# Aerial Mapping of Mining Sites for Environmental Reclamation

A Major Qualifying Project (MQP) Report
Submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements
for the Degree of Bachelor of Science in

Mathematical Sciences,
Robotics Engineering

By:

Yiyi Wu

Project Advisors:

Berk Calli
Robert Krueger
Sarah Olson

Sponsored By:

Ghana Project Center

Date: July 2023

# Abstract

This project aimed to create a detailed map of a pilot mining site in Ghana utilizing drone imagery and employing photo stitching techniques. An OpenCV [1] stitching package was initially used to stitch the aerial photos together, yielding satisfactory results overall. However, challenges arose in areas of the site with extensive tree coverage, leading to sub-optimal stitching outcomes. To address this issue, a MATLAB algorithm was developed, leveraging GPS data and edge detection techniques to enhance the stitching process for images containing trees. Although the resulting map was not flawless, significant improvements were observed in areas characterized by dense tree coverage. This project highlights the potential of integrating MATLAB algorithms to enhance the accuracy and completeness of aerial mapping in challenging environments.

# Contents

# List of Tables

# List of Figures

# 1 Introduction

Mining activities, while crucial for economic development, often leave behind significant environmental challenges, including land degradation and ecosystem disruption. One effective strategy for mitigating these issues is through the process of site reclamation, which aims to restore the land to a functional and sustainable state. In the context of a pilot mining site in Ghana, this project focuses on the ultimate goal of planting and growing bamboo to facilitate the reclamation process. However, before the reclamation efforts can commence, it is crucial to create an accurate and detailed map of the site, which necessitates the utilization of drone technology and advanced image stitching techniques.

## 1.1 Project Objectives

The primary objective of the current project is to create a comprehensive map of the pilot mining site, encompassing its entire extent and capturing crucial details of the terrain. This map serves as a foundational component for the subsequent reclamation efforts, providing an essential reference for planning, monitoring, and implementing the bamboo planting process. By accurately mapping the site, potential areas for reclamation can be identified, and suitable strategies for planting and growing bamboo can be devised.

### 1.1.1 Utilizing A Drone

To achieve the goal of mapping the mining site, the project employs the use of drone technology. Drones offer a unique advantage in capturing high-resolution aerial imagery, allowing for a broader perspective and more precise documentation of the site's characteristics. By employing a drone equipped with a high-quality camera, multiple photographs can be captured from different angles and altitudes.

### 1.1.2 Computer Vision: Image Stitching

Individual images obtained from the drone require integration to create a cohesive and detailed map of the entire site. This is accomplished through the process of image stitching, which involves aligning and merging the individual photographs to form a seamless composite image. Image stitching techniques, such as those offered by an OpenCV stitching packages [4], are commonly utilized for this purpose, enabling the creation of panoramic images by automatically aligning and blending the overlapping areas of the photos.

## 1.2 Challenges and Limitations

While the utilization of drone technology and image stitching techniques presents an efficient approach to mapping the pilot mining site, certain challenges and limitations need to be considered. The presence of complex terrain, such as dense vegetation and rugged topography, can pose difficulties during the image stitching process. In particular, capturing accurate images of areas with dense tree cover can present challenges in achieving seamless integration due to the intricate nature of the foliage.

To address this limitation, additional measures will be implemented. This project will explore the development of a MATLAB algorithm that integrates GPS data and edge detection techniques to improve the stitching process for tree photos. By incorporating location information and enhancing the identification of edges within the images, a broader and more accurate representation of the site can be achieved.

In conclusion, the primary goal of this project is to create a comprehensive map of the pilot mining site in Ghana, laying the foundation for subsequent bamboo planting and reclamation efforts. Through the use of drones and advanced image stitching techniques, a detailed understanding of the site's terrain will be achieved, aiding in the planning and execution of the reclamation process.

# 2 Background

## 2.1 History of Gold Mining in Ghana

In Western Africa, there is a rich history of gold mining. From pre-colonial times to the present, the Ancient Kingdom of Ghana, the former Gold Coast Colony, and Ghana today, are heavily involved in mining activity and relevant events.

Ghana is home to some of the world's most significant gold deposits, formed by tectonic processes and erosion over millions of years. Gold trading with the Arab world started in the 9th century AD when North African merchants discovered the gold-rich region and established a monopoly. The Ancient Kingdom of Ghana was a powerful and wealthy state from the late thirteenth to the fifteenth century, which taxed and exported gold and other goods [5].

Pre-colonial mining in Ghana was done with simple methods, such as using pans and sieves to extract gold from streams and rivers, then smelting to produce gold ornaments. When the British set foot on the ground of Ghana, scientific methods of mining were also introduced. In the late 1800s, the implementation of advanced scientific mining techniques in Ghana allowed European mining corporations to dominate the industry, effectively barring Ghanaians from active participation in gold mining except for roles as laborers and artisanal miners [6].

In December 2018, after the ban on gold mining was lifted, small-scale mining rose in rural towns of Ghana. The Minerals Commission established Satellite Offices in various mining districts to provide support to the District Offices responsible for overseeing mining operations. However, the challenges faced by regulatory agencies in Ghana including the remote, inaccessible, and geographically dispersed nature of small-scale mining operations, which make it difficult for them to monitor these activities effectively. There was also the concern of security threats faced by officers, as well as the presence of illegal mining operations [7].

## 2.2 Artisanal Gold Mining Process

Artisanal and small-scale gold mining (ASGM) is a widespread activity in Ghana and has been so for centuries [8]. The process involves various methods that range from rudimentary to technologically advanced, and the scale of operation varies significantly.

The initial process begins with the identification of potential mining sites, usually located near or in river bodies where gold deposits are suspected. Once the site is selected, prospecting, the first step of gold mining, begins [9]. This involves the use of simple tools to dig small pits or trenches to confirm the presence of gold.

Subsequent to prospecting is the extraction phase. Typically, miners dig into the earth's crust using basic tools such as pickaxes and shovels. The extracted material is then sifted and washed to separate the heavier gold particles [10]. The use of mercury amalgamation is a common practice in Ghana, despite its detrimental environmental and health effects [11]. The gold-mercury amalgam is then subjected to heating, which evaporates the mercury, leaving behind the gold.

Reclamation, the final step in the mining process, is often poorly practiced or entirely overlooked in the artisanal mining sector in Ghana, contributing significantly to environmental degradation [12].

## 2.3 Environmental Impact of Gold Mining

Small-scale mining activities contribute to nearly a third of Ghana's total gold production [13], but the prevalence of illegal mining presents a significant challenge. Gold mining has had a markedly adverse impact on the environment, leading to land degradation, water bodies becoming muddy and contaminated with heavy metals.

As one of the most commonly seen ways of mining, artisanal miners typically create numerous shallow pits spread out over extensive areas of land. They extract and process gold on-site before moving to the next location, often neglecting to refill the excavated pits. Additionally, they frequently employ mercury to extract gold from sediments, which can result in significant health concerns and prolonged soil and water pollution. Researchers from NASA, United States universities, and Ghanaian government agencies employed satellite data to evaluate the extent of vegetation degradation resulting from artisanal mining in the southwestern region of Ghana, where most of the country's gold mining occurs. Their findings indicate that artisanal mining activities caused approximately 24% of the region's vegetation loss between 2005 and 2019 [14].

## 2.4 Human Health Effects of Gold Mining

The contamination of soils in gold-mining regions by heavy metals is a a well-known phenomenon. This contamination arises from various mining activities such as milling, grinding, ore concentration, and the disposal of tailings, all of which contribute to the use of heavy metals in mining environments. These toxic substances pose a threat to living organisms, and numerous studies have provided evidence linking their association with atmospheric particulate matter to respiratory and cardiovascular diseases. Consumption of contaminated food crops has also been reported to result in severe health problems and even fatalities in humans. The intake of high levels of heavy metals through the food chain can trigger allergic reactions and contribute to the formation of complex toxic compounds, ultimately causing significant harm to biological functions [15].

## 2.5 A Sustainable Solution: Cultivating Bamboo in Ghana

Bamboo is an indigenous plant in Ghana, and as some recent research showed that bamboo can help the reclamation of the land, it presents a viable solution for remediating soil in abandoned mining sites. The growth of bamboo in such areas will hold significant benefits for both the environment and human health.

Moreover, sustainability is crucial, particularly when considering the needs of the local community. Therefore, implementing a localized approach becomes imperative, involving the utilization of indigenous plants and objects, employing methods that are locally friendly, and respecting local conventions. By embracing these principles, bamboo cultivation emerges as an effective and culturally appropriate strategy to address soil remediation challenges, fostering environmental restoration and promoting the well-being of the local population.

### 2.5.1 Previous Research of Bamboo in Soil Remediation

According to a recent case study done by Bian et al [16], bamboo is a rapidly growing woody grass species, exhibiting a remarkable ability to achieve significant biomass productivity within a short time frame. Bamboo displays vigorous growth and attains its full size in approximately 120–150 days, while shoots typically require five years to develop into mature culms. Unlike many other woody plants commonly employed for remediating heavy metal-contaminated soil, bamboo species offer several additional advantages. These include a high tolerance to heavy metals, considerable absorption capacities, notable efficacy in heavy metal removal, and ease of cultivation. Heavy metals tend to accumulate predominantly in the root system of bamboo tissues. Although bamboo does not possess hyperaccumulation properties, it remains highly valuable for phytoremediation purposes owing to its impressive productivity, rapid growth cycle, and substantial economic worth.

### 2.5.2 Akyem Dwenase: The Local Village

Dwenase (Figure 1) is a village in the Akyem area, and the pilot site of this project belongs to this village. When driving, it takes about 10 minutes to reach the abandoned mining site from the village. The village gets its water from a river nearby, and a river nearby the mining site. We bought our drinking water from a shop in the next village, and the kids from Dwenase would ask for the tap water from us, which comes from the water tower in the village sourcing from the rivers. As discussed in the previous section, heavy metals is harmful to the human body, and the kids in the village are exposed to the polluted water in many ways. The reclamation of the mining site can lower the risk of exposing to heavy metals, let along the other benefits.

Figure 1: A view of the village from the water tower.

## 2.6 Mapping a Pilot Site in Ghana

It was important to identify the concentration of the hazardous elements in different areas, which was the task of the biochemical part of this project. Combining the data collected with a most up to date map would be helpful to present the data, bamboo planting planning, and the ability to pinpoint exact locations.

A drone was used to take photos of the site, and two stitching algorithms were used to combine the photos to a complete map of the site.

The drone that was used to take the photos was DJI Mavic 2 pro. It has a default flight height limit of 137-meter, maximum flight distance of 18-kilometer under no wind conditions, and a camera. The camera has lens with field of view of about 77-degree, still Image size of 5472*3648, resulting in about 20-million pixels.

## 2.7 Photo Stitching

Photo stitching involved a few essential segments: determining the region(s) of interest (ROI); finding feature-points; matching the feature-points; producing transformation matrices based on the matched features; stitching the images.

All the steps above are crucial, but in this project, modifications were only made to determine the ROI, as it affects the following step of finding the feature-points. A flowchart of the process is shown as Figure 2.

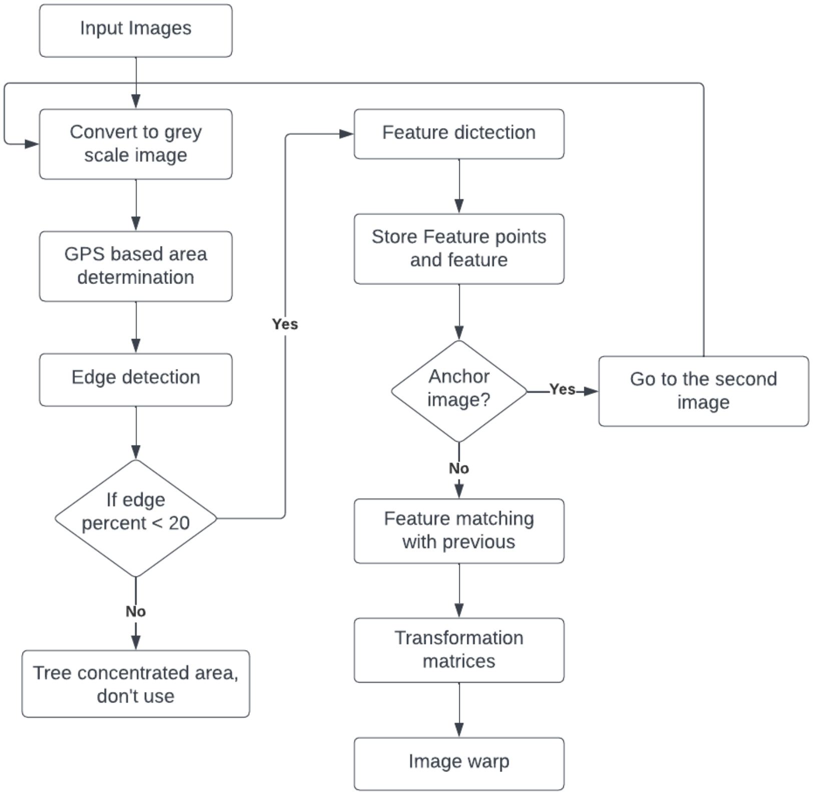

Input Images
Convert to grey scale image
GPS based area determination
Edge detection
If edge percent < 20
Yes
No
Tree concentrated area, don't use
Feature dictection
Store Feature points and feature
Anchor image?
Yes
Go to the second image
No
Feature matching with previous
Transformation matrices
Image warp

Figure 2: A flowchart of the photo stitching process used in this project.

### 2.7.1 Determine the Region of Interest (ROI)

In most cases, ROI is not specified. Feature detection algorithms would just take in the whole image and find feature-points. Specifying the ROI tells the feature detection algorithm to only look at the specified region and find feature-points in that region. Figure 3 shows an example of the strongest 20 feature points.

In this project, the mining site was surrounded by trees and the stitching algorithm kept finding feature-points in the trees, which were not good feature-points and confused the algorithm. How the feature-points were determined will be explained in the later sections, but in our case, it was easier to specify the ROI than to modify the logic of feature detection.

