DECISION SUPPORT SYSTEM FOR TEA PLANTATIONS

K.M.C. Jayasanka (150251P)

A.S.K. Jayasena (150256K)

S.M. Kahawala (150284T)

K.V.I.H. Samaranayake (150547E)

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

December 2019

APPENDIX A: CONTRIBUTION MATRIX

This research project has four independent sections as Autonomous Navigation and Precision Terrain Following of the Drone, NDVI Camera Development, Segmentation and Evaluation of Tea Leaf Image, Map Generation and Graph Generation of the Plantation.

All the work which has been done regarding each stream is reported in separate sections and the responsible group member for each section is as follows.

Member	Contribution
K.M.C. Jayasanka (150251P)	Section 3.4
A.S.K. Jayasena (150256K)	Section 3.1
S.M. Kahawala (150284T)	Section 3.3
K.V.I.H. Samaranayake (150547E)	Section 3.2

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our supervisors Dr. Chandana Gamage and Dr. Sulochana Sooriyaarachchi for the support and guidance given for the research work, the motivation, and immense knowledge provided throughout the project. Also, we would like to express our sincere gratitude to Dr. Chathura De Silva for the advice and suggestions given during the implementation of this research as an external advisor.

Also, we would like to thank Dr. Chathura De Silva and Mr. Isuru Wijesinghe for evaluating the project by giving necessary insights about the relevant areas such as image processing and machine learning. Their feedback enlightened us to make this research a success. Also, we are much grateful to Dr. Kutila Gunasekara, Dr. Adeesha Wijesiri and Eng. Nalin Karunasignhe for the advice given during the progress meetings and discussions.

We take this opportunity to thank Dr. Charith Chithraranjan for Coordinating the CS4202 – Research and Development Project. His guidance was valuable support for us to complete the project with required requirements and within the given time frame.

Next, we would like to thank IntelliSense laboratory systems engineer Mr. Randika, Microsoft high performance laboratory systems engineer Mr. Darshana and department studio systems engineer Mr. Sanjaya for the help given to make this project a success.

Our grateful thank goes to Department of Computer Science and Engineering of the University of Moratuwa, all the academic and non-academic staff members for the immense support given in numerous ways including but not limited to provide required hardware components for this project.

Finally, we would like to thank all the people who helped us during this research in every aspect to make this research project a success.

ABSTRACT

Using modern technology to optimize traditional techniques can enhance the quantity and the quality of the agricultural products. This project is based on applying modern-day technologies to utilize available resources optimally and maximizing the harvest based on real-time data. This project highlights the possibility of using emerging drone technology combined with multispectral imaging technology to create more advanced and reliable decision supporting system which will improve the final output of tea cultivations in both quantitative and qualitative aspect.

The project consists of two major sections as, data collection platform and the data analysing platform. Data collection platform will be based on multispectral aerial image capturing. The target is to create a robust, fully autonomous data collection system at a lower production cost. Data analysing part will perform the harvest evaluation and update the traditional tea plucking tables and generate a field map to pluck the tea harvest optimally.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DEFINITION OF TERMS

UAV: Unmanned Aerial Vehicle

FPGA: Field Programmable Gate Array

HDL: Hardware description languages

BRAM: Block Random Access Memory

LUT: Look Up Table

ROM: Read Only Memory

SSD: Sum of Squared Difference

SAD: Sum of Absolute Difference

MMCM: Mixed-Mode Clock Manager

VGA: Video Graphics Array

LIDAR: Light Detection and Ranging

SNR: Signal to Noise Ratio

FOV: Field of View

GPU: Graphical Processing Unit

I2C/ IIC/ I^2C: Inter IC Communication

PID: Proportional Integral Derivative control

ROS: Robot Operating System

LQR: Linear Quadratic Regulator

MPC: Model Predictive Control

PWM: Pulse Width Modulation

NDVI: Normalized Difference Vegetation Index

VHDL: Very High-Speed Integrated Circuit Hardware Description Language

USB: Universal Serial Bus

CPU: Central Processing Unit

HLS: Hue, Lightness, Saturation colour space

HSV: Hue, Saturation, Value colour space

RGB: Red, Green, Blue colour space

ROI: Region of interest

SIFT: Scale Invariant Feature Transform

LoG: Laplacian of Gaussian

DoG: Difference of Gaussian

RANSAC: Random Sample Consensus

1 INTRODUCTION TO PICKTEA

1.1 Overview

Tea plantations are currently managed mainly through a manual process which is highly inefficient. For a country like Sri Lanka, where the labour cost is comparatively high, it is not optimal to be highly dependent on manual decision-making system for a critical export product of the country without considering real time data.

Our solution is to maximize the yield of tea by an Unmanned Aerial Vehicle (UAV) based **monitoring system** and a **decision support system** to allocate physical and human resources while suggesting the next best option that the management personnel can possibly take.

- Quality is more preferred in tea products in today's market. The demand for higher quality tea is higher than the supply.
- Automating the process by introducing machines for plucking will reduce the quality of tea therefore, it would not an option. Instead, controlling and optimizing the conditions will lead to high quality products
- Precise monitoring and analysing are the most prominent factors in quality controlling in tea plantations.
- Lower altitude UAV based data capturing integrated with a data analysing platform can use for monitoring and get real time information for decision making.
- This project will be focused on developing the fundamental components that are required for a complete decision support system.

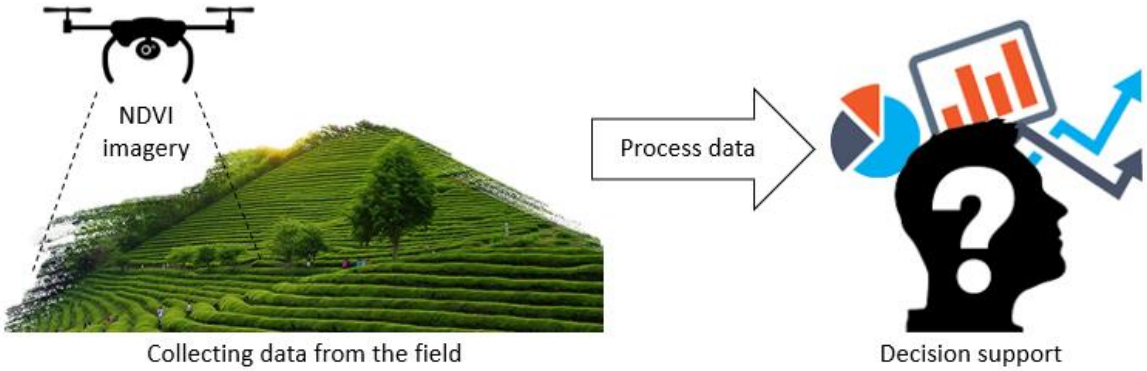


Figure 1.1 System Overview

1.2 Problem statement

Traditional plucking techniques are still used in the Tea industry. But currently, the available number of workers for the initial stage of the tea production (plucking) is inadequate to supply the demand in the world market. **Plucking at the correct time** and **managing the labour force** can be optimized with the help of modern technology. Improving the yield by deciding the best time to pluck a section of the estate and managing the labours through the best-decided sections is the major concern of this project. In this way, everyday production of tea can be estimated before plucking and allocation of resources can be done accordingly.

1.3 Motivation

Among the leading export industries in Sri Lanka, tea industry carries a significant role in export revenue generation in the country [1]. The focus of this research is to increase the quality and efficiency of the tea industry by providing updated information about the cultivation to facilitating the decision-making process.

Tea plantation management decisions include maintaining the estates, deciding and making plucking tables, maintaining the quality of the plants and leaves and managing the limited labour force within the estate.

Currently, every decision is made using the expertise and intuition of the authorised person. Since there are no facts for justification of such decisions, a lot of wastage of human and physical resources takes place during the process. By a systematic approach driven on real time data, this wastage can be minimized, and the harvest can be further optimized.

1.4 Objectives

The major objective of the project is to model a systematic approach to collect real time data from the fields and make accurate decisions to

- Optimize the yield of tea harvest according to ambient conditions.
- Develop a decision support system to utilize
    - The labour
    - Resources
- Manage plucking schedules and tables
- Cutting periods (Maintaining optimal tree heights)
- Fertilizing and pest control
- Reduce the experience requirement for maintenance and management
- Develop a prediction model to identify the age of a shoot.
- Predetermine the expected harvest from the estate with past data.

2 LITERATURE REVIEW

2.1 Introduction

Most of the decision support systems proposed by researchers are based on the soil data and climatic data to find the patterns in the yields. Khalid [2] proposes a machine learning-based approach to predict harvest based on climatic patterns and using PH level details of soil. The major intention of this research is to find out the best-suited environment conditions for the tea cultivations. Baruah et al. [3] also propose a similar system with a statistical data-based approach. The ultimate goals of these researches are to identify the most favourable conditions to get the maximum yield from tea plantations. In the literature, it is rare to find researches that have been done to optimize the resource management of the tea plantations and pre-plan the process dynamically based on the harvest. This research project is done in order to reduce the gap that leads to wastage of human effort and resources.

This research consists of two major sections. Data capturing platform and the data analysing platform which will be discussed in separate sections.

2.2 UAV obstacle avoidance and terrain following

Data capturing is convenient when it is done using UAV mounted multispectral camera systems for this kind of application. The UAVs requires highly skilled pilots to control in such an inhospitable environment like a tea estate. In order to remove the human factor from this application and introduce autonomous controlling mechanisms to the UAV, two challenges should be addressed in this specific application. The first challenge is the obstacle avoidance, and the other one is high precision terrain following.

Researchers propose different approaches to archive obstacle avoidance capabilities to their rovers. LIDAR based systems, CPU based image processing systems [4], Stereo vision-based systems and FPGA based stereo vision systems [5] are the more prominent type of solutions that can be found in the academia.

Most commonly available solution proposed by researchers for achieving high precision terrain-following ability to UAVs it to have a gimble mounted distance measuring sensor to pre-estimate the trajectory based on the moving velocity of the UAV [6].

2.3 NDVI imagery

Aerial imagery which based on different colour bands carry additional information than typical TrueColor images. There are several indices, generally known as vegetation indices that developed for efficient monitoring of cultivations [7]. Among those indices, Normalized Difference Vegetation Index (NDVI) shows a prominent relationship with the plant's chlorophyll content [8].

Implementing image processing algorithms in FPGAs have several challenges to overcome. Constraints related to limited memory bandwidth, timing and having to implement lower-level functionalities are the most common considerations [9] that a computer engineer must pay extra attention when the development is done on FPGA.

2.4 Yield evaluation

There have been many researchers conducted related to plant phenotyping which consists of research in the plant disease identification, leaf counting, leaf segmentation and measurement of plant growth. Huang [10] proposed a method for automatic identification of plant species. Manuel Grand-Brochier [11] depicted the studies to extract the tree leaves from natural images by applying various segmentation methods. Tang [12] proposed an algorithm to extract the leaf region from the images with complex background.

There are only a few works of literature available related to tea leave evaluation. Peidi and researchers [13] have proposed a variation of k-means clustering to tea bud recognition. Xiaojing and researchers [14] propose an efficient method to extract the leaf region and count the number of leaves in digital plant images.

2.5 Segmentation and graph generation

Automatic Panoramic Image Stitching Using Invariant Features [15] is a seminal paper which introduces a methodology to stitch multiple images to generate a panoramic image. SIFT [16] keypoints are used to calculate homography between images using RANSAC method robustly. Efficient Graph-Based Image Segmentation [17] is introducing a bottom-up approach for segmentation of images using graphs. Even though several edge detection algorithms exist, they are not suitable for an accurate graph representation of an image.

2.6 Summary

UAV based precision agricultural approaches enable the acquisition of real-time environmental data. Such a system can deliver a variety of IoT services and applications related to field management and resource allocation by capturing and analysing images from great heights. Real-time obstacle avoidance is essential functionality for autonomous controlling of UAVs. Stereo vision with LIDAR based mechanism to facilitate autonomous controlling is one of the main approaches that can be followed.

Imaging mechanism is an important component in a remote sensing system. Vegetation index which based on Leaf Nitrogen Content is a standard measure for the agriculture domain. Generating an image based on vegetation index involves the processing of raw pixel data and for that, a processing platform is required [18].

The final stage of the monitoring system contains the data analysing mechanism. For that, an evaluation mechanism to give a qualitative and quantitative analysis is required. For the evaluation of the captured image data, a segmentation-based approach was prominent.
Representation of the information must be performed in a human readable format. A graph representation of the vegetation and a map to include the real-world information would be a better option to consider.

3 METHODOLOGY

In this section, we propose our methodology for developing the tea plantation monitoring and decision support system. The system is consisting of two main parts. First part is data capturing and monitoring while the other part is data analysing and decision supporting.

Under the first section, a stable platform for UAV based data capturing is developed. That includes an FPGA based obstacle avoidance mechanism and a Precision terrain following mechanism. Apart from that, an imaging mechanism specialized for agricultural monitoring is also developed.

Decision supporting is obtained by processing and analysing the collected data. There the main focus is pointing towards measuring the quality of tea leaves and generate a general overview of the estate by a map representation.

Therefore, the project is divided into 4 main tasks. That is,

- Task 1: Hardware section of the Drone
- Task 2: Hardware section of the Camera
- Task 3: Evaluating a tea leaf image
- Task 4: Map generation and Segmentation

These four tasks are described separately in sections 3.1, 3.2, 3.3 and 3.4 respectively.

3.1 Hardware Part 1: Autonomous Drone

3.1.1 Introduction

This section is totally focusing on the data gathering unit of the system which is the drone. For the autonomous navigation part, two major requirements are addressed under this section. Real-time obstacle avoidance and high precision terrain following are the requirements addressed in this section.

For the obstacle avoidance, the proposed method is based on stereo vision. A depth map created using the stereo vision can be used to identify obstacles.
For the precision terrain following gimble mounted distance measuring sensor to measure the distance to the ground and based on the velocity of the UAV controlling the tilt angle of the gimble mechanism is proposed as the solution.
Obstacle avoidance and precision terrain following were done separately and hence there are two separate design and implementation sub sections available in section 3.1.

3.1.2 Literature Review

Usage of unmanned aerial vehicles for the collection of data for agriculture is more common in the present. Many complex movements and hovering over required places make multirotor the best candidate for collection of data for a decision support system for Tea cultivation. For a closer inspection of the tea plants and capturing images for the system are the main tasks of the multirotor that we are using in this project. Tea plantations in Sri Lanka are situated in uneven slopes of lands. Most of the plantations have trees and other similar obstacles that can cause problems to the low altitude autonomous flight missions.

Although there are implementations for maintaining constant altitudes using optical sensors, Open source implementation for high precision terrain following is not available. For obstacle avoidance, there are implementations done using single point detecting LIDAR sensors. They are not practical for navigation in a tea plantation because trees and other obstacles in the plantations can be easily missed by a single point detecting sensor. Implementations done using stereo cameras are available, but they require a lot of processing power.
For this project, this literature survey was done in order to find a low cost, robust, real-time and lightweight implementation for obstacle avoidance and terrain following for multirotor.

3.1.2.1 Obstacle avoidance for UAVs

Using camera images to detect obstacles is a very cost-effective approach when comparing the price of the sensor with LIDARs. But huge onboard processing power is required for the image processing. Group of researchers from the Institute for Visual Computing, Switzerland have conducted a research to implement the processing on Field Programmable Gate Array (FPGA) based system which is more cost effective and powerful solution. Their objective was to implement the system to operate at 60Hz at 640x480 image size. Although they are planning to develop on FPGA, their algorithms have been implemented on a mobile CPU with the intention of porting the algorithms to an FPGA. Power requirement for the implementation is rated at 5Watt and the payload size for the processing unit is 50gramms [19].

Another group of researchers in their publication explains their implementations of obstacle avoidance algorithm based on disparity maps, planned approach for short distance avoidance waypoints in the map space and proposed hardware design. Group of researchers from Jet Propulsion Laboratory, California Institute of Technology has used stereo vision-based map generation for the obstacle avoidance [20]. But they are not using any application specific platforms such as FPGA. In the publication, they have included sections about related work from literature and their methodology. A significant amount of work has been done in academia related to obstacle avoidance using single axis scanning LIDARs and single point scanning LIDARs [20]. But the major limitation of them is the lack of the second axis. Although using stereo vision is much better than single point scanning or single axis scanning LIDAR sensors most of the related work has been done with the assumption of having an unlimited amount of processing power on board [19].

Another group of researchers who were developing an obstacle avoidance system by directly using off the shelf stereo camera to create the disparity map and create a point cloud using the disparity map. Bumblebee stereo camera has been used to create the depth map during this research which was done at Delhi college of engineering [21]. This approach contains some issues related to the implementation due to lack of accessibility to the internal hardware level algorithms for optimizations. And also, off the shelf stereo cameras are more expensive than normal camera sensors. The benefits of this approach are, directly control algorithms and point clouds generations can be done using the processed disparity map from the camera. This also requires a significant amount of processing power because this involves a lot of image processing tasks onboard.

Researchers are using Unmanned ground vehicle equipped with a normal desktop computer loaded with OpenCV library running on C++ environment to create the real time point cloud [21]. The system is very bulky and in payload wise, it is not practical for an unmanned aerial vehicle. Implementation wise a better approach has been followed by a group of researchers from the University of Science and technology, Algeria [5]. This research is well structured, and they are using a hardware-based approach for the data processing. Using FPGA and parallelism in the hardware implementation and using custom algorithms to do the task more efficiently perform better compared to a software-based approach [5].

Almost all the publications related to Stereo matching and disparity generation is based on one single theory. It is calculating the offset of objects that are visible to the two cameras. Different researchers follow different approaches to implement this fundamental theory. The major issue is that, because we must use two cameras, even though the two cameras are having the same hardware and manufactured in a single assembly line followed by the same process, they have certain dissimilarities. The sensors have been soldered by hand or pick and place machines, therefore they have small deviations in the orientation, that makes it harder to calculate the offset of an object concerning the other camera. To overcome this issue pre-calibration of the cameras should be done. To archive this when capturing images, an image rectification sequence should be followed.

In the literature, there are implementations that have been done using Xilinx Virtex II FPGA using the Sum of Absolute Difference (SAD) algorithm [5]. The frame rate is at 30FPS with the image resolution of 270x270 pixels [22]. Another group of researchers have been able to archive 19FPS at an image pair resolution of 640x480 pixels with 80 disparity levels [23]. Researcher Han has proposed a system with 60 frames per second at a resolution of 640x480 pixels and with a maximum disparity level of 128 pixels. Another group of researchers from Canada have implemented a system with 10 frames per second at the resolution with 640x480 pixels [24] using the SAD methodology combining with the three levels of recursion developed by another research group [25]. Most of these frame rates and the resolution are determined by the capability of the processing unit. In case of an FPGA number of logic units determine these factors. The maximum clock frequency supported by the camera sensor also affects the system capabilities and the processing speed. Other related works like using GPU for the processing Yang and group were able to archive 4.8 frames per second at 512x512 pixel resolution at disparity range of 100 pixels [26]. From almost all the available literature, hardware approaches

have given the most advanced output compared with other software-based approaches. But during the implementation, lot of considerations and advanced algorithmic techniques should be used since hardware implementation is not straight forward as a software approach. Several researchers that have used FPGAs for the implementation have presented their Device utilization summary. Selecting a FPGA for this task can be done with comparing the results from those researches.

Device utilization summary from researchers from Algeria is stated in Table 3.1 [5],

Table 3.1 Device utilization summary

	Slices	**Flip flops**	**4 input LUTs**	**IOBs**	**RAMs**	**GCLK**
Available resources	63168	126336	126336	768	552	32
Initial disparity map	17274 (28%)	25695 (21%)	16433 (13%)	342 (44,5%)	144 (26%)	7 (22%)
Refinement (Median Filter)	3123 (5%)	4032 (3,2%)	3528 (2,8%)	729 (95%)		1 (3%)
Final disparity map	29226 (47%)	30735 (25%)	32336 (26%)	342 (44,5%)	385 (70%)	14 (44%)

Summary of the stereo implementation available in the literature is in Table 3.2,

Table 3.2 Stereo implementation summary in the literature

Author	**Time (frames/s)**	**Algorithm**	**Image Size**	**Window Size**	**dmax**	**et**
Miyajima et al. [23]	19	SAD	640 x 480	7x7	80	2 FPGA
Niitsuma et al. [27]	30	SAD	640 x 480	7 x 7	27	1 FPGA
Han et al. [28]	60	SAD	640 x 480	7 x 7	128	1 FPGA
Yi et al. [22]	30	SAD	270 x 270	-	34	1 FPGA
Woodill et al. [29]	30	SAD	512 x 480	7 x 7	52	1 FPGA
Hadji et al. [30]	35,6	SAD	320x240	7 x 7	25	1 FPGA

Ambrosch et al. [31]	60	SAD-CT	750 x 400	23 x23	60	1 FPGA
Khaleghi, et al. [24]	10	SAD	640 x 480	3x3	30	1 FPGA
Larabi, et al. [5]	136.8 110	DSI DSI+MF	450 x 375	7 x 7	59	1 FPGA
Perri et al. [32]	25,6	SAD	512x512	5x5	255	1 FPGA
Murphy et al [33]	150	Census. Transf.	320 x240	13x13	20	1 FPGA
Masrani et al. [34]	30	LW Phase Corr.	640 x 480	-	Dyn.	4 FPGA
Naouli et al. [35]	130	Census.Trans	640x480	7x7	64	1 FPGA
McCullagh [4]	30	SAD	640x480	7x7	48	CBE
Zhang et al. [36]	62,5	Census.Trans	640x480	-	60	1 FPGA

SAD in Table 3.2 refers to the Sum of Absolute Difference algorithm, which is the simplest algorithm. The SAD algorithm requires well rectified images because the search space is limited to an epi polar line. The epi polar line is the straight line of intersection of the epi polar plane with the image plane [37]. The epi polar plane is the plane defined by a 3D point and the optical centres [37]. In Table 3.2 d_{max} value represents the extend that we search for a matching area in the next image.

