UNSYS digital

Journal of Unmanned System Technology

Analysis of Grass Area Aerial Image Set

Sutthiphong Spot Srigrarom, Yap Zhan Hong, Lee Meng Da, Cheong Kah Yuen

Aerospace Systems, University of Glasgow, Singapore

Abstract—In the recent years, the emergence of drone industry have expanded its utilities purpose in the commercial world. When coupled with computer vision, it can provide useful analysis of the aerial image set taken from the drone, and have since benefited across various industries such as agriculture and land surveillance. For a land surveillance mission, not only does it have the possibility of producing results with higher accuracy, it will also reduce the number of manpower and time required if similar tasks were carried out manually. Singapore is known as a City in a Garden due to the huge amount of greeneries spread across the country. Many of these greeneries consists of large empty grass patch which need maintenance work by hired contractors and visual checks by agency officers every few weeks. With a few thousands site large and small, the amount of manpower required is tremendous. In this paper we propose a solution that can resolve the manpower issue and offer visual analysis through computer vision methods.

Keywords— computer vision, aerial surveillance, kw3

DOI: 10.21535/just.v7i2.1023

I. Introduction

The workflow of the proposed systems consists of using a quadcopter, DJI F450, to first conduct an autonomous flight mission, capturing aerial images of a grass patch. After which the image set is used as input into a Matlab program whereby a 3-Dimension (3D) model can be generated for analysis, using a popular computer vision method, Structure from Motion (SfM) algorithm. Next the 3D model will go through a proposed RGB filter to identify the difference between short or dense long grass, resulting in a percentage of cut grass and a coloured visual model to highlight the areas of which the grass is long. The whole process within Matlab is done through a single run without the need to extract and import data from one software to another.

The main objective of SfM algorithms is to extract 3D information from a set of 2D images. Most SfM algorithms currently shares the same basic processing workflow of first detecting 2D features from every images, match 2D features among the images, construct tracks from the matches, and lastly to generate SfM model from the tracks. A track contains information of the 3D coordinates of a reconstructed 3D point, corresponding to the 2D coordinates of the images. The algorithms used within the SfM workflow are Minimum Eigenvalue for feature detections[1], Kanade-Lucas-Tomasi (KLT)[2] for feature matching, M-estimator Sample Consensus (MSAC)[3] to filter outliers and Triangulation Algorithm[4][5] to reconstruct 3D model. An addition function, Camera Calibrator application, within Matlab is used to obtain the parameters of the camera used.

Different plants species or different density of the same species can results in different shade of green[6]. This make it possible to implement a RGB analysis of the reconstructed 3D model by going through a RGB filter to detect specific colour range. However, there are certain limitations that can cause varying results such as weather or lighting conditions. It can be seen in other paper that shrubs with considerable height have proven to be difficult and expensive[7]. The target of analysis of this project which is grass, is of a much smaller scale and therefore further research for a more precise reconstruction and analysis needs to be done. Despite the limitations, through our results it is shown that the method is visible and has potential to be implemented for commercial use in the future.

II. Related Research

Aerial photography has played a vital role in ecological management for the past few years. It enables the assessment of terrain and landscape changes. The two most commonly used methods are vertical and oblique aerial photography, but both methods cause errors such as relief displacement and are required to be rectified before any of their captured information can be accurately evaluated. However, with the correction of these errors, this alternative remote sensing approach has proven to be more cost effective for its various applications such as assessment of vegetation changes and forest inventory in comparison to implementation of Light Detection and Ranging (LIDAR) or through satellite imaging.

Nonetheless, it is not possible to retrieve targeted information from the pictures directly. The images from aerial photography have to undergo digital image processing and application-appropriate filters to access desired data. There are research done to prove that even though LIDAR is the preferred technological method to produce 3D visualisations, there are certain advantages to using photogrammetry method such as SfM[8]. A few of these advantages include economic efficiency, single workflow in varying areas (outdoor or indoor), photogrammetry flight mission can be completed much faster than LIDAR mission, and accuracy of 3D points are often better than ≈ 1 pixel.

There are many ways to stitch the collection of images together. The straightforward way will be to combine them

Corresponding author: Sutthiphong Spot Srigrarom
(e-mail: spot.srigrarom@glasgow.ac.uk)

This paper was submitted on July 16, 2018; and accepted on April 8, 2020.

through image stitching; direct or feature-based. It compares the pixels' intensity or image features and stitches images with sufficient similarities[9]. With images stitching, a bird's eye view of the landscape can be obtained through acquiring multiple partitioned views of the whole area. However, the most direct method may not be the best in the case of this project. In the case where the camera's view is not perpendicular to the ground, such areas will appear to be denser as compared to perpendicular areas and subsequently cause inaccuracy during the analysing stage.

Multiple research done have used 3D reconstruction of aerial images for reconstructing landscapes of buildings or vegetation[10][11][12]. This shows that SfM algorithm is an established method of obtaining landscape information from images. However, the main disadvantage of SfM is that there cannot be any obstacles obstructing a particular view; the area behind or underneath the obstacle cannot be constructed and will be left with an empty hole (no points can be plotted). In the case of reconstructing grass patches, it is relatively unaffected by such a disadvantage.

Canopy height model algorithm is another method that can be used to identify the height of vegetation[13]. It is mainly used to calculate the height of trees by finding the difference of two different models; Digital Terrain Model and Digital Surface Model. The DTM is the bare earth extraction of the 3D model by classifying the ground points (lowest 3D points) after the segregation of the model into multiple smaller area. Thereafter the difference between the highest points of the 3D model within each segregated area known as DSM, and the ground points of DTM will result in the height of the trees. However, the algorithm was thought not to be suitable as the height of grass are often very short (usually in cm), and therefore, will require extremely high resolution of images in order to achieve the necessary richness of 3D points to identify the height differences.

III. Aerial Photogragphy

Aerial photography is defined as pictures are taken from an elevated plane with an airborne camera. From the comparison of vertical and oblique aerial photography in [14], vertical aerial photography is chosen for its ability to display open areas with precision while maintaining uniform scale. This project utilizes an Unmanned Aerial Vehicle (UAV), DJI F450 quadcopter, equipped with a PixHawk fight controller, 3DR UBlox GPS and compass module, 433 MHz Telemetry, FPV 2 axis brushless gimbal, and Firefly 7s camera.

In order to conduct full autonomous flight mission, QGroundControl (QGC), an open-source Ground Control System is used to control, observe, and plan the flight mission. The functions within QGC enable us to set waypoints as flight path for the quadcopter, specify the locations for the camera to trigger automatically, and adjust flight parameters such as speed, altitude, and home location which is the location for the quadcopter to land after completing its mission. By selecting the amount of overlay we want between the images, QGC can calculate the optimal trigger locations for the camera. The parameters set for the flight mission and the targeted flight area are presented in Table 1 and Figure 2.

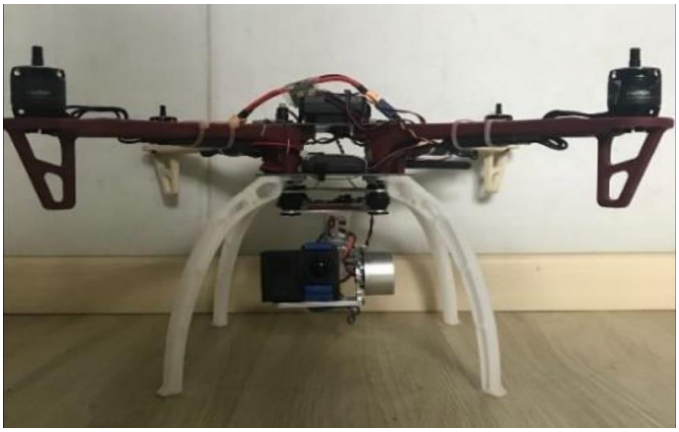

Figure 1 DJI F450 Quadcopter with Firefly 7s camera

Table 1 QGroundControl flight parameters

Parameters	Value
Flying speed	0.5 m/s
Flying altitude	3 m
Trigger Distance	0.5 m
Angle	1°
Spacing	3 m
Turnaround distance	1 m

Figure 2 Waypoints set in QgroundControl as flight path

In total three set of images are collected for different purpose, and are listed as Set-A, B, C for ease of future referencing. Set-A shown in Figure 3, consists of four images of a small grass area, captured by an iPhone 7Plus on ground. The purpose of this set to supply the Matlab program with a small sample data to test the functionality. Set-B shown in Figure 4, consists of twenty four images of a knoll, captured by an iPhone 7Plus mounted on the quadcopter. The purpose of this set is to supply small sample data while finding the optimal parameters within both QGC and Matlab program. Set-C shown in Figure 5, consists of fifty images of a large grass landscape, captured by a Firefly 7s camera mounted on the quadcopter. The purpose of this set is to find out the visibly of implementing the program on the propose target area of grass patch and find out if the results are acceptable. All three sets of images were able to produce positive results.