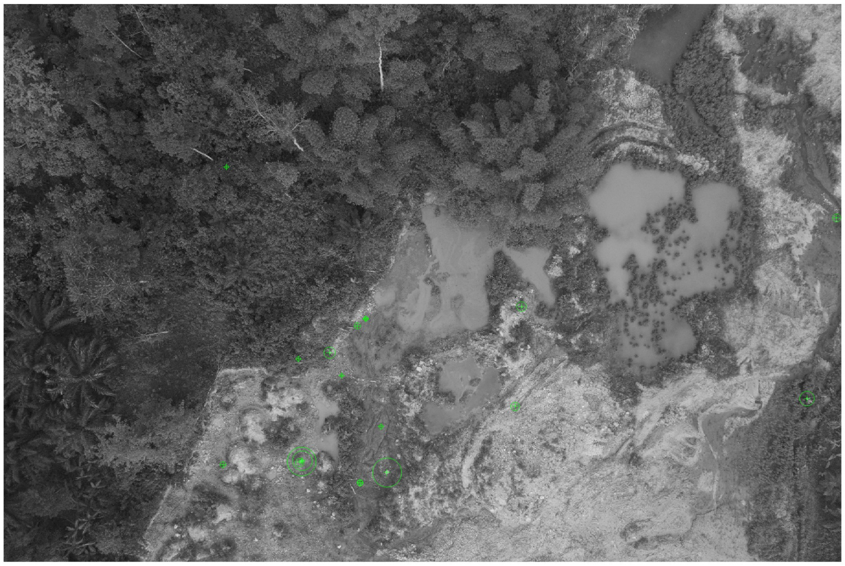


Figure 3: 20 Strongest feature points in a gray-scale photo.

### 2.7.2 Edge Detection algorithms

When it comes to edge detection algorithms, several popular techniques are commonly employed, each with its own characteristics and advantages. In this section, four widely used edge detection algorithms will be examined: Sobel edge detection, Prewitt edge detection, LoG (Laplacian of Gaussian) edge detection, and Canny edge detection.

The Sobel operator uses two 3x3 convolution kernels, one for horizontal changes and one for vertical. The kernels are applied to the image separately, such that two measurements of the gradient will be produced in each horizontal and vertical direction ($Gx$, $Gy$). Suppose the image to be processed is I, then the horizontal changes $G_x$ is expressed as:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix},$$

and the vertical changes $G_y$ is expressed as:

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}.$$

As shown in the expressions ($Gx$,$Gy$) above, rotate one kernel by 90 degrees, results in getting the other.

The magnitude of the gradient at each point and the orientation can then be found by combining them [17]. Using that, the Sobel operator finds high changes in gradient, which indicates a major change or an edge in the image.

The Prewitt operator, like the Sobel operator, uses two 3x3 convolution kernels, one for horizontal edges and one for vertical edges. Although it is similar to the Sobel operator, it does not place as much emphasis on pixel intensity changes at the diagonal directions [18].

"The Laplacian is a 2-D isotropic measure of the 2nd spatial derivative of an image” [19]. Similar to Sobel, the Laplacian operator identifies areas in the iamge where there are rapid changes in intensity. The Laplacian operator is usually applied to the image after smoothing it with a Gaussian filter. The Laplacian $L(x, y)$ of an image $I(x, y)$ of pixel intensity values is defined as:

$$L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}.$$

The Canny edge detector, proposed by John Canny in 1986, is known for its low error rate, meaning that it effectively detects true edges and rejects non-edges. It is also recognized for its ability to accurately mark edges and maintain localization by detecting the edges as close as possible to the real edge [20]. Canny edge detection is a multi-step algorithm, and a quick breakdown of the steps are:

1. Noise filtration: Canny edge detection uses Gaussian filter to compute a mask, and uses it by standard convolution methods, meaning the mask covers a square of pixels at a time. The sensitivity to noise is higher when the width of the mask is smaller.

2. Gradient of the image: To find the gradient of the image, the Sobel operator was used in both horizontal

and vertical direction to get first derivative in both directions as $Gx$ and $Gy$. The edge gradient and direction for each pixel can be found with the following formulas:

$$EdgeGradient(G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle(\theta) = \arctan\left(\frac{G_y}{G_x}\right).$$

3. Non-maximum Suppression: Non-maximum suppression follows the edge in the edge direction if a pixel is not considered to be an edge, that pixel's value will be suppressed to be 0.

4. Hysteresis thresholds: Hysteresis uses a high and a low threshold to eliminate making an edge look inconsistent. If a pixel has a value greater than the high threshold it is marked as an edge pixel, and the pixels connected to it that have values greater than the low threshold are also considered as an edge. But anything in between the high threshold and low threshold or lower than low threshold, it is considered non-edge.

The four operators each has their own advantages and disadvantages. Sobel is computationally inexpensive; Prewitt is similar to Sobel, but places less emphasis on pixel intensity changes at the diagonal directions; LoG is more correct finding edges but more computationally expensive compared to Sobel and Prewitt; Canny often yields the best results but it is the most computationally expensive. Figure 4 is a photo of a dough that has been converted to gray-scale, Figure 5 is the results of applying the four edge detection operators.

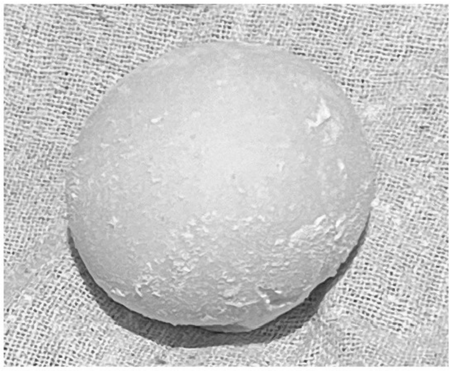


Figure 4: Photo of a dough that was converted to gray scale image.

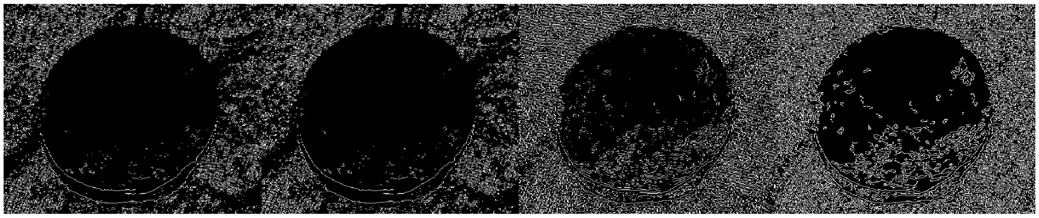


Figure 5: Montage of four kinds of edge detection methods, left to right are: Sobel, Prewitt, LoG, and Canny.

### 2.7.3 Feature Detection

A feature-detector is an algorithm designed to identify points of interest in an image, also known as feature-points, interest-points, or key-points. These features are typically detected as corners, blobs, edges, junctions, or lines. The features are small regions that exhibit variations in properties, different algorithms use different properties such as brightness or color, in comparison to the surrounding areas. Differences between the types of features is whether the distinct regions are more likely to be that kind of feature. For example, as shown in Figure 6, a corner-feature detecting algorithm FAST finds pixel regions (points) with large differences in brightness compared to their neighboring pixels and identifies as corner points [2]. Once the features are detected, they are described using various methods that capture the unique patterns exhibited by the pixels surrounding them. This step is referred to as feature description, as it assigns a distinct identity to each feature, allowing for effective recognition and matching [21].

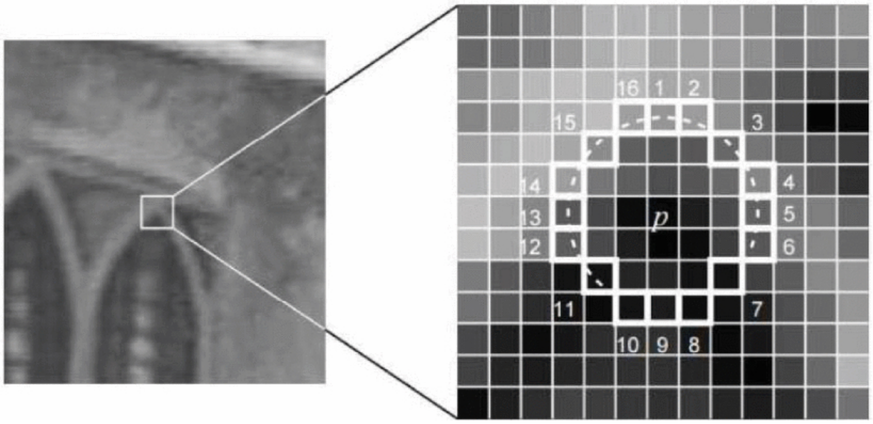


Figure 6: A key point detected with FAST feature detection. Taken from [2].

There are many methods for feature detection. The ones that were used/tested during this project are Binary Robust Invariant Scalable Key-points (BRISK), Scale-Invariant Feature Transform (SIFT), and Speeded-Up Robust Features (SURF). BRISK detects the features as corners, SIFT and SURF detects the features as blobs. All these algorithms have their own feature-descriptor and process their own merits and demerits, thus the term feature-descriptor would be implied in the rest of the report, as no customization was done for the project. Feature points possess a distinct quality in that they remain consistent regardless of changes in the image, changes of transformations such as rotation, scaling, translation, or distortion (such as projective transformation or homograph), the same feature points present in the original image should still be identifiable in the modified image when compared.

#### 2.7.3.1 BRISK Feature Detection Overview

BRISK is a binary descriptor that is efficient and robust to various image transformations. It combines the FAST keypoint detector and the BRIEF descriptor, which significantly improves upon the speed of other feature detection algorithms [3]. The algorithms contains three main steps:

1. Scale-space construction: BRISK constructs a scale-space using a method similar to the Difference of Gaussians used in SIFT. However, instead of using Gaussian filters at various scales, BRISK uses a simple resampling strategy to save computational time.

2. Keypoint detection: BRISK uses FAST (Features from Accelerated Segment Test) as a basis for its corner detection. The pattern is a circular arrangement of points sampled at different radii corresponding to different scales. Local maxima over scale and space are selected as keypoints. Keypoints are further refined through subpixel localization and scale estimation.

3. Keypoint descriptor: For each detected keypoint, BRISK calculates a binary descriptor. This descriptor is built by sampling intensity pairs around the keypoint and comparing their values. The descriptor is rotation invariant, as it rotates the sampling pattern according to the orientation of the keypoint. Figure 7 illustrates $N$ locations equally spaced on circles concentric with the keypoint.

#### 2.7.3.2 SIFT Feature Detection Overview

SIFT[22] is a highly robust method for detecting and describing local features in images. It is based on Difference-of-Gaussian (DoG) which is an approximation of Laplacian of Gaussian (LoG). The SIFT algorithm can be broken down into four main steps:

1. Scale-space Extrema Detection, identify key points that are invariant to scale changes.

2. Key point Localization.

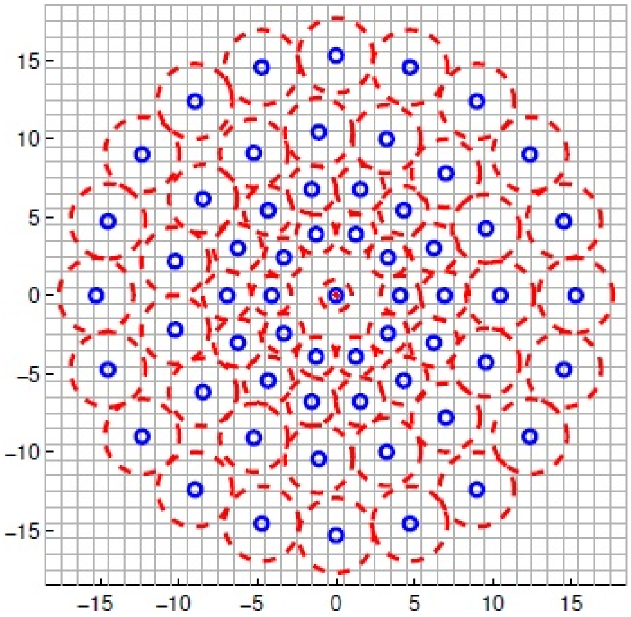


Figure 7: Keypoint pattern, taken from [3].

The BRISK sampling pattern with $N = 60$ points: the small blue circles denote the sampling locations; the bigger, red dashed circles are drawn at a radius $\sigma$ corresponding to the standard deviation of the Gaussian kernel used to smooth the intensity values at the sampling points. The pattern shown applies to a scale of $t = 1$.

3. Orientation Assignment.

4. Key point descriptor.

The scale space $L(x, y, \sigma)$ of an image $I(x, y)$ is defined as a convolution of the image with a Gaussian kernel $G(x, y, \sigma)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y).$$

The DoG is the difference between two successive Gaussian blurred images which can be calculated with:

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y).$$

#### 2.7.3.3 SURF Feature Detection Overview

SURF is a robust and efficient feature detection and description algorithm designed to improve upon the speed and performance of previous methods like SIFT (Scale-Invariant Feature Transform) while maintaining robustness to scale and rotation changes [23]. The SURF algorithm can be divided into three main steps:

1. Scale-space construction: SURF creates a scale-space using box filters that approximate the Laplacian

of Gaussian (LoG) used in SIFT. The advantage of box filters is that they can be computed quickly using integral images.

2. Key point detection: Potential key points are those points where the determinant of the Hessian matrix is maximum. The Hessian matrix at scale $\sigma$ at a point $(x, y)$ in the image $I$ is defined as:

$$H(x, y, \sigma) = \begin{bmatrix} L_{xx}(x, y, \sigma) & L_{xy}(x, y, \sigma) \\ L_{xy}(x, y, \sigma) & L_{yy}(x, y, \sigma) \end{bmatrix}.$$

   Here, $L_{xx}$, $L_{xy}$, and $L_{yy}$ are the second order partial derivatives.

3. Key point descriptor: SURF uses wavelet responses in horizontal and vertical directions around the keypoint to create a descriptor. For rotation invariance, the dominant orientation is estimated and the descriptor is aligned to it.