From the above publications, most popular approach was using Sum of absolute different approach with window size of 7x7. The main reason for that is the simplicity of the algorithm and the results are better compared with the processing power required. The d_{max} value depends on the image resolution and the application.

3.1.2.2 Precision terrain following for UAVs

Currently, commercial opensource low-cost terrain-following mechanisms are not available for UAVs. In the literature also there is a very limited number of concepts are available. There are lot of implemented methods to maintain a constant altitude during hovering. The same implementation cannot be used in the terrain following to the unpredictability of upcoming uneven terrain. This feature is useful for applications like precision agriculture that have been

cultivated in uneven lands. In relevant to this project, tea plantations are more commonly cultivated in uneven lands with a lot of variations. Therefore, in order to plan the upcoming flight trajectory, we must get an idea about the upcoming nature of the terrain.

Different researchers propose different methods for terrain following. Using a known elevation map of the area is proposed by researchers Samar, Rehman [38] ,Kosari, Moghsoudi [39] and Asseo [40]. The major issue of this method is that the UAV cannot react to real-time changes in the field. Researchers ([41], [42] and [43]) also propose vision-based terrain mapping procedures. This is very computationally demanding task compared with the objective. LIDAR and laser range finders have become more compact and low cost at present. They work in different environment conditions giving better results. Several researchers [44] and [45] have proposed using Spinning LIDAR systems to create a full 3D terrain map. This also requires lot of computational power as it deals with a point cloud. Using fixed LIDAR sensors will measure the altitude behind the drone when the UAV is moving forward. (Due to pitching of the drone). Researchers from Corvidae Technology proposes a gimble mechanism to fix the range finder so that the mechanism can directly face down during the complex movements of the UAV [46].

Clark and Roberts [46] proposed the method of using a camera gimble which will tilt the range finder according to the speed of the UAV. They were using the ROS platform to pass the controlling instructions to the flight controller board. The trajectory is planned at every instance depending on the reading of the range finder and the speed of the UAV. They have defined two mathematical functions to control the altitude and the tilt angle of the gimbal. Testing of the system has been done on DJI assist simulator.

AlQahtani and the researchers [47] proposed the same technique more focusing on the mathematical background rather than the implementation point of view. In their model, they create the trajectory by getting readings of the range finder at different angels. They haven't mentioned about the implementation strategy. Trajectory planning is done under two major modelling techniques. Trajectory will be derived from the rangefinder readings and then to smoothen the path Gaussian filter is applied. Then the resultant is passed through cubic spline in order to generate feasible way points. As the controlling algorithms, researchers are addressing problems related to proportional derivative integral control (PID), linear quadratic regulator (LQR) and model predictive control (MPC).

Khademi and the researchers [48] are discussing more into their mathematical model for terrain following which involves the direct transcriptional method. They have done a detailed analysis of the dynamics of the UAV and the limitations. Then the problem is modelled, and the solution is proposed with the direct transcriptional method. The simulation of their model has been done using MATLAB.

In LIDAR based methods hardware implementation is straight forward, but the mathematical modelling is complex. Most of the researchers have been limited to the simulation stage. But LIDAR based trajectory planning is more accurate, economical and compatible with payload restriction than Vision based, elevation map and spinning LIDAR methods.

3.1.3 Research Methodology - FPGA depth map generation

3.1.3.1 Design

According to the data gathered from the literature review and from the conclusion it was clearly highlighted that the FPGA based systems can outperform other approaches in similar tasks. The final goal of this section of the project is to create a low cost, reliable stereo vision based optical avoidance system. Based on the above facts, FPGA was selected as the platform for this project. To define the FPGA circuit Hardware Describing Language (HDL) approach was selected because this project involves complex arithmetic and operations. To get the abstract idea about the project, the project is divided into subcomponents as illustrated in Figure 3.1.

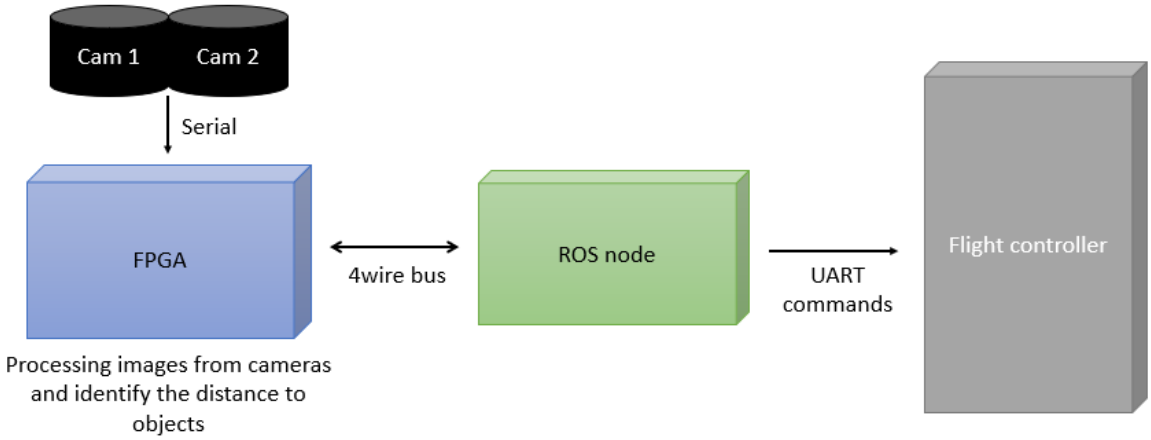


Figure 3.1 Major components of obstacle avoidance

Two identical camera modules are used to capture stereo image streams and an FPGA is used to process both the video streams to generate the depth map. Then based on the proximity of

the obstacles that can be identified by the depth map, controlling signals will be passed to a Linux based ROS node where it is directly communicating with the flight controller and it will command the flight controller to change the trajectory.

From the major breakdown of the project parts, the most important part is the FPGA based disparity generation part. From the details gathered from the literature review components and the FPGA board was selected. Selected FPGA Basys3 is consisted with Artix7 chipset which packs33,280 logic cells in 5200 slices (slice contains four 6-input LUTs and 8 flip-flops), 1800Kbits of fast block RAM,90 DSP slices, Internal clock speeds exceeding 450 MHz and 12-bit VGA output which is ideal for displaying the output. The selected camera module OV7670 carries 640x480 size digital photosensitive array which supports a maximum frame rate of 30fps.

.

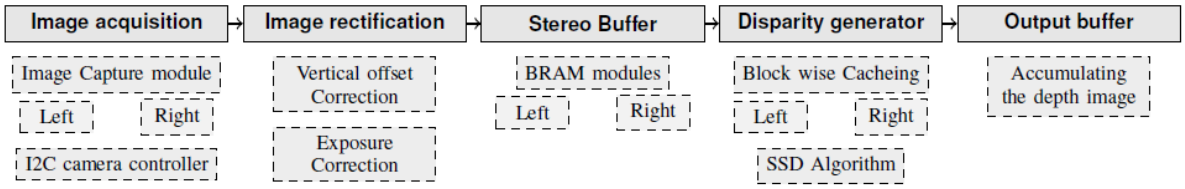


Figure 3.2 Design pipeline and sub tasks

Based on the requirements for the purpose and the characteristics of the selected hardware initial design pipeline (Figure 3.2) for the FPGA was created. From the design pipeline, the priorities were given to the components in the following order,

1. Disparity generator
2. Image acquisition
3. Buffers
4. Image rectification

Python programming language was selected for the rapid prototyping and Verilog and VHDL was selected as the hardware describing languages.

Based on the priority of modules the project was separated into 3 major parts.

1) Functional verification stage.
2) Stereo Camera integration and testing stage.
3) Disparity implementation on FPGA.

Under the functional verification stage, the most prominent design pipeline element is the disparity generator module. This requires a prototyping model using python, Simulation model using HDL and Improved and optimized final version to be run on the FPGA. For the implementation Sum of Squared Difference (SSD) algorithm was used due to its lower complexity compared with the output that was highlighted in the literature review summary.
With the Stereo camera integration and testing stage major concern is to interface the camera with the FPGA, physical mounting of two cameras, configuring cameras individually and getting the visual output from the VGA output of the FPGA.

The third part Disparity implementation on FPGA is the final integration of both the parts into an optimized system. This includes camera image rectification, buffering intermediate results, memory management, Results comparison with other's project results in literature and final optimizations.

3.1.3.2 Implementation

Implementation was started with the first stage that was proposed in the design stage, which is the functional verification stage. The initial task was to create a simulation model that supports behavioural verification of the depth map implementation. To start the project an image reading module using Verilog was created. A sample image was converted to a hexadecimal document where each line represents a value in each pixel. This was fetched using HDL and a bitmap file was created inside the simulation environment using Verilog. Then this image was saved into the computer and it could be opened using a bitmap viewer software. During this implementation, all the timing diagrams and the outputs were directly saved to the computer. The operating frequency was selected as 50MHz.

Extending the previous task then famous TSUKUKBA images were taken as the benchmark images for the rest of the implementation. TSUKUBA images are the most rectified and well calibrated stereo image pair that is used in academia as the benchmark for the stereo vision applications.

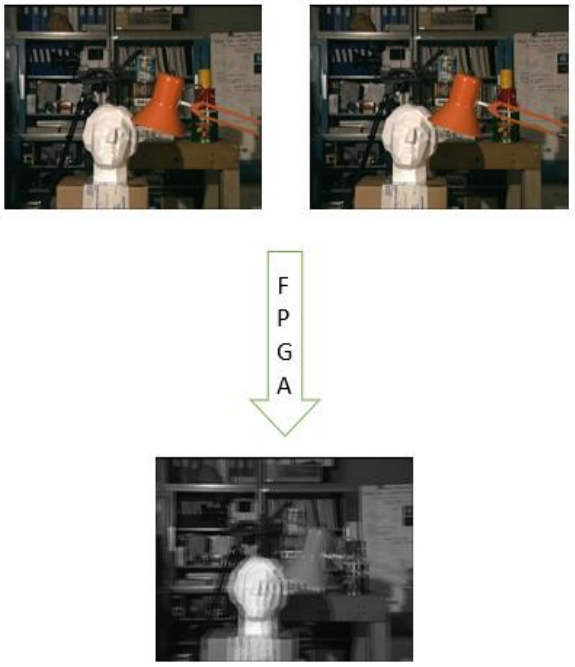


Figure 3.3 TSUKUBA image pair input and output from Verilog simulation

According to the literature review, for the disparity implementation colour is not feature. Therefore, to read both the images, initially, two images image were converted into greyscale. Then the images were converted into separate text files with hexadecimal values so that each pixel is represented as a hex value.

A simulation HDL circuit was created to read the two images and create an average image by taking the average of each corresponding pixel values in two images. For this task *Image_acquisition*, *Image_average* and *Image_write* Verilog modules were created. For the simulation to be operational without an FPGA a test bench file was also created. Figure 3.3 illustrates the TSUKUBA image pair and FPGA simulation generated Image average output. Simulation timing diagrams are shown in Figure 3.4.

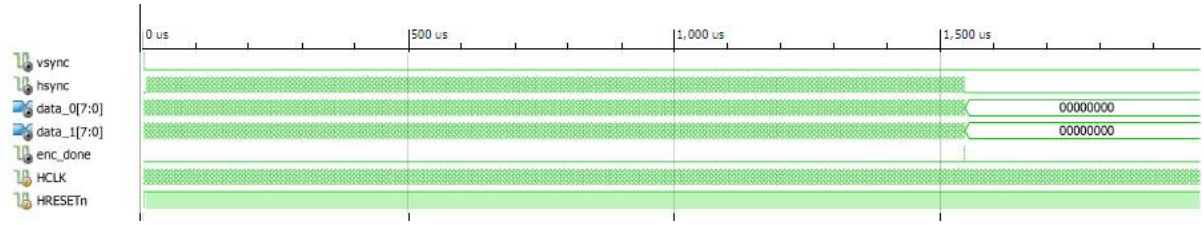


Figure 3.4 Simulation timing diagram for image reading and averaging

This simulation was done under Clock frequency of 20ns = 50MHz (Basys 3 support 100MHz physical clock max)

$$1frame/1.6mS=625\ fps \quad \text{— Eq 3.1}$$

According to the timing diagram (Figure 3.4) this operation takes less than 1.6mS for entire reading, averaging and writing operations of the images at 50MHz that means, theoretical processing speed is 625fps (Equation 3.1). The images that were used for these operations were 320x240 and each pixel value consisting 8bits.

After a successful simulation of the initial Verilog implementation, the disparity generator was taken into implementation.

3.1.3.2.1 Theory behind the Disparity calculation implementation

For the disparity implantation, the theory is straight forward, But the implementation technique can differ. If we take the birds eye view of the setup the Figure 3.5 sketch depicts how the two cameras are situated with respect to the object at (X, Y). From simple geometry, Equation 3.2 and Equation 3.3 can be derived for left and right cameras respectively. Combining Equation 3.2**Error! Reference source not found.** and Equation 3.3 we get the Equation 3.4. S implifying the Equation 3.4 gives the depth value Equation 3.5 (Y).

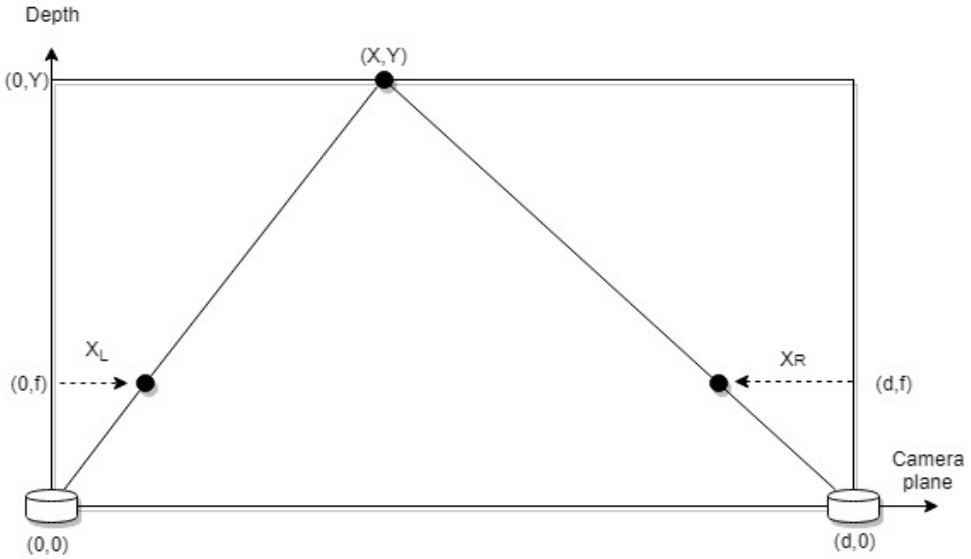


Figure 3.5 Disparity implementation setup

$$\frac{X^L}{f} = \frac{X}{Y} \quad \text{— Eq 3.2}$$

$$\frac{-X^R}{f} = \frac{d - X}{Y} \quad \text{— Eq 3.3}$$

$$\frac{X^L - X^R}{f} = \frac{X + d - X}{Y} \quad \text{— Eq 3.4}$$

$$Y = \frac{df}{X^L - X^R} \qquad \text{— Eq 3.5}$$

3.1.3.2.2 Rapid Test implementation on Python

The initial attempt of making the depth map using Verilog failed due to the inability of rapid prototyping in Verilog. Therefore, Python was used as the prototype development tool. As explained in the design section the Sum of Squared Difference was used for the similarity calculation to identify the same object in both left and right images.

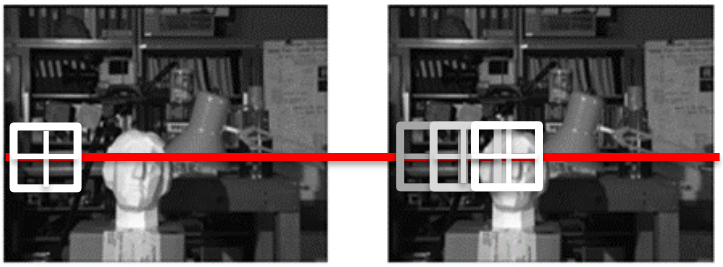


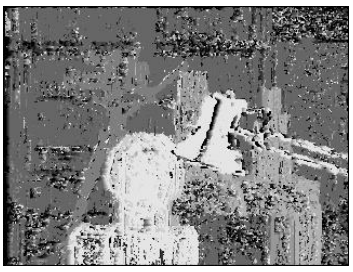
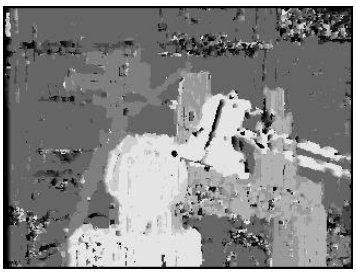
Figure 3.6 Epi-polar line (RED) and comparing window (White)

$$SSD(x, y, d) = \sum_{n=1}^{w} |L(x, y) - L(x - d, y)|^2 \qquad \text{— Eq 3.6}$$

When traversing the right image, the pixel which returns the lowest SSD value is selected as the offset value and used to calculate Y using the SSD calculation equation (Equation 3.6).

Python prototype was completely implemented using basic data structures and without any external libraries. Python prototype results with the elapsed time are given for comparison in Table 3.3.

Table 3.3 Python rapid test implementation results

Window Size	**3x3**	**5x5**
Python Results		
Elapsed time	7.67S	26.79S
Search space	10 pixels along epi polar line	10 pixels along epi polar line

Several tweaks were done to reduce the Processing time. By limiting the search space range from 0-10 by 4-10 same results were obtained within 4.85S. This offset value is depending on the cameras, the distance between the cameras and the distance to the objects from cameras. Colour heat map of the python output can be found in Figure 3.7.

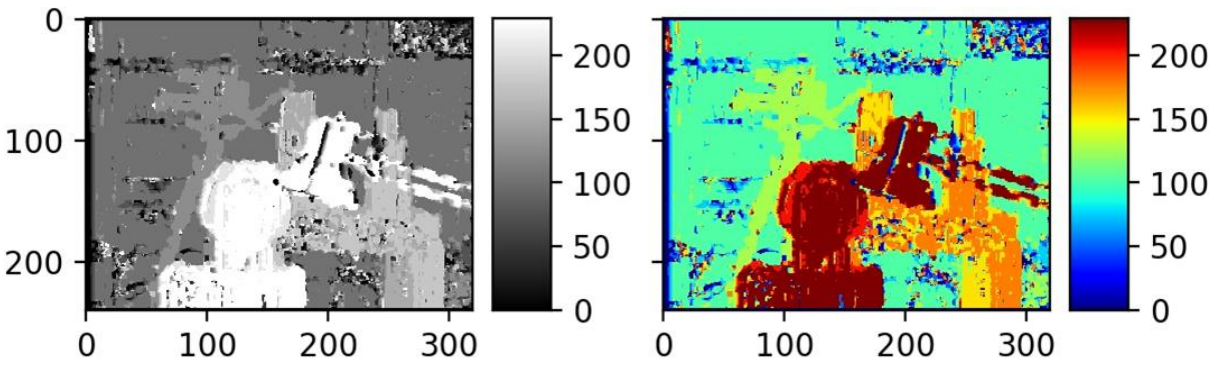


Figure 3.7 Colour interpretation of the output

3.1.3.2.3 Verilog Disparity implementation and the flow chart

Based on the Python implementation, a flow chart (Figure 3.8) was created for the hardware description implementation.

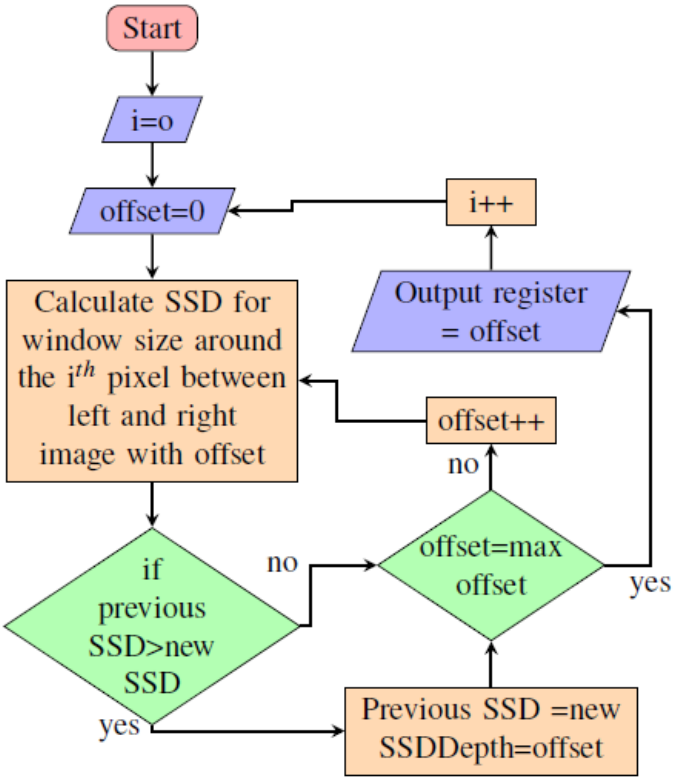
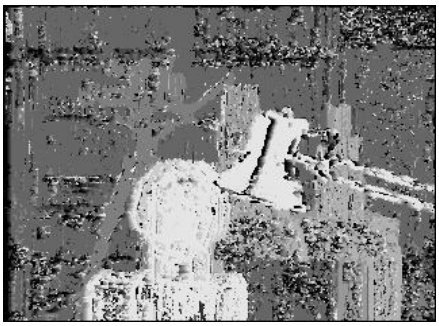
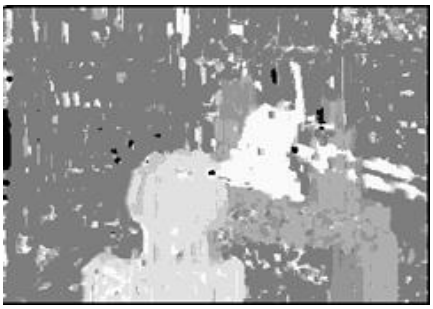
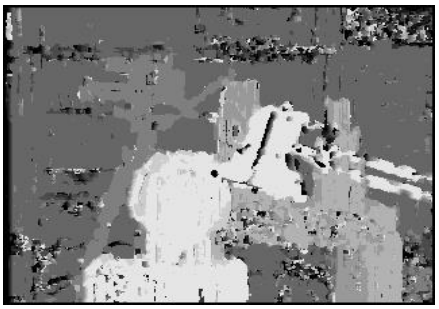
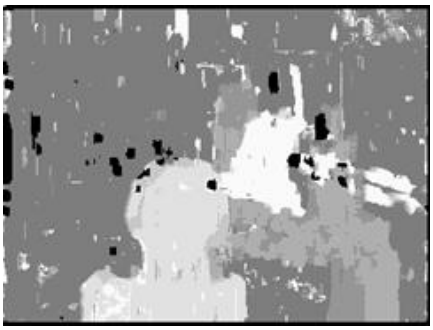
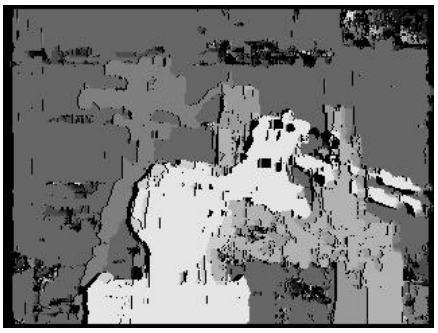


Figure 3.8 Verilog Disparity implementation flow chart

To speed up the process calculations were done to two pixels simultaneously. All the behavioural simulations were done after implementation to verify the functionality. There were issued related to clock synchronizations between modules. After significant troubleshooting, the errors were found and fixed.

