, the atmosphere, or even minerals on the planet's surface. It is primarily through USGS that we receive all the data we know about Mars and the moon.

The way that USGS is able to analyze more than just what space looks like is through all the extra data embedded inside these images. When most people think of pictures, they think of something interesting to look at, but images can be so much more. The way technologies like thermal imaging work is by capturing the thermal data that we can't see, and converting it to fit into the RGB spectrum that we can see as humans. Now expand this idea a hundred times, the cameras in space capture images with hundreds of what are called channels. All we can see with our eyes is the visual light spectrum, but these cameras capture a much wider range. USGS then moves some different parts of the data into the visual spectrum and we can then see some thermal images, how hot areas are, or even small radiation from certain minerals, so we can see what is on the planet's surface. This is important for possible future terraforming of other planets since when we see what's happening on the surface we can relate it to Earth and then see what could be done for when we are able to accomplish such things as colonizing other planets. More modern and viable uses would be weather patterns and planetary mapping. Also by observing other planets we can learn more about the solar system and hopefully expand outward into understanding the galaxy.

Due to the massive amount of data that is stored in these images, they tend to be anywhere from a few gigabytes in size all the way up to terabytes. The resolution of these images is massive as well so analysts can clearly zoom in far into the planet's surface. Then thanks to the massive size of these files they can't be opened with traditional image viewers. There are few image readers out there for analysing this kind of data and most of them are slow. USGS must generate what is called pyramids from the image. What's important to know about pyramids for this project is that they take a long time to generate, and they make the images even larger than they start out as. These pyramids are used to allow analysts to read images quickly since it generates multiple images stacked on top each other so a viewer only needs to read small parts of the image pyramid at a time.

With our stochastic image reader we plan to make up for the difficulties and challenges on time they are having with the large images cut drastically. With our method we plan to cut the times at least in half. This will make it so the analysts no longer have to wait for such extreme time periods to observe the data from the images. Also with our reader being integrated into GDAL it will have access to all its features for images so our reader will be able to zoom and see images with better detail and be modified for a clearer observation.

Our Goal

Our goal is to make a reader that begins to push this pyramid preparation technique closer to being obsolete. We want to make a reader that can quickly read an image without the need to do any preparation on the image. Without any image preparation, most readers will read the entire image and average the image data to fit the screen resolution as seen in Figure 1. This process is extremely slow for large images.

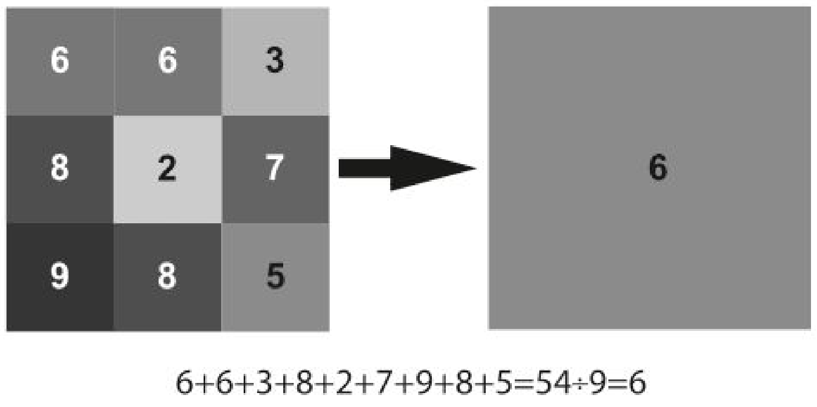


Figure 1: Demonstrating the process of pixel averaging

With our software, USGS analysts will be able to view their large images with ease. They may still generate pyramids for their images since it may be more accurate, but they may use our reader to search for points of interest while the pyramids are being generated. This will speed up the time it takes to gather important information from the images returned from space. Ultimately, our reader will increase productivity and allow us to learn more about space faster than we are now.

Our Solution

We will implement a reader inside of GDAL that will randomly sample parts of the image. An example of this is demonstrated in Figure 2. Then from the sampled parts of the image we'll generate the new image to show on the screen, by sampling the image instead of averaging the whole thing, the process should dramatically speed up.