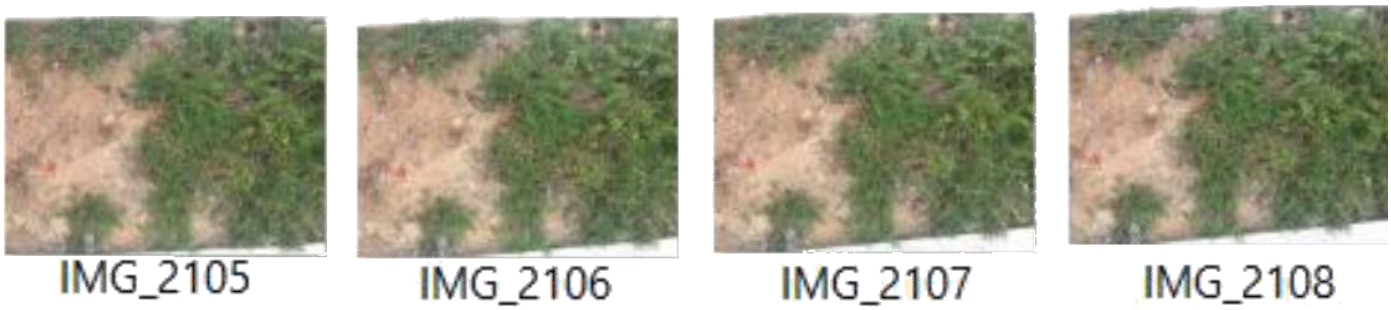

Figure 3 Photo collection (4) of small grass patch

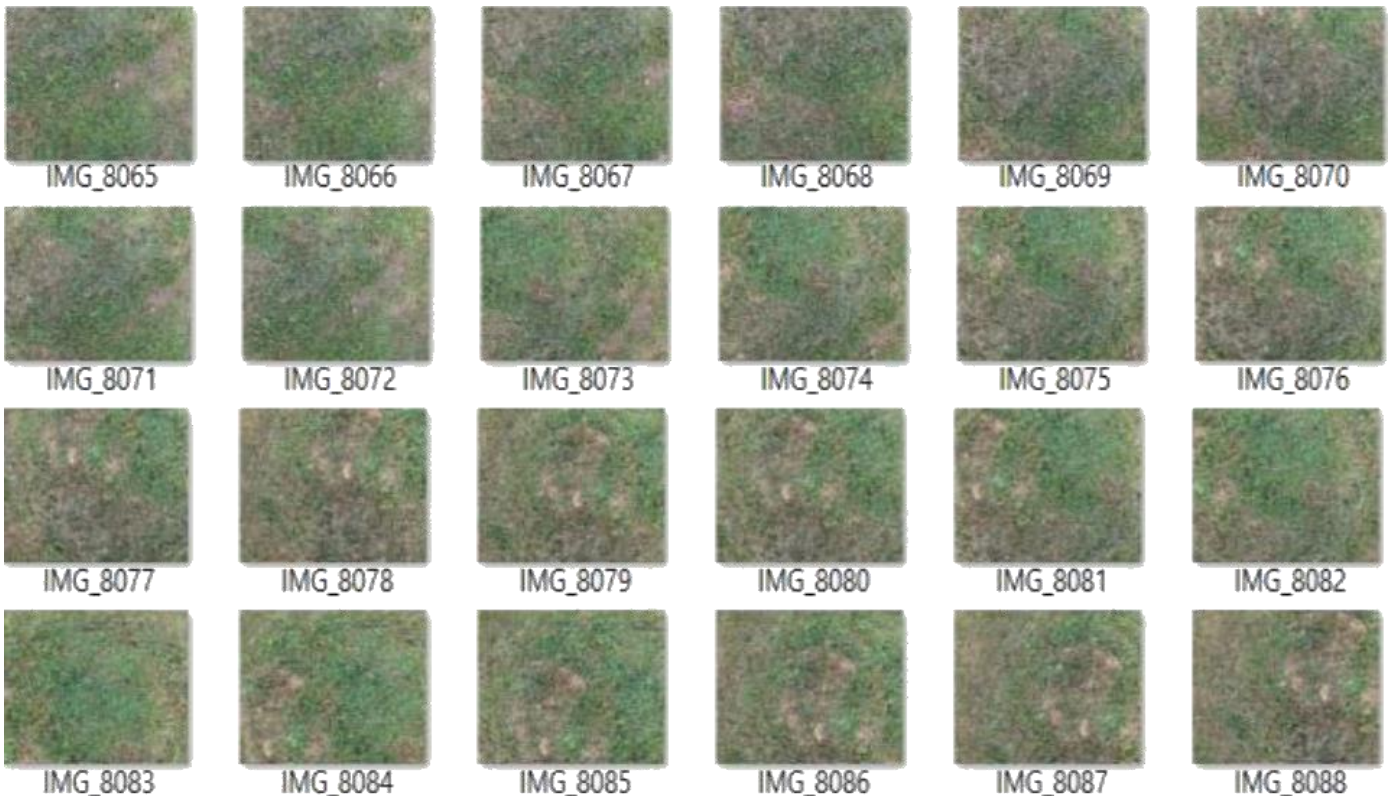

Figure 4 Photo collection (24) of a knoll

Figure 5 Photo collection (50) of a large grass landscape

IV. MATLAB PROCESS WORKFLOW

Computer Vision System is one of the toolbox offered by Matlab that is used heavily in for our program. Some of the functions that it provide are feature detection, extraction, matching, as well as object detection and tracking. Most of the different algorithms required for the program to work can be found within the toolbox. However, the function of RGB filter used for analysis had to be coded by us as Matlab currently do not have such a function. Before the reconstruction of the 3D model, the calibration of the camera is required. This is to extract the necessary camera parameters that will later be used for the triangular algorithm. For this step, the Camera Calibrator application in Matlab will be used. Thereafter, the image set will be input into the program to run through the rest of the process for generation and analysis of the 3D model. The breakdown of the entire process of the Matlab program can be seen in Figure 6.