### 2.7.4 Transformation Matrix

After having determined the matched features, the information of the feature points on the original photos is used to find a transformation matrix that will transform the photos to match the anchor photo, usually the first photo. The most commonly used transformation method is Affine transformation.

As explained in the documentation of OpenCV, A transformation that can be represented as the combination of matrix multiplication (representing linear transformation) and vector addition (representing translation). Then Rotations, translations, scale operations can be expressed by the Affine Transformation with a 2×3 matrix[24].

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}_{2\times 2}$$

$$B = \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix}_{2\times 1}$$

$$M = \begin{bmatrix} A & B \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} & b_{00} \\ a_{10} & a_{11} & b_{10} \end{bmatrix}_{2\times 3}$$

Since the feature points are distinct in the way that it is consistently detectable despite the transformation of the photos, two photos containing the same features can then be matched and the relationship between the photos can be described by the Affine transformation matrix.

# 3 Methodology

The pilot mining site selected for this project has remained abandoned for approximately a year, with no ongoing mining activities taking place. Given the previous research highlighting bamboo's capability to extract hazardous elements from soil, it becomes crucial to assess the concentration of these elements across various areas of the land. This assessment will enable the formulation of a targeted planting plan for bamboo cultivation.

The objective of this project is to generate an aerial map of the mining site by capturing and stitching together aerial photos obtained using a drone. In the future, the map will integrate the biochemistry aspect of the larger project by identifying contaminated areas and quantifying their concentrations. This section presents the process, starting from establishing the project goal and implementing the necessary methods to achieve the desired outcome.

## 3.1 Multiple Solutions and Directions

The project is a short-term project with a duration of one and a half months. Due to the time constraints, certain solutions such as building a sampling robot are not feasible. However, to achieve the ultimate goal of reclamation through bamboo growth, it is crucial to have a detailed map of the pilot mining site for future development.

One of the existing problem is that the satellite images available for the mining site were outdated and insufficiently detailed due to excessive scaling, rendering them inadequate for accurate representation of the land. As shown in Figure 8, the mark is around the entrance of our pilot site, and at the top of the screenshot is another adjacent mining site. The satellite image is outdated as it shows all green where in reality it has been deforested.

Considering both the time constraints and the project goal, we have opted to employ a drone to obtain aerial images and create an accurate map. This approach will provide a more precise and comprehensive understanding of the land's characteristics, facilitating the achievement of the objectives.

### 3.1.1 Writing Software for the Drone

To create a map using photos, it is essential for the captured images to have overlapping areas and for the drone's flight path to cover the desired perimeter comprehensively. Numerous software applications

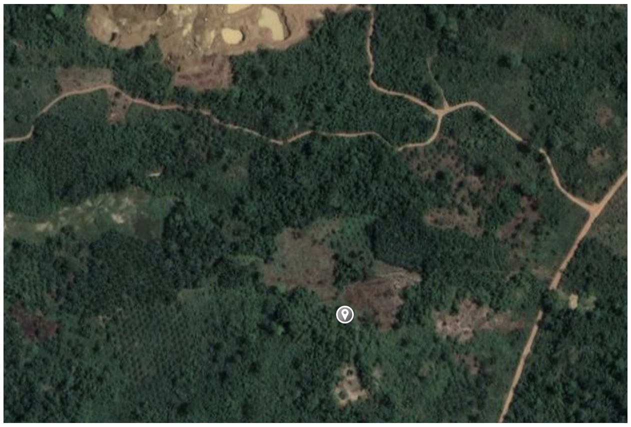


Figure 8: Screenshot from Google maps satellite, the mining site is still green.

are available for mapping purposes with drones; however, most of them require a subscription.

For this project, I utilized the DJI Mavic 2 Pro drone. DJI offers a mobile API that supports both Android and Apple devices, which prompted me to explore the possibility of developing an autopilot software. This software would be designed to plan the drone's flight path based on the desired area and capture photos automatically, ensuring the photos have overlapping areas.

Despite my efforts to work with the API and develop the autopilot software, I soon realized that the scope of this task was too substantial to accomplish within the project's timeframe. As a result, I had to explore alternative approaches to achieve the desired mapping objectives.

### 3.1.2 Create a Map with Photo Stitching

An alternative approach we considered was to utilize the GPS information of the photos and anchor them to mapping software as a base layer. During the Christmas break, I had the opportunity to test another drone in China and attempted to match the captured photos with mapping software ArcGIS. Unfortunately, ArcGIS did not support photos as an input format, rendering it unsuitable for our needs. Another option explored was Google Earth's API, but it appeared to be a complex undertaking that would require more time and resources, which made it a lower priority.

Following further investigation, we decided to focus on finding a solution for stitching the photos together. During this exploration, I discovered an OpenCV package on GitHub that perfectly served this purpose. I will provide more detailed information about this package in a subsequent subsection of this report.

### 3.2 Ghana Field Investigations

One notable aspect of this project is that it involved conducting fieldwork in Ghana, which provided a unique and hands-on experience. The pilot site selected for the project was located approximately a 10-minute drive away from the village. To ensure smooth communication and avoid any misunderstandings, it was highly recommended to accompany a local guide during the field visits. The village's chief's brother was our guide, a photo is included as Figure 9.

It was intriguing to observe that the mining site appeared distinct when viewed from the air compared to the ground. This aerial perspective inspired the idea of creating a stitched map that would offer an overview of the site and facilitate better planning for future planting endeavors. By visualizing the entire area through a comprehensive map, it would be easier to strategize and make informed decisions regarding the cultivation process.

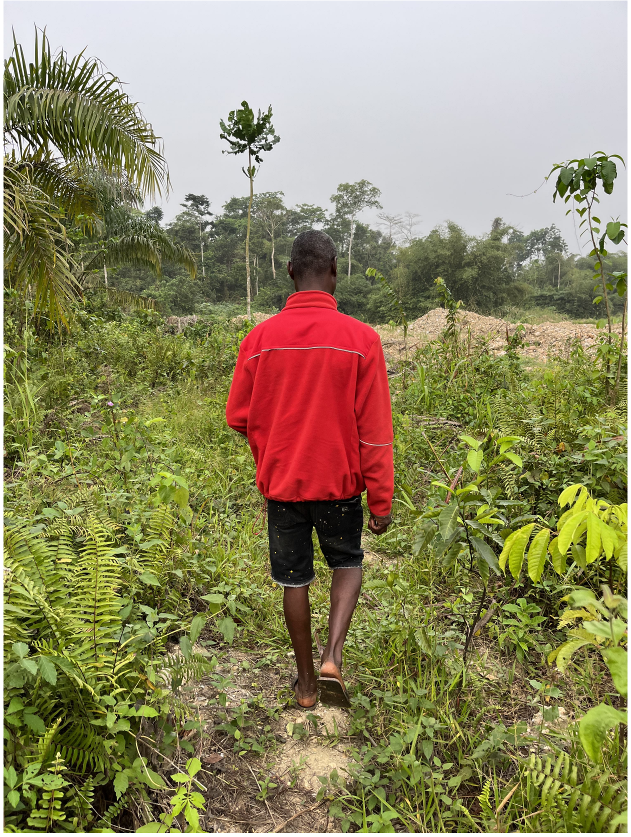


Figure 9: A photo of the village's chef's brother approaching the mining site.

### 3.3 Concept of Photo Stitching

Photo stitching falls within the realm of computer vision and image processing. When working with colored images, which consist of three scales and values for RGB (red, green, and blue), they are inherently three-dimensional. To simplify the stitching process, the images are conventionally converted to gray-scale, which is one-dimensional scale ranging from 0-255. There are several ways to convert RGB to gray scale [25]. The one MATLAB in the built in function `im2gray` is Luminosity method, where an RGB image to R, G and B have different weight as the following [26]:

$$grayscale = 0.299 * R + 0.587 * G + 0.114 * B.$$

Figure 10 is an example of original RGB images to be stitched.

Once the images have been converted to gray scale as in Figure 11, the stitching algorithm utilizes a feature detection algorithm such as BRISK or SURF. These algorithms identify thousands of feature points within each image. Figure 11 displays the top 20 strongest feature points for each image.

Next, as illustrated in Figure 12, the matching feature points between the images are connected using yellow lines. The two images are aligned on top of each other, with blue lines representing one image and red lines representing the other.

Finally, Figure 13 depicts the final image after combining the two stitched images together. The visible seams in the top right and bottom left corners signify the geological differences between the two images and the transformation matrix applied to the second image to align it with the first. By default, the resulting image is cropped to form a rectangle. However, to showcase the entire process and facilitate debugging, the side seams were intentionally left in place.

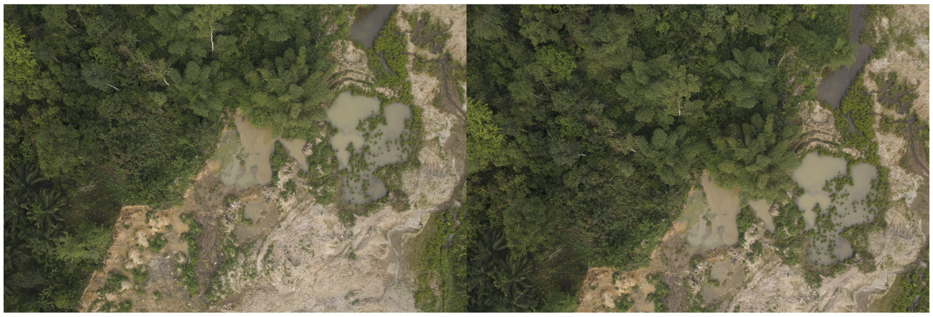


Figure 10: Two photos to be stitched in the original stage.

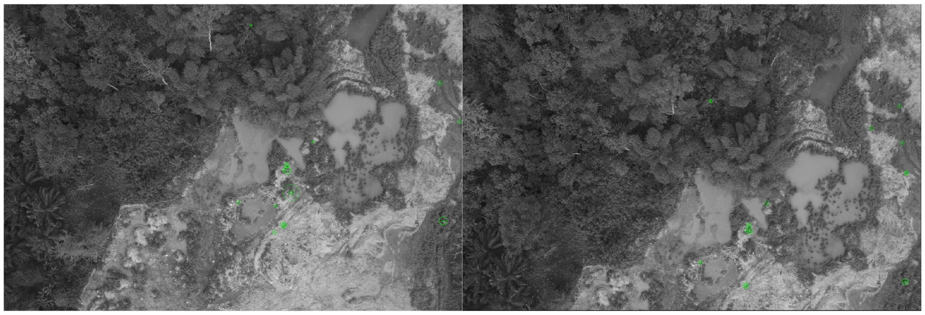


Figure 11: Photos in Figure 10 converted to gray-scale and showing 20 strongest feature points.

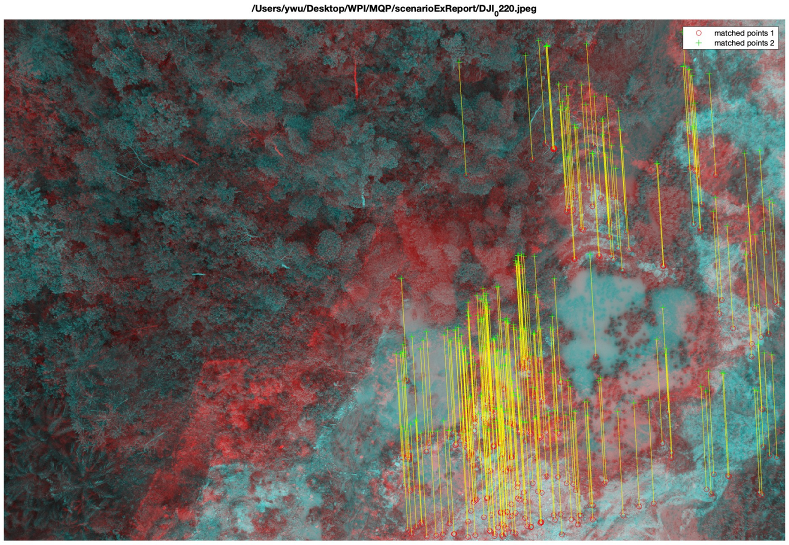


Figure 12: Two photos overlapping, showing the matching feature points with the yellow lines indicating pairs.

## 3.4 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source software library that specializes in computer vision and machine learning tasks. To facilitate the image stitching process, I discovered a GitHub package that integrates the functions of OpenCV [4]. This online hosting service for software development provides convenient features for selecting various parameters such as the feature detection algorithm, confidence threshold, and warp type.

The available feature detection algorithms include ORB, SIFT, BRISK, and AKAZE. The confidence threshold is set to discard the photos below the threshold. For example, if the threshold is 0.6, then if a photo doesn't have a confidence rate higher than 0.6 with all other photos, that photo is excluded. It can

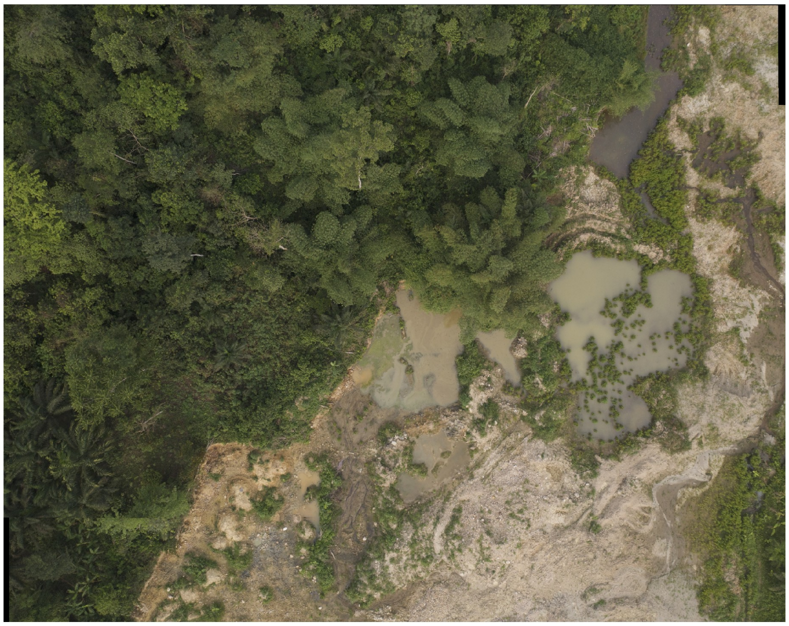


Figure 13: Result of the two photos.

be set lower if the matching process proves to be challenging. Regarding the warp type, numerous options are available, but for this particular case, the relevant choices are plane, affine, and cylindrical. The optimal combination of these settings is depicted below:

```
confThres = 0.4 #lower if hard to match
detectMethod = "brisk"
warpType = "plane"
waveCorrect = "vert"
matcherType = "affine"
```

The wave correct setting determines whether the panorama is horizontal or vertical.