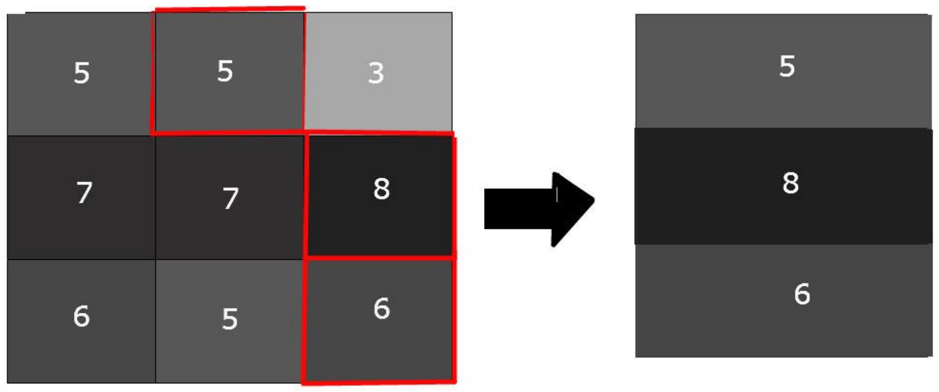


Figure 2: Demonstrating our method of stochastic sampling

This software could help reduce the image loading for all large images used by everyone making it so that no one has to wait extremely long times to see, analyze, or edit a large image. Several companies could use this software such as Google for their Google Earth images making it so that they can load and examine parts of the planet at much quicker rates and map out the area easier. For people it makes the wait time on the images significantly shorts so rather than taking eight hours to wait on an image this software could possibly get it down to two save six hours so the people like the workers at the USGS would have six more hours to look at the image and collect the data in that time rather than just waiting.

Functional Requirements

The functional requirements for our product are as follows:

1. Input
    1.1. Terminal Input
        1.1.1. User can enter the following parameters via command line:
            1.1.1.1. file to open
            1.1.1.2. window width
            1.1.1.3. window height
            1.1.1.4. sampling rate (int)
            1.1.1.5. verbose mode (boolean)
    1.2. GUI Input
        1.2.1. User will be able to interact with the window with the following controls:
            1.2.1.1. Enter/Exit zoom mode
            1.2.1.2. Enter/Exit pan mode
            1.2.1.3. Increase sampling rate
            1.2.1.4. Decrease sampling rate
            1.2.1.5. Stop sampling
            1.2.1.6. Open new file
2. Usage Modes
    2.1. Zoom mode
        2.1.1. User may zoom in on the image.
        2.1.2. User may zoom out from the image.
    2.2. Pan mode
        2.2.1. User may drag the image while it is zoomed.
    2.3. Idle mode
        2.3.1. Image quality will improve over time in this state.
3. Output
    3.1. Create a window to display the image.
    3.2. Interactive controls for viewing the images.

Environmental Requirements

1. Cross Platform
    1.1. Ability to run on Windows
    1.2. Ability to run on Mac
    1.3. Ability to run on Linux

2. Software Libraries and Programming Languages
    2.1. Software Libraries
        2.1.1. OpenGL for cross platform functionality
        2.1.2. GLEW (GL Extension Wrangler) for modern OpenGL Functionality
        2.1.3. GLFW (OpenGL framework) for cross platform windowing
        2.1.4. GDAL for reading image files
    2.2. Programming languages
        2.2.1. C
        2.2.2. C++
        2.2.3. Python
3. Other external constraints
    3.1. Our team will only be using free and open source software (FOSS) to create the reader and viewer.
    3.2. Supported image formats (courtesy of GDAL)
        3.2.1. http://www.gdal.org/formats_list.html
        3.2.2. http://www.gdal.org/ogr_formats.html

Non-Functional Requirements

The reader must perform faster than readers that are currently available.

The most important part of our reader is that it needs to render gigapixel images much faster than the current standard. The current method is to build pyramids to render the images slowly after which they are available to be viewed. We are using random sampling (stochastic sampling) to drastically increase the speed in which images are rendered. A current render of an image not using stochastic sampling takes around one to ten minutes. With stochastic sampling we plan on rendering images in about ten seconds.

The reader must produce a visible, quality image.

Our reader must also produce an image of an acceptable quality. The pyramid method of rendering already produces high quality images which are to our sponsors standards. We don't expect our reader to produce an image of the same quality, but we do expect it to meet the standards that our sponsor desires. Stochastic sampling with a small sample size could result in a rendered image with very poor quality. Our reader must have a minimum sample size where the quality is at its lowest acceptable standard.