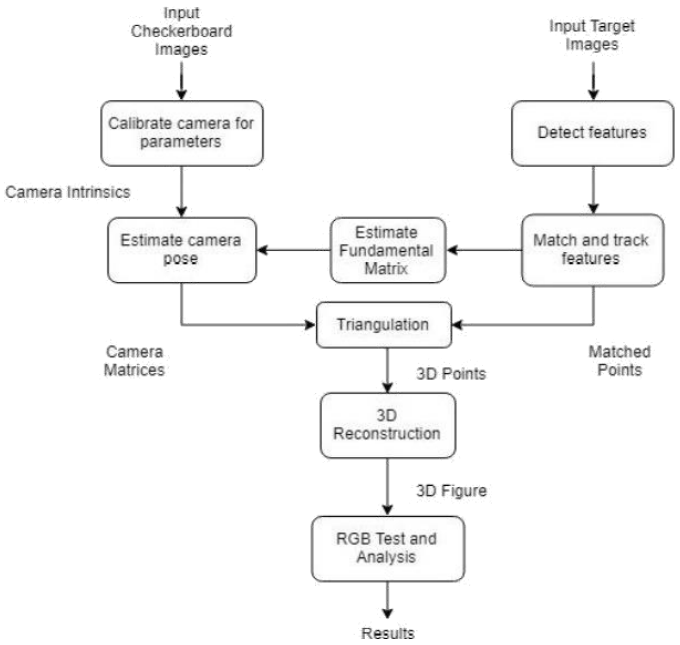

Figure 6 Matlab process workflow

V. MINIMUM EIGENVALUE ALGORITHM

The first step of the SfM algorithm is to detect features in target images. SfM algorithm have been established well over ten to twenty years. Since then, many feature detection algorithm have been developed such as Scale-Invariant Feature Transform (SIFT), Speed Up Robust Feature (SURF), Features from Accelerated Segment Test (FAST), Oriented FAST and Rotated BRIEF (ORB), Maximally Stable Extremal Regions (MSER), Harries Corner Detection, and Minimum Eigenvalue[15], each having its own advantage and disadvantage. For this project, the requirement for accuracy outweighs the need for speed, therefore, based on comparison done in another research paper[16], the Minimum Eigenvalue algorithm is chosen or its ability to detect the most number of feature points per image. The Minimum Eigenvalue algorithm was developed based on Harris Corner Detector. However, the reason why the Minimum Eigenvalue is used more often is because of the slight variation in the selection criteria between the two algorithms, which allows the algorithm to perform better.

The definition of detecting features of an image mean to extract corners using computer vision algorithms and infer the features of the image. Corners represents image locations that have intensity that are large and changes in multiple directions. They are great as features because they are uniquely identifiable across multiple images. The sum-of-squared-difference (SSD) can be used to represent the change in intensity along a direction.

$$D(x,y) \approx \big(I(i+x,j+y) - I(i,j)\big)^2 \quad \textbf{(1)}$$

where $D(x,y)$ is the difference between original and moved window, x,y is the window's displacement in x and y direction respectively, and I is the intensity at position (x,y). Applying Taylor theorem to (1) will result in,

$$D(x,y) \approx [x \quad y]\left(\sum \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}\right)\begin{bmatrix} x \\ y \end{bmatrix} \quad \textbf{(2)}$$

$$G = \sum \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \quad \textbf{(3)}$$

The applications of eigenvalues and eigenvectors are numerous. Other than being widely known as a characteristic root, it can help to determine the window's suitability. Changes in intensity of certain direction are characterized by matrix G in (3). In Minimum Eigenvalue algorithm case,

$$G = \sum \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \quad \textbf{(4)}$$

By selecting a minimum eigenvalue threshold, points out of undesirable range can be filtered out and more precise feature points can be obtained. The eigenvalues of matrix G will then go through the selection criteria for determining the corners, which is summarized in Table 2.

Table 2 Minimum Eigenvalue Algorithm criteria

Corner Criteria	
λ_1 and λ_2 are close to zero.Flying speed.	No corner is found.
λ_1 and λ_2 differ by a large margin.	An edge is found.
λ_1 and λ_2 are far greater than zero.	Corner is found.

VI. KANADE-LUCAS-TOMASI FEATURE TRACKER

After the features of every images in the set have been extracted, we need an algorithm to match the features across the images while yielding the best results. Two algorithm had the potential of complete the task; Kanade-Lucas-Tomasi (KLT) Feature Track and Smallest Univalue Segment Assimilating Nucleus (SUSAN). Comparison between the two algorithms were made[17] and KLT was chosen due to the ability to detect

and match "weak" features. KLT has an advantage over SUSAN in terms of computational complexity as well. Using Newton-Raphson method, features across images are tracked by minimizing the difference between the two windows.

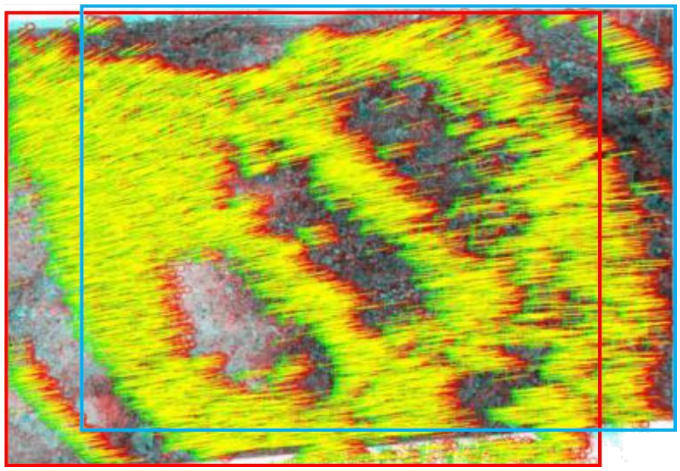

Figure 7 Tracking of features between two images from Set-A

As the camera moves from original position to the next within a short interval, the images taken from these positions will strongly relate to one another but with a slight change due to the displacement. KLT finds this displacement through the shift in the images by using the comparison of the found corner features in the images. In order to track the features from current to past images, the measurement of match between these fixed-size windows will be determined by the minimization of SSD intensity across the images. The displacement that needs to be solved is defined as the one that minimizes the SSD. Due to small inter-frame motion between images, an approximation can be made for the current window in the image by a translation of the previous window of the previous image. The intensity of the current image in the translated window can be represented the same as the original window plus a residue term. As the displacement vector is small, the intensity function can be represented as,

$$I(x-d) = I(x) - g \cdot d \tag{5}$$

where d is the vector of displacement and g is the spatial gradient of image intensity. The residue term can then be defined as,

$$\begin{aligned} \epsilon &= \int_W \left(I(x) - g \cdot d - J(x)\right)^2 \cdot w \cdot dx \\ &= \int_W (k - g \cdot d)^2 \cdot w \cdot dx \end{aligned} \tag{6}$$

where $k=I(x)-J(x)$, w is a weighting function usually equals one in simplest case, and W is the feature window. The minimization can be achieved by differentiating (6) with respect to d and the result will be set to zero.

$$\int_W (k - g \cdot d) \cdot g \cdot dA \tag{7}$$

The term d is assumed to be within W and $(g \cdot d)=(ggT)d$, which leads to,

$$d\left(\int_W g \cdot g^T \cdot dA\right) = \int_W k \cdot g \cdot dA \tag{8}$$

Expression (8) represents the basic step of the algorithm tracking procedure and it can be rewritten as $(G) = e$ where, G is the matrix mentioned in previous section and e is a vector representing the difference between two frames of window. Both matrix G (from one frame of window) and vector e (from difference between two frames of window) are computable, which leaves displacement d as the solution to (8). Figure 7 shows the results of matched features between two images from Set A.

Figure 8 Checkboard images captured by iPhone 7Plus

Figure 9 Checkboard images captured by Firefly 7s

VII. Camera Parameters

The Camera Calibrator application in Matlab is used to estimate parameters such as camera intrinsic, extrinsic, and lens distortion. These parameters are necessary for the triangulation algorithm. The steps involving the Camera Calibrator application include capturing 20 images of checkerboard with the camera used for taking the image Set-A to C (iPhone 7Plus and Firefly 7s), shown in Figure 8 and Figure 9. Then they are loaded into the application, and calibration will take place to produce the following results shown in Figure 10 to Figure 11.