Table 3.4 Verilog implementation results

Window Size	HDL Results	Python results
3x3		
5x5		
7x7		

There are minor dissimilarities in the output images that have been introduced due to the differences in bit lengths and word sizes in both the architectures. If we assume the ultimate possibility of parallelism on FPGA without considering the internal gate delays FPGAs can process outputs with different window sizes in the same amount of time theoretically. Timing diagrams for the Verilog implementation are available in Figure 3.9 and Figure 3.10.

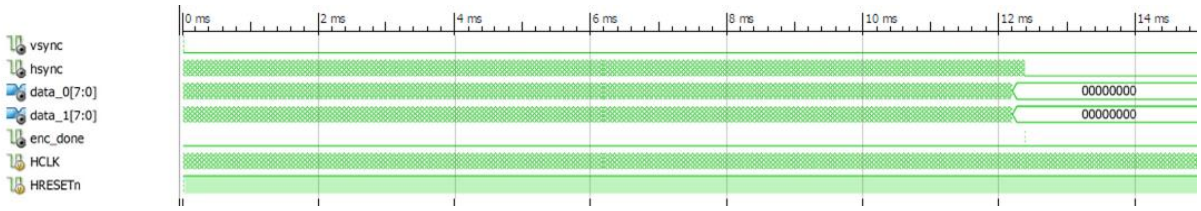


Figure 3.9 Verilog simulation results window size 5x5

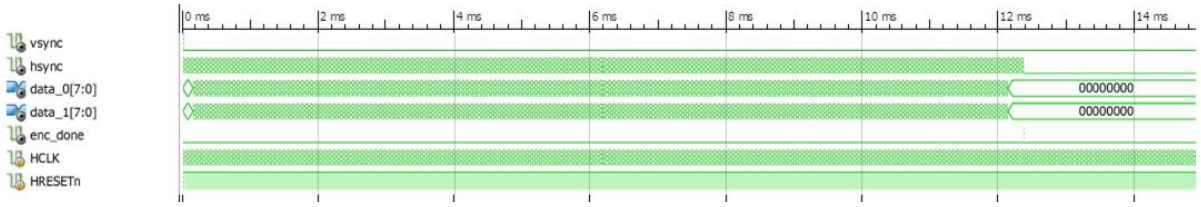


Figure 3.10 Verilog simulation results window size 7x7

$$\frac{1frame}{12.5mS} = 80\ fps\ at\ 50mHz \qquad \text{— Eq 3.7}$$

According to the simulation, it was observed that processing a single pair of images can be done under 12. 5mS.When this is converted to the number of frames that can be processed in a second it counts to 80fps at 50mHz clock frequency — Eq 3.7 Equation 3.7). After behavioural verification is completed the HDL was optimized to run on FPGA. VHDL was used for this and the flag registers were used for the loop implementation. With the completion of Depth map implementation in VHDL, successfully completed the stage one of three, functional verification and proceeded to the next, Stereo Camera integration and testing stage.

3.1.3.2.4 Stereo Camera integration and testing stage

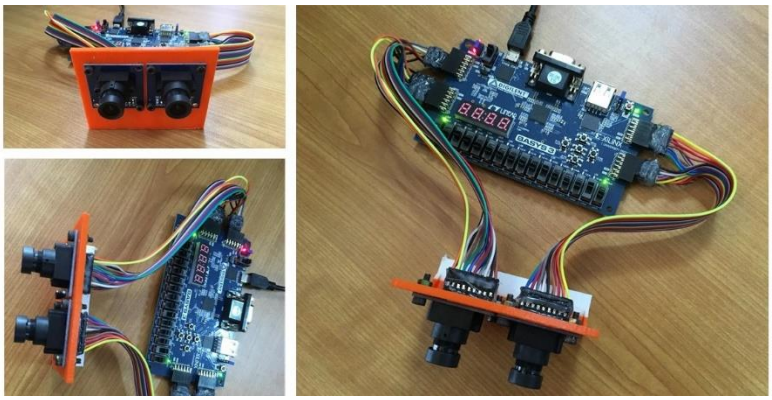


Figure 3.11 Complete prototype setup with 3D printed adjustable camera mount

After finalizing the algorithm and verifying the functionality on the HDL simulation environment the camera integration stage was started. This focuses majorly on image acquisition part of the design pipeline. After a thorough walkthrough on the OV7670 datasheet, the initial wiring diagram was created for the setup. Cameras can be configured using I2C. For that, a simple I2C emulator was created using hardware description language. The camera should be feed with an external clock and the minimum, typical and maximum supported frequencies were 10Mhz,24Mhz and 48Mhz. But in the datasheet, it was specified as the maximum fps supported by the camera as 30fps. Therefore, the 25Mhz clock was generated from the FPGA to drive the cameras.

A single camera driving module was created initially and tested the output using the VGA output of the FPGA. For the VGA output, Xilinx VGA output core was used directly. The camera was transmitting images in raw byte format using an 8bit bus in RGB (555) format. To reduce the workload on the FPGA the camera was configured to output a YCbCr image and hence Y channel is exactly same as the greyscale image it was directly used for processing. The same camera module was duplicated and two separated I2C channels were created to configure two cameras separately. Finally, a single image was created by averaging each corresponding pixel and displayed via the VGA output.

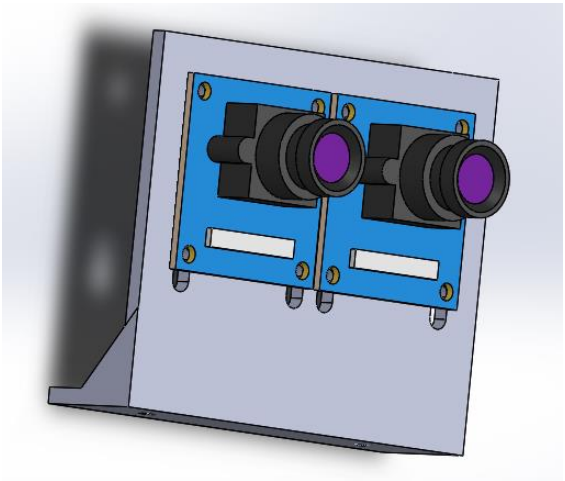


Figure 3.12 CAD model created using Solidworks

Figure 3.13 Stereo image Average output on VGA display

After HDL implementation resource utilization summary for the Image averaging on Basys 3 FPGA is available in Table 3.5.

Table 3.5 Resource utilization table for image average on FPGA

Resource	Utilization	Available	Utilization %
LUT	218	20800	1.05
FF	344	41600	0.83
BRAM	19.50	50	39.00
IO	42	106	39.62
BUFG	5	32	15.63
MMCM	1	5	20.00

With the successful completion of the Stereo Camera integration and testing stage then proceeded to the third and final stage.

3.1.3.2.5 Stage of Disparity implementation on FPGA

This stage focuses on combining stage 1 and 2 results, image rectification, image buffers in the design pipeline and further optimization of the system. The acquired images from the two cameras were stored in two separate Block RAMs (BRAMs) in FPGA. But BRAM on FPGA does not support parallel access to its content. Only sequential access is provided. Therefore, in order to system become faster, a cache was created. This cache can prefetch the data from the BRAM during other operations. The Cache was created consuming the LUTRAM. Basys 3 FPGA was having a very limited amount of LUTRAM. Therefore, it was impossible to cache a single image pair to the available LUTRAM.

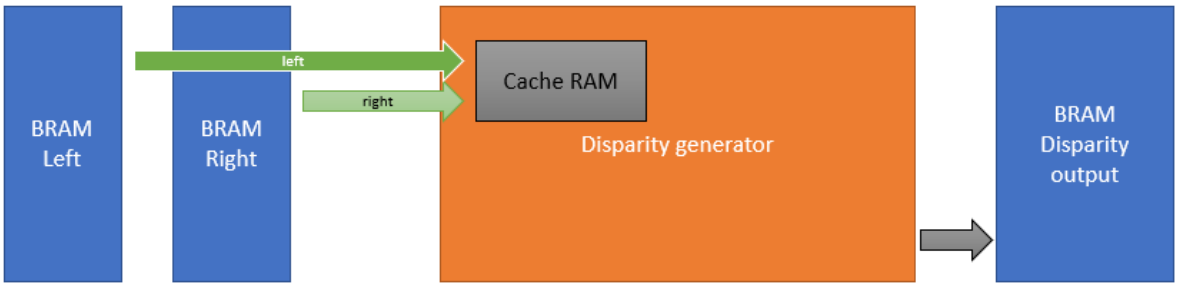


Figure 3.14 Cache RAM abstract overview

Basys 3 FPGA contains 9600 LUTRAM units which not enough even to fit 160x120 size images. Figure 3.15 summarizes the resource requirement for 160x120 size images and LUT and LUTRAM count available in the Basys 3 FPGA is not enough for generating the depth map. In order to do the processing without affecting the processing time, cache area should be reduced. This requires changing the comparator circuit for the SSD.

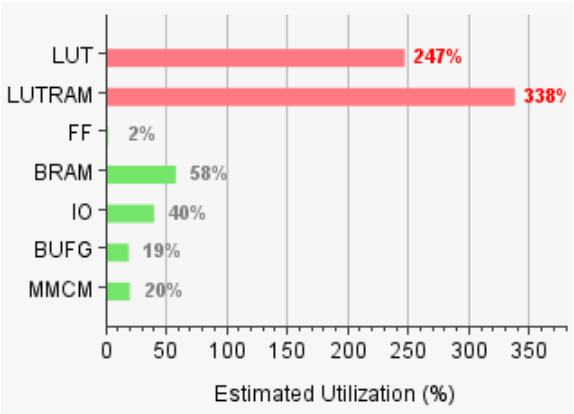


Figure 3.15 Resource requirement estimate for 160x120 size images

3.1.3.2.6 Block wise Disparity generation to reduce the usage of LUTRAM.

Caching each epi-polar line time to time increases the processing time further. Therefore, a chunk of epi-polar lines was cached at one time without caching the entire image.

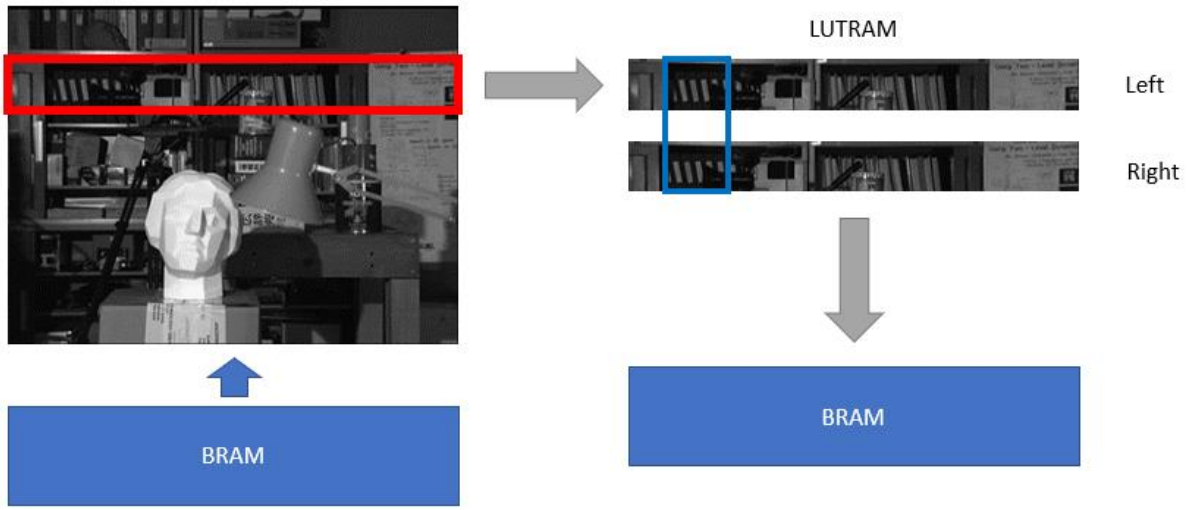


Figure 3.16 Block wise comparator circuit abstract

With the block-wise caching approach two 320x13 size image blocks are separately fetched and the comparator circuit was designed to accommodate 3x3 size window and two extra epi-polar lines. The extra lines were fetched to compensate the first and last line comparison.

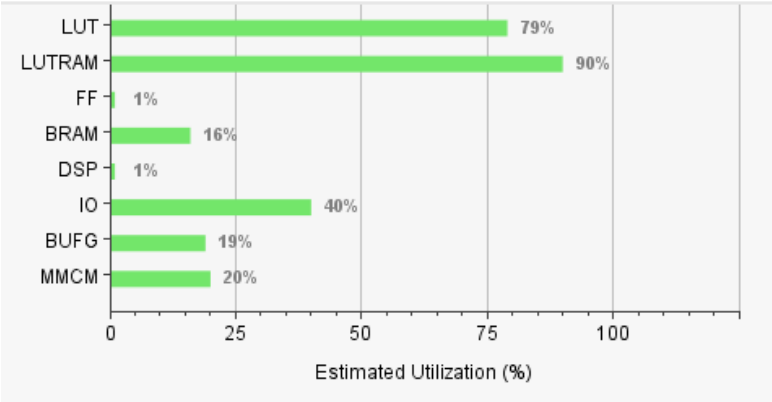


Figure 3.17 Resource utilization after block-wise caching

Although the integration on FPGA is finished there were several issues related to camera inputs and calibrations. Two cameras did not contain any sort of fisheye effect but there were alignment issues. Auto exposure settings were enabled in both camera sensors by default and there was a huge imbalance of exposure as a result of this. This affected severely on the generated disparity output.

To address these issues an image rectification module was required. Therefore, an Image rectification module was created and added in between to the left, right image buffers and disparity generator. The horizontal and vertical offsets were corrected by shifting the cache fetching addresses via image rectification module. Exposure was corrected by configuring the left and right cameras separately.

After the successful completion of the third stage, Disparity implementation on FPGA, the setup was tested in different environment conditions. The camera sensors perform well in well lighted environments, but the system fails to identify features on the objects when light is limited. This depends directly on the sensitivity of the selected camera module.

Figure 3.18 Cricket bat holding example (Left: Average image | Right: Disparity image)

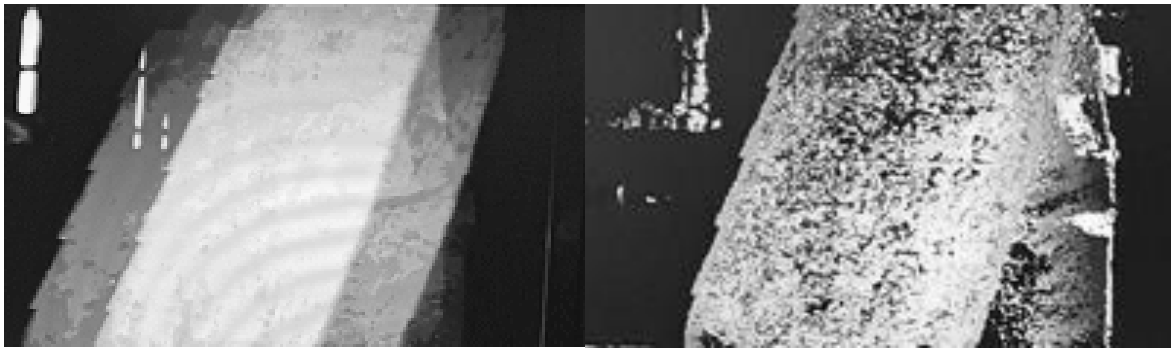
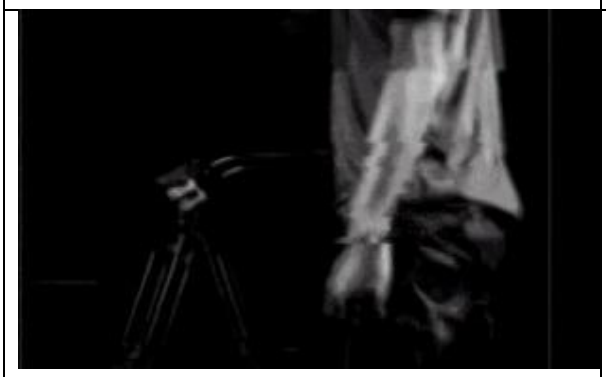
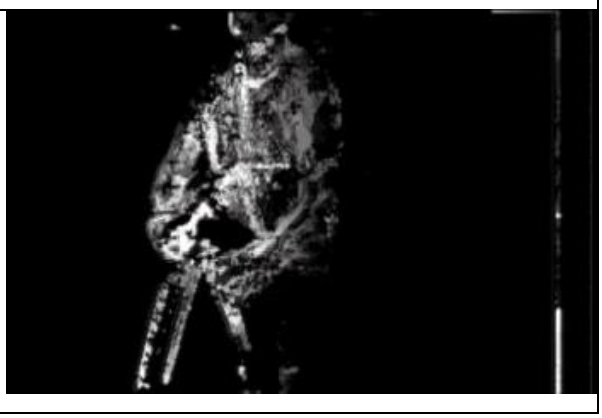


Figure 3.19 Cricket bat close-up example (Left: Average image | Right: Disparity image)

Table 3.6 An action sequence captured using the FPGA system

Average image	Disparity Image
	
	
	

3.1.3.2.7 ROS node implementation

The output disparity image is 320x240 resolution 8bit image. Altogether there are 614,400bits. To transmit this data to an external device at 25fps it requires 15,360,000bps speed communication medium. But since we are utilizing all the available PMOD connectors on the Basys 3 FPGA it is impossible to use a parallel bus to transmit the data to an external device. Therefore, serial Universal Asynchronous Receive and Transmission (UART) unit was created. UART support a variety of baud rates 115,200bps was selected for this communication. This baud rate is not enough for entire depth map transmission. Therefore, a summarized image was transmitted to the ROS node.

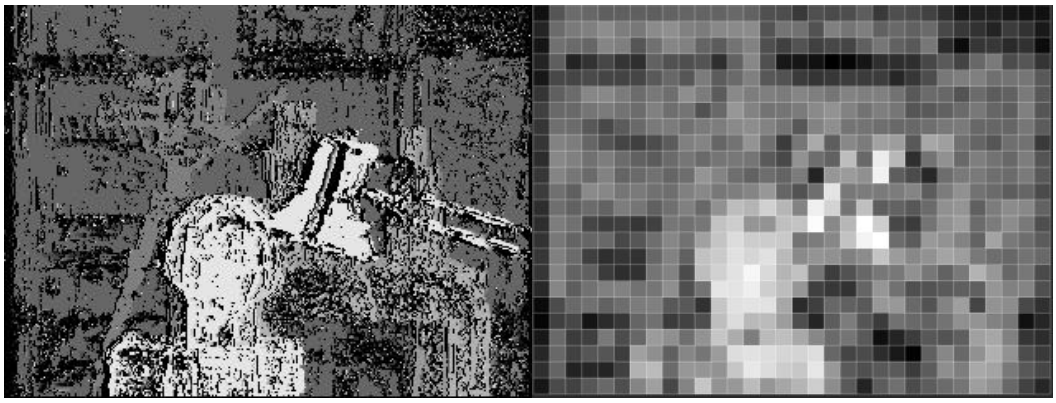


Figure 3.20 Left: 320x240 8bit disparity image | Right 32x24 8bit summarized disparity image

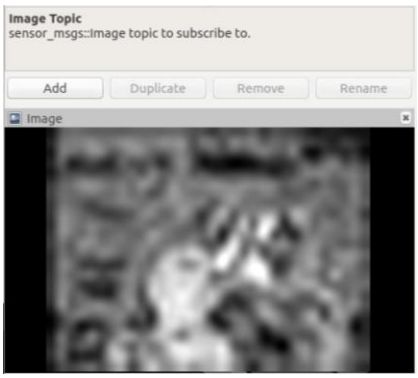


Figure 3.21Image topic RViz output received by the ROS node

The summarized image was created using the selecting max from each 10x10 pixel window. The images are transmitted as bit values and using a python script running on the RaspberyPi single board computer which runs the ROS node the 32x24 image is created back. Figure 3.21 shows the output image captured by the RVis visualizer.