During the photo capturing process, I followed a zigzag flight path to ensure overlapping areas between consecutive images. However, there were instances when I took photos that were not in sequence, requiring their exclusion for the package to function correctly.

While the package worked well with the mining site, it encountered difficulties when dealing with the trees surrounding the area. The mining site was encircled by trees, and these trees potentially provided indications of the location of rivers. Given that the mining activity had disrupted the rivers, the presence of more rivers near the mining site was expected.

## 3.5 MATLAB

The OpenCV package demonstrated satisfactory performance in capturing the mining site; however, our objective was to include more of the surrounding areas. After conducting research, we discovered that leveraging the GPS information from each photo could assist the feature detection algorithms in narrowing down the regions of interest. It is challenging for algorithms to detect and match features accurately when dealing solely with photos of trees. Therefore, to facilitate feature detection, all the photos were captured to include areas other than just trees.

MATLAB provides an example of photo stitching on their official website [27], and I utilized the code provided. Although the code worked well with the photos, the feature detection algorithm consistently detected the strongest feature points within the tree area. To address this issue, we implemented the utilization of GPS information, followed by incorporating edge detection techniques to narrow the region of interest.

### 3.5.1 Implementation of Meter Per Pixel (mpp) and GPS

The meter per pixel (mpp) represents the measurement in meters that is equivalent to a single pixel in a photograph. To determine this value, trigonometry is employed using the drone's height at the time of photo capture and the camera's field of view (FOV). In this project, the drone's camera had a FOV of 77 degrees as mentioned in Section 2.6, and the height remained consistent at 103 meters. By applying the appropriate Equation 1 as depicted in Figure 14, the length $l$ of the photo in meters can be calculated. Furthermore, the photo size is 5478*3648 pixels, we can find meter per pixel by dividing $l$ by the photo size.

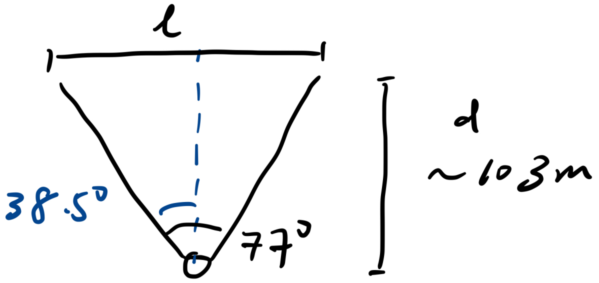
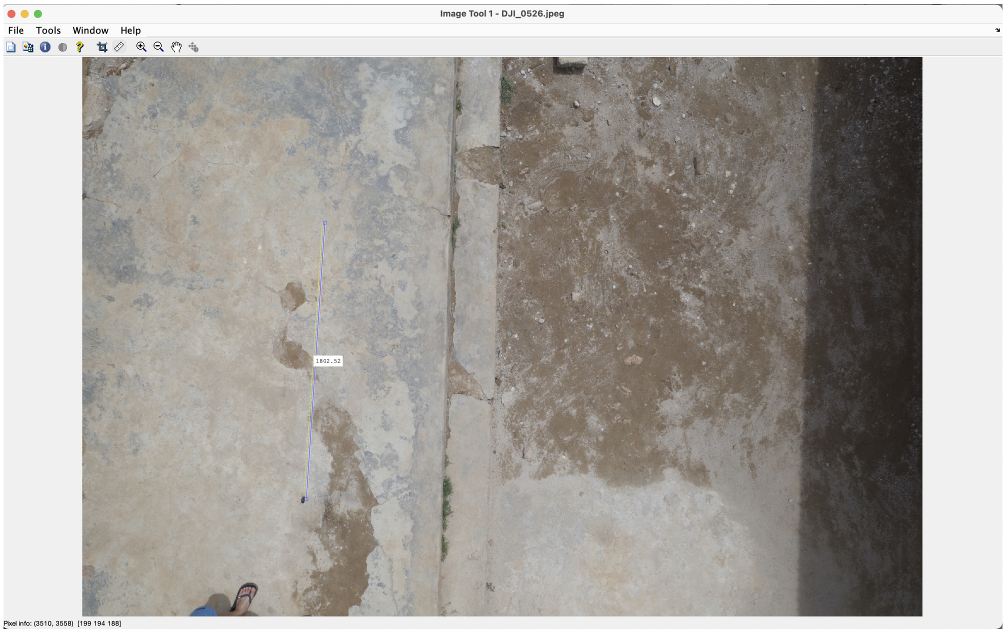


Figure 14: Sketch of the relationships between drone height ($d$), FOV, and length ($l$).

$$\frac{\phi}{2} = \arctan\left(\frac{l}{2d}\right) \tag{1}$$

To verify the accuracy of the calculations for determining the mpp, a test photo was taken. A 6-ft ruler was placed on the ground and expanded, while the drone was flown to a height of 4.7 meters to capture the photo. The photo was then opened in MATLAB to measure the length of the ruler in pixels, providing a supplementary verification of the correctness of the mpp calculation.



Pixel info: (3510, 3558) [199 194 188]

Figure 15: Test photo with a 6-ft ruler on the ground opened in MATLAB add-on application image viewer. The pixel value of the ruler is 1802.52.

The following code uses Equation 1 with the drone specification and camera frame converting rate from a website [28] that demonstrates how to find the meter per pixel value. The pixel value of the image was obtained from MATLAB add-on application image view, it is a few pixel less than the official specification of the drone.

```
% the variables of finding cm/pixel, 28mm equivalent with full frame 36*24
   mm
droneh = 103; %drone height in meter
fieldx = 2 * droneh * tand((2*atand(36/(2*28)))/2);
fieldy = 2 * droneh * tand((2*atand(24/(2*28)))/2);
pixelx = 5464; %pixel number in x direction
pixely = 3640; %pixel number in y direction
mperpixy = fieldy/pixely;
mperpixx = fieldx/pixelx;
```

In the process of identifying the overlapping region between two photos, GPS information was utilized. The photos were input into MATLAB, then the longitude (latitude likewise) were read and obtained by the following lines:

```
longitude = imfinfo(char(siteScene.Files(j))).GPSInfo.GPSLongitude; % read the longitude of the photo
decimalLongitude = longitude(1) + (longitude(2)/60) + (longitude(3)/3600); %convert from degree, second to decimal
longInfoRef = imfinfo(char(siteScene.Files(j))).GPSInfo.GPSLongitudeRef; %read the reference, E or W
if longInfoRef == 'W'
    decimalLongitude = -decimalLongitude; %If it's W, then it's negative in decimal.
end
```

With the longitude (and latitude) found for each photo, the differences in longitude between every two photos were determined.

The mpp value obtained for the drone photo, which remained constant throughout the project due to the consistent height and FOV, was applied. By utilizing the conversion of one degree of longitude (111139 meters), along with the mpp, the differences between the two photos in terms of pixels could be calculated, considering that the region of interest (ROI) was defined in pixels.

### 3.5.2 Implementation of Edge Detection

There exists a range of edge detection algorithms, each suitable for different purposes. After experimenting with various methods, it was determined that the Canny edge detection algorithm was the most suitable for identifying trees.

MATLAB provides a built-in function for performing edge detection using different methods. However, additional steps were involved in implementing the process of identifying the photo block that primarily contained trees.

Firstly, the photo was divided based on the GPS difference described in Section 3.5.1, with the overlapping region being defined as the area of interest. Next, the photo was transformed into edges using the Canny edge detection method. Subsequently, the photo was equally divided into four quadrants, and the percentage of edges in each quadrant was calculated by determining the ratio of non-zero elements to the total number of elements in that quadrant. Since the edge photo only consists of two values (0 and 1), this calculation was straightforward.

Next, an edge threshold value was established. If the percentage of edges in a particular quadrant exceeded this threshold, the corresponding area and any preceding areas or succeeding areas depending on the highest edge ares would not be considered as regions of interest. This approach was adopted because MATLAB's feature detection function only allows for a single region of interest, and it was anticipated that the tree area would predominantly be located on one side of the photo. Consequently, this process effectively eliminated a significant portion of the tree regions.

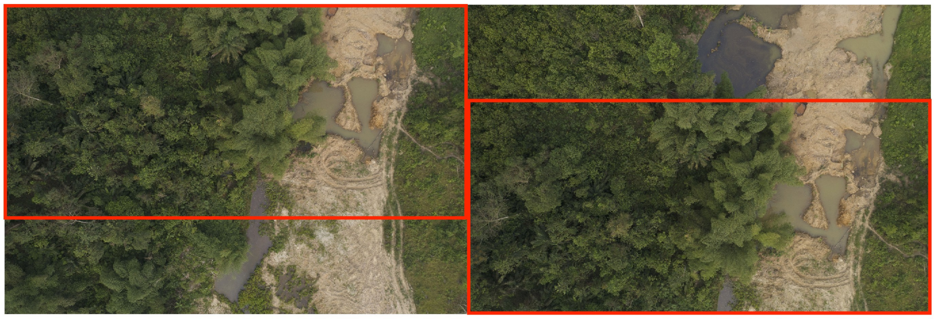


Figure 16: Two consecutive photos in their original state, with the red boxes indicating the overlapping areas.

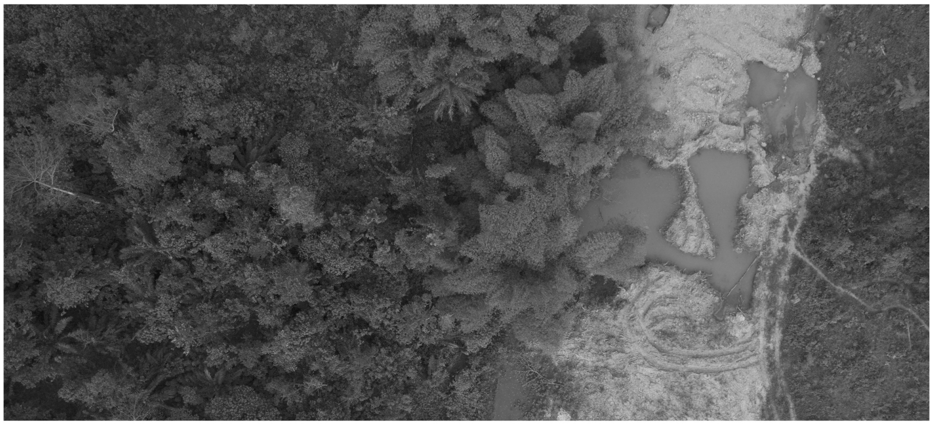


Figure 17: Photo on the left after cropping by GPS information.

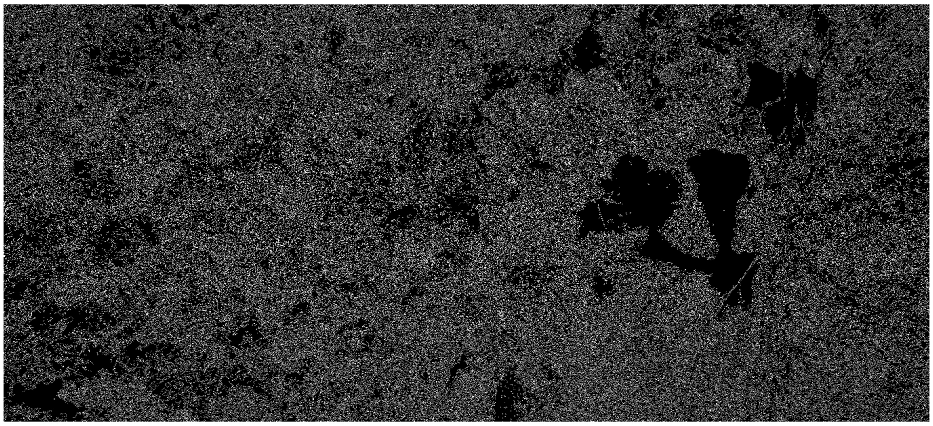


Figure 18: After performing Canny edge detection.

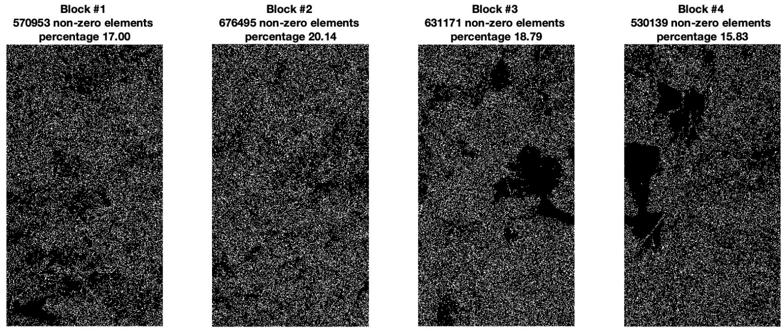


Figure 19: The photo divided into quarters by column, percentage indicating the edge percentage in the area.

# 4 Results

The results section presents the outcomes obtained form both OpenCV and MATLAB. This section presents a comprehensive analysis of different feature detection methods and settings used for image stitching. The aim of this analysis is to evaluate the performance and effectiveness of various techniques in producing the best matching result when encountering trees. Multiple feature detection algorithms and parameter settings were examined, the results of each method and setting were assessed based on time, alignment accuracy, and visual quality of the output in different scenarios. Every resulting image presented in this section is not cropped to show the most features and details.