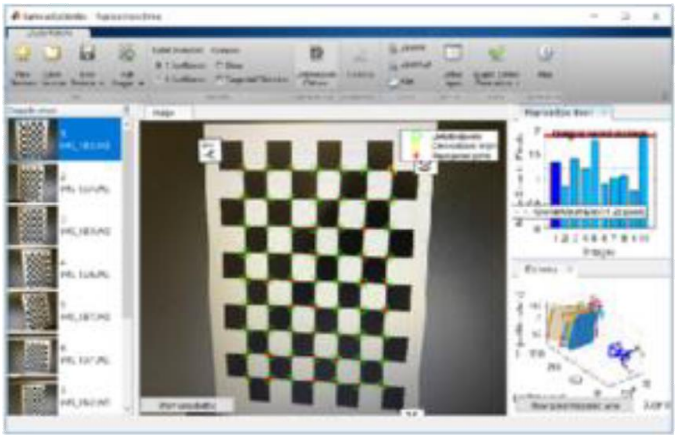

Figure 10 Calibration process of iPhone 7Plus

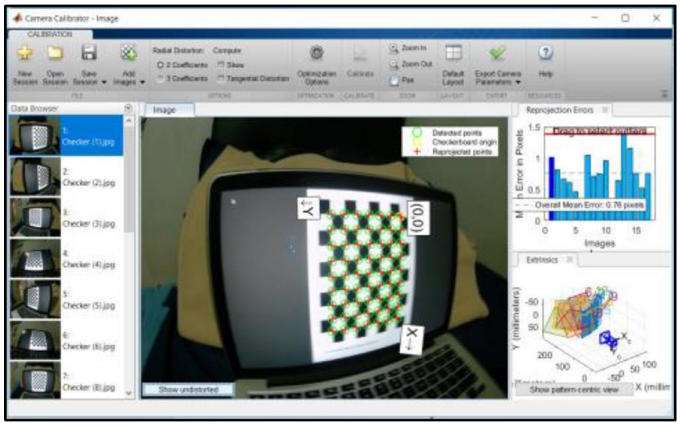

Figure 11 Calibration process of Firefly 7s

This method was completed manually from input of images to camera calibration before exporting and importing camera parameters back to Matlab scripts. For a more automated approach, the application can also be used through a more programmatic manner. Through the programmatic way of using Matlab functions of the application, camera calibration geometric parameters such as intrinsic, extrinsic and lens distortions of the camera can also be found for usage in a later stage without the need of exporting and importing back into the main Matlab script. The intrinsic matrix K found for iPhone 7Plus rear camera,

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad \textbf{(9)}$$

$$K_{\text{iP7+}} = \begin{bmatrix} 3301.741901 & 0 & 1484.983003 \\ 0 & 3536.728071 & 2054.134609 \\ 0 & 0 & 1 \end{bmatrix}$$

The intrinsic matrix K found for Hawkeye Firefly 7S,

$$K_{\text{HF7S}} = \begin{bmatrix} 775.920345 & 0 & 788.762178 \\ 0 & 839.901956 & 588.678406 \\ 0 & 0 & 1 \end{bmatrix}$$

where fx is X-Axis focal length in pixels, fy is Y-Axis focal length in pixels, cx approximately half of image's width in pixels, and cy approximately half of image's length in pixels.

Intrinsic matrices are related to the specifications of the camera in use and can be directly utilised. However, extrinsic parameters contain the world points or camera's physical orientation and differ for different images. Hence, extrinsic parameters of the checkerboard images found through the camera calibration application cannot be used for the targeted images and have to be acquired through an alternate way mentioned in another section later.

VIII. M-Estimator Sample Consensus

In order to achieve 3D reconstruction, the intrinsic projective geometry of the images from the camera is required as well. It is dependent on the camera's intrinsic parameters (intrinsic matrix found above) and its relative pose. When two pictures of the same target from different views are taken, the intrinsic project geometry, also known as epipolar geometry describes how they are related[20]. The fundamental matrix covers this intrinsic geometry, and is different from the essential matrix. The fundamental matrix contains corresponding pixel coordinates points (matches) between two images. However, some of corresponding points are considered outliers (incorrect matches), and an estimating algorithm can be used to estimate the fundamental matrix that contains good corresponding points, which in our case we applied the M-Estimator Sample Consensus algorithm (MSAC).

Although there are many minimal solvers such as the Eight-Point algorithm to estimate the fundamental, however they have the critical assumption of no outliers and are not equipped to filter out any. MSAC, which is a modified version of Random Sample Consensus (RANSAC), is an iterative method designed for estimating general parameters, removing outliers, and handling a large amount of input data[18]. The framework of RANSAC starts solving for outliers problem by working with a randomly selected small subset of feature matches. Then it assess the error residuals for the remaining matches using the solution obtained from the previous step. The consensus set for this solution is the small subset of matches with error residuals below certain predefined threshold. Lastly is to repeat the steps a number of times and the solution that provide the largest consensus set will be selected as an estimation of the fundamental matrix[19].

The main difference between RANSAC and MSAC is the ability to achieve a more robust estimation with no additional processing load[3]. One issue with RANSAC is that robust estimation can be very poor if the threshold T for considering inliers (error below threshold) is set too high. For RANSAC, it has a minimum cost function of,

$$C = \sum_i p(e_i^2) \quad \textbf{(10)}$$

where e is error residual, and $p(\,)$ is corresponding points,

$$p(e_i^2) = \begin{cases} \mathbf{0}, & \boldsymbol{e^2 < T^2} \\ \text{constant}, & \boldsymbol{e^2 \geq T^2} \end{cases} \quad \textbf{(11)}$$

Expression (11) can be explained as each outlier records a constant penalty while inliers records zero. This leads to a

problem when $T2$ is large enough, then all solutions will cost the same because all feature matches would be considered as inliers, and this will result in poor estimation. MSAC can solve this issue can amending expression (11) to,

$$p(e_i^2) = \begin{cases} \boldsymbol{e^2}, & \boldsymbol{e^2 < T^2} \\ \boldsymbol{T^2}, & \boldsymbol{e^2 \geq T^2} \end{cases} \quad \textbf{(12)}$$

In expression (12), outliers will still record a penalty but inliers will be recorded based on how small the errors are. The smallest cost function computed will be the best estimation of the fundamental matrix. Hence, an accurate estimate of the fundamental matrix containing good inlier points can be obtained by running the found matched features from two images through MSAC.

IX. Extrinsic Parameters

As mentioned above, through the Camera Calibrator application, we are unable to use the extrinsic parameters found there as it differs for every images. The extrinsic parameters expresses the relationship between the world coordinate system and the camera frame. Having the intrinsic matrix of the cameras and the fundamental matrix, we can obtain the essential matrix. Essential matrix contains corresponding points of images as well, but the points are in normalized image coordinates. We can compute the extrinsic matrix by first decomposing the essential matrix into a rotational matrix and a skew-symmetric matrix. The skew-symmetric matrix is a 3×3 matrix that is correspondence to the translation vector,

$$S = \begin{bmatrix} 0 & -s_3 & s_2 \\ s_3 & 0 & -s_1 \\ -s_2 & s_1 & 0 \end{bmatrix} \quad \textbf{(13)}$$

Expression (13) represents the basic form of a skew-symmetric matrix and the corresponding translational vector will be in the form of ($s_{1,2}$,s_3). The rotational matrix and translation vector together describes the motion of the camera while capturing the two image views. The method used to decompose a given essential matrix is called the singular value decomposition (SVD). SVD is a mathematical topic of Linear Algebra, which can be used to decompose any matrix A into the following form,

$$A = U \cdot \Sigma \cdot V^{\mathrm{T}} \quad \textbf{(14)}$$

U and V are orthogonal matrices and Σ contains non-zero entries ($\lambda_1, \ldots, \lambda_n$) which are called the singular values of matrix A, and they are positioned in the main diagonal of matrix Σ. Using SVD, the essential matrix is decomposed into the form same as expression (13). From there, further factorization can achieve the form of essential matrix E which is the product of skew-symmetric matrix S, and rotational matrix R. There are a few papers explaining clearly the derivation of the decomposition of essential matrix using SVD[21][22]. After obtaining the rotational matrix and translational vector, we can reconstruct the 3D model by triangulation. As the rotational matrix and translational vector changes every two images, only the first set of matrix and vector from the first two images from Set-A is shown below to indicate how the matrix and the vector should look like,

$$\text{Rotational} = \begin{bmatrix} 0.9994 & -0.0333 & -0.0076 \\ 0.0333 & 0.9994 & 0.0058 \\ 0.0074 & -0.0060 & 1.0000 \end{bmatrix}$$

$$\text{Translational} = [-0.5327 \quad -0.5606 \quad 0.6340\,]$$

X. Triangulation Algorithm

Triangulation refers to solving for the position of a point in space given its location in two images taken with cameras with known intrinsic and extrinsic parameters. Through the intersection of two know rays in space, the position of the point can be obtained in absence of noise. However, the rays do not usually meet when noise is present and therefore, it is critical to find the best point of intersection. The triangulation method used for this project was introduced by Richard I. Hartley and Peter Sturm[4]. It is referred as the Polynomial method and is more optimal than other triangulation method because of its capability to seek the global minimum of the cost function while using a non-iterative algorithm.