3.1.3.2.8 Output results of FPGA

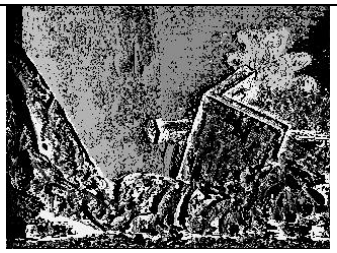
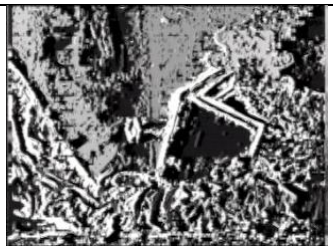
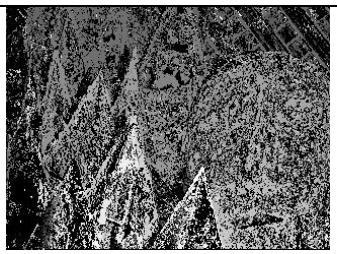
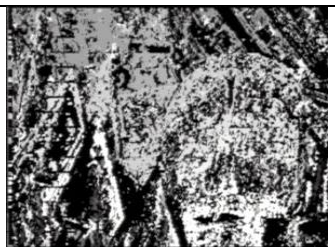
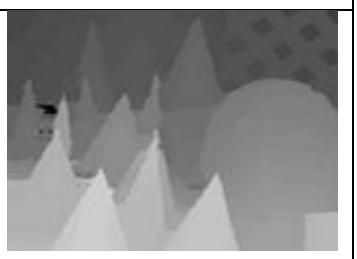
Functional verification of this project was implemented based on TSUKUBA images that is popular among the researchers. But on the FPGA it was directly ported to camera images. Therefore in order to compare the results small modification had to be done. Instead of BRAM used by the cameras to store images, two separate ROMS were created and they were programmed with testing image sets.

Information and the previews of the testing images are available in Table 3.7. Output results comparison with python output and the ground truth is available in Table 3.8.

Table 3.7 Dataset used to evaluate results and their sources

Image	Left image	Right image	Source
Tsukuba			Computer Vision Laboratory University of Tsukuba, Japan https://home.cvlab.cs.tsukuba.ac.jp
Toys			MIT Computer Science & Artificial Intelligence Lab https://people.csail.mit.edu/
Mask			MIT Computer Science & Artificial Intelligence Lab https://people.csail.mit.edu/

Table 3.8 FPGA output compared with ground truth and python output

Image	Python output	FPGA output	Ground truth
Tsukuba			
Toys			
Mask			

3.1.3.3 Summary

FPGAs are a different platform from GPUs and CPUs. Support high clock frequencies than typical microcontrollers. This project was a video and image processing project which was done on FPGA platform. Since FPGAs can be programmed specifically for the task, they are much more efficient in routine tasks. This can be clearly seen when we compare the python implementation time and FPGA implementation time (Table 3.9).

Table 3.9 CPU and FPGA time comparison

Platform	CPU	FPGA
Algorithm	SSD	SSD
Programming Language	Raw Python 3.6	Verilog
Clock speed	2.3GHz (Windows) (No background applications)	50MHz
Elapsed time	7.67s	12.5ms

This project was a great experience for understanding the complex logic circuit generation using hardware descriptions and optimizing them, implementation of standard communication protocols using digital circuits (I2C/ UART) and troubleshooting them and register level operation understanding on digital circuits.

As Register Transfer Level (RTL) engineers, It required lot of experience to predict the required resource amount for a certain application, But with a detailed literature survey it is possible to create a rough estimation about the project where in this project FPGA board selection was done based on literature survey.

3.1.4 Research Methodology – Precision Terrain Following

3.1.4.1 Motivation

Major problem with fixed distance measuring sensors used for the height estimation is, when there is motion due to the tilt created by the UAV, it creates false reading. If these readings are used for the height estimation during that occasion it creates huge mismatches with other sensor data during the sensor fusion. To avoid this problem typically distance measuring sensors are disabled after a certain tilt angle. Therefore, although there are open source implementations for the precision low height position holding there are no similar implementations for precision low height terrain following. If we disable the tilt angle limitation in existing system, the trajectory of the flight cannot maintain the even distance with the ground and eventually error will accumulate when climbing up hill and there is a possibility of crashing the UAV.

3.1.4.2 Design

According to the literature survey there are several approaches followed by different researches. For this application a more commonly used gimbal mounted distance sensor approach will be followed.

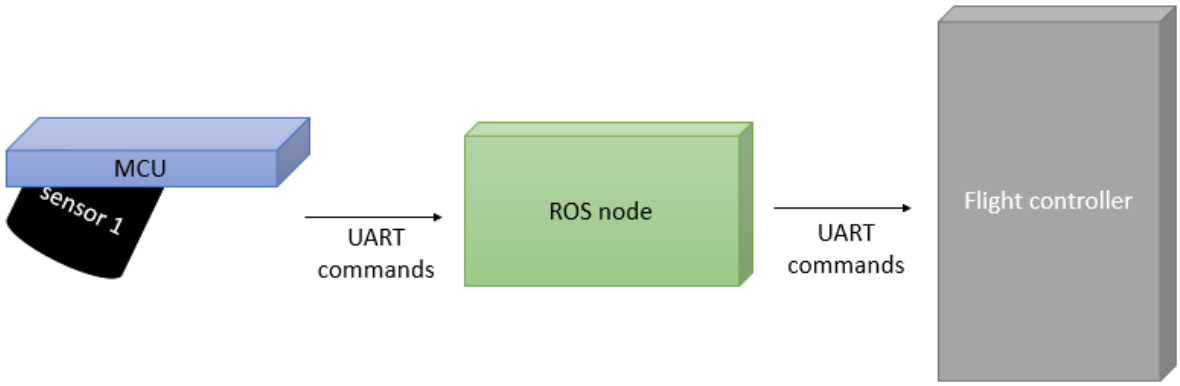


Figure 3.22 Major components of precision terrain following

Sensor 1 in the Figure 3.22 is the LIDAR sensor which will be mounted on a gimble system. Microcontroller (MCU) is responsible for reading the range finder value and controlling the gimble mechanism. The data will be passed to the flight controller via the same ROS node used for the obstacle avoidance. The flight trajectory calculation is done based on LIDAR sensor data. The gimble will control the tilt angle of the sensor depending on the velocity of the drone that is communicated to micro controller via the flight controller.

The components for this project were selected based on the sampling frequency required for a UAV that is having a flight controller processor working at 168MHz and can travel at a limited velocity of $25Ms^{-1}$. Based on the data gathered from available products that support precision position holding, the distance measuring sensor and the micro controllers were selected. TF02 LIDAR is a centimetre level accurate distance measuring sensor which is capable of sampling the readings at 100Hz-500Hz. Its capable of measuring distances in the range of 0.4m to 22m. Operating voltage is 5V with the peak demand for the current is 3A. For the micro controller 32bit ARM STM32F1 board was selected. Since it is 32bit it is directly compatible with most of the open source flight controller platforms.

There are three major sections in precision terrain following part of the project.

1) Custom gimble controller and hardware integration
2) ROS integration with the flight controller

3.1.4.3 Implementation

The implementation was started from the gimble controller, for the prototype system in order to compensate the pitch and roll of the UAV two actuators are required, for that purpose two 9g analog servo motors and to measure the tilt angle of the UAV GY-271 gyroscope and

accelerometer two in one module were used. GY-271 module uses I2C protocol for the communication and configuration.

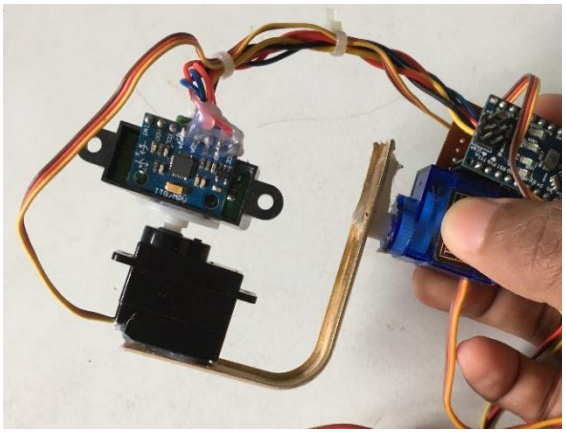


Figure 3.23 Prototype hardware unit in levelled situation

Microcontroller was programmed to read the tilt angel measurement from the gyroscope and control the actuators using Pulse width modulation signals (PWM). After setting up the hardware, during the first test it was observed that controlling requires intermediate PID controller to smoothen the motion. The microcontroller was adjusted to dynamically tilt with the changing roll and pitch velocity. These parameters should be tuned correctly relative to the application hardware.

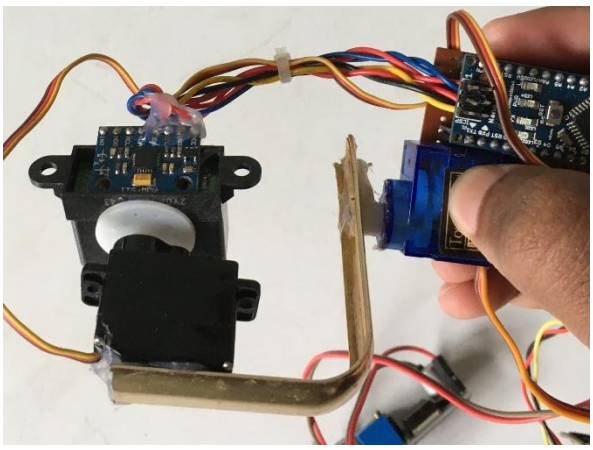


Figure 3.24 Reaction of the system to a roll orientation change

ROS integration was straight forward in this application by using the same ROS environment that was used with the depth map to publish data to a topic where then this data is communicated to the Flight controller.

3.1.4.4 Summary

Actuators that were used for this project have a larger responding time where the simulation environment requires more faster response to the environment. The distance measuring sensor used for the prototype was sharp short-range distance measuring sensor which was having lower signal to noise ratio and the sampling interval of this sensor was considerably low for the requirement. Therefore, the prototype should be improved with better sensors for the real application.

3.2 Hardware Part 2: NDVI Camera Module

3.2.1 Introduction

3.2.1.1 Background

This section is focusing on capturing the plantation field and generating the "Normalized Difference Vegetation Index" (NDVI) for the captured area of vegetation. In precision agriculture, NDVI can be used to differentiate plants and soil in a cultivated area. Mainly because there is a significant linear relationship between NDVI image and leaf chlorophyll content as well as leaf nitrogen content. The calculation is based on the vegetation reflectance at the visible colour band and near-infrared band.

NDVI is one of the main metrics that can be used for analysing the quality of vegetation. In most remote sensing applications, NDVI images are widely used. In this project, usage of lower altitude NDVI images for vegetation quality assurance is elaborated and tested the results. To integrate the entire system into a single platform, development should be carried out in an FPGA platform. For the proof of concept stage, a generic FPGA development board is used.
Implementing image processing algorithms in FPGAs require specific data structures and sometimes caching mechanisms. Also, there are some constraints associated with FPGA based programs and those constraints depend on the application requirements.
This section will describe the development of the platform for providing standardized information for the final decision support system to come up with better decisions.

3.2.1.2 Motivation

The main drawback related to satellite-based multispectral imagery is the very higher cost of acquiring such images. One alternative would be getting a multispectral camera that directly generates NDVI or similar vegetation indexed image. As undergraduates, we had thought of delivering a complete system in a single embedded system for aerial monitoring of tea cultivations is Sri Lanka. Because of that, the idea of implementing a low-cost NDVI imaging system on an FPGA was developed.

3.2.2 Literature Review

Modern-day agriculture is more focussed on quality enhancement using precise control mechanisms. Many kinds of research have been conducted in order to achieve precise controlling of conditions and reduce the cost and impact of external factors [49]. In precision

farming, various types of information from heterogeneous sources are combined to increase the total input-to-output ratio of cultivation [50]. Measuring the nitrogen content and the chlorophyll content involves determining the amount of biomass and usage of visual footages to measure those factors that can also support in decision making based on the intensity of vegetation cover of the area [51]. Several index ratios of some colour channels of an image had reflected information about the nitrogen content of the plant and among them, the Normalized Difference Vegetation Index (NDVI) shown a greater correlation with chlorophyll content [8]. Although NDVI is widely used in remote sensing using satellite imagery, it is also potentially feasible to use NDVI images captured in lower altitudes as an overall summarization of the cultivation. Another advantage of using NDVI is that it omits the illusions caused due to various lighting conditions. That is, NDVI operation is based on absorption ratios of red band and near-infrared band by chlorophyll molecules [49]. Therefore, it is more effective than measuring the surface reflection of the visible band of light.

Capturing NDVI images is generally performed in satellite scale altitudes and the insufficiency of satellite-based remote sensing for precision agriculture is attributed to the low accessibility of high-resolution imagery. Typical multispectral images that were taken at satellite altitudes are very expensive and cannot afford a general use case. Also, satellite images may have interferences due to the cloud cover. UAV borne imagery using off-the-shelf cameras can more appropriately address many of the imaging needs required for precise monitoring, low area landholding, and variable planting cycles scenarios in agriculture. The ability to precisely target locations and times for mapping allows us to collect data even in irregular climate patterns. Using such aerial images, several vegetation indices can be calculated, Visible Atmospheric Resistant Index (VARI), normalized excess green index (ExG), normalized green-red difference index (NGRDI) and Triangular Greenness Index(TGI) are purely based on true colour (RGB) images while Normalized Difference Vegetation Index (NDVI) is based on both RGB and Near Infrared images [7].

In a typical NDVI camera, two aligned charge coupled device (CCD) chips being used in order to capture the visual band and near-infrared band. However, the cost of such a device is much higher due to its requirement of precise optical alignment. As alternatives for this expensive NDVI camera, Rabatel *et al.* developed a concept in which by using a single standard camera, getting an NDVI image as the output. The methodical approach is based on capturing simultanious images from both red colour band and near-infrared band. Modification of

bandpass filters of the standard camera was the implementation strategy they have followed [52]. With that, the bottleneck due to the optical path alignment has been completely eliminated.

Another approach to address the NDVI image acquisition is by using a multispectral camera and a proper calibration mechanism. There are even commercialized products that follow this concept, as well [53].

Since NDVI calculation is a point operation, measurement of crop growth status is still a bottleneck when it needs to achieve Realtime performance in precision agriculture. In order to achieve this, a hardware-accelerated embedded processing unit can be used. Field Programmable Gate Arrays are very efficient in parallel execution of instructions, and hence most image processing algorithms can get a considerable advantage when executing an FPGA [54]. FPGA's performance gain is obtained by bypassing the fetch-decode and execute overhead of regular general-purpose processors and by getting the full usage of parallelism support of the hardware design [9].

Therefore, FPGA based multispectral camera system for NDVI acquisition would give better results in evaluating and analysing the crop growth status. Zhang Ze et al. developed a multispectral camera with two CMOS sensors to address the issue of measuring the growth status of a plantation. They have identified that the usage of discrete points to represent a small area in a plantation zone is problematic because even at two adjacent points, the growth status may be widely different. Instead of using discrete information, the proposed methodology is processing the continuous data to get precise information about the vegetation status. They used a Nios II processor and developed a VHDL based driver module to control 2 CMOS sensors and then process the captured data in the Nios II processor and stored in a CompactFlashTM Card. The non-compressed Bitmap image format is used throughout the development, and the resultant image is also stored in a bitmap format. The main operation and data flow between the image sensor and the controller are controlled via three synchronization signals, clock, horizontal synchronization signal, and the vertical synchronization signal [18].

Even though FPGAs provide spacial and temporal parellelism, there are certain constraints and general guidelines that need to adhere. Some of those constraints can force the developer to reformulate the image processing algorithms. Classifying the flow of operation and then

addressing the constraints is one way of implementing image processing algorithms in FPGAs. The processing model can be either stream, offline, or hybrid mode. In each process model, different constraints affect significantly.

In stream processing, memory bandwidth creates a massive burden because data has to be processed as arrived. On the other hand, in offline processing, the memory access speed is limiting the speed of execution of the algorithm. Hybrid mode is a mixture of the above two approaches, and in that, there is the luxury of adjusting the fraction of timing and memory constraints as preferred [9]. Another constraint that occurs when implementing image processing algorithms in FPGA is that most frequently used hardware description languages(HDL) only supports raw characters such as ASCII. Because of that, the direct manipulation of images in standard formats(jpeg, bmp,..) is not straightforward. For that, a separate mechanism to translate the standard representation to the binary information with ASCII characters in the hexadecimal format has to be implemented. After doing the conversion, the binary file can be applied as a vector to the operating HDL modules [55].

3.2.3 Research Methodology

3.2.3.1 Design

The design was carried out under formal methodology. Work has been started after setting up the domain as Sri Lankan tea plantations and modern imaging technologies. There we did a thorough review of many works of literature focussing on various subdomains. Under NDVI camera development, there were many kinds of research that had been conducted focussing on the conceptual background behind NDVI and precision agriculture, whereas some of those regarding the agricultural aspect and some regarding the technical aspect. After a careful analysis of the literature, we were able to come up with a good understanding of the existing gaps present in our area of focus. After selecting relevant papers, we divided existing literature into four main areas and then explored and summarized the approaches and get an understanding of those areas.

The subsections include,

1. Usage of NDVI in vegetation monitoring
2. Strategies to calculate and process NDVI images
3. Perform image processing in FPGAs
4. Interfacing the camera into the FPGA

We referred to research papers on the aspects mentioned earlier and identified relevant literature which has similar approaches. On the other hand, we were aware of the limitations that researchers have mentioned in their papers as well.

With that understanding, the project could be placed between the third and second areas we discussed as subsections under our area of focus. That is, this NDVI camera development includes performing image processing tasks within an FPGA while implementing a mechanism for the efficient calculation of NDVI. Throughout the development, a significant effort had to put in order to implement an interface to transfer data to the FPGA. Therefore, the interfacing part is also added as a section of the primary design.

Apart from that, since the project is more into an empirical approach, support functionalities also have an important impact towards the project. Camera interfacing can be taken under that category as well because there needs to have a robust interfacing mechanism in order to carry out the development. Standard communication must be considered because the project is initially developed as a proof of concept, and later, it should be deployed as a commercialized product.

According to the literature discussing the image processing implementation in FPGAs and interfacing mechanisms, the high-level architecture of the system was designed, as shown in Figure 3.24. The functional description of each internal module of FPGA implementation is discussed in Section 3.2.3.3.

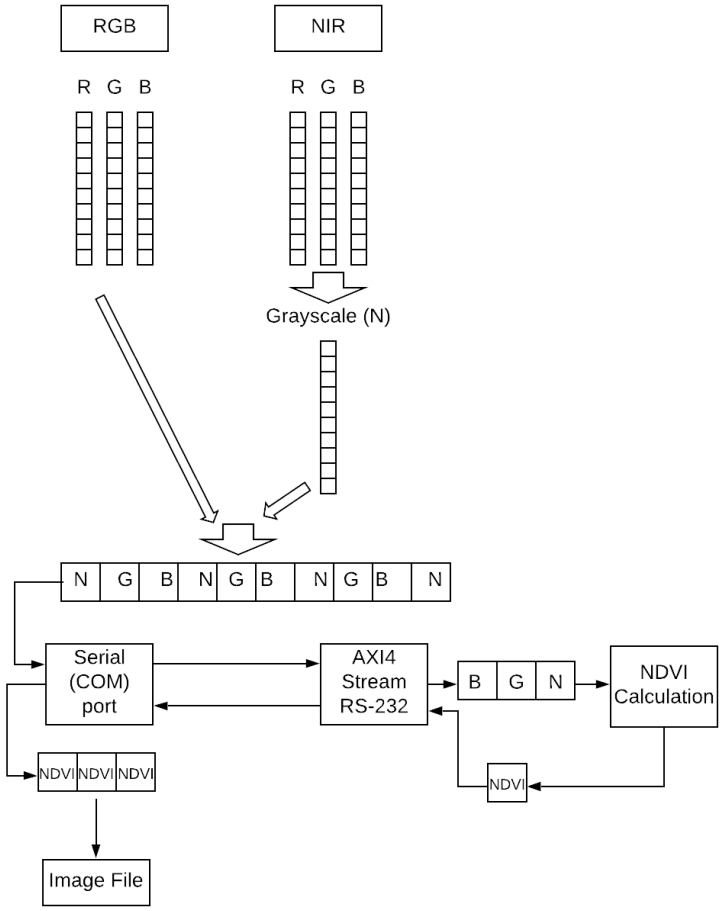


Figure 3.25 System Design of FPGA-NDVI device

From the information available in the literature and the developer community, components and the FPGA development board were selected. Selected Basys3 development board is consisted with Artix7 FPGA chip which packs33,280 logic cells in 5200 slices (slice contains four 6-input LUTs and 8 flip-flops), 1800Kbits of fast block RAM, Internal clock speeds exceeding 450 MHz and a Digilent USB-JTAG port for bitstream programming and communication. BASYS3 board also consists of a USB-UART bridge which can be used as a serial communication interface. The selected camera module, AR0230 RGB and IR Dual output camera carry 1920X1080 size digital photosensitive array, and the camera supports the USB2.0 High-Speed interface. The module can capture images with a minimum illumination of 0 lux by the IR sensing element. However, for this application, the low light operating capability is not required because the system will be deployed in an outdoor environment. Also, the Signal

to Noise Ratio of the sensor is 41dB concerning the optical power, which is an acceptable level of noise for a general image processing application.

3.2.3.2 NDVI – Normalized Difference Vegetation Index

Normalized Difference Vegetation Index (NDVI) image carries a relatively large portion of information than regular RGB image. Therefore, in order to support the decision-making process in tea cultivation, NDVI information about the explored area will be captured.
Two cameras which sensitive to two different illumination bands used for raw data capturing. A simple RGB image sensor and Near Infrared (NIR) sensor are used and with that, both visible and NIR bands can be acquired simultaneously. After that, based on those footages, the NDVI image can be constructed.

There are different approaches to calculate NDVI values. Each approach has different characteristics, and the selection should be made based on the application conditions. For example, the cases where we need to differentiate land and vegetation in an image taken from a higher altitude and, to differentiate variations in vegetation have different aspects being highlighted. Extra attention must be pay in aligning multispectral images because NDVI calculation is a point operation.