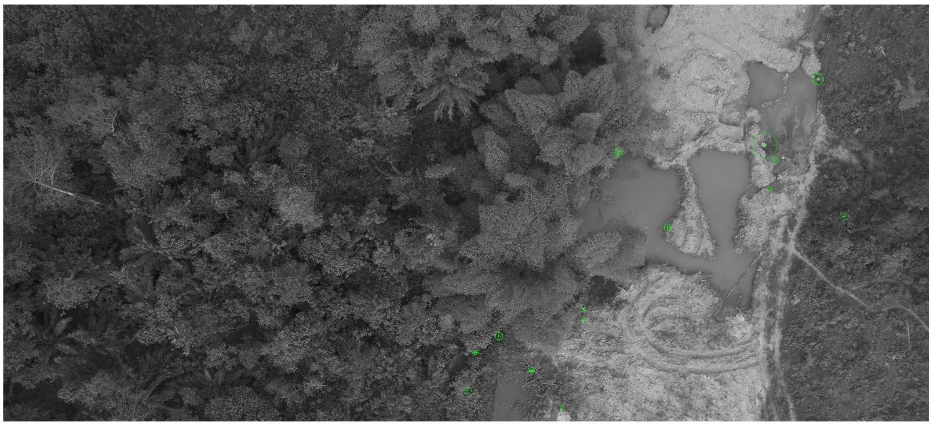


Figure 20: 20 strongest feature points after performing GPS and edge detection.

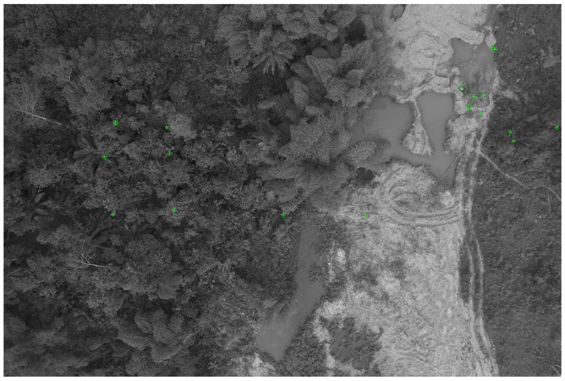


Figure 21: 20 strongest feature points of the original photo converted to gray-scale.

## 4.1 OpenCV Results

The previous section highlighted four feature detection options provided by the OpenCV package for image stitching: ORB, SIFT, BRISK, and AKAZE. However, this section focuses on the evaluation of the first three techniques. Two distinctive scenarios were tested for each method - one featuring fewer trees and the other predominantly filled with trees. In both scenarios, images were manually selected to ensure continuity, with the beginning and end of each sequence specifically designated within the code.

Table 1 provides the runtime for each method across the two scenarios. Each test was conducted ten times to increase result reliability. Figure 22 depicts the original photos used in the 'fewer trees' scenario, while the images used in the 'mostly trees' scenario are showcased in Figure 23. The detailed matched features of the three feature detection methods can be found in Appendix A.

First, the ORB feature detection method was tested with the less tree scenario, with confidence

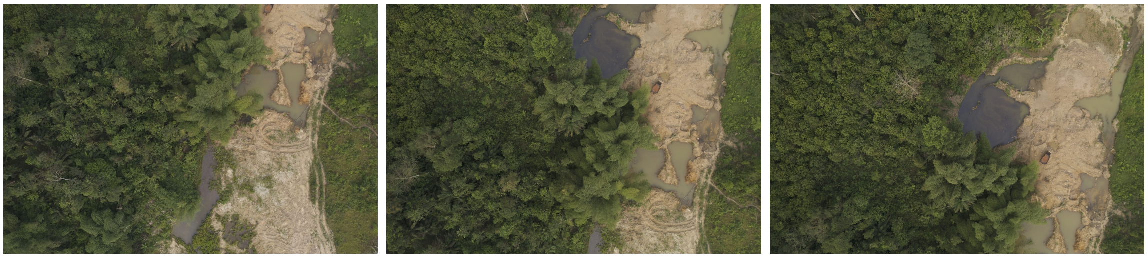


Figure 22: The three photos for the less tree scenario.

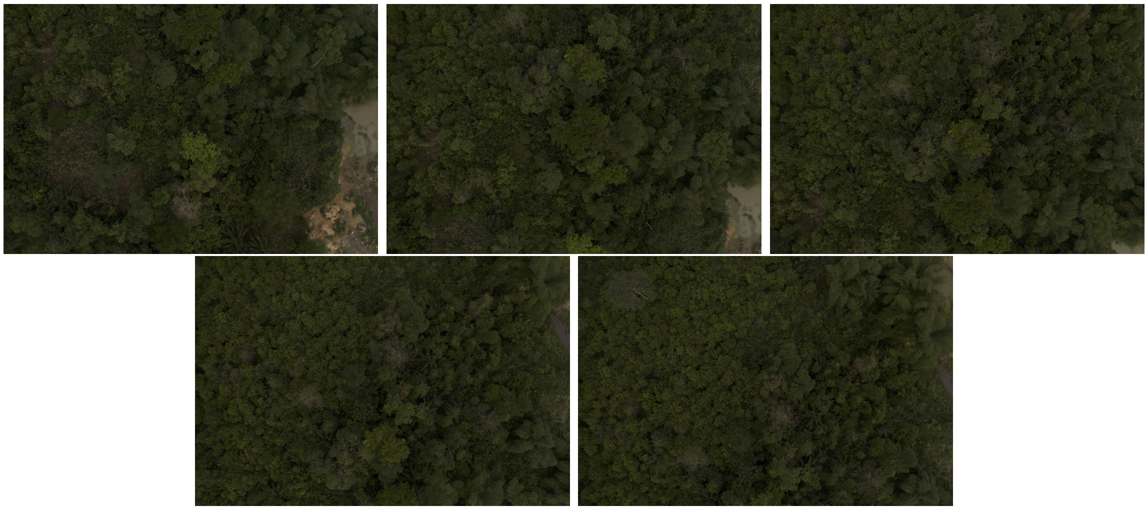


Figure 23: The five original photos for the mostly tree scenario.

threshold set to 0.6. The feature points are shown in Figure 24. The inliers are the ones that matched correctly, they are calculated using the random sample consensus (RANSAC). The confidence of the match [1] is calculated by:

$$confidence = \frac{\text{number of inliers}}{(8 + 0.3 \times \text{number of matches})},$$

and the resulting matrix of photo matching confidence is

$$\begin{bmatrix} 0 & 1.67155425 & 0.93406593 \\ 1.67155425 & 0 & 1.37466307 \\ 0.93406593 & 1.37466307 & 0 \end{bmatrix}.$$

The matrix above represents each photo's matching confidence with all other photos, forming a square matrix. The diagonal entries are zeros as there is no self-matching. The higher the matching confidence, the more features are matched in two photos. So from the matrix above, photo 1 and photo 2 has a considerably good match, photo 2 and photo 3 also has a considerably good match since these are photos with a lot of non-tree features.

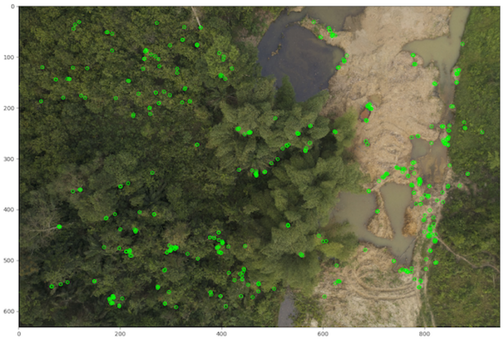


Figure 24: OpenCV ORB feature detection feature points of the first photo in the more tree scenario.

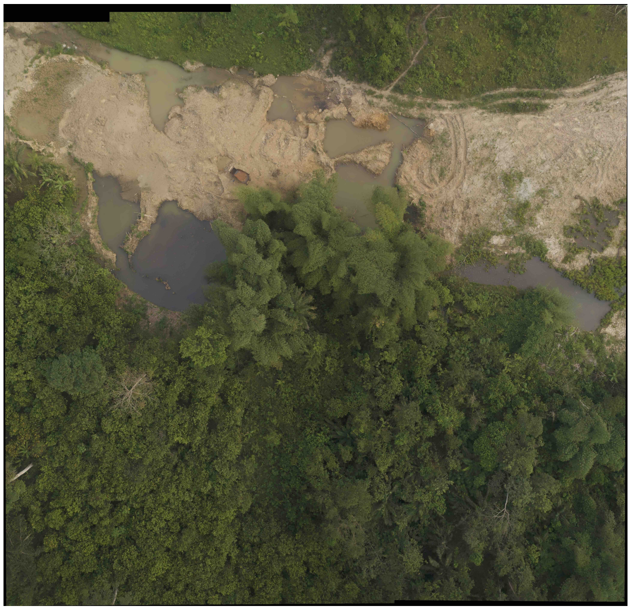


Figure 25: OpenCV ORB feature detection result.

With the mostly tree scenario, the resulting image is shown in Figure 26 and the confidence matrix of the ORB feature detection with the same threshold is:

$$\begin{bmatrix} 0 & 0.46979866 & 0.32786885 & 0.4 & 0.96446701 \\ 0.46979866 & 0 & 0.38167939 & 0.3539823 & 0.3539823 \\ 0.32786885 & 0.38167939 & 0 & 0.90909091 & 0.34246575 \\ 0.4 & 0.3539823 & 0.90909091 & 0 & 0.86486486 \\ 0.96446701 & 0.3539823 & 0.34246575 & 0.86486486 & 0 \end{bmatrix}.$$

Next, we examined the BRISK feature detection method. Figure 27 illustrates the feature points

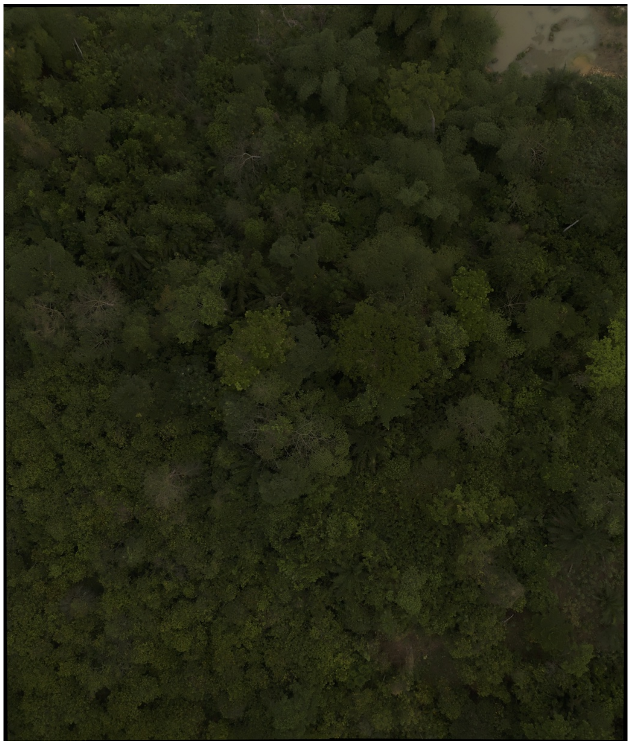


Figure 26: OpenCV ORB feature detection result.

identified from the first photo in the scenario with more trees. The confidence matrix was computed as follows:

$$\begin{bmatrix} 0 & 2.76538805 & 2.14099217 \\ 2.76538805 & 0 & 2.18506998 \\ 2.14099217 & 2.18506998 & 0 \end{bmatrix}.$$

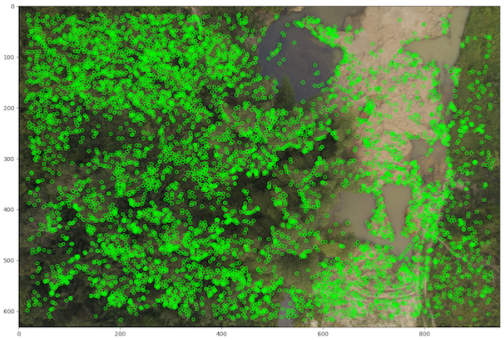


Figure 27: OpenCV ORB feature detection feature points of the first photo in the more tree scenario.

For the same 'mostly trees' scenario, the resultant image can be viewed in Figure 29. The confidence

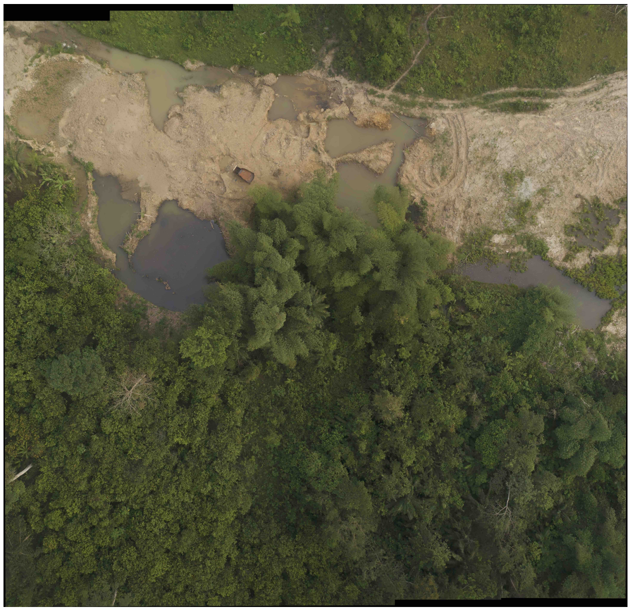


Figure 28: OpenCV BRISK feature detection result.

matrix for the BRISK feature detection method at the same threshold level is provided below:

$$\begin{bmatrix} 0 & 0.36363636 & 0 & 0.37383178 & 1.00671141 \\ 0.36363636 & 0 & 0.40816327 & 0 & 0 \\ 0 & 0.40816327 & 0 & 1.26696833 & 0.48076923 \\ 0.37383178 & 0 & 1.26696833 & 0 & 1.44329897 \\ 1.00671141 & 0 & 0.48076923 & 1.44329897 & 0 \end{bmatrix}.$$

Finally, we investigated the SIFT feature detection method. The SIFT feature points, extracted from the same photo, are presented in Figure 30. It is observable that SIFT yields fewer feature points than BRISK but more than ORB. The corresponding confidence matrix is as follows:

$$\begin{bmatrix} 0 & 2.60820046 & 2.17142857 \\ 2.60820046 & 0 & 2.16685979 \\ 2.17142857 & 2.16685979 & 0 \end{bmatrix}.$$

In the scenario with a predominance of trees, the SIFT feature detection method was unable to complete the process with a confidence threshold of 0.6. Upon adjustment to 0.5, the algorithm successfully concluded, with the resulting image exhibited in Figure 32. The corresponding confidence matrix for this

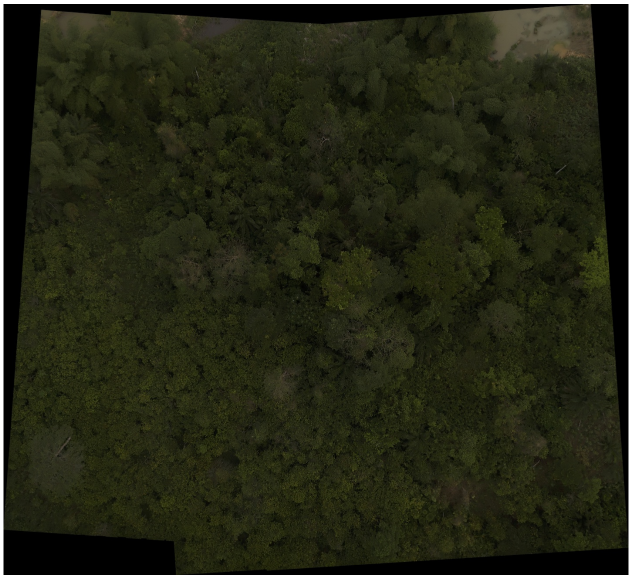


Figure 29: OpenCV BRISK feature detection result.