At this stage it is critical to have the camera matrices and fundamental matrix. A pair of matching points (i and i') will have its corresponding rays meet in space if the following relationship with fundamental matrix F is satisfied,

$$i' F\, i = 0 \quad \textbf{(15)}$$

However, due to digitization errors, often times the features are displaced from its correct position as the images are subjected to Gaussian noise, the actual corresponding points ($\hat{\imath}$ and $\hat{\imath}'$) should still be near to the measured points (i and i'). Therefore, we need to solve for the points ($\hat{\imath}$ and $\hat{\imath}'$) that minimize the sum-of-squared-distances while still subjected to the epipolar constraint in expression (15) The expression that define the relation between (i and i') and ($\hat{\imath}$ and $\hat{\imath}'$) is as follows,

$$d(i, \hat{\imath})^2 = d(i', \hat{\imath}')^2 \quad \textbf{(16)}$$

When the epipolar constraint is met, it means that any pair of points are on the corresponding pair of epipolar lines (λ and λ') in the two images. Instead of searching for points around the images to minimize the sum-of-squared-distances, the method search for those points along the epipolar lines. Expression (16) is then modified to,

$$d(i, \lambda)^2 = d(i', \lambda')^2 \quad \textbf{(17)}$$

The outline of this triangulation method to minimize expression (17) begins with parametrizing the epipolar lines of first image by t which results in (t). Then it compute the corresponding epipolar line $\lambda'(t)$ in second image using the fundamental matrix F. Next is to express expression (17) as a function of t, and lastly is to solve for t which will minimize expression (17). In order to simplify the expression, a rigid transformation is made to put both (i and i') at the origin in homogeneous coordinates $(0,0,1)^{\mathrm{T}}$. The epipoles at both images are also given the coordinates on x-axis of $(1,0,f)^{\mathrm{T}}$ and $(1,0,f')^{\mathrm{T}}$. This will lead to the fundamental matrix F having a form of,

$$F = \begin{bmatrix} f\,f'd & -f'c & -f'd \\ -f\,b & a & b \\ -f\,d & c & d \end{bmatrix} \quad \textbf{(18)}$$

The epipolar line on the first image calculated to be the cross product of $(0,,1)^T$ and $(1,0,f)^T$ as it is considered to cross the point and the epipole of the first image. The corresponding epipolar line on the second image is calculated using the fundamental matrix F. The total sum-of-square-distances then given as,

$$s(t) = d(i,\lambda(t))^2 + d(i',\lambda'(t))^2 = \frac{t^2}{1+f^2\,t^2} + \frac{(c\,t+d)^2}{(a\,t+b)^2+f'^2\,(c\,t+d)^2} \quad \textbf{(19)}$$

The maxima and minima of (t) will occur when $s'(t)$ equals zero. Hence, the objective of this triangular method is to solve for the absolute minimum of expression (19) using elementary calculus.

So far the processes mentioned since section V are only for a pair of images. Within the Matlab script, a loop had to be implemented to apply the algorithms to every subsequent pairs of images. Once the 3D points are found, they are plotted onto 3D coordinate system, seen in Figure 12 to Figure 17. However, these points originally do not have their original colours and follows Matlab default colour map. The 3D points are then given back their colour from the original images to provide actual captured colours.