In most cases, to calculate the NDVI, the following equation is used. Here the difference between the visible band and near-infrared band is considered.

$$NDVI = \frac{NIR - VIS}{NIR + VIS} \qquad \text{— Eq 3.8}$$

$$VIS = \frac{(Blue + Green)^2}{(Blue - Green)^2} \qquad \text{— Eq 3.9}$$

Since NDVI shows a significant difference in reflection between the red channel and near-infrared channel, we can also use the following equation to calculate the NDVI value.

$$NDVI = \frac{RED - NIR}{RED + NIR} \qquad \text{— Eq 3.10}$$

To cater the final integration requirement, FPGA based image construction approach is proposed. Since the cameras having standard USB interface for communication, a separate entity is required for protocol translation.

AXI4 Stream Protocol

AXI is first introduced as a part of ARM AMBA (Advanced Microcontroller Bus Architecture) 3.0, which is a family of microcontroller busses. AXI4 is included in AMBA version 4.0, which is released in 2010.

In streaming devices, the main objective is to provide a steady flow of high-speed data. Therefore, one new block of data is transferred in every clock cycle. Apart from that, to reduce the overhead, streaming busses do not have addressing, which means streaming connections are point to point. AXI4-Stream follows master-slave communication channel with unidirectional flow of data.

Xilinx has adopted the AXI4-Stream protocol for their FPGA devices (beginning with Spartan-6 and Virtex-6), which enables configurable multiple masters to multiple slaves (up to 16 X 16) capable cross point switch. Some other key features and benefits of AXI4-Stream are,

- Synchronous and asynchronous clock rate conversion
- Support for multiple clock domains
- Arbitrary data byte width conversion

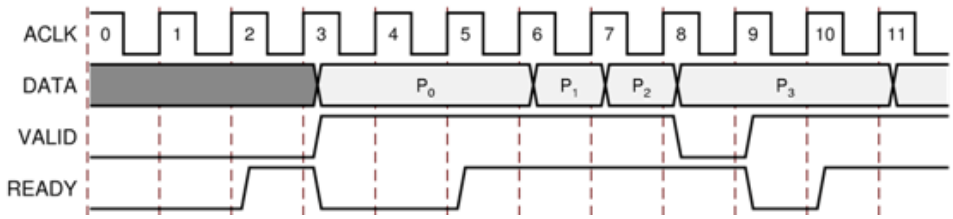


Figure 3.26 Ready/Valid handshake, start of a new frame

The transmission is started after the master(sender) sends the TVALID signal and the slave(receiver) responds with TREADY signal. At the last byte of the stream, the sender will raise the TLAST flag. In between TREADY and TLAST signals, TDATA width of bits is transferred per clock cycle. A new data can appear on the bus only when TVALID and TREADY were both asserted in the previous clock cycle.

RS-232 Protocol

RS232 is a serial information transfer protocol standard that details how a stream of data bits is sequentially transmitted into a wire. The protocol defines the order and meaning of each bit. The transmitted byte is not synchronized to the receiver. Which makes RS232 is an

asynchronous protocol. For this reason, the interface at each end of the communication link must be set up at the same rate so that each serial decoder chip can decode the serial data stream (Baud Rate).

The AXI4-Stream IP core has an RS232 module as a built-in component. With the built-in RS232 module, a serial communication interface can be implemented in FPGA's end. Then the FPGA can communicate serially with any device which supports RS232 protocol via the USB port.

As a support component to the system, an external device needs to be configured to convert the captured image data into raw bytes and transmit to FPGA as serial data (UART, SPI). Besides, the same device can be used if there is a buffering requirement that occurs in the development phase.

In the further development stage, functionality can be extended to have real-time NDVI output. To do that, a stable mechanism should be implemented for static image processing, and then with minimal modifications to the internal modules; we can obtain real-time stream processing capability. Since an external module is used to convert standard media format into raw bytes, extending the functionality to stream processing needs a different conversion mechanism. For cater the stream processing requirement, an interface with parallel data links would be preferred.

3.2.3.3 Implementation

FPGA design contains a hierarchical structure. The topmost level is responsible for handling inputs and outputs to the real world and integrate all the secondary level components. In the developed system, the top module maintains buffers for incoming serial data and forward received data towards relevant components via internal signals.

Clock domains are defined in the space of the top module and for that, a clock divider component has been instantiated. Xilinx IP catalogue contains standard clock divider circuits, and for this project, the standard 100MHz clock is used for the AXI4-Stream component, and 50MHz clock is used for the core calculation modules.

The core functionality is implemented in the NDVI calculation module. The module takes three colour channels (NIR, Blue, Green) as 8-bit vector inputs and returns corresponding NDVI value as an 8-bit vector. Since arithmetic operations to calculate the NDVI value are implemented as a state machine, a transmission done signal is used to notify the top module to initiate a data sending step.

In addition to those functional components, an accumulation module is implemented to store calculated NDVI values. For that, a block memory component is instantiated, which is having an 8-bit wide word size and 76800 (320 X 240) addresses. This accumulation module can be used as a source to output the complete image. Because general-purpose FPGAs do not have a memory access mechanism and therefore, a caching strategy needs to be implanted. Lookup tables would be a suitable methodology to consider. As LUTs are dynamically updated during the execution, a constant memory access time can be maintained throughout the system. Data transmission between the modules and the incoming bus width would be 8bit wide.

The data transferring technique is another primary consideration of the system. Since there are multiple types of devices work together in this system, implementing support for standard communication interface would be a good option to consider. In order to implement a communication interface, the IP catalogue of Artix-7 was referred and supported protocols for BASYS 3 were studied. From the available options, the AXI4-Stream IP core is selected to implement the UART communication channel towards the FPGA. The interfacing module is directly connected to the primary input and output signals of the system. Apart from that, internal data buffers used to feed data to the interface and receive incoming data to the system.

3.2.3.3.1 **Functional Verification - NDVI Image Generation**

Initial validation of the NDVI calculation has implemented in python, and the following results had been obtained. For each technique, the same source images have been used, and each one is compared with an image captured from a standard NDVI specialized camera.

Figure 3.27(a) – RGB source image

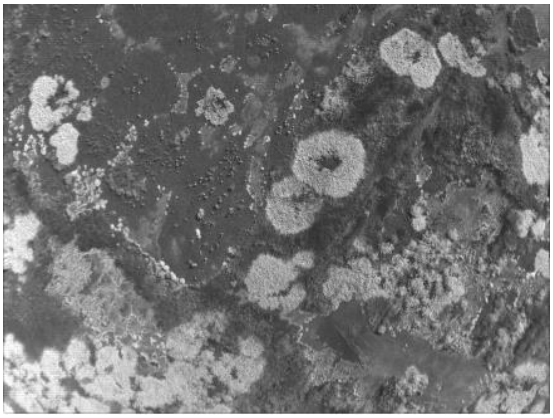


Figure 3.27(b) – NIR source image

Figure 3.27 Source Images

Figure 3.29 contain the results of different NDVI calculation approaches and enhancement techniques.

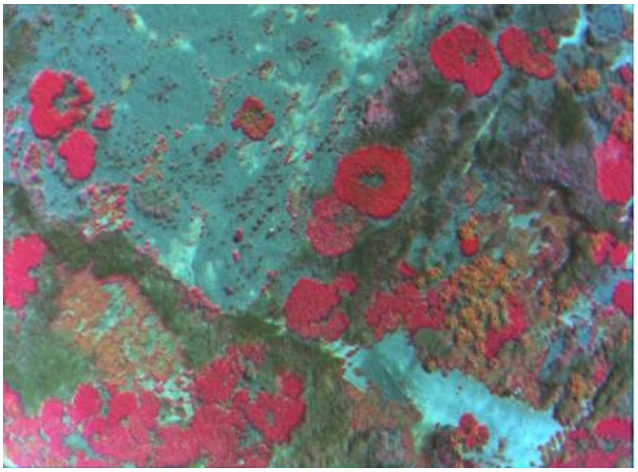


Figure 3.28 Image Captured from NDVI Specialized Camera

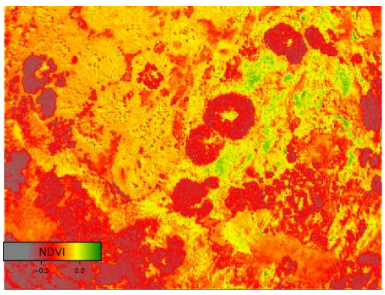


Figure 3.29(a) Normalized image

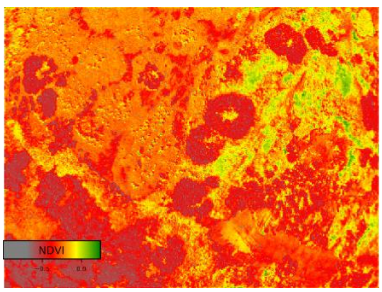


Figure 3.29(b) Approach 1

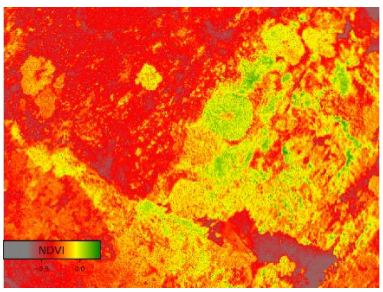


Figure 3.29(c) Approach 2

Figure 3.29 Resultant NDVI Images

Figure 3.29(a) is obtained by normalizing the image, which is in Figure 3.29(b) and that was constructed based on Equation 3.8 and Equation 3.9. Though the full range of NDVI lies in between -1 and +1, active photosynthesis corresponds only in the range between 0.3 and 0.9, therefore normalizing the calculated values would give a better representation of the NDVI image.

Figure 3.29(c) is obtained based on Equation 3.10 which is not as good as the earlier approach. However, still we can use enhancement techniques to obtain different features compared to the other approach.

Figure 3.30 shows an example of the usage of NDVI and we can differentiate the healthy plants and weaker plants, considering its' NDVI values.

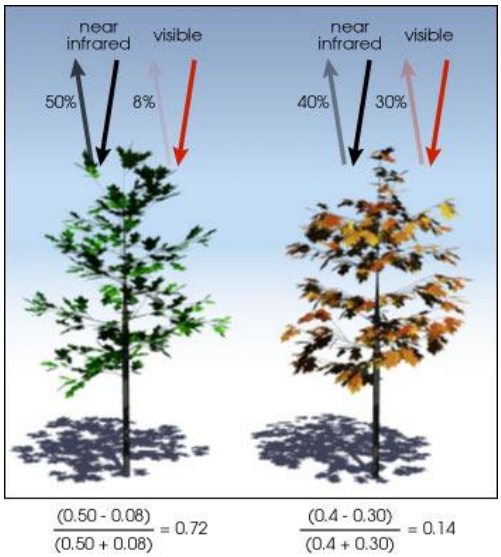


Figure 3.30 NDVI calculation example

3.2.3.3.2 FPGA Implementation - NDVI Calculation Module

The main calculation is done in the NDVI calculation module. Throughout the development time, several modules have been implemented which were following different data structures and different calculation techniques. The HDL program is developed in a way that with minimal modifications, the NDVI calculation module can be changed.

The input variables to the NDVI calculation module would be three vectors, each having an 8-bit width and the output is returned as a vector of 8bit width. The first input vector is representing the NIR value of a specific pixel and the other two vectors are carrying the green and blue colour channel information, respectively. Similarly, the output vector is returning the calculated NDVI value for the pixel, and using this value, a grayscale image having a single channel can be constructed.

NDVI calculation modules use different data structures to process image data, such as standard logic vector representation and integer type representation. Each representation has unique advantages and disadvantages when implementing a certain logic function.
Using standard logic vector representation is a straightforward approach to process which does not require any type conversion. Simple operations such as addition and subtraction can be performed easily with vectors and the structure of the function is easily understandable. Nevertheless, there can be situations that may lead to overflow the values because vectors have a predefined number of bits that can be used in calculations.

The integer representation is an important feature in VHDL because in typical hardware description languages, primitive types definitions are not supported. This feature gives freedom to the developer to look at the problem in the programming point of view and come up with a solution. When the integers have been used to store data, the developer does not have to pay much attention towards overflows because integers have a range from $-(2^{31}-1)$ to $+(2^{31}-1)$. However, with integer representation, additional type conversion step needs to be added because the input and output data will follow the vector representation.

Arithmetic operations related to NDVI calculation are involved with floating-point numbers as well. Therefore, arithmetic operations with floating-point numbers are performed using separate components. The implemented sequence of operations to calculate NDVI value is mentioned in Figure 3.31.

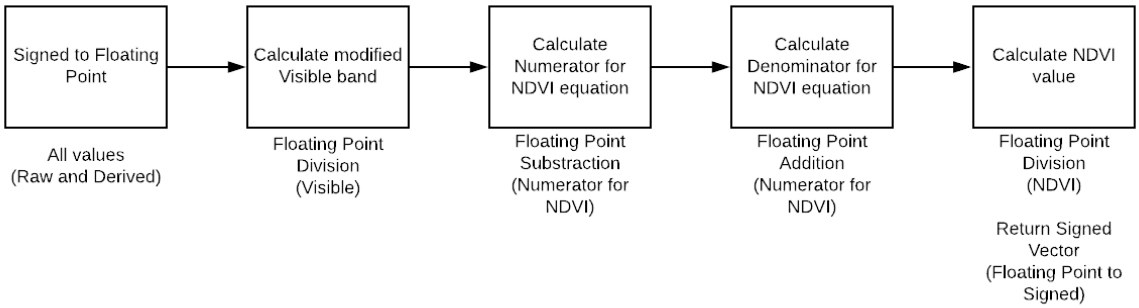


Figure 3.31 NDVI Calculation Sequence

Since the NDVI calculation is done as a point operation, steps are executed sequentially. To achieve the sequential execution, calculation steps were defined in a single process block, which is triggered per each clock pulse. When there is a serialized code, the convention is to use variables instead of signals to hold the internal values. Apart from that, value assignment to variables will be performed without delay, whereas in signals, there will be an associated delay due to wire.

Table 3.10 Results Table

Image	Results Received	Discussion
		Thresould added (0.5) to generated NDVI image. Leaves containing high chlorophyll amount are highlighted.
		Same threasould (0.5) added. Noise removal has to perform.

		0.5 threashould. Low noise level because the image contain less weeds.

3.2.3.3.3 Floating Point Arithmetic in FPGA

NDVI calculation involves floating-point numbers and therefore implementing a synthesizable calculation module using built-in primitives would be a challenging task. For that, separate routines have been implemented to carry out the floating-point arithmetic operations.

The module adheres to the IEEE 754 compliant of single precision floating point numbers, where a floating-point number is represented as a 32-bit value. The first bit is reserved for the sign (positive or negative) bit followed by an 8-bit exponent and a 23-bit mantissa.

The component includes the floating-point addition, floating point division and two conversion functions to convert data between standard vector data and 32-bit floating-point numbers.

Floating-point addition and subtraction are done using the same component and for the subtraction, the input value for the second operand should be passed as a negative signed value. The ADD_SUB component is functioning as a state machine with five states. Since the addition operation requires the exponents of the two operands to be equal, an exponent alignment step has been performed and shift the mantissa accordingly. After the addition operation, a post normalization step is added to handle overflows and in that, a left shift is performed according to the position of leading one.

The floating-point division component has a separate subcomponent to perform the mantissa division. Mantissa division functionality is also implemented as a state machine with five states. Those states include an idle state, operand validation state, performing and verifying state, normalization state, and finish state.

3.2.3.3.4 UART Interface in FPGA

One of the main challenges faced when implementing the system in an FPGA is to have a robust communication channel to receive and transmit image data. Since the selected USB cameras could not directly connect to the development board, a standard interface should be implemented within the FPGA. Since BASYS3 has an inbuilt AXI4-Stream IP core, a UART communication can be implemented. In order to instantiate an AXI4-Stream module, several internal signals required to maintain the functionality. Those internal signals and the intended use of those signals are described in Table 3.11.

Table 3.11 Internal Signals of UART Interface

Name of the Signal	Usage
clk_uart	Global clock. All signals are sampled on the rising edge of clk_uart.
Rst	Global reset signal. Working as active-LOW
RS232_TX / RS232_RX	Enable signals for individual ends (rx / tx)
rx_aresetn / tx_aresetn	Reset signal for individual ends (rx / tx)
rx_tvalid / tx_tvalid	tvalid indicates that the master is driving a valid transfer.
TDATA	TDATA is the primary payload that is used to provide the data that is passing across the interface. The width of the data payload is 8 bits. [7:0]. Two such signals exist for two ends.
TREADY	Used in slave mode. Indicates that the slave can accept a transfer in the current cycle. Two signals exist.

To verify the functionality of the communication interface, a simple grayscale conversion module is implemented and tested the operation. In the grayscale conversion module, the arithmetic sum of three colour channel values will be taken as the grayscale value.

3.2.3.3.5 FPGA Implementation – Image Acquisition module

The first part that was implemented in the FPGA was the image acquisition module. It is the interfacing module between the image capturing device and the FPGA processing system. The image acquisition process implemented using Verilog Hardware Description Language, and the behaviour of the acquisition module is described using a state machine representation.

The operation of the acquisition module is controlled using four flags.

- Start bit
- Vertical synchronization flag
- Horizontal synchronization flag
- Done flag

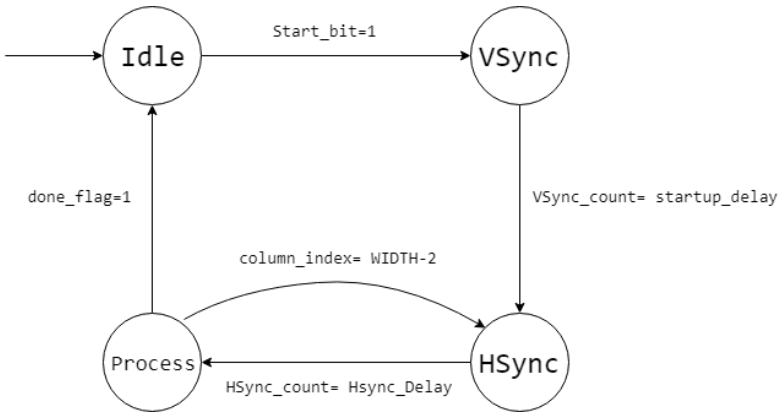


Figure 3.32 State Diagram for Image Acquisition Module

Since the image acquisition is made row by row, a horizontal synchronization flag is required to signal the module that the data for a row is transmitted and ready for processing.

This v_sync, h_sync-based approach is easy to understand when there are complex architectures. However, for our purpose, a Mealy machine can be used as the base architecture for the image acquisition module.

According to the initial plan, the imaging system was supposed to use two OV7673 RGB cameras and remove the IR filter from one camera and develop the multispectral camera. But there has been a critical requirement to keep one optical path in both sensors and if this approach is used, there will be a mismatch in two optical paths. Using this approach, images can be transmitted to the FPGA by 8-bit width bus and camera configuration can also performed by internal logic of FPGA circuit. Since BASYS3 has two PMod interfaces, both NIR and RGB images can be taken simultaneously with this approach.

Despite that, there has been an uncertainty in capturing NIR images by the same sensor. There was no clear way of removing the IR filter and there was an associated risk in modifying the

tiny hardware components of the camera. Since then this approach is neglected and continue with a different method in image acquisition.

3.2.3.3.6 External Image Buffer and Camera Interfacing Component

In the development stage, the protocol translation is done using MATLAB. The functionality is implemented in a MATLAB live script, which can be executed on a computer. The MATLAB script includes the camera calibration settings, image acquisition, and transmission to the FPGA via COM port and receives the processed image and writes the received image into the storage. In addition to that, this intermediate program can be used as a buffer for the images because, for the processing, it takes an average of 20 seconds per image due to the communication bottleneck in the serial port.

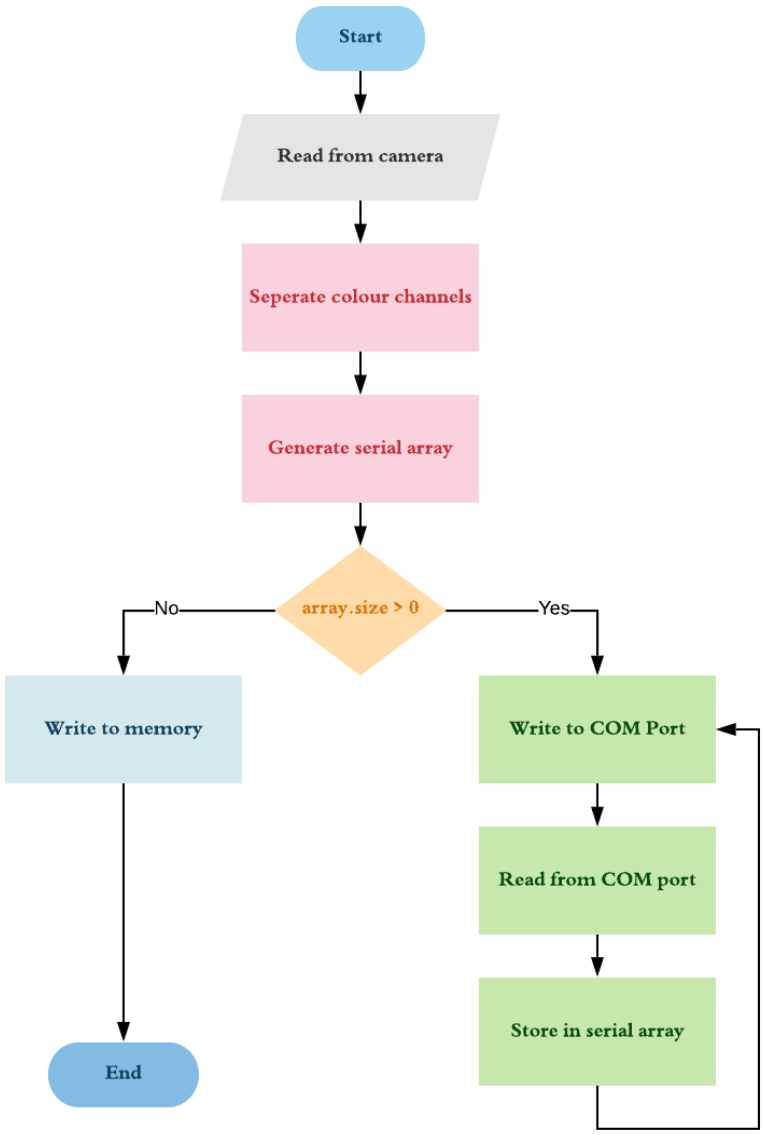


Figure 3.33 Flow Chart for image serialize section

3.2.3.3.7 Non-Functional Components

A memory block which is having 120*160 addresses and 8bit width has been defined in order to store the generated NDVI values. This module can be used as an output accumulator and further extend to get the complete image.