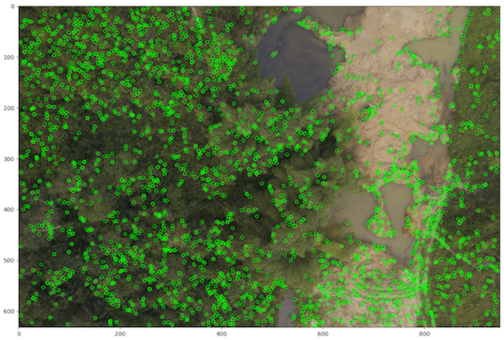


Figure 30: OpenCV SIFT feature detection feature points of the first photo in the more tree scenario.

adjustment is provided below:

$$\begin{bmatrix} 0 & 0.4950495 & 0 & 0 & 0.4950495 \\ 0.4950495 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.83969466 & 0 \\ 0 & 0 & 0.83969466 & 0 & 0 \\ 0.4950495 & 0 & 0 & 0.76335878 & 0 \end{bmatrix}.$$

Below are the results of 27 consecutive photos stitched together using BRISK, which has the best resulting image, with confidence threshold being 0.6, and plane warp type.

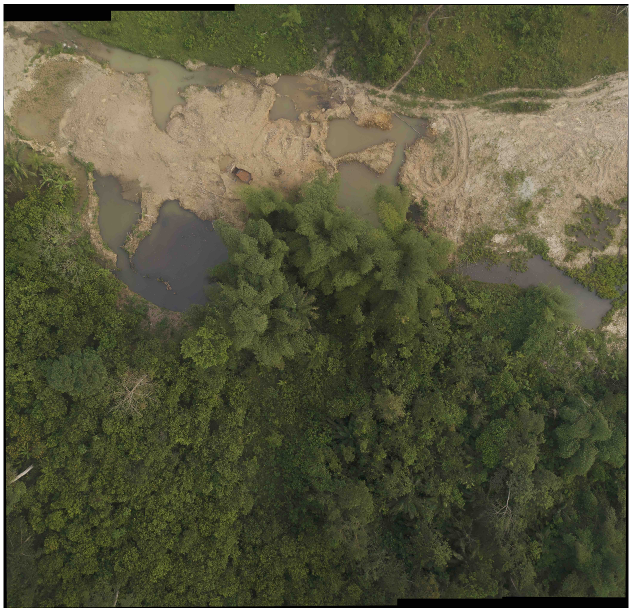


Figure 31: OpenCV ORB feature detection result.

| | ORB | SIFT | BRISK |
|---|---|---|---|
| less tree | 21.69s | 20.95s | 29.39s |
| more tree | 14.52 | N/A | 28.13s |

Table 1: Run time of each method with three photos run 10 times, confidence threshold 0.6.

## 4.2 MATLAB Results

The OpenCV package yields satisfactory results in scenarios with fewer trees. The intent behind using MATLAB is to incorporate GPS data and edge detection to aid the feature detection algorithms in identifying more effective feature points. Consequently, tests are conducted using primarily tree-dense images across different scenarios. Each scenario consists of manually selected photos to maintain continuity. However, the functionality to skip inconsistent photos has yet to be implemented.

Initially, fourteen photos, previously utilized with OpenCV (as shown in Figure 33), were selected to evaluate the MATLAB algorithm using both BRISK and SURF feature detection methods. The resultant images are depicted in Figure 34 and Figure 35, respectively. Notably, the seams are more pronounced than those produced by OpenCV. This is because OpenCV includes a class that identifies the line of transition between images with minimal interference.

In contrast, MATLAB employs a built-in function, `imwarp`, which applies a geometric transformation to the input image. The input image can be numeric, logical, or categorical, with the transformed image returned as output, denoted by the variable B [29]. This scenario serves to test the fundamental functionality

Figure 32: OpenCV SIFT feature detection result.

| | photo 1 | photo 2 | photo 3 | photo 4 | photo 5 | photo 6 | photo 7 |
|---|---|---|---|---|---|---|---|
| BRISK: | 5314 | 1987 | 3577 | 2872 | 1763 | 6481 | 7972 |
| SURF: | 7267 | 4308 | 7903 | 8622 | 6197 | 14167 | 18210 |
| | photo 8 | photo 9 | photo 10 | photo 11 | photo 12 | photo 13 | photo 14 |
| BRISK: | 1401 | 1070 | 2482 | 5428 | 6162 | 3800 | 3832 |
| SURF: | 6962 | 2967 | 5950 | 9362 | 7455 | 8649 | 8335 |

Table 2: Feature points detected by the feature detection methods of each photo.

of the MATLAB algorithm, given that its primary purpose is to stitch images with a predominance of trees.

The number of feature points of each feature detection method are shown in Table 2. The edge ratio (the number times 100% is the percentage) of each quadratic area of each photo are given in Table 3. The differences in longitude and latitude between every two photos are presented in Table 4. The run time of using BRISK method was 139.29$s$ and the run time of SURF method resulted in 150.86$s$.

To further force the feature points towards the non-tree area, the images were divided into six sections, with the feature points depicted in Figure 40. The figure reveals that all feature points are leaning more towards the non-tree features. The edge rate is presented in Table 7, where, yet again, the area on the

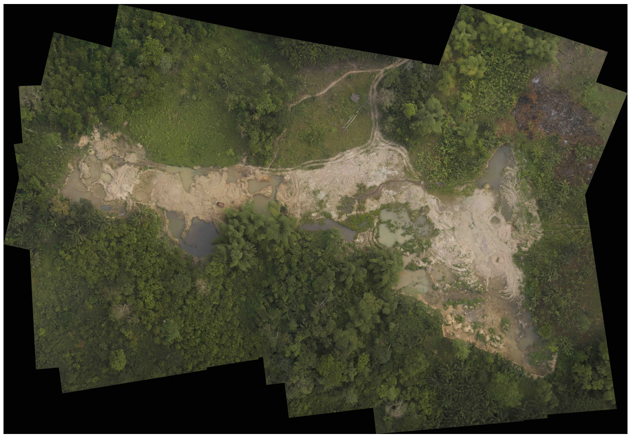


Figure 33: OpenCV result with BRISK feature detection method.

| | | | | |
|---|---|---|---|---|
| photo 1: | 0.1665 | 0.1863 | 0.2076 | 0.1685 |
| photo 2: | 0.1688 | 0.2211 | 0.1679 | 0.1362 |
| photo 3: | 0.1587 | 0.2106 | 0.1756 | 0.1487 |
| photo 4: | 0.1618 | 0.2029 | 0.1773 | 0.1372 |
| photo 5: | 0.1860 | 0.2105 | 0.1800 | 0.1432 |
| photo 6: | 0.1862 | 0.1952 | 0.1960 | 0.1548 |
| photo 7: | 0.1548 | 0.1902 | 0.1793 | 0.1674 |
| photo 8: | 0.1737 | 0.2067 | 0.1928 | 0.1752 |
| photo 9: | 0.1994 | 0.2085 | 0.1431 | 0.1507 |
| photo 10: | 0.1954 | 0.2141 | 0.1511 | 0.1515 |
| photo 11: | 0.1684 | 0.2114 | 0.1758 | 0.1344 |
| photo 12: | 0.1620 | 0.2063 | 0.1989 | 0.1409 |
| photo 13: | 0.1560 | 0.2084 | 0.1654 | 0.1464 |
| photo 14: | 0.1647 | 0.2152 | 0.1683 | 0.1521 |

Table 3: Edge ratio of each quadratic area of each photo.

| photo | 1-2 | 2-3 | 3-4 | 4-5 | 5-6 | 6-7 | 7-8 |
|---|---|---|---|---|---|---|---|
| longitude*0.003 | 0.3528 | 0.1222 | 0.0306 | 0.1583 | 0.2417 | 0.0417 | 0.0833 |
| latitude*0.003 | -0.1 | -0.0333 | -0.2056 | -0.0444 | -0.0667 | -0.2306 | -0.2444 |
| | 8-9 | 9-10 | 10-11 | 11-12 | 12-13 | 13-14 | |
| longitude*0.003 | 0.2583 | 0.2278 | 0.225 | 0.3056 | -0.0583 | 0.0611 | |
| latitude*0.003 | -0.0778 | -0.0639 | -0.0667 | -0.0889 | -0.2083 | -0.0167 | |

Table 4: Longitude and latitude differences of every two photos.

extreme right has the lowest edge percentage.

Although the feature points were directed towards better features, the program was unable to produce a result due to a MATLAB error indicating "too many elements in input 1" when utilizing the `step` function.

Given that neither algorithm yielded satisfactory results, the final two images of the scenario were

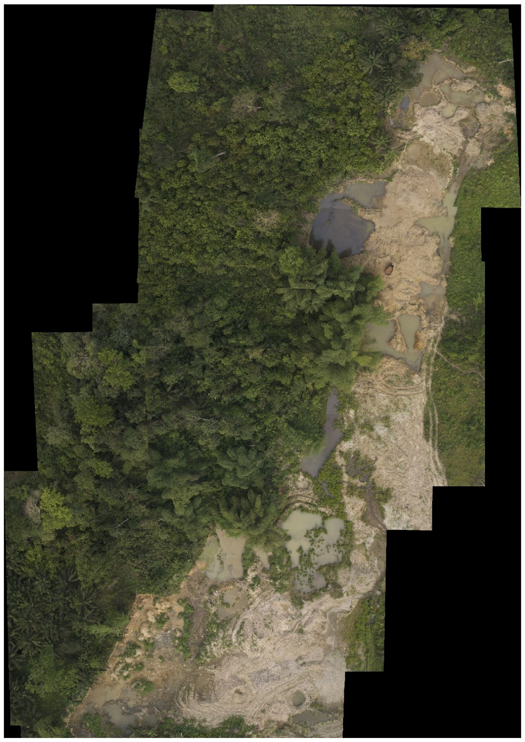


Figure 34: MATLAB result with BRISK feature detection algorithm, edge threshold is 20%.

| | photo 1 | photo 2 | photo 3 | photo 4 | photo 5 |
|---|---|---|---|---|---|
| SURF: | 102 | 67 | 143 | 146 | 125 |

Table 5: Feature points detected by the SURF feature detection methods of each photo.

| | | | | |
|---|---|---|---|---|
| photo 1: | 0.1875 | 0.2138 | 0.2000 | 0.1570 |
| photo 2: | 0.2065 | 0.2193 | 0.1941 | 0.1665 |
| photo 3: | 0.1982 | 0.2117 | 0.1764 | 0.1499 |
| photo 4: | 0.2033 | 0.2119 | 0.1833 | 0.1476 |
| photo 5: | 0.2062 | 0.2270 | 0.1995 | 0.1568 |

Table 6: Edge ratio of each quadratic area of each photo for 4 blocks of more tree scenario.

| | | | | | | |
|---|---|---|---|---|---|---|
| photo 1: | 0.1838 | 0.2022 | 0.2160 | 0.2073 | 0.1861 | 0.1421 |
| photo 2: | 0.2000 | 0.2212 | 0.2174 | 0.1975 | 0.1828 | 0.1607 |
| photo 3: | 0.1939 | 0.2112 | 0.2099 | 0.1802 | 0.1664 | 0.1431 |
| photo 4: | 0.1997 | 0.2164 | 0.2068 | 0.1874 | 0.1695 | 0.139 |
| photo 5: | 0.2004 | 0.2255 | 0.2240 | 0.2061 | 0.1796 | 0.1488 |

Table 7: Edge ratio of each quadratic area of each photo for 6 blocks of more tree scenario.