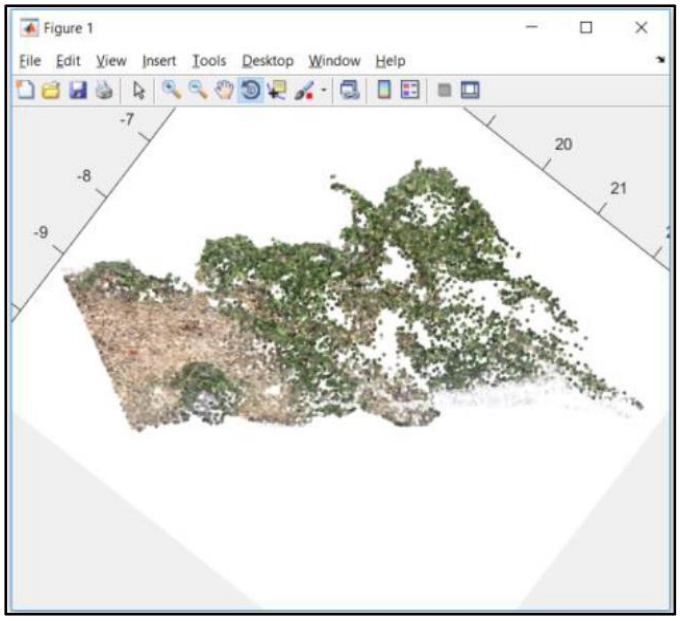

Figure 12 Reconstruction 3D model of Set-A

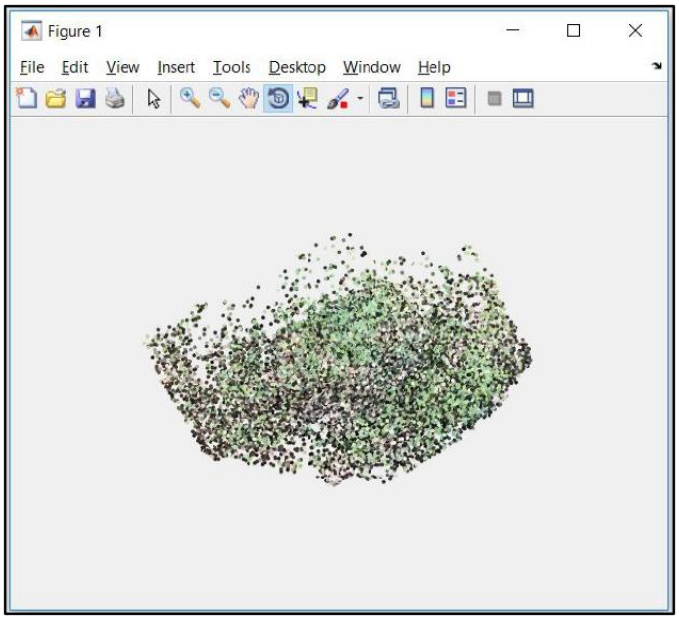

Figure 14 Reconstruction 3D model of Set-B

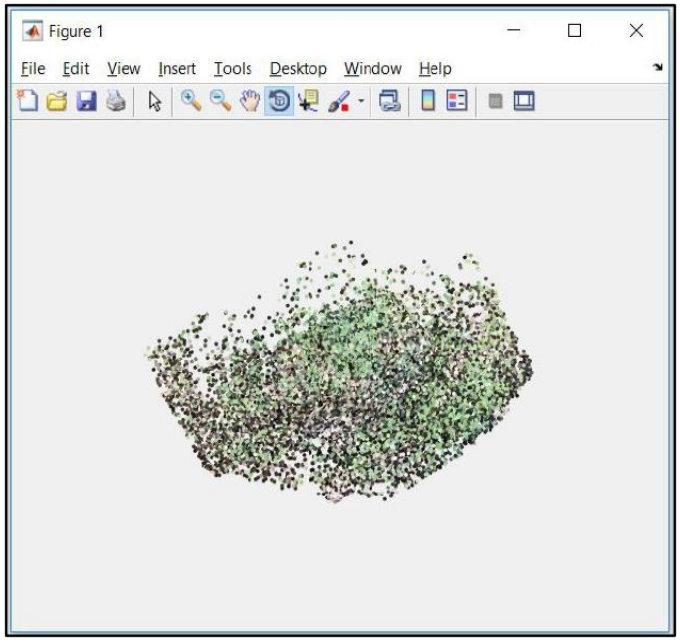

Figure 13 Another view of Set-A 3D model

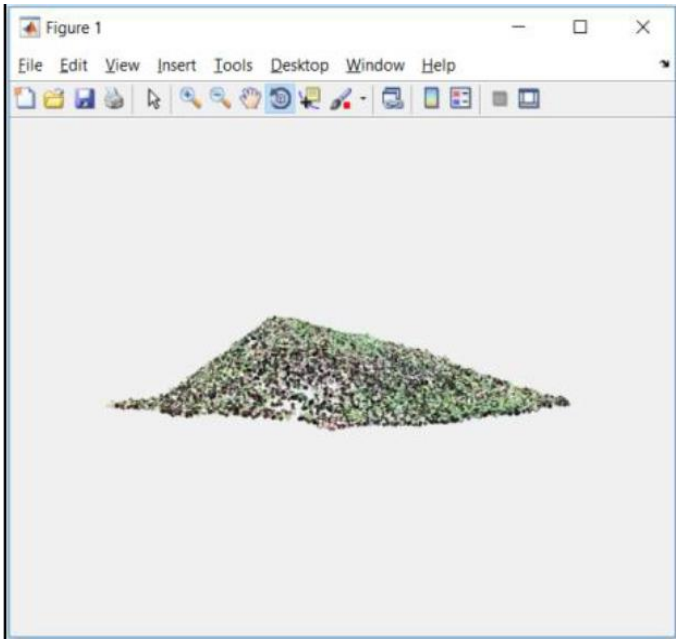

Figure 15 Another view of Set-B 3D model

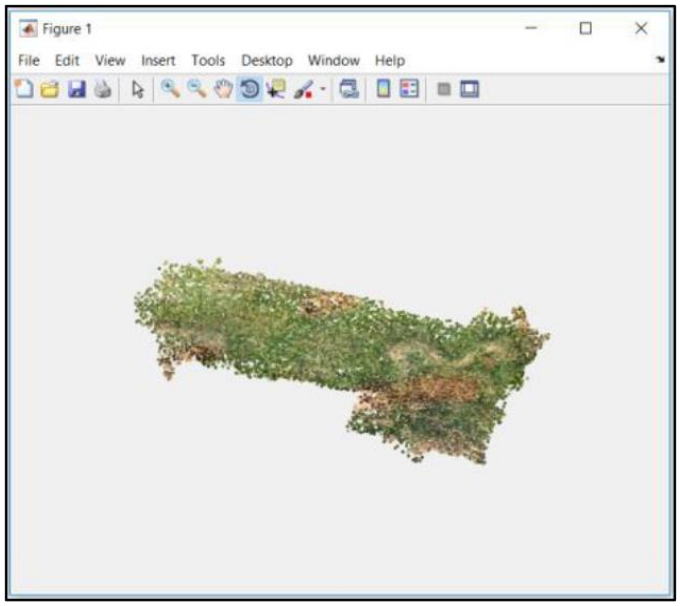

Figure 16 Reconstruction 3D model of Set-C

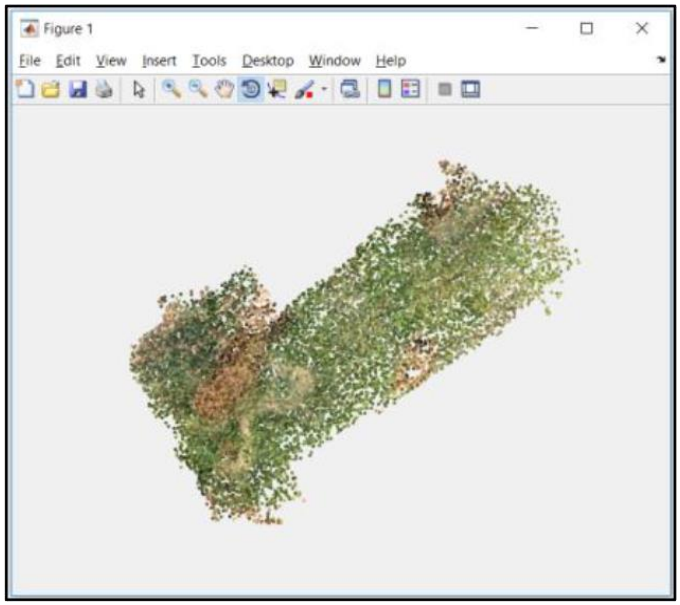

Figure 17 Another view of Set-C 3D model

XI. RGB ANALYSIS

Even though there are many different colour models such as Black & White, Greyscale, RGB, Cyan-Magenta-Yellow-Black (CMYK), and Hue, Saturation and Value (HSV), RGB was chosen as it can clearly pick up the colour green which is the main colour for our target of analysis, grass, and the 3D models from triangulation had the colour information of every vertexes in RGB model. For the RGB colour model, each colour, red, green and blue, has a possible integer between 0 and 255 which contributes to the final colour. When all three channels are close to or have the same values, it will belong to the greyscale. For example, from (0, 0, 0) to (255, 255, 255), the final colour will be black and gradually move towards white as shades of grey. RGB analysis for 2D images involve finding the RGB matrix for the target image and producing colour statistics of the image. Since the plotted 3D points have image's original colour, this makes a RGB analysis possible.

Knowing that different plants or the density of plants can have different colours and different shades of green, it is possible to find the RGB difference of cut and uncut areas [6]. By searching through the points according to an acceptable range of Red, Green and Blue for properly or badly cut grass, the number of such acceptable points can be found. A grass colour model will be RAL 6010, with RGB value of (72, 111, 56), and this model is used for this project to test the functionality and visible of doing a RGB analysis on 3D models. This is just an example with the assumption of this grass species cut below certain height will have colour similar to RAL 6010. An advantage of RGB analysis over grass area is that even if nearby structures are included into the 3D figure from the images taken, they can also be filtered out appropriately using such RGB filtering as long as they contrast from grass RGB values. A filter is designed to go through all of the vertexes of the 3D models to highlight any 3D points that is within a range of RAL 6010, by replacing that 3D point with the colour red. Based on the amount of 3D points highlighted, a percentage of cut grass in the area can also be calculated. The 3D model results are represented in Figure 18 to Figure 23. Figure 24 shows the Matlab processes as it run through the program and the last line shows the calculated percentage of grass cut in the area.