The memory block is a having simple dual-port configuration in which read and write operations can be performed separately. The writing port is working in standard 100MHz clock, and the reading port can operate with a different clock. As long as the clock rate of the reading port is lower than the writing rate, the processed image can be extracted row by row.

3.2.4 Summary

Vegetation index-based imaging is useful in remote sensing and precision agriculture applications. Among several vegetation index concepts, NDVI has a higher relatability with the quality of vegetation depending on the chlorophyll content of leaves. Usage of specialized imaging mechanism will ease the segmentation tasks that are based on the quality of the cultivation.

The advantage of parallelism to increment the throughput is significant in FPGAs. For real-time applications that use FPGA's as the processing platform, communication protocols having relatively higher bandwidths are required to interface the platform with external entities. Since FPGAs itself does not implicitly support many standard communication interfaces, custom made interfaces or vendor-specific interfaces must be used.

Because VHDL has several in-build data types such as integer and real numbers, it has become much easier in implementing logic systems in a much readable manner. When performing numeric calculations in FPGA logic circuits, programmers must pay extra attention towards scaling and overflows of the values as well as making the designed component a synthesizable one. There can be instances that the functional verification can be performed in a simulated environment using built-in data types such as "real", but when implementing such logic circuit in actual hardware, real valued operation must be implemented in a synthesizable manner using custom supporting modules.

3.3 Software Part 1: Evaluating a Tea Leaf Image

3.3.1 Introduction

This section focuses on evaluating a tea leaf image using colour as its features. It consists of two major components

1. Pluckable tea leaf segmentation.

   Segmenting out pluckable tea leaves (Tea buds + high grade leaves + low grade leaves) from other leaves. Segmenting leaves vs segmenting part of leaves is a major challenge when comparing with the research on this area. In tea leaves quality of the leaves degrade with depth, since here we are using vision as the feature, we need to come up with a good methodology to segment leaves with a good accuracy.

2. Evaluation of segmented leaves.

   Numerical evaluation for the segmented pluckable leaves. Can be mapped to days left to pluck or as a goodness measure. This evaluation will be used as the quality of the tea leaf image.

3.3.2 Literature Review

There have been many researchers conducted related to plant phenotyping (rapidly emerging research area concerned with quantitative measurement of the structural and functional properties of plants), which consist of research in the plant disease identification, leaf counting, leaf segmentation and measurement of plant growth. In modern days plant phenotyping has been done in a controlled environment or in the field. Many researchers have contributed a lot to plant phenotyping. Huang [10] proposed a method for automatic identification of plant species. Manuel Grand-Brochier [11] depicted the studies to extract the tree leaves from natural images by applying various segmentation methods. Tang [12] proposed an algorithm to extract the leaf region from the images with complex background. Shen [56] proposed an automated system based on computer vision technology for counting the soybean leaf aphids. Cerutti [57] proposed a parametric active polygon model. Above is some examples related to researches in plant phenotyping.

3.3.2.1 Image segmentation

Image segmentation is one of the most used process in image processing systems such as image retrieval, pattern recognition, object detection, medical imaging and video surveillance.

The outcome of segmentation is mainly used for understanding the image and entity recognition through the identification of the ROI (region of interest). Image segmentation is the process of partitioning a digital image into multiple segments. Survey on image segmentation [58] show that the following methods are been used in the modern days of machine vision to segment out images.

- Edge based
- Threshold
- Region based
- Clustering
- Watershed

Edge based techniques: This technique is based on locating the boundaries of a targeted object within the image. It segments the image by identifying the intensity differences at the boarders. Sobel [59], canny [60], Laplacian and fuzzy logic are some common techniques that used for edge detection.

Threshold based techniques: This method based on threshold value calculated by converting image to binary image. Thresholding methods includes Global and Local thresholding, which having methods like Otsu global thresholding [61] and Adaptive local thresholding [62].

Region based techniques: This method partitions an image into different same types of regions. Pixels of same type identified and grouped together into the same type of regions. The Main types of Region based Segmentation are Region Growing, Region Splitting [63] and Region Merging.

Clustering based techniques: Clustering is the technique used for segmentation, in which image is first converted into histogram and then clustering is performed on it. One of the basic clustering algorithms is K-means [64], which is intended for segmentation in textured images. It clusters same pixels to k-segments of the image.

Watershed based Segmentation: The watershed based [64] method uses the concept of topological analysis. In this the intensity represents the basins. The watershed methods think about the gradient of image as topographic surface. The pixels having more gradient are represented as boundaries which are unbroken.

3.3.2.2 Leaf segmentation

The above five methods can be used for segmenting leaves from background or segment part of leaves from remaining area.

Nowinski and researchers [65] has used Region Based segmentation techniques to find out the disease regions. Out of the three basic Regions based segmentation techniques such as Region Growing, Region Merging and Region Splitting they derived methodologies which is used to select the best one. Based on timing analysis and Quality Metrics, it was found that Region growing had the maximum peaks representing distinct regions with least discrete entropy and highest grey level energy as compared with mean-shift segmentation methods. Hence it was derived that Region Growing Segmentation can be utilized as the segmentation strategy for further processing.

Darshana and Research group [14] has proposed an unsupervised, programmed K-means clustering algorithm to play out the division assignment of wheat leaf scab image in view of the Lab colour space. The investigations on wheat leaf images with three basic sicknesses demonstrated the acceptable execution of their technique as far as precision, productivity, and automation.

Xiaojing's research [13] propose a k-means clustering based tea bud recognition method. They use HIS colour model for further enhancement of the feature differentiation. An improved k-means clustering algorithm has been used to identify and separate the tea buds. They have also suggested the HSV colour model to further enhanced the differentiation. They have received good result for k =3 with good accuracy.

Xianwei and researchers [66] propose an efficient method to extract the leaf region and count the number of leaves in digital plant images. The proposed method has three steps. The first step involves a new statistical based technique for image enhancement. The second step involves in the extraction of leaf region in plant image using a graph-based method. The third step involves in counting the number of leaves in the plant image by applying Circular Hough Transform (CHT). The proposed work has been experimented on benchmark datasets of Leaf Segmentation Challenge (LSC). The proposed method achieves the segmentation accuracy of 95.4%.

Table 3.12 gives a comparison for the different techniques when used for leaf segmentation.

Table 3.12 Comparison between segmentation techniques

Segmentation technique	Advantages	Disadvantage
Edge based	Good for images with better contrast between objects.	Not suitable for images with too many edges.
Threshold based	Does not require any prior information of the image. Fast and simple implementation.	Highly noise sensitive.
Region based	Works well for images with good contrast between regions.	Time and space complexity is high.
Clustering based	For small k values computationally fast. Eliminates noisy regions successfully.	Difficult to predict k. Computationally expensive for a larger k value.
Watershed	Results are more stable compared to other methods.	Complex calculations in gradient calculation,

3.3.3 Research methodology

3.3.3.1 Design

Proposed method for Pluckable tea leave segmentation.

Figure 3.34, a mechanism is proposed to segment out tea buds from rest of the image.

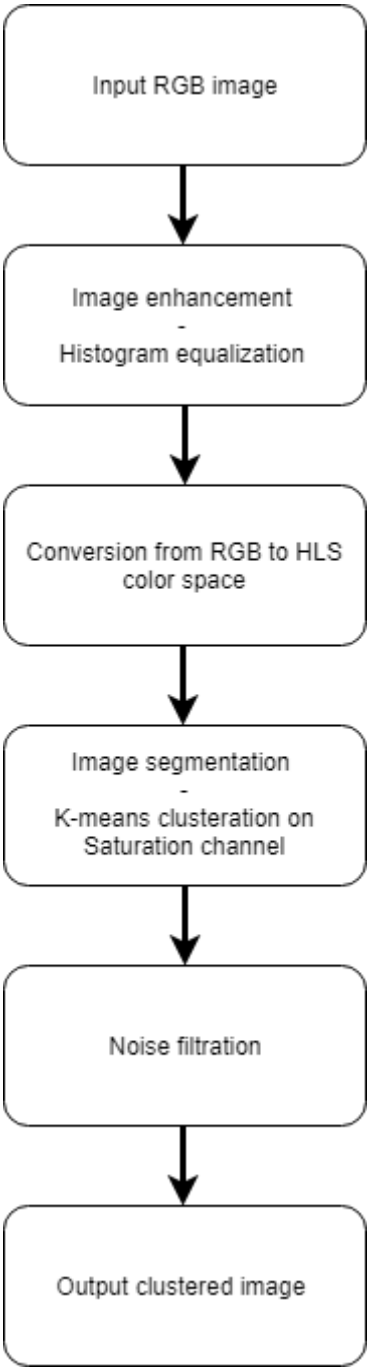


Figure 3.34 proposed method for leaf segmentation

3.3.3.1.1 Image enhancement

For the input image first step was to equalize the histogram to enhance the features of the image. Histogram Equalization is a computer image processing technique used to improve contrast in images. It accomplishes this by effectively spreading out the most frequent intensity

values, i.e. stretching out the intensity range of the image. This method usually increases the global contrast of images when its usable data is represented by close contrast values. This allows for areas of lower local contrast to gain a higher contrast. For colour images (RGB), it is not desirable to equalize three channels separately and then synthesize them, because histogram equalization is not a linear operation. So, the image is first converted into YCbCr colour space and then equalizing the histogram of Y channel (brightness) then converted back again to the RGB colour space.

Figure 3.35 shows where histogram is equalized on a lightness channel.

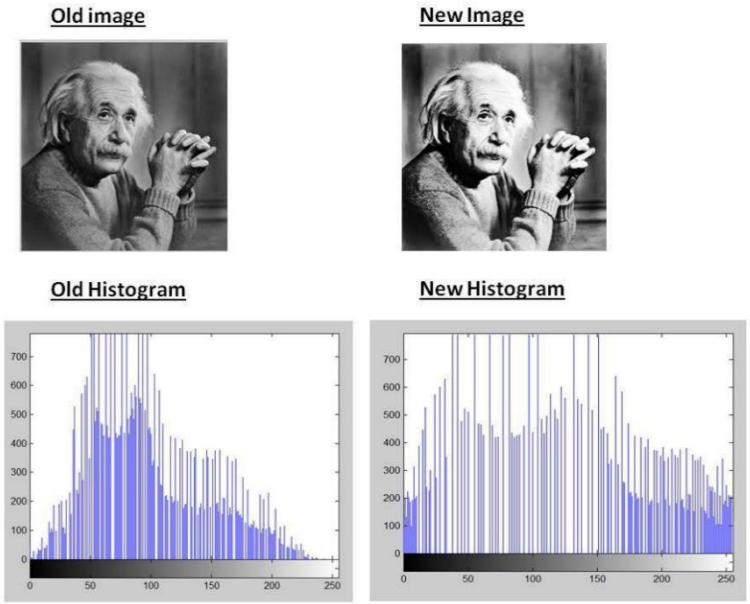


Figure 3.35 histogram equalization on lightness channel

3.3.3.1.2 Conversion from RGB to HLS

For the histogram equalized image next step was to convert it to a right colour model. Choosing the right colour model can reduce the complexity of the tea bud recognition algorithm as well as it can increase the efficiency of the algorithm. With the continuous development of computer vision technology, researchers have proposed many colour models.

These colour models include RGB models, HSI models, HSVmodels, etc. In digital image processing, more are used as RGB colour models and HSI colour models. HSI model was

proposed by H. A. Munseu in 1915. It reflects the way the human visual system perceives colour. It perceives colour by three basic characteristics: hue, saturation and intensity. Which H defines the wavelength of the colour, called Hue; S represents the depth of the colour, called Saturation; I represent Intensity or Lightness. Choosing HIS colour model here can reduce the effect of light intensity change on colour judgment. I have chosen HSL (hue, saturation, and lightness) colour space.

HSL values are calculated as below,

$$R' = R/255,\ G' = G/255,\ B' = B/255$$

$$Cmax = \max(R', G', B'),\ Cmin = \min(R', G', B') \quad \text{— Eq 3.11}$$

$$\Delta = \text{Cmax} - \text{Cmin} \quad \text{— Eq 3.12}$$

Hue calculation:

$$H = \begin{cases} 0^\circ & \Delta = 0 \\ 60^\circ \times \left(\frac{G'-B'}{\Delta} mod 6\right) & , C_{max} = R' \\ 60^\circ \times \left(\frac{B'-R'}{\Delta} + 2\right) & , C_{max} = G' \\ 60^\circ \times \left(\frac{R'-G'}{\Delta} + 4\right) & , C_{max} = B' \end{cases} \quad \text{— Eq 3.13}$$

Saturation calculation:

$$S = \begin{cases} 0 & , \Delta = 0 \\ \frac{\Delta}{1-\lvert 2L-1 \rvert} & , \Delta <> 0 \end{cases} \quad \text{— Eq 3.14}$$

Lightness calculation:

$$\text{L} = (\text{Cmax} + \text{Cmin}) / 2 \quad \text{— Eq 3.15}$$

3.3.3.1.3 Image clusteration

Next step was to apply k-means clustering to segment out the image to pick out the pluckable tea leaves. The K-Means algorithm is an unsupervised clustering analysis algorithm, in which the K value is defined according to the specific situation. The K-mean value is divided into samples according to the nearest neighbor's iterative rule without any other prior knowledge.

When the number of iterations and moving centroids increases, the clustering function converges to the end of the K-Means algorithm to complete the classification.

Steps in K-Means algorithm:

1. Choose the number of clusters K.
2. Select at random K points, the centroids.
3. Assign each data point to the closest centroid using Euclidean distance.

$$\text{distance} = \sqrt{(R_2 - R_1)^2 + (G_2 - G_1)^2 + (B_2 - B_1)^2} \qquad \text{— Eq 3.16}$$

4. Compute and place the new centroid of each cluster.
5. Reassign each data point to the new closest centroid. If and reassignment took place, go to step 4, otherwise, end.

3.3.3.1.4 Noise filtration

To compensate the noise in the image two steps were proposed. First is a simple thresholding based on the green pixel value. This helps in reducing the non-greenish cluster points. Thresholding value is decided based on the camera used.

$$f(i,j) = \begin{cases} f(i,j) & \text{if, } f(i,j)[G] > T \\ 0 & else \end{cases} \qquad \text{— Eq 3.17}$$

Second step is to remove the connected components which is below a given threshold. Threshold value is decided based on the image resolution.

The connected components labeling operator scans the image by moving along a row until it comes to a point *p* (where *p* denotes the pixel to be labeled at any stage in the scanning process) for which *V={1}*. When this is true, it examines the four neighbors of *p* which have already been encountered in the scan (*i.e.* the neighbors to the left of *p*, above it, and the two upper diagonal terms). Based on this information, the labeling of *p* occurs as follows:

- If all four neighbors are 0, assign a new label to *p*, else
- if only one neighbor has *V={1}*, assign its label to *p*, else

- if more than one of the neighbors have *V={1}*, assign one of the labels to *p* and make a note of the equivalences.

After completing the scan, the equivalent label pairs are sorted into equivalence classes and a unique label is assigned to each class. As a final step, a second scan is made through the image, during which each label is replaced by the label assigned to its equivalence classes.

3.3.3.2 Implementation

Following input image in Figure 3.36 is taken through the above process and the results are being showed and explained in this section.

Figure 3.36 Input image

For the input image first step was to equalize the histogram to enhance the features of the image.

Figure 3.37 Histogram equalized image

As we can see from the Figure 3.37 histogram equalized image tea buds are further highlighted. Furthermore, the edge details have being enhanced. One slight disadvantage here is that the effect of sunlight also has being enhanced.

Next step is to apply the HLS colour filter on the histogram equalized RGB image.

Figure 3.38 HLS converted image

As we can see in Figure 3.38 tea buds are further highlighted with a bright yellow colour. Since this is a three-colour space if we try to get a clusteration on this it can be prone to errors. For an example, since we are using Euclidean distance to get the distance between two points, ({0,0,255},{0,255,0},{255,0,0}) all these pixels have same Euclidean distance from pixel ({0,0,0} or {255,255,255}). To avoid this possible error the option was to go for a single channel image for clusteration. This further reduces the computing power as well. Figure 3.39 shows the HLS channels separately.

Figure 3.39 Separate channels (From right to left H, L, S)

If we look carefully on the separate H, L and S channels we can see that S channel (Saturation) has the bud details close to the HLS converted image, but it has some random noise. So, the target was to cluster the saturation channel image and then compensate the noise using above mentioned techniques.

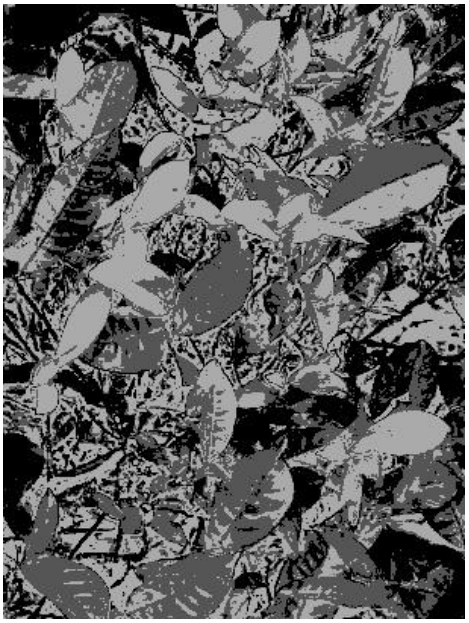


Figure 3.40 Clustered saturation space image

Figure 3.40 is the results that were received when clustering Saturation space image with k =3. A modified K-Means algorithm were designed to successfully segment tea buds from tea leaves. K-means algorithm were modified in such a way where we can initialize the starting centroids. After many experiments and explorations, it was found that when K=3 and [0,127,255] centroid selected as initial centroids, the buds of the separated tea leaves could be well identified.

Figure 3.41 is the cluster results when applied on the original image.

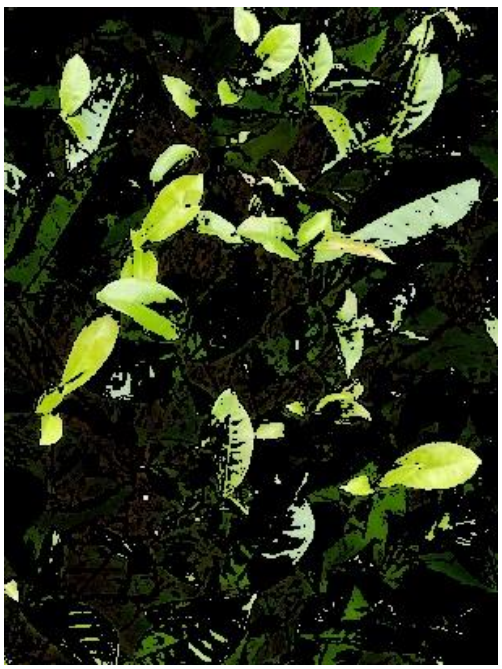


Figure 3.41 Cluster results applied on the original image

As we can see in the Figure 3.41 tea buds are clearly captured but there are lot of noise in the clustered image. Next step is to reduce the noise with mentioned noise filtration techniques.

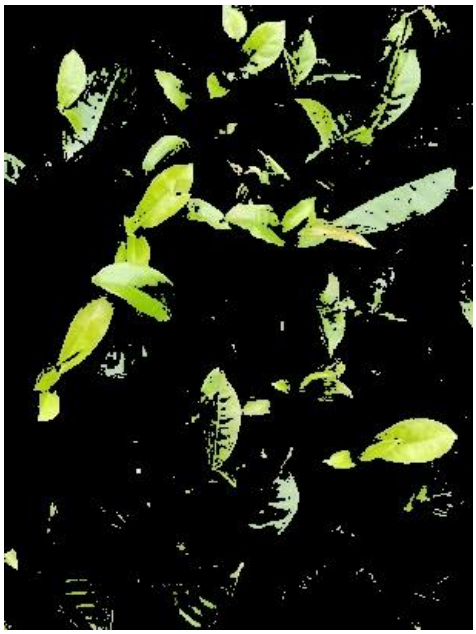


Figure 3.42 Cluster results filtered with the thresholding method

Figure 3.42 shows the results that we received after applying the first noise filtration technique. Image pixels were thresholded by a number compared with the green pixel value. As we can see it still contains some random noise in the image. Figure 3.43 is the final segmentation results that were received after applying the final noise filtration method. Which is the connected component removal below a given threshold.

Figure 3.43 Cluster results filtered with the connected component removal method

As we can see in the Figure 3.43, lot of noise is being removed and the tea buds are clearly segmented as desired. Final part of this section is to give a numerical evaluation of segmented image. For this I have used a percentage wise evaluation.

For a given image frame,

$$Numerical\ Evaluation = \frac{Tea\ buds\ coverd\ area}{Total\ Image\ area} \times 100\ \% \quad \text{— Eq 3.18}$$

For the above image,

Numerical evaluation = 14.729%

i.e. 14.7 percent of the image is covered with tea buds.