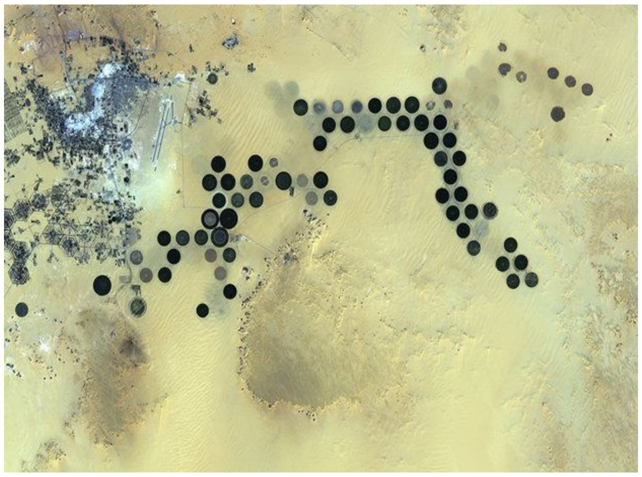
The emphasis of our reader is on speed, not quality, but both of these factors are important for the project.

Potential Risks

The biggest risk for our reader is getting the stochastic sampling rate incorrect. Our reader relies upon a good balance in the sampling between number of samples taken and time needed to render the image. The number of samples taken will directly impact the quality of the image that is rendered and the time needed to render that image. To get a high quality image we will need more samples but more samples will lead to a longer rendering times.

Since our goal is to increase the speed of rendering, we want to take as few samples as possible while still maintaining a certain level of image quality. The effects of this risk could lead to our software either producing an image that is unviewable due to its low quality or it could lead to render times that are not fast enough to warrant using the software. This risk could make our software unusable or inferior to current options.

There is also a risk in a high color variation within the sampling which can also lead to slower reading. Our reader will take into account the color variation of the image when it is calculating the sampling rate needed for a quality image. With an image that has a low color variation, as seen in Figure 3, less samples would be needed to render an accurate representation. This would lead to fast render times while maintaining viewability. As color variation increases, the sampling rate needed will also increase to maintain a minimum level of quality. Figure 4 is an example of possible high color variation. This could lead to longer render times than desireable for these images.



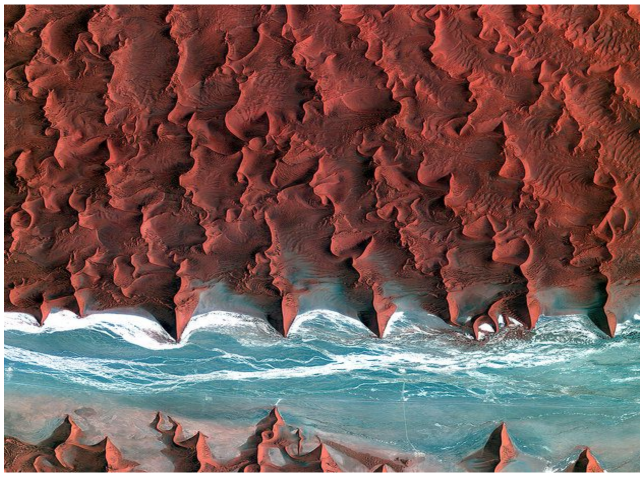
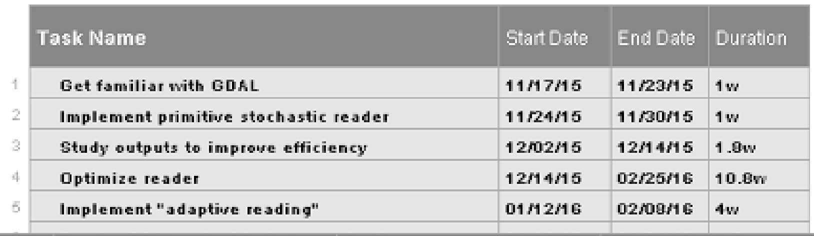


Figure 3: Man-made oasis in the Sahara Desert	Figure 4: Namib desert coastline
Image Credit: European Space Agency	Image credit: European Space Agency
Image URL: http://www.esa.int/Our_Activities/Observing_the_Earth/Earth_from_Space_Sahara_oasis	Image URL: http://www.esa.int/Our_Activities/Observing_the_Earth/Earth_from_Space_Dune_45