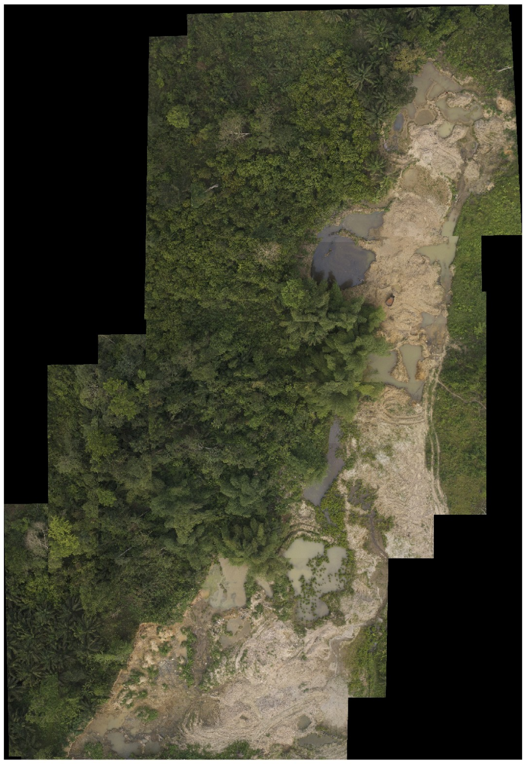


Figure 35: MATLAB result with SURF feature detection algorithm, edge threshold is 20%.

| | photo 1 | photo 2 | photo 3 |
|---|---|---|---|
| SURF: | 77 | 28 | 75 |

Table 8: Feature points detected by the SURF feature detection methods of each photo.

excluded due to their minimal non-tree features. OpenCV generated the same result with a confidence threshold set at 0.4, as illustrated in Figure 38. MATLAB, on the other hand, although yielded a flawed outcome, it maintained the non-tree features as shown in Figure 39. The feature points detected in each image are tabulated in Table 8.

The images had few matched sets; the first and second images had three matched sets, as did the second and third. The matched images are displayed in Figure 39. As can be seen, there are only three matched sets, with one set from each image appearing misaligned.

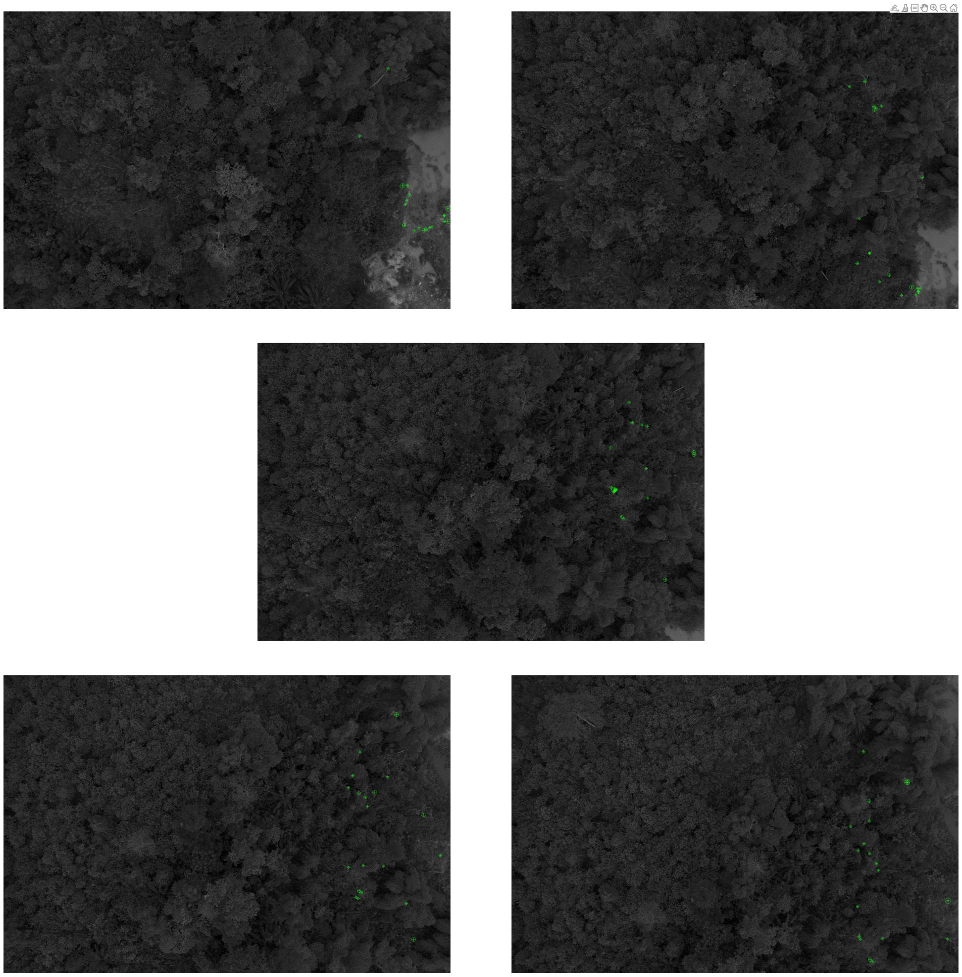


Figure 36: The five photos for the mostly tree scenario with SURF detection and 4 blocks of edge detection.

# 5 Discussion

In this section, the results obtained from the project will be analyzed and contextualized in relation to the research objectives.

## 5.1 OpenCV Results Discussion

As presented in the previous Section 4.1, the resulting images and confidence matrices for each feature detection method were obtained. It is apparent that among the three methods - SIFT, BRISK, and ORB - BRISK exhibits the highest matching confidence overall.

In the more optimal scenario with fewer trees, Photo 1 has a higher match to Photo 2 than to Photo 3, and Photo 3 has a higher match to Photo 2 than to Photo 1, as anticipated. This is attributed

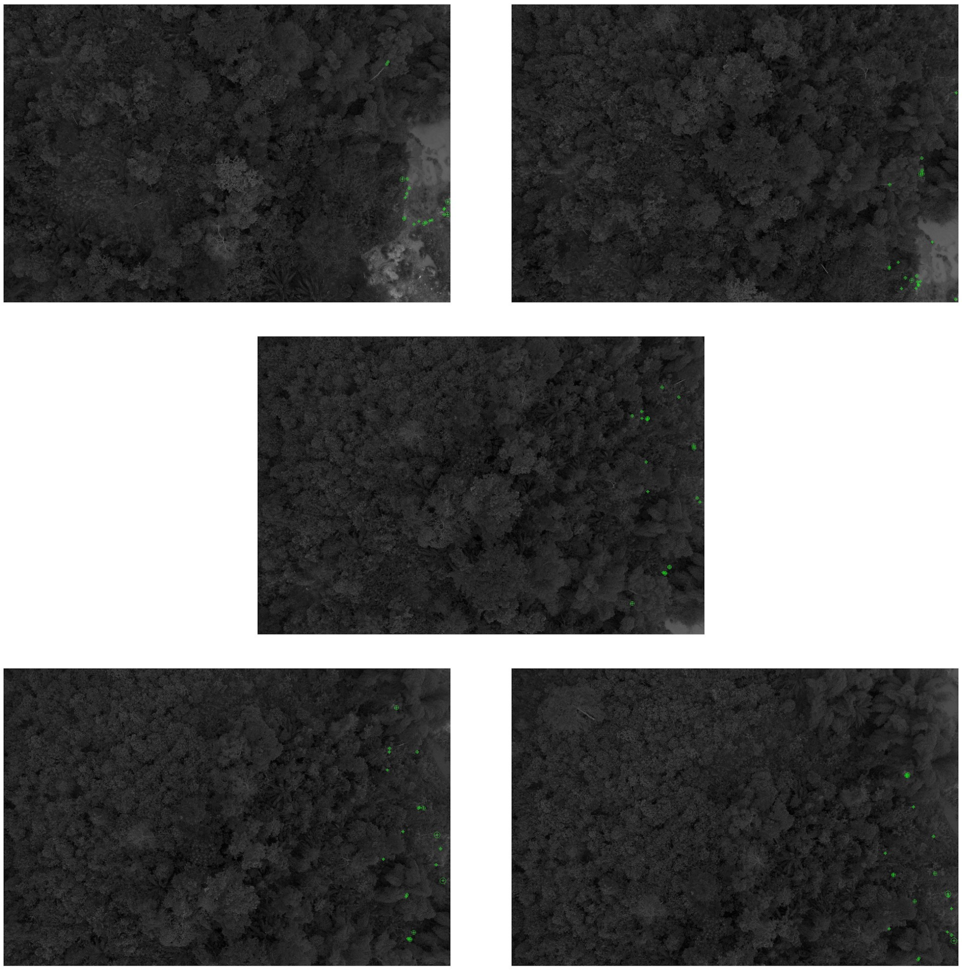


Figure 37: The five photos for the mostly tree scenario with SURF detection and 6 blocks of edge detection.

to the sequential nature of the photographs, with each image overlapping with its immediate predecessor and successor. Consequently, Photo 3 should exhibit minimal or no overlapping areas with Photo 1, thereby implying a higher matching confidence with Photo 2. Given that the matching confidence is identical between two photos, the one with a higher confidence with Photo 2 is not as significant.

It's evident that in the less ideal scenario, dominated by trees, the resultant images were inaccurate. The largest soil part evident in the first and second photos of Figure 23 is absent in all resulting images, regardless of the method employed. The confidence matrices exposed this issue. The OpenCV package includes a step that identifies a subset of the input images. Images falling below the threshold are discarded, while those deemed a "good match" are grouped into a subset to be stitched. In addition, the order of the images may change.

All the methods matched Photo 1 to Photo 5, Photo 3 to Photo 4, and Photo 4 to Photo 5. The

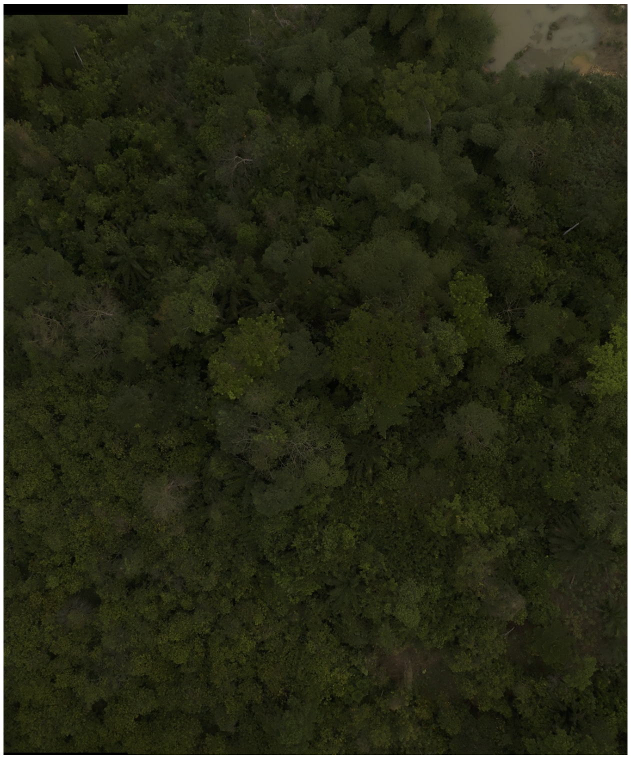


Figure 38: OpenCV result of 3 photos of more tree scenario using BRISK with 0.4 confidence threshold.

matrices indicate that Photo 1 has the highest matching confidence with Photo 5, Photo 2 does not surpass the confidence threshold, Photo 3 exhibits the highest confidence rate with Photo 4, and Photo 4 has the highest matching rate with Photo 5. Although the specific figures vary by method, the overall pattern is consistent.

## 5.2 MATLAB Results Discussion

The results encompassed several aspects: the number of detected feature points, the edge percentage of the tree area, the differences in longitude and latitude between consecutive photos, and the runtime. Each aspect will now be examined in detail.

As discussed in the previous Section 4.2, the OpenCV package includes a class called `SeamFinder`. In contrast to MATLAB's `imwarp` function, which performs image transformations, `SeamFinder` identifies irregular lines known as seams to optimize image warping.

To comprehensively evaluate the results and analyze the effectiveness of the feature detection methods, several aspects were considered. Firstly, the number of feature points detected for each algorithm was examined. Additionally, the calculation of the percentage of the tree area covered by edges was performed.

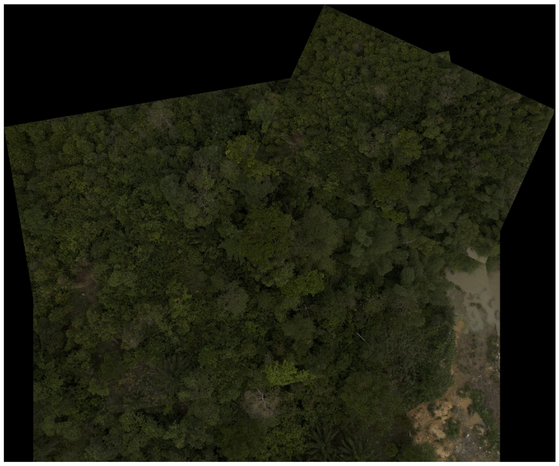


Figure 39: MATLAB result of 3 photos of more tree scenario using SURF.

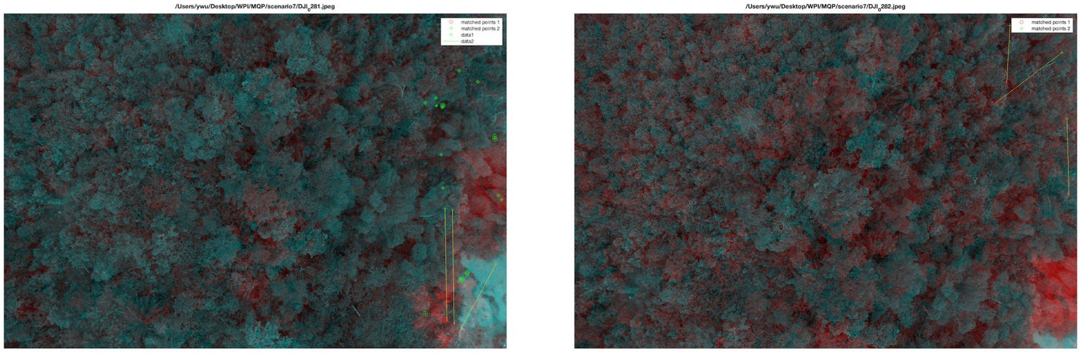

/Users/ywu/Desktop/WPI/MQP/scenario7/DJI$_0$281.jpeg
matched points 1
matched points 2
data1
data2
/Users/ywu/Desktop/WPI/MQP/scenario7/DJI$_0$282.jpeg
matched points 1
matched points 2

Figure 40: Matched points of the 3 photos of the more tree scenario.