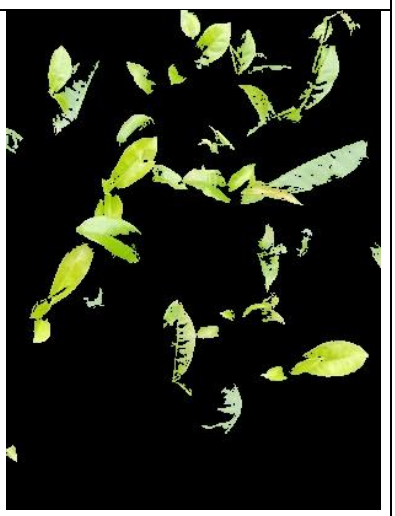
3.3.4 Results and Discussion

3.3.4.1 Results

First the results of the proposed method were compared with the ground truth of the image in Table 3.13. Image was taken from a mobile phone camera with a sensor of Sony IMX 519. Original resolution of the camera is 1080 x 2280.

Image is capped at a 305 X 407 resolution in the test image.

Table 3.13 Results comparison

Image	Ground truth	Results received
		
Numerical evaluation	10.370%	14.729%

If we compare the ground truth with the result image, we can see that it has identified all the tea buds, but it has also identified non pluckable tea leaves as tea buds. Thus, the methodology has a good recall, but it has a low precision. Identified non tea bud leaves are those with a sunlight reflection on top of them. The methodology has false classified the sun light reflected non tea buds as tea buds.

To check the dependability of the proposed method, it was tested on three data sources. And the result is given in the Table 3.14.

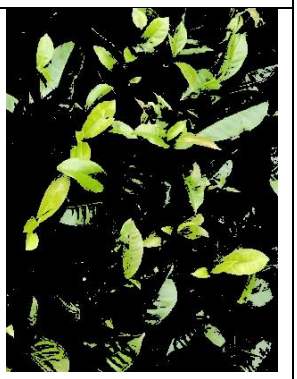
Table 3.14 Dependability comparison

Image	Results received	Camera and location.
		Camera - One plus 5T with a Sony IMX 519 sensor. Location - Galle
		Camera - iPhone 6 Location - Horana
		Source - https://www.thebrooke.org/our-work/nepal/journey-nepal-tea-estate-pictures
		Source - https://www.thebrooke.org/our-work/nepal/journey-nepal-tea-estate-pictures

As we can see in Table 3.14, proposed methodology is mostly independent of the image source (camera and the location). It has given considerable good results in all the cases.

Finally, the results are compared with another implementation proposed by Peidi Shao and team [13]. Results are shown in Table 3.15.

Table 3.15 Results comparison with available literature

Image	**Ground truth**	**Proposed methodology**	**Peidi et al [13] implementation**
	 NE - 10.370%	 NE - 14.729%	 NE - 25.943%
	 NE - 24.821%	 NE - 22.123%	 NE - 28.783%

**NE - Numerical evaluation

From the Table 3.15, it is clear that the proposed methodology performs well in tea bud segmentation. It has outperformed the implementation of peidi and team's method for tea bud recognition, by a considerable margin.

3.3.4.2 Summary

The object of evaluation of a tea leave image can be useful in various ways. For example, if we mapped the whole tea estate and then use this on top of it, we can get a yield estimation of the estate, we can search for anomalies of yield in the estate, we can even get the fertilizer requirements of the estate.

Proposed method consists of series of steps including image enhancement, colour space conversion, clusteration and set of filtration techniques. As shown on the results the proposed method performs well and does not depend on the image source or location. It has outperformed the available literature related to this object by a considerable margin.

Major drawback of the proposed methodology is the false segmentation of the sunlight reflected leaves as tea buds. With the filtration techniques the impact of that has being minimized.

3.4 Software Part 2: Map generation and Segmentation

3.4.1 Introduction

In this section, map generation and graph generation using images are discussed. Literature review, design, implementation and conclusion for each part is given in separate subsections.

Agricultural image processing is one of the main heavily researched areas in recent times. In the agricultural domain, image processing has been utilized in different ways to identify the crop, plant, leaves, flower, fruits etc. as well as to identify the diseases. To improve agriculture product efficiency, it is essential to have reliable systems.

Objectives of this section

- Generate a map of tea cultivation
    - Using captured images of the tea estate, generate a map which can be used to segmentation and visualization.
- Segment tea estate to a graph
    - Represent estate in a graph to mathematically represent tea cultivation so optimization algorithms can be used for further processing.

3.4.2 Literature Review

Map generation

Map generation using multiple images is a common image processing task that is highly researched by many researchers over the last four decades. For this, we can take two approaches. First one is stitching images together. The second approach is by generating a point cloud. Since it is efficient to compute, image stitching method is selected for this project. To stitch two images together, we need to first identify similar points in both images and match them accordingly. When selecting those points, it is necessary to understand that when stitching, there can be relative rotations, scale changes, illumination changes, noise and deformations in comparing images. Good feature matching algorithm should be able to identify features in such a way that those effects are minimized in feature description.

Scale Invariant Feature Transformation (SIFT) [16] is a method proposed by Lowe in 2004 to match and detect objects in given images efficiently. In this paper, the author explains how this process is done in nearly real-time in four steps. First, scale-space extrema detection is done to identify potential interest points that are invariant to scale and orientation. It is important to detect features in a way that even after some rotations or scale changes or changes in 3D viewpoint, they can be differentiated and identified correctly so that we can match them efficiently. To achieve this, the author uses the difference of Gaussian function. The second step is keypoint localization. From previously identified interest points, most stable points are chosen, and a detailed model is fit to determine location and scale. The third step is the orientation assignment. Based on local gradient directions, one or more orientation is assigned to each feature point. All further processing is done relative to this assigned rotation to make features rotation invariant. The fourth and final step is to describe those points in a way that allows a significant amount of local distortions and illumination changes.

Oriented FAST and Rotated BRIEF (ORB) [67] is an alternative to SIFT which is free to use with fast matching times. They use FAST keypoint detector with a slight modification to also track rotational component of a feature. Unlike the previous method used in SIFT, this method is computationally efficient. The only downside is it only returns just one dominant orientation compared to many rotations in SIFT key point. For keypoint descriptor, authors use BRIEF, which is very sensitive to rotations. They propose a modified method to make BRIEF rotation aware.

Automatic Panoramic Image Stitching using Invariant Features [15] is a method to stitch multiple images using SIFT features as feature detection. In this paper, the authors give a solution to automated panoramic image stitching. Using this method it can even identify and stitch unordered image sets. A ratio test is applied to matching features to make later calculations more efficient. Homography estimation is done using the RANSAC algorithm to efficiently eliminate outliers and consider most possible inliers for the result statistically. After this step Bundle Adjustment. Bundle Adjustment is performed to reduce any error accumulated from pairwise homography calculation. This is basically adjusting all images at once using image features. Next step is to Gain Compensation to minimize intensity mismatch between overlapping pixels while stitching. To further reduce this mismatch, authors use Multi-Band Blending, which is blending overlapping images using pyramid image structure to create a smooth blend.

Segmentation and Graph Generation

Generation of a graph using an image is also a heavily researched area. With our application of tea plantation, we must identify tea planted areas clearly and segment them using colour and features such as edges. For navigation and many other applications, graph generation is used.

Multichannel Image Segmentation Algorithm for Agricultural Crop Detection [68] is proposing a solution to segment images when given a border, and it takes colour into account when segmenting, unlike many other methods available for segmentation. Since this method requires a predefined boundary, it is not suitable for our application. Normalized Cuts and Image Segmentation [69] paper proposes a technique that is to look from the big picture to downward rather than traditional pixel value comparison. They propose a graph-theory based algorithm to group areas into a graph. Contour Detection and Hierarchical Image Segmentation [70] paper present a way of reducing image segmentation problem into contour detection problem. Authors have also introduced a high-performance contour detector which combines both local and global features. Random Walks for Image Segmentation [71] paper also presents a novel method for segmentation when a user has defined starting pixel values. Then for each pixel, the probability is calculated for each class, and highest class is assigned to the pixel. Since this method requires initial classes for a few pixels, this is also not suitable for our application. Region Competition [72] is a statistical-based algorithm which is derived from minimizing a generalized Bayes criterion using the variation principle. One advantage of this algorithm is that it is guaranteed to converge into a local minimum. This algorithm also requires an initial seed to work. Unsupervised Segmentation of Colour-Texture Regions in Images and Video [73] also introduces a new method for image segmentation. In the proposed JSEG method, there are two main steps. First is colour quantization which is quantization of colours into different classes to use for differentiating regions in an image. Then image-pixels are replaced by colour labels. This is called class-map of image. Then the region growing method is used to segment image.

Efficient Graph-Based Image Segmentation [17] paper discusses a method for image segmentation using a greedy approach to connect graph vertices which have low difference using a threshold. This paper also solves gradient issues when segmenting using traditional segmentation methods. In this paper, they also show how global properties are preserved when segmentation is done using this algorithm.

3.4.3 Research methodology - Map generation

The objective of map generation is to take multiple aerial images as input and generate a map of the area covered by those images as output to further segment and use it for graph generation in the next section.

Two main approaches in the literature for map generation

- Stitching images
    - Using video captured from drone camera, the generation of the map is done by stitching selected frames using matching and aligning.
- Point cloud method
    - Frames are processed for distinct features, and a point cloud is generated. Using point cloud to generate a top view of the scene.

3.4.3.1 Design

From the above approaches, stitching method is chosen for implementation since it requires less computational power compared to point cloud method. Stitching one image to another can be achieved using two primary techniques. Those are feature-based methods and image region matching methods. From the methods mentioned above, feature-based matching is more robust and dependable compared to another method for the agricultural domain. There can be differences between consecutive two images, such as contrast difference, scale difference or rotations. By trying to find distinct features in images first and then trying to match them using those matching features gives better results.

Several problems are identified during the initial implementation process and suggested solutions are presented in Table 3.16.

Table 3.16 Problems and Solutions identified in Map Generation

Problem	Solution
Stitching image resolution on memory	Keep images in pyramid structure
Difficult to down sample and feature match	
Large resulting image size and resolution	Keep only partial images in memory
GPS mapping	Generate initial empty image map

3.4.3.2 Implementation

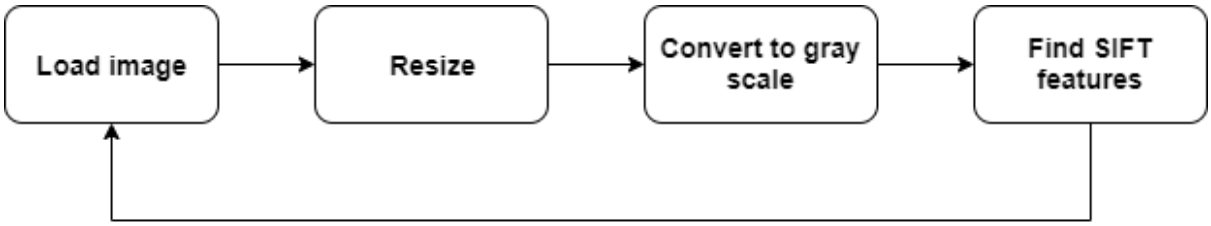


Figure 3.44 SIFT feature identification

Features are matched in images one by one and results are stored in the hard disk. When feature matches are needed, relevant files are loaded from hard drive and processing is continued. Initial images are resized to 20% of the original size before feature matching since original image resolution is high in the dataset. After features are matched and images are stitched result is again resized by 50% of the original for better representation.

Initially, a new data structure is created to hold images in a grid type space sorted by GPS coordinates. The idea was to match images that are only nearby in GPS coordinates from the original image. The main problem of that approach is that images can have rotations, GPS data mismatch or missing grid images that can result in non-matching images to compare with each other when the matching algorithm is applied. To efficiently calculate features, all images are converted to grayscale. This reduces computation time significantly. Approach to keep partial images in memory is not required when the calculation is done on smaller size images. Later we decided to go with simple image stitching one after another after the failure of the initial idea. This gave better results compared to GPS data grid system.

SIFT (Scale Invariant Feature Transform) method is selected for feature finding since it can robustly find unique features in images which can be used to match confidently. Figure 3.44 shows how features are found using the algorithm. There are many online resources available for this algorithm since it is used widely in state-of-the-art systems which require feature identification and matching. Some of many use cases of SIFT feature matching technique are object detection, motion tracking and creating 3D structure from multiple images.

There are four main steps in the SIFT process to identify and describe distinct features of an image.

- Scale space extrema detection
- Keypoint localization
- Orientation assignment
- Keypoint descriptor

Blurring is a convolution of Gaussian operator with the image, as shown in Equation 3.19.

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad \text{— Eq 3.19}$$

The symbols:

L- resulting image

G- Gaussian Blur operator

I- input image

Gaussian Blur operator is equal to Equation 3.20.

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad \text{— Eq 3.20}$$

Scale space extrema detection

Scale-space is a progressively blurred out image set. In SIFT each image is up-sampled first to make it double the size of original and blurred to remove aliasing effects. This gives more overall keypoints. After five blur levels, starting image is down sampled to half size and again progressively blurred for five times. This one set of five blurred images is called an octave. The whole process is continued until four octaves are generated with five blur level images.

To calculate extrema, Laplacian of Gaussian (LoG) operation can be used. to calculate LoG, the second-order derivative of an image must be calculated. This locates edges and corners on the image. Those are high interesting points to start feature finding. The only problem is calculating the second-order derivative is computationally intensive. In SIFT LoG is

approximated differently. By taking the difference between two consecutive scales of scale-space (DoG) gives a near approximation to LoG, as shown in Figure 3.45.

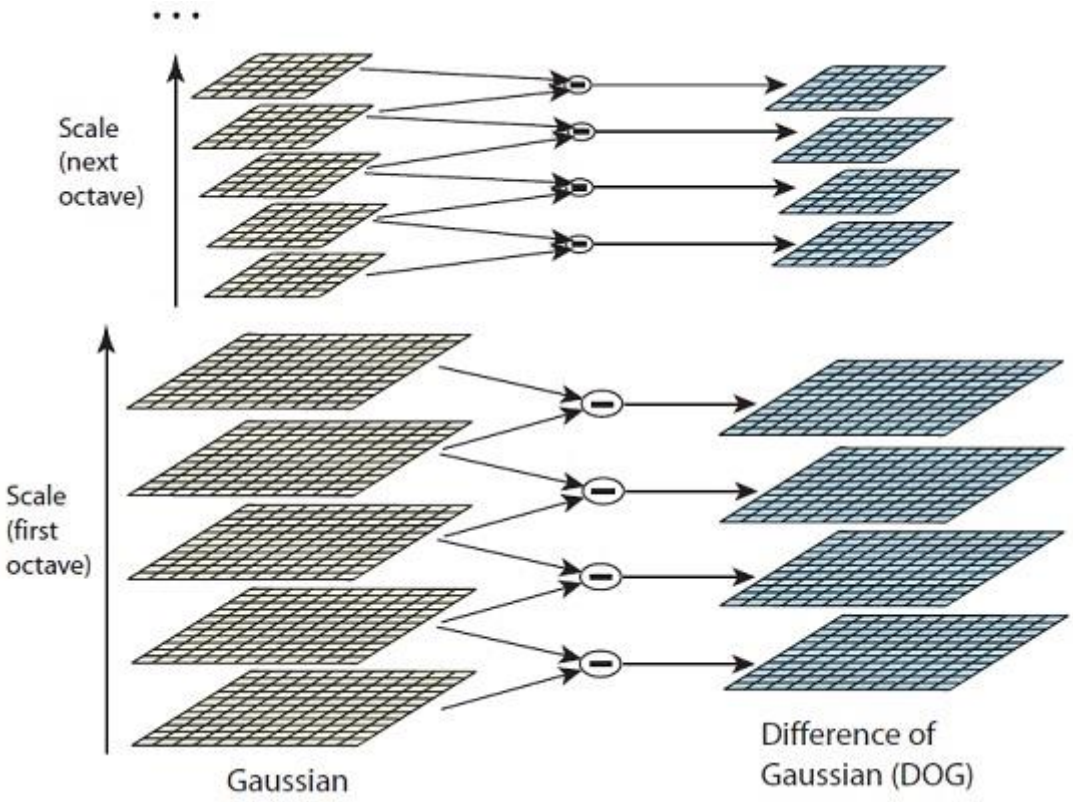


Figure 3.45 Scale space, octaves and DOG

Keypoint localization

To find keypoints, extremes should be identified in the above DoG images and find subpixel maxima or minima locations of them. To identify extremes, each pixel is checked with neighbours of the same image and same pixel area in above and below image of scale-space.

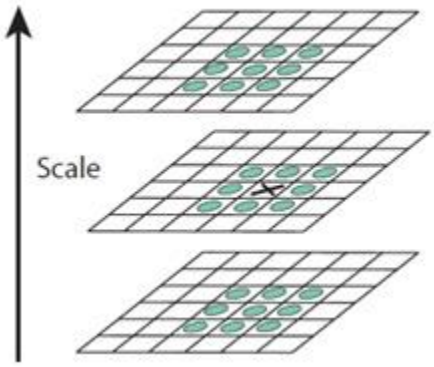


Figure 3.46 neighbours of a pixel

There are 26 neighbours to each pixel, as shown in Figure 3.46. Pixel is considered extrema if all 26 neighbours are either higher than the current pixel value or lower than current pixel value.

Since images are a discrete representation of actual scene using pixels, we can try to locate subpixel maxima/minima values corresponding to above-identified extremes. Using the available pixel data, subpixel values are generated. This is done by the Taylor expansion shown in Equation 3.21 of the image around the identified extrema.

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^\mathbf{T}\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x} \quad \text{— Eq 3.21}$$

Now we have a large number of keypoints generated through the previous step. Keypoints which lie on edges and keypoints having low contrast are useless for feature matching since they cannot be uniquely identified. To remove edge features, we calculate two gradients at keypoint, which are perpendicular to each other. If we analyse the image region around keypoint, there can be several possibilities for gradient values,

- Both gradients are low – this means no significant change in either direction
- One gradient is large – this means the feature is on an edge
- Both gradients are large – this means the feature is on a corner

Since corners are great features, we check each gradient for threshold and only selected keypoints are selected for the next stage as stable keypoints.

Orientation assignment

For this step, gradient direction and magnitude are calculated for all pixels around the keypoint. SIFT paper uses a histogram with 36 bins. So, for every 10 degrees, there is one bin (total of 360 degrees). After all, orientations are calculated and inserted to a histogram, most prominent orientations are assigned to each keypoint, and all future calculations are done using this orientation. This gives rotation invariance to these features.

Keypoint descriptor

The final step is to generate a unique way of describing each keypoint, while easy calculation and matching can be performed. In SIFT paper 16x16 window around the keypoint is taken and broken into sixteen 4x4 windows as shown in Figure 3.47.

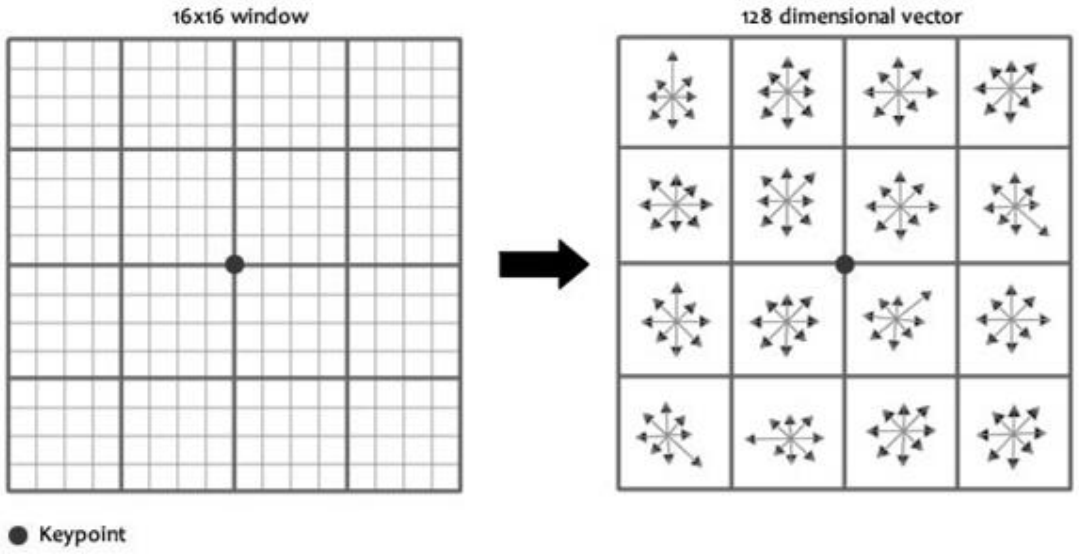


Figure 3.47 keypoint and neighbourhood

Then gradient magnitudes and orientations are calculated for each part of 4x4 windows. These orientations are put into an 8-bin histogram, as shown in Figure 3.48.

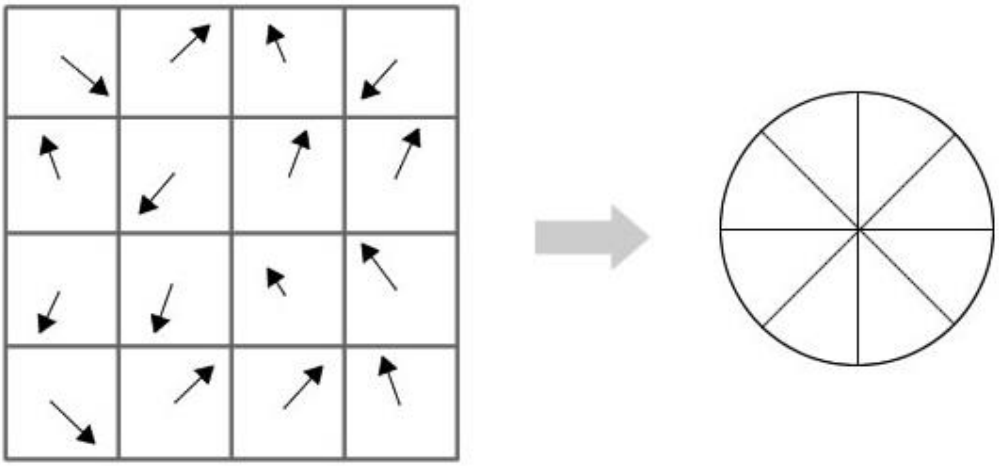


Figure 3.48 pixel gradients are used to generate 8-bin histogram

After doing this for all 16 pixels in a single region and calculating 8 numbers, repeat it for all other 15 windows. This gives 16x8 = 128 numbers as a result. This 128-length vector is used as the descriptor for the feature keypoint.