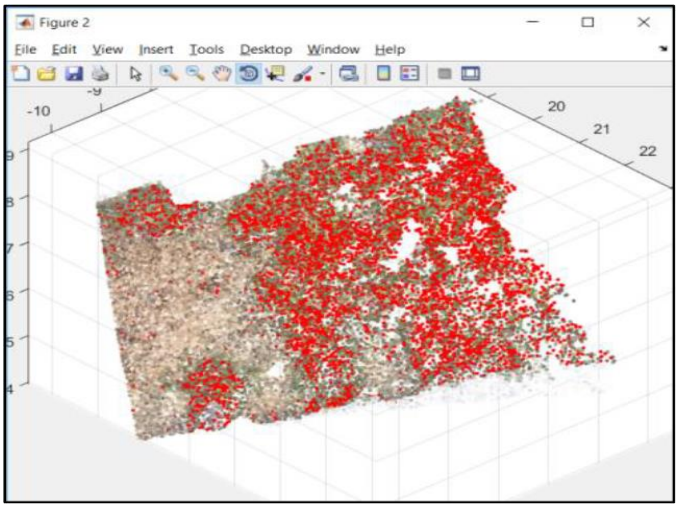

Figure 18 RGB analysis of Set-A

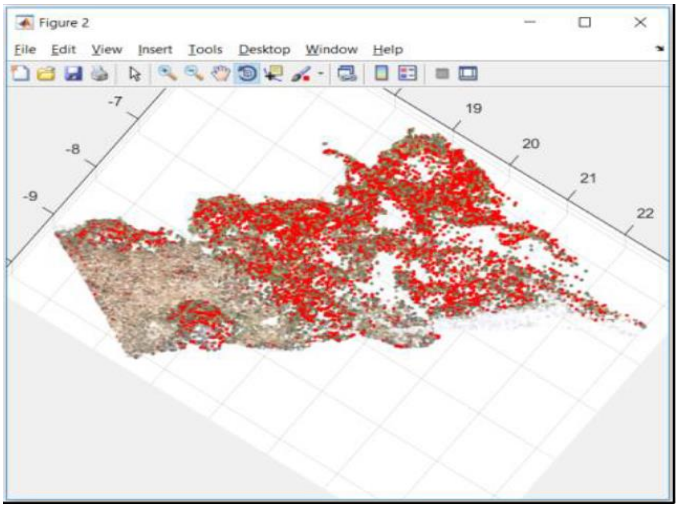

Figure 19 Another view of RGB analysis of Set-A

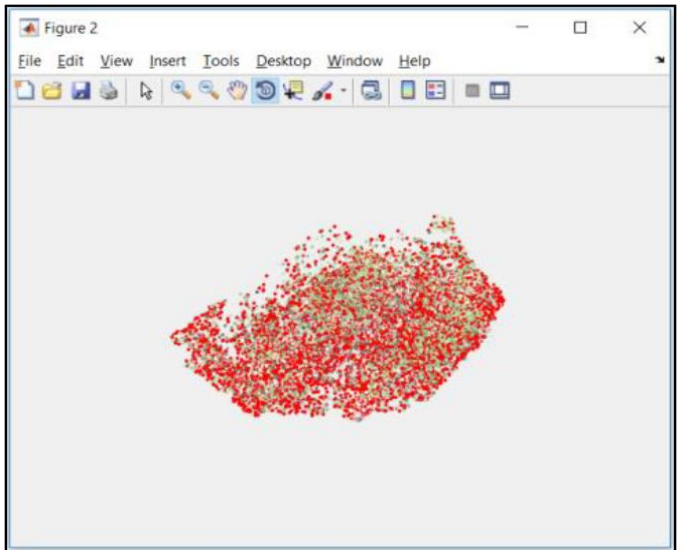

Figure 20 RGB analysis of Set-B

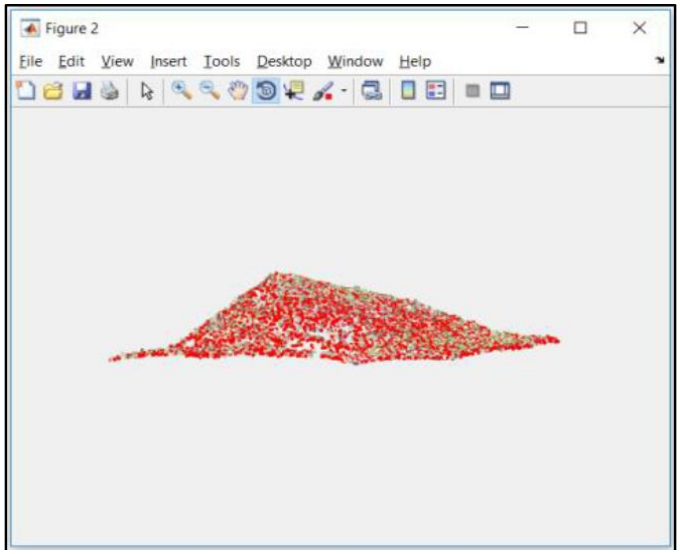

Figure 21 Another view of RGB analysis of Set-B

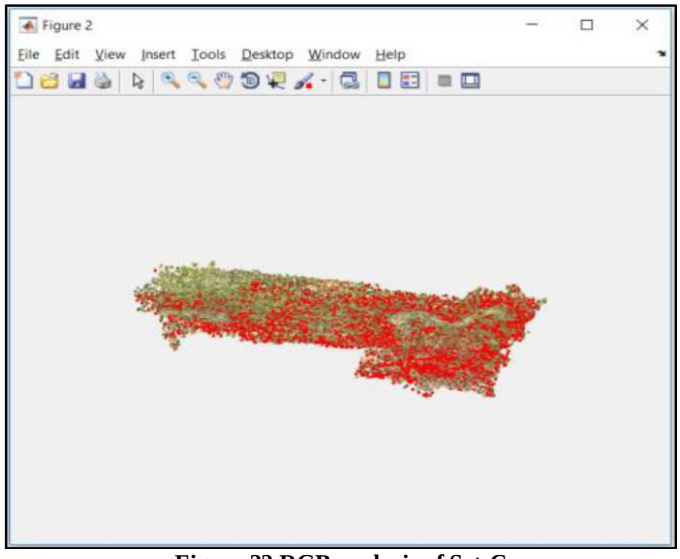

Figure 22 RGB analysis of Set-C

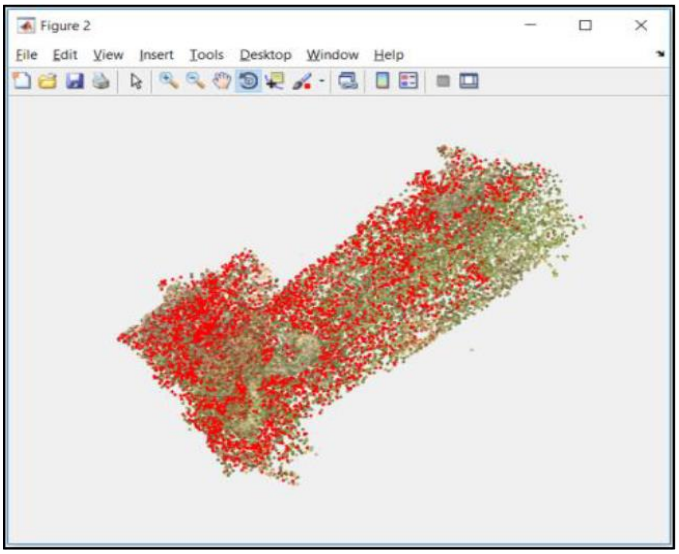

Figure 23 Another view of RGB analysis of Set-C

```
>> FypFINAL
Number of Images: 4
1--2
No. of matched points: 23893
2--3
No. of matched points: 23357
3--4
No. of matched points: 22223
  pointCloud with properties:

     Location: [22265×3 single]
        Color: [22265×3 uint8]
       Normal: []
    Intensity: []
        Count: 22265
      XLimits: [-12.3186 -4.1797]
      YLimits: [-9.7371 -5.2003]
      ZLimits: [18.3030 21.3529]

  pointCloud with properties:

     Location: [22265×3 single]
        Color: [22265×3 uint8]
       Normal: []
    Intensity: []

  pointCloud with properties:

     Location: [22265×3 single]
        Color: [22265×3 uint8]
       Normal: []
    Intensity: []
        Count: 22265
      XLimits: [-12.3186 -4.1797]
      YLimits: [-9.7371 -5.2003]
      ZLimits: [18.3030 21.3529]

Percent of proper cuts = 73.393218%
```

Figure 24 Matlab program process of Set-A

As seen from the differences between the original reconstructed 3D models and the visual representation of RGB analysis, a grass area of specific colour can be differentiated from the rest. However, the system can only fully identify between cut and uncut only if the colour model for the filter is chosen accurately based on plant species. Therefore, there is a need to create a database based on plant species and their colour matrix when it is cut and uncut.

XII. Enhancement

The frames of a video are of lower resolution than that of photos due to frame rate during video recording, but if the resolution of videos can be made equal to those of photos, videos instead of photos can be use as input instead. We designed a simple program to process the video after input to capture frames of video as images based on certain settings such as overlay required or number of images per second.

The main benefit of using video as input is that it will simplify the entire workflow of the system. Currently for a successful 3D reconstruction from an image set, the images must have a certain overlays between one another, and if that overlay setting do not result in a satisfactory 3D model, the new flight mission have to be carried out again with a new overlay setting. With video there is no need for repeated flight mission as the overlay setting can be readjusted on the video. The quality of a single image can differ from another as the camera might lose focus during the flight as well. There are also required additional hardware for Pixhawk to trigger the camera, such as specialised trigger cable that is sold only at certain stores. Video taking will only require 2 trigger; to start and to stop. Video input will also allow the centralization of keying parameters entire within Matlab itself, instead of keying in image and 3D reconstructions parameters in multiple software right now (QGroundControl and Matlab). Figure 25 shows the result of reconstructed 3D model of a watch with the video of it as the input to the Matlab program.