This provided insight into the extent of coverage achieved by each method.

Furthermore, the differences in longitude and latitude between consecutive photos were analyzed. These variations offered valuable information regarding the spatial relationships and overlaps between the images.

Lastly, the runtime of the algorithms was meticulously recorded, documenting the processing and stitching time required by each method. By considering all these aspects collectively, a thorough evaluation of the results could be obtained.

In scenario 1, it is observed that the SURF feature detection method detects more features than the BRISK method (as shown in Table 2), but it also exhibits a noticeably longer runtime, which aligns with expectations.

When considering the differences in longitude and latitude between consecutive photos, it is evident that the differences in longitude are significantly greater than the differences in latitude. This is due to the fact that one degree of longitude is equivalent to approximately 111,139 meters.

All photos in this scenario include a portion of the mining site, indicating that the mining site is likely situated on one side of the photo. As indicated in the results section and shown in Table 3, a threshold value of 0.2 appears to be effective in determining which side of the photo to use.

The resulting images obtained from the two feature detection methods do not seem to exhibit significant differences.

In the second scenario, the results were flawed. The feature points were deliberately directed towards non-tree elements, which unfortunately resulted in a reduction in their number. This was primarily due to a diminished Region of Interest (ROI) that restricted the scope of the algorithm to a narrower field. The shrinkage in the quantity of feature points leaded to a considerable decrease in the number of matched sets. This was an undesirable outcome, as a high number of matched sets is generally indicative of a more precise and effective stitch.

Despite these shortcomings, the scenario presents a significant number of unexplored opportunities. There are several other adjustable parameters within the algorithm that have not been manipulated or experimented with. These represent potential avenues for tweaking and improving the results obtained. Indeed, one of the defining strengths of this algorithm lies in its high capacity for customization. It allows for an array of parameters to be individually adjusted, thereby making it possible to finely tune its performance to suit specific requirements and contexts.

The significant drop in the number of feature points and matched sets in this scenario is not definitive proof of the algorithm's limitations. Instead, it brings attention to the fact that there's substantial scope for refining the algorithm by exploring and adjusting other parameters. This inherent flexibility is promising as it suggests that, with the right configuration, the algorithm can be optimized to yield improved results, thereby enhancing the overall quality of its output.

# 6 Conclusion

In this project, we employed a drone to capture aerial photographs of a mining site in Ghana and utilized a OpenCV stitching package to stitch the images together. While the OpenCV package yielded satisfactory results overall, it encountered difficulties when processing images with dense foliage, particularly in areas abundant with trees. To overcome this limitation, we developed a MATLAB algorithm that integrated GPS data and edge detection techniques to enhance the stitching quality specifically for tree photographs.

The integration of GPS data and edge detection within the MATLAB algorithm allowed for more refined ROI and identifying non-tree areas, leading to improved stitching outcomes. While the results were not flawless, the algorithm demonstrated promising advancements in overcoming the challenges posed by tree-rich scenes. This development highlights the significance of leveraging multiple algorithms and techniques to address constraints encountered during image processing tasks.

The project findings emphasize the importance of considering environmental factors and specialized algorithms when dealing with diverse landscapes and challenging scenarios. By acknowledging the limitations of the initial approach and proactively developing a tailored solution, we were able to enhance the stitching process for images featuring dense foliage. However, further refinements and optimizations could be explored in future iterations to achieve even more accurate and seamless stitching outcomes.

Overall, this project serves as a stepping stone toward improving the efficiency and accuracy of aerial image stitching methodologies for mining site documentation. The combination of drone technology, Python stitching algorithms, and the introduction of a tailored MATLAB algorithm demonstrate the potential for enhanced image processing in challenging environmental conditions. Continued research and development in this field will undoubtedly contribute to the advancement of mining site assessment and monitoring techniques, facilitating better decision-making processes and promoting more sustainable mining practices.

# Appendices

## A Appendix A: OpenCV Matching Results

The next nine figures are the matched features of the three feature detection methods. Figure 41, 42, and 43 are features from ORB feature detection. Figure 44, 44, and 44 are features from SIFT feature detection. The next three figures Figure 47, 48, and 49 are from BRISK feature detection.

The last two figures are from the "bad scenario", only photo 2 and photo 3 has features that are matched, further more, SIFT failed to match any features (see Figure 50 for ORB feature detection and Figure 51 for BRISK feature detection).

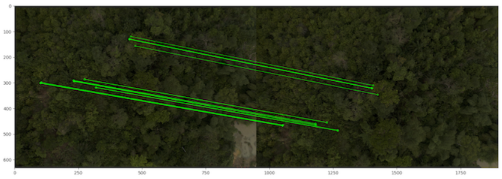


Figure 41: OpenCV ORB feature matching of photo 1 and photo 2.

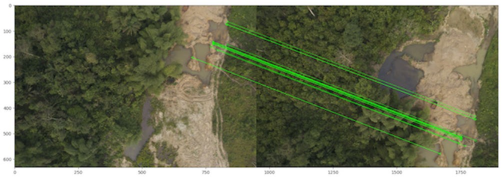


Figure 42: OpenCV ORB feature matching of photo 1 and photo 3.

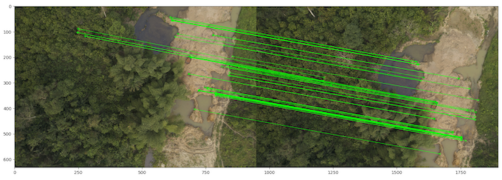


Figure 43: OpenCV ORB feature matching of photo 2 and photo 3.

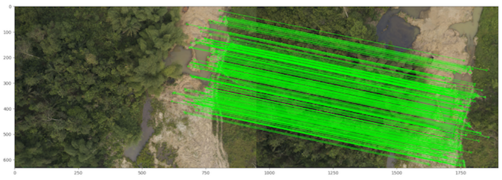


Figure 44: OpenCV SIFT feature matching of photo 1 and photo 2.

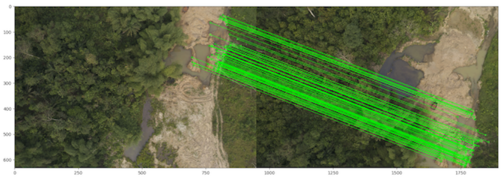


Figure 45: OpenCV SIFT feature matching of photo 1 and photo 3.

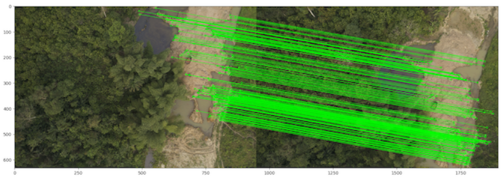


Figure 46: OpenCV SIFT feature matching of photo 2 and photo 3.

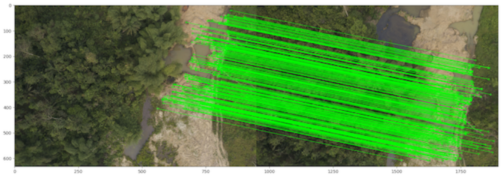


Figure 47: OpenCV BRISK feature matching of photo 1 and photo 2.

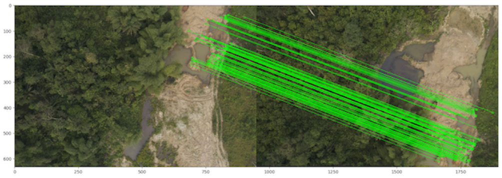


Figure 48: OpenCV BRISK feature matching of photo 1 and photo 3.

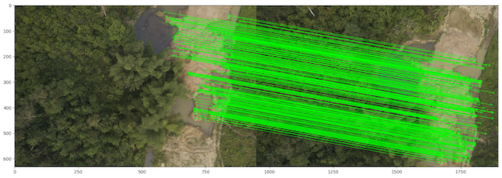


Figure 49: OpenCV BRISK feature matching of photo 2 and photo 3.

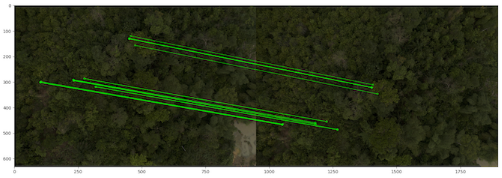


Figure 50: OpenCV ORB feature matching of photo 2 and photo 3.

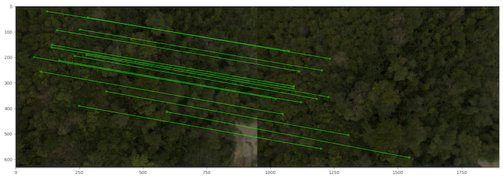


Figure 51: OpenCV BRISK feature matching of photo 2 and photo 3.

# References

[1] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[2] T. Zhang and M. Zhu, "Gps-assisted aerial image stitching based on optimization algorithm," in *2019 Chinese Control Conference (CCC)*, Jul 2019, p. 3485–3490.

[3] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International Conference on Computer Vision*, 2011, pp. 2548–2555.

[4] W. Lukas, "stitching," https://github.com/OpenStitching/stitching, 2023.

[5] G. Hilson, "Harvesting mineral riches: 1000 years of gold mining in Ghana," *Resources Policy*, vol. 28, no. 1, p. 13–26, Mar 2002.

[6] E. Ofosu-Mensah Ababio, "Historical overview of traditional and modern gold mining in Ghana," *International Research Journal of Library, Information and Archival Studies*, vol. 1, p. 6, Jan 2011.

[7] B. Kansake, R. Kudzawu-D'Pherdd, H. Tetteh, S. Dagbui, and E. Mends, "Artisanal and small-scale gold mining in Ghana -past, present and future," Mar 2021. [Online]. Available: https://www.researchgate.net/publication/356388265_Artisanal_and_Small-Scale_Gold_Mining_in_Ghana_-Past_Present_and_Future

[8] G. Hilson, "A contextual review of the Ghanaian small-scale mining industry," *Mining, Minerals and Sustainable Development*, vol. 1, no. 76, 2001.

[9] G. Hilson and C. Potter, "Why is illegal gold mining activity so ubiquitous in rural Ghana?" *African Development Review*, vol. 15, no. 2-3, pp. 237–270, 2003.

[10] E. A. Ofosu-Mensah *et al.*, "Historical overview of traditional and modern gold mining in Ghana," *International Research Journal of Library, Information and Archival Studies*, vol. 1, no. 1, pp. 006–022, 2011.

[11] N. Basu, E. Clarke, A. Green, B. Calys-Tagoe, L. Chan, M. Dzodzomenyo, J. Fobil, R. Long, R. Neitzel, S. Obiri, E. Odei, L. Ovadje, R. Quansah, M. Rajaee, and M. Wilson, "Integrated assessment of artisanal and small-scale gold mining in Ghana—part 1: Human health review," *International Journal of Environmental Research and Public Health*, vol. 12, no. 5, p. 5143–5176, May 2015. [Online]. Available: http://dx.doi.org/10.3390/ijerph120505143

[12] B. A. Teschner, "Small-scale mining in ghana: The government and the galamsey," *Resources Policy*, vol. 37, no. 3, p. 308–314, 2012.

[13] M. Hall, "Ghana nature hit hard by gold mines - but future looking up," *Deutsche Welle*, Feb 2018. [Online]. Available: https://www.dw.com/en/ghanas-environment-hit-hard-by-rampant-gold-mining-but-theres-hope-on-the-horizon/a-42323012

[14] K. Patel, "The large footprint of small-scale mining in ghana," *NASA Earth Observatory*, Jun 2021. [Online]. Available: https://earthobservatory.nasa.gov/images/148434/the-large-footprint-of-small-scale-mining-in-ghana

[15] C. K. Bempah and A. Ewusi, "Heavy metals contamination and human health risk assessment around obuasi gold mine in Ghana," *Environmental Monitoring and Assessment*, vol. 188, no. 5, p. 261, May 2016.

[16] F. Bian, Z. Zhong, X. Zhang, C. Yang, and X. Gai, "Bamboo – an untapped plant resource for the phytoremediation of heavy metal contaminated soils," *Chemosphere*, vol. 246, p. 125750, May 2020.

[17] OpenCV, "Opencv: Sobel derivatives," https://docs.opencv.org/3.4/d2/d2c/tutorial_sobel_derivatives.html.

[18] R. Gonzalez, R. Woods, and S. Eddins, *Digital Image Processing Using MATLAB.* Pearson Prentice Hall, 2004. [Online]. Available: https://books.google.com/books?id=YYuJQgAACAAJ

[19] R. Maini and H. Aggarwal, "Study and comparison of various image edge detection techniques," *International Journal of Image Processing (IJIP)*, vol. 3, 2004.

[20] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[21] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, Mar 2018, p. 1–10.

[22] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, p. 91–110, Nov 2004.

[23] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Computer Vision – ECCV 2006.* Springer Berlin Heidelberg, 2006, pp. 404–417. [Online]. Available: https://doi.org/10.1007%2F11744023_32

[24] OpenCV, "Opencv: Affine transformations," https://docs.opencv.org/4.x/d4/d61/tutorial_warp_affine.html.

[25] P. Antoniadis, "How to convert an rgb image to a grayscale | baeldung on computer science," https://www.baeldung.com/cs/convert-rgb-to-grayscale, Feb 2022.

[26] T. M. Inc., "Convert rgb image to grayscale - matlab im2gray," Natick, Massachusetts, United States, 2023. [Online]. Available: https://www.mathworks.com/help/matlab/ref/im2gray.html

[27] ——, "Feature based panoramic image stitching," Natick, Massachusetts, United States, 2023. [Online]. Available: https://www.mathworks.com/help/vision/ug/feature-based-panoramic-image-stitching.html

[28] D. Silcock, "Calculating meters per pixel from aerial photographs," Jul 2014. [Online]. Available: https://www.drewsilcock.co.uk/meters-per-pixel

[29] T. M. Inc., "Apply geometric transformation to image - matlab imwarp," Natick, Massachusetts, United States, 2023. [Online]. Available: https://www.mathworks.com/help/images/ref/imwarp.html