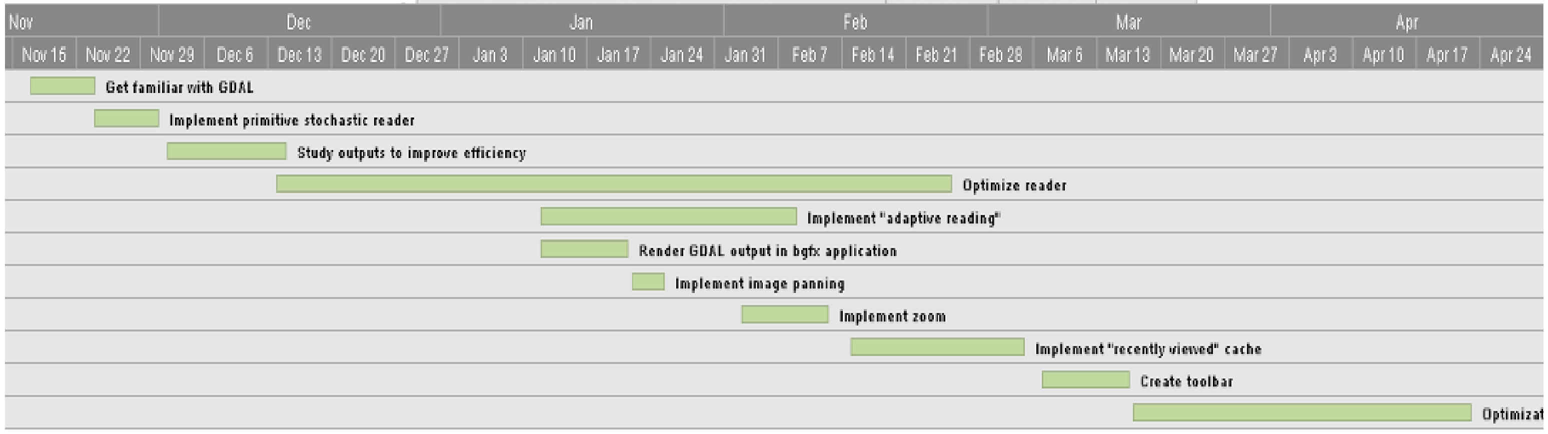
Project Plan

1. Get familiar with GDAL
2. Implement a primitive stochastic reader
3. Study the reader’s outputs to find the most efficient outputs based on viewing area
    a. Through trial and error we will find the optimum settings to achieve a desired balance between quality and speed.
4. Implement “adaptive reading” feature
    a. This feature will analyze a section of the image and its color variation. Depending on the level of the variation the sampling rate will change. There will be a higher sampling rate for higher color variation and a lower rate for lower variation.
5. Optimize the reader
    a. Work on fixing bugs and speeding up the rendering process.
6. Render the GDAL output in OpenGL application
7. Implement image panning
    a. Panning will take an input from the user’s keyboard or other methods to allow the user to pan around the image while zoomed in.
8. Implement image zoom
    a. Image zoom will also take a user input, either from the keyboard or a mouse wheel to allow the user to view a section of the image with more clarity.
9. Implement a “recently viewed” cache
    a. This will help speed up load times while viewing as the recently viewed images can be quickly added back when needed.
10. Create a toolbar for the GUI
    a. A toolbar will allow for ease-of-use while interacting with our software.
11. Optimization/Optional Features

Schedule of plans and completion dates



	Task Name	Start Date	End Date	Duration
1	Get familiar with GDAL	11/17/15	11/23/15	1w
2	Implement primitive stochastic reader	11/24/15	11/30/15	1w
3	Study outputs to improve efficiency	12/02/15	12/14/15	1.8w
4	Optimize reader	12/14/15	02/25/16	10.8w
5	Implement "adaptive reading"	01/12/16	02/08/16	4w



Glossary

GDAL - Geospatial Data Abstraction Library

A library for reading and writing geospatial data forms. Used by many different applications requiring vector data formats such as Google Earth.

Stochastic Sampling -

A form of random sampling where an object to be sampled is split into sections. Each section has random samples pulled from it to represent the entire section. Those samples are then re-arranged to form the newly rendered image.