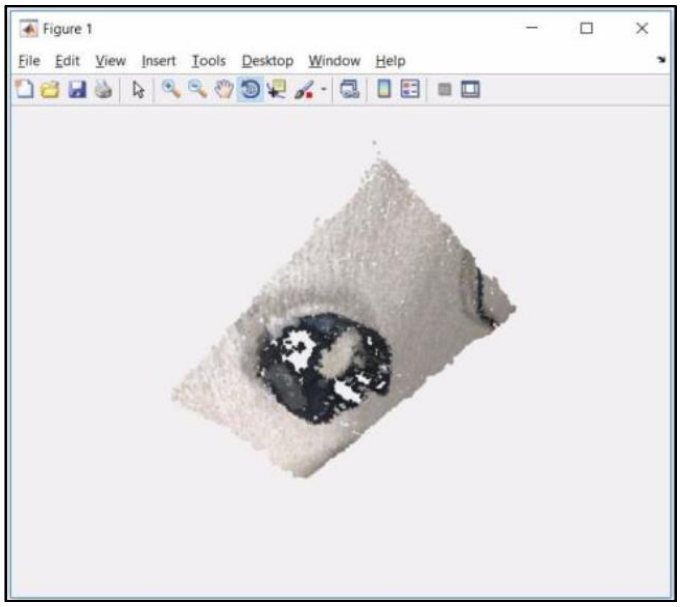

Figure 25 Reconstructed 3D model of a watch using video input

XIII. Conclusion

With the progress of the project so far, it is obvious that accuracy can be further improved. There are numerous ways to improve the processes such as using Global Positioning System as another constraint. Additional filters to exclude nearby structures can be added to prevent inaccurate findings. More experiments can be done to find a better shade of green to determine the filter range for a more accurate result based on the plant species. It can be seen that brightness in the images captured are not taken into account during this process, meaning that RGB should be converted into HSV to be tested. In the foreseeable future, a mobile application can even be made to allow ease of access to final results.

From the process and results obtained, it can be seen that 3D reconstruction through structure from motion ensures the wholeness of the targeted grass patch. As seen from the results, accuracy of the figure depends on the views of the targeted area provided and number of 3D points found through matched points in triangulation. However, it can be observed that there are cases where matching failed to occur, causing the unfilled holes or blunders in the reconstruction. Despite the failed matches, the current existing points are able to provide the average colour of a portion of the area of the entire plot which allows acquisition of the percentage of cut or uncut grass as well as a visual of maintenance results within targeted area. Hence, it is concluded that as long as there are sufficient views which are able to cover all angles, it will be possible to analyse the whole field in mention.

References

[1] C. Tomasi, T. Kanade “Detection and Tracking of Point Features” Technical Report CMU-CS-91-132 1991. Available: https://cecas.clemson.edu/~stb/klt/tomasi- kanade-techreport-1991.pdf

[2] J. Shi, C. Tomasi “Good features to track” IEEE Conference on Computer Vision and Pattern Recognition (CVPR94) 1994. Available: https://pdfs.semanticscholar.org/2eb6/74b39b2e2505663c75f92d1ccffb4311cb4b.pdf.

[3] P.H.S Torr, A. Zisserman “MLESAC: A new robust estimator with application to estimating image geometry” Oxford University 1995. Available: https://www.robots.ox.ac.uk/~vgg/publications/2000/Torr00/torr00.pdf.

[4] R. Hartley, P. Sturm “Triangulation” Computer Vision and Image Understanding Volume 68 Issue 2 1997, Pg 146-157. Available: https://users.cecs.anu.edu.au/~hartley/Papers/triangulation/triangulation.pdf.

[5] R. Hartley, A. Zisserman “Linear triangular methods” Multiple View Geometry in Computer Vision Second Edition 2004, Pg 312-321. Available: http://cvrs.whu.edu.cn/downloads/ebooks/Multiple%20View%20Geometry%20in%20Computer%20Vision%20(Second%20Edition).pdf.

[6] D. Kendal, C.E. Hauser, G.E. Garrard, S. Jellinek, K.M. Giljohann, J.L. Moore “Quantifying Plant Colour and Colour Difference as Perceived by Humans Using Digital Images” PMC 2013. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3748102/

[7] J.K. Gillan, J.W. Karl, M. Duniway, A. Elaksher “Modeling vegetation heights from high resolution stereo aerial photography: An application for broad-scale rangeland monitoring” Journal of Environmental Management 2014, pp 226-235

[8] F. Leberl, A. Irschara, T. Pock, P. Meixner, M. Gruber, S. Scholz, and A. Wiechert "Point Clouds: Lidar versus 3D Vision" Photogrammetric Engineering & Remote Sensing Volume 76 2010, Pg 1123-1134. Available: https://pure.tugraz.at/ws/portalfiles/portal/1372794

[9] S. Pravenaa, R. Menaka "A Methodical Review on Image Stitching and Video Stitching Techniques," International Journal of Applied Engineering Research Volume 11, no. 5, 2016, Pg 3442-3448.

[10] C. Baillard, H. Maitre, "3-D Reconstruction of Urban Scenes from Aerial Stereo Imagery: A Focusing Strategy," Computer Vision and Image Understanding Volume 76 no. 3, 1999, Pg 244-258.

[11] S. Zlatanova, J. Paintsil and K. Tempfli, "3D Object Reconstruction From Aerial Stereo Images" International Institute for Aerospace Survey and Earth Science (ITC). Available: https://otik.uk.zcu.cz/bitstream/11025/15944/1/zlatanova_98.pdf

[12] S. Mizoe, Y. Yaguchi, K. Takahashi, K. Ota, R. Oka "Reconstructing 3D Land Surface From a Sequence of Aerial Images," IAPR Conference on Machine Vision Applications, Nara Japan, 2011.

[13] M. Mohan, C.A. Silva, C. Klauberg, P. Jat, G. Catts, A. Cardil, A.T. Hudak, M. Dia "Individual Tree Detection from Unmanned Aerial Vehicle (UAV) Derived Canopy Height Model in an Open Canopy Mixed Conifer Forest" MDPI Forests 2017, 8, 340. doi:10.3390/f8090340.

[14] "Desert View Aerial Photography" 2013. Available: https://dvaerialphoto.com/oblique-vs-vertical-aerial-photography-infographic/

[15] P. King, B. Anstey, A. Vardy "Comparison of feature detection techniques for AUV navigation along a trained route" IEEE 10.23919/OCEANS.2013.6741023, 2014.

[16] F. Ali, S.U. Khan, M.Z. Mahmudi, R. Ullah, "A Comparison of FAST, SURF, Eigen, Harris, and MSER Features" International Journal of Computer Engineering and Information Technology Volume 8, no. 6, Pg 100-105 2016.

[17] M.O. Fril, E. Jones, M. Glavin, C. Hughes, "Comparison of Feature Detection Methods for an Automotive Camera System" ISSC 2007. Available: https://www.researchgate.net/publication/251403979_Comparison_of_Feature_Detection_Methods_for_an_Automotive_Camera_System

[18] M.A. Fischler, R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography" Communication of ACM Volume 24, no. 6 1981. Available: https://www.sri.com/sites/default/files/publications/ransac-publication.pdf

[19] "Lecture 7: RANSAC and Minimal Solvers" 2013. Available: http://www.maths.lth.se/matematiklth/personal/calle/datorseende13/notes/forelas7.pdf

[20] R. Hartley, A. Zisserman "Epipolar Geometry and the Fundamental Matrix" Multiple View Geometry in Computer Vision, Cambridge, Cambridge University Press, Pg 219-243 2004.

[21] G.H. Georgiev, V.D. Radulov "A Practical Method for Decomposition of the Essential Matrix" Applied Mathematical Sciences, Volume 8, 2014, no. 176, 2014. Available: http://www.m-hikari.com/ams/ams-2014/ams-173-176-2014/georgievAMS173-176-2014.pdf

[22] H. Stachel "Descriptive Geometry Meets Computer Vision — The Geometry Of Multiple Images" 12TH International Conference On Geometry And Graphics, 2006. Available: https://pdfs.semanticscholar.org/093b/e28210c3e41725265d0d4dcba39877d278a2.pdf