When matching feature points with each other, a ratio test is used before homography calculations in the next section to eliminate false matches. This is done by taking the ratio between close two matches to the same feature. If this ratio is too large, this means that it is not a good match.

Since the order of images is known, feature matching has been optimized to only match features with the previous image to reduce computational time. After identifying matching feature points between two images, the RANSAC algorithm is used to get rid of outliers and calculate homography between two images. Four pairs of correspondences are used to estimate homography in each RANSAC iteration shown in Figure 3.49.

Figure 3.49 RANSAC to find homography

Random Sample Consensus (RANSAC) is a robust estimation procedure that uses randomly picked set of correspondences to estimate image transformation parameters. This is used to calculate the homography matrix explained in the next section. To explain RANSAC using simple example data distribution shown in Figure 3.50 is selected. Most of the data points follow a linear pattern which is called inliers and some data points are outside of the linear model, and they are called outliers.

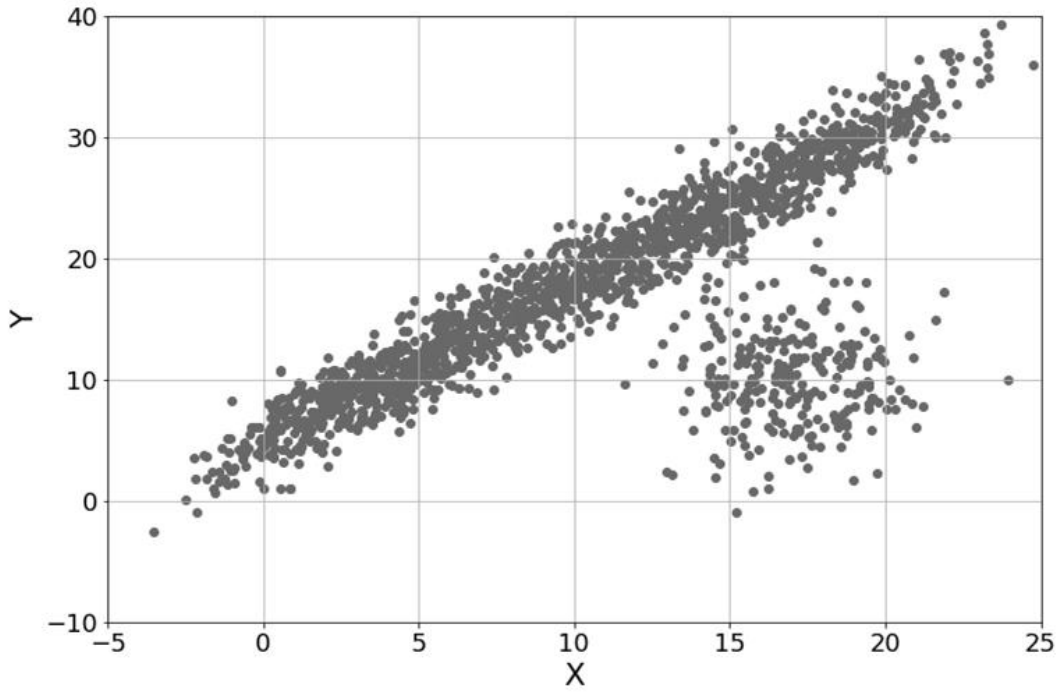


Figure 3.50 sample data distribution

If we want to fit a line for this dataset (line is taken for simplicity) results shown in Figure 3.51 can be observed.

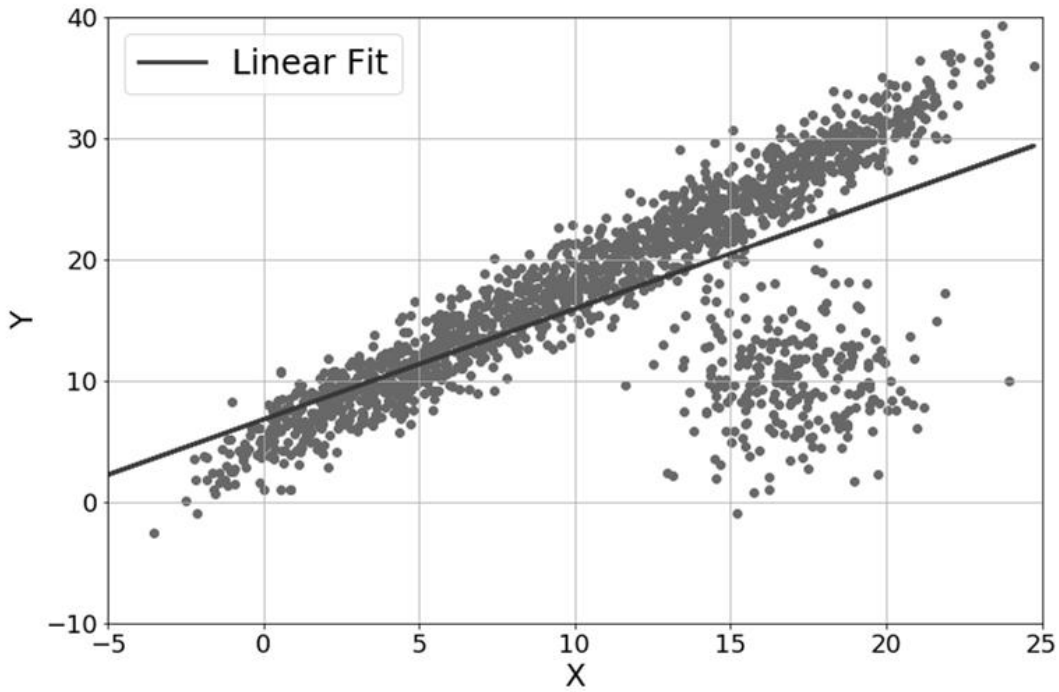


Figure 3.51 Linear fit considering all data points

The best fit line is moved to the side with the effect of outliers. If we can discard those outliers and consider only inliers, the result would be much better. One approach for achieving above is using RANSAC method introduced by Fischler and Bowles in 1981. There are four main steps to this algorithm which are listed below.

1. Randomly select a predetermined number of points from the data set. These points are used to fit a curve to our data set (in above example a line. So, we select two points).
2. Use randomly selected points to fit a line
3. Then data points are checked for distance with the line for a threshold. If data points are within threshold those are considered inliers. Else they are considered outliers. Count of inliers for each sample is recorded.
4. Repeat all above steps until one of the following conditions are met.
    - Iteration limit reached (predetermined) → return sample with most inliers detected
    - Enough inliers are detected for a sample → return current sample

Stitching algorithm initially assumes that image capturing is started from a corner of the land. Features of the first image are then calculated and stored. Next, second image features are calculated and compared with first image features to identify matching keypoints. Then

homography between the first two images is calculated using matching points. Using that homography matrix, those two images can be aligned with matching features correctly.

There are different types of transformations available. Translation, Rotation, Affine and Projective are some of them. All these transformations can be achieved by multiplying a relevant matrix with the image matrix. The homography is a combination of one or more transformations from one projective plane to another.

After features are calculated for the third image, those features are matched with features of the second image and outliers are discarded with the use of the same method as before. Then homography is calculated between image two and three. Result of multiplying homography between image one and two, with homography between images two and three is homography between images one and three (H12 x H23 = H13). By using this property, homography between image one and every other image is calculated. Then using those results, every image is warped, as shown in Figure 3.52 and aligned relative to the first image to get the final stitched image.

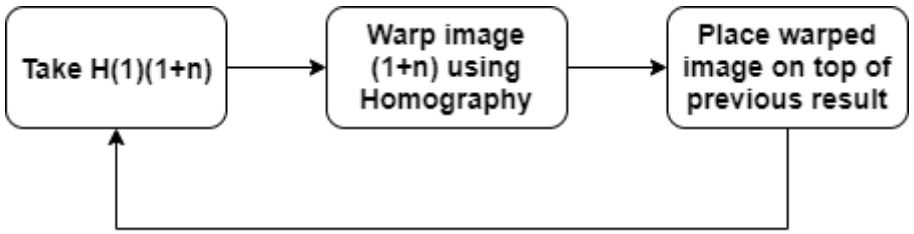


Figure 3.52 image warping and stitching

H(1)(1+n) means homography between images 1 and 1+n

3.4.3.3 Results of Map Generation

Figure 3.53 Feature points matched using SIFT

Figure 3.53 shows identified feature points in two images and matching features between them.

Figure 3.54*(a) Initial image*

Figure 3.54(b) Second image

Figure 3.54 First two images used to stitch

Figure 3.55(a): Figure 3.54(b) is warped with homography *Figure 3.55(b): Figure 3.54(a) is overlapped with warped image*

Figure 3.55: images in Figure 3.54 stitched

Figure 3.56(a): next image to stitch *Figure 3.56(b): image warped with homography*

Figure 3.56: next image is warped to stitch with result in Figure 3.55

Figure 3.54 shows first two images used for feature matching. Homography matrix between these two images (H12) is calculated using the RANSAC algorithm and the second image is warped accordingly, as shown in Figure 3.55(a). Next step is to overlap the warped image with the first image to get the stitched image shown in Figure 3.55(b). Next step is to calculate homography(H23) between Figure 3.54(b) and Figure 3.56(a). Then by multiplying H12 with H23, H13 is obtained. Using H13, Figure 3.56(a) is warped before stitching it with the previous result, as shown in Figure 3.56(b).

Figure 3.57: results from Figure 3.55 and Figure 3.56 merged

Using the result in Figure 3.55(b) and Figure 3.56(b), we can overlap them to obtain final stitched, as shown in Figure 3.57.

3.4.3.4 Summary

Image stitching is a problem with several answers, and each of them has its advantages and disadvantages. With the feature-based approach, more continuous results can be obtained with trading off computation time. Many feature-based techniques depend on robust feature detector and descriptor to achieve optimal results. SIFT is an excellent feature detection technique that can be used in near real-time applications. There exist many more feature detectors that can work in real-time. However, their accuracy is not on par with the SIFT detector. Image stitching can be further improved using image blending techniques. This can be used to eliminate seams that are created when stitching images. Bundle adjustment is another way to optimize stitching alignment globally using features of images. Gain compensation can be used to reduce differences between brightness level differences between stitching images.

3.4.4 Research methodology - Segmentation & Graph generation

The objective of Segmentation and Graph Generation is to take the stitched map from the previous section as input and produce a segmented image with graph representation of the image as output.

3.4.4.1 Design

Initially, canny edge detection is used for edge detection. There are five main steps for Canny edge detection.

Steps of Canny edge detection

1. Apply Gaussian filter
2. Find gradients of the image
3. Apply non-maximum suppression
4. Apply double threshold
5. Suppress all the other edges that are weak and not connected to strong edges.

These are the implemented Smoothing and Difference filters for future requirements

- Gaussian filter
- Mean filter
- Median filter
- Laplace filter

Initial approach for segment detection is given in Figure 3.58.

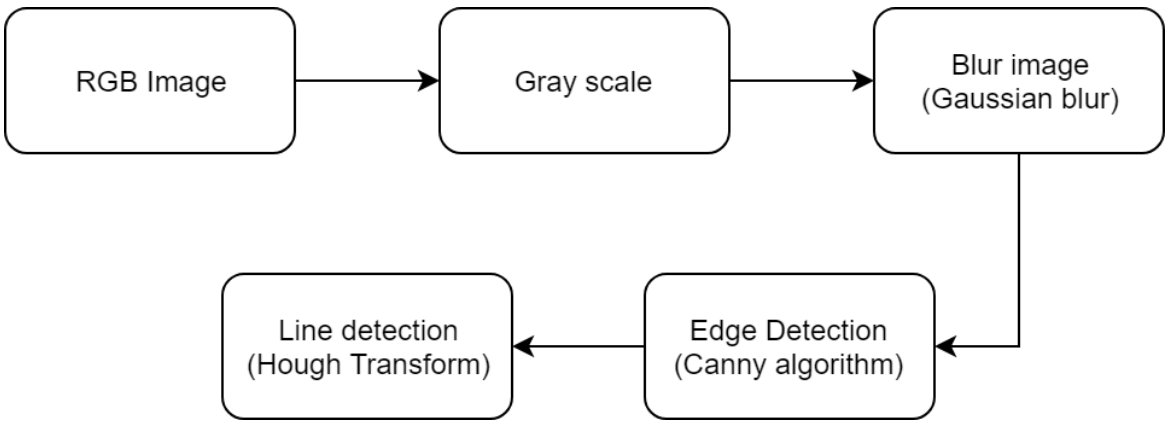


Figure 3.58 flow diagram of contour detection

As shown in Figure 3.58, the RGB image is converted to Grayscale image first. Then the image is blurred to reduce noise because canny edge detection is susceptible to image noise. Blurring image makes its edges harder to detect by regular differentiation operators. Since the Laplacian filter is used in canny edge detection, the effect of blurring is minimized for edge detection. Next, detected edges are used for line detection with Hough Transform.

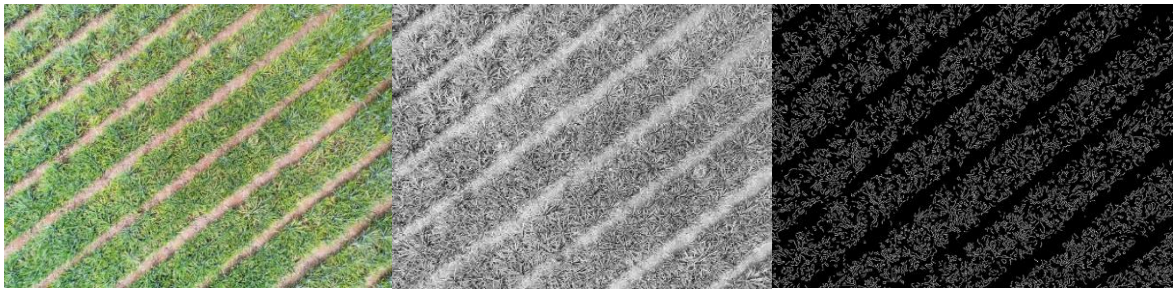


Figure 3.59(a) RGB Image *Figure 3.59(b) Grayscale Image* *Figure 3.59(c) Edge detection with Canny*

Figure 3.59 Edge detection flow

Figure 3.60(a) RGB Image *Figure 3.60(b) Gaussian Blur* *Figure 3.60(c) Edges detected*

Figure 3.60 Edge detection on RGB images

As shown in Figure 3.60, we can also use RGB images to edge detection. Features in all three layers are considered when blurring and edge detection when using this approach. For further improvement, we can also use the HSV colour format since it uses only one channel to represent colours. This is easy for computations and all colour components are considered in one feature detection. In Figure 3.60, it shows edge detection done in tea plantation and segmentations are identified using canny edge detection.

Canny edge detection is excellent for edge detection in images. But it is hard to process the result of canny edge detection to identify uneven segments that a tea plantation can contain and map them to a graph. Even though we can use Hough transform to detect lines and pre-defined shapes, it is not suitable for detection of unknown shapes.

Efficient graph-based image segmentation algorithm proposed by Felzenszwalb et al. uses the opposite approach that we initially took. Using this approach graph is first generated and then using that graph image is segmented. It was an efficient and fast technique that can be easily implemented and tested for our purpose.

3.4.4.2 Implementation

Algorithm for Efficient graph-based image segmentation is shown below. V in the algorithm is considered as pixels and E is some measure of dissimilarity between connecting two pixels (intensity difference is taken here). What this algorithm shown in Figure 3.61 does is selecting pixels with less than threshold intensity differences to segments in such a way that intensity difference between different components in the image is larger and within the same component, intensity difference is lower.

The input is a graph $G = (V, E)$, with n vertices and m edges. The output is a segmentation of V into components $S = (C_1, \ldots, C_r)$.

0. Sort E into $\pi = (o_1, \ldots, o_m)$, by non-decreasing edge weight.
1. Start with a segmentation S^0, where each vertex v_i is in its own component.
2. Repeat step 3 for $q = 1, \ldots, m$.
3. Construct S^q given S^{q-1} as follows. Let v_i and v_j denote the vertices connected by the q-th edge in the ordering, i.e., $o_q = (v_i, v_j)$. If v_i and v_j are in disjoint components of S^{q-1} and $w(o_q)$ is small compared to the internal difference of both those components, then merge the two components otherwise do nothing. More formally, let C_i^{q-1} be the component of S^{q-1} containing v_i and C_j^{q-1} the component containing v_j. If $C_i^{q-1} \neq C_j^{q-1}$ and $w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$ then S^q is obtained from S^{q-1} by merging C_i^{q-1} and C_j^{q-1}. Otherwise $S^q = S^{q-1}$.
4. Return $S = S^m$.

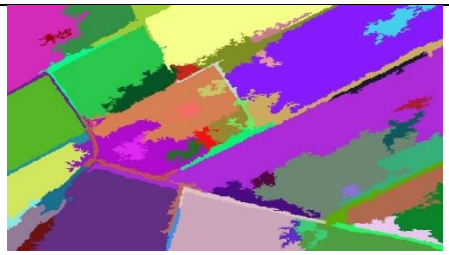
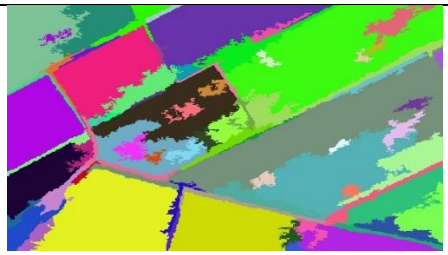
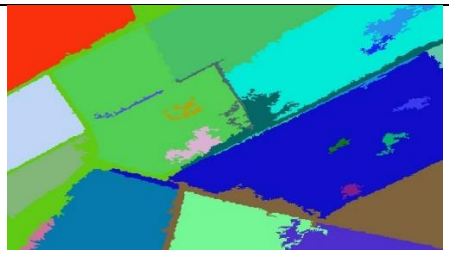
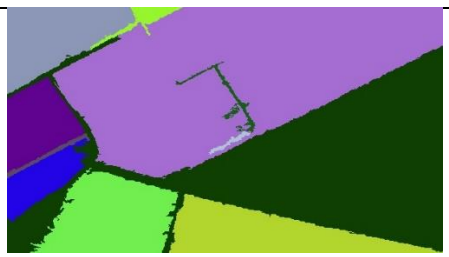
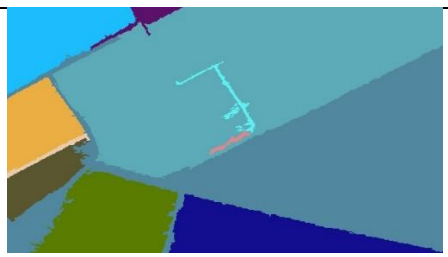
Figure 3.61 Algorithm for Efficient Graph-based segmentation

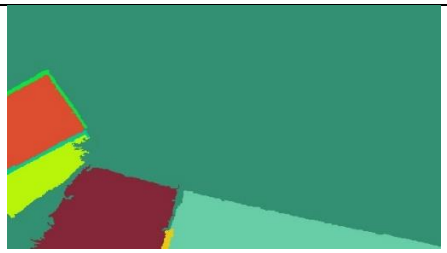
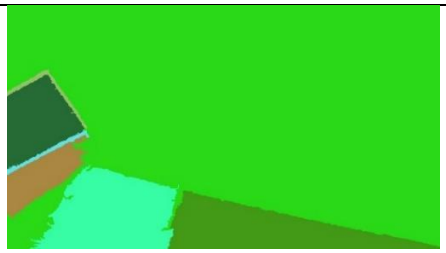
Since RGB images have three channels, this calculation is done for all three layers, and if all layers do not agree for a certain boundary, it is discarded from the result.

3.4.4.3 Results of Image Segmentation

Figure 3.62 Reference image used for graph generation

Table 3.17 Results obtained for different threshold values

Threshold	Min component = 200	Min component = 300
50		
100		
150		

200		

3.4.4.4 Summary

Image segmentation is a unique challenge that should be solved differently depending on the application. It is essential to represent image segments in a graph for mathematical representation. Graph structure can be used to do optimization tasks between each node even though we can use edge detectors to identify edges in images by reliable accuracy, converting that data to a graph is not accurate. Graph-based approaches for segmentation are the bottom-up approach for a given problem. Since we only require graph-based representation for our application, it is more efficient and accurate to use this approach. Possible improvement for the current implementation is to give segment labels to graph while generating. This can be used to optimize further algorithms used on this graph representation if we can name certain segments as roads, cultivation, rocks etc.

4 CONCLUSION

This research project proposes a novel method of optimizing available resources in tea cultivations. Four separate sections discussed above address different components of the overall system with in-depth analysis for their implementation. Almost all the systems discussed in the above sections were able to archive their expected outcome at the end of the research. All the prototype hardware developed during this project can be designed into more compact and robust production-ready embedded systems. FPGA based obstacle avoidance system is completely successful with generating identical results with other systems available in academia, utilizing a minimal amount of resources. Although Precision terrain-following model needs more improvements to its hardware, it highlighted the required implementation strategy for the specific feature. Using a domain-specific imaging mechanism can be used as a good foundation for segmentation-based analysis. Regarding the agriculture domain, NDVI is a prominent index to consider. If image data can be loaded to the processing FPGA platform in an efficient manner, high performance in calculation can be achieved. Evaluation of the tea leave image can be used in many ways to get meaningful insights. Such as yield estimation of the estate, search for anomalies of yield in the estate, fertilizer requirements of the estate. The proposed method for tea leave image evaluation has given near ground truth results surpassing available literature in that domain. To avoid the false segmentation of sunlight reflected leaves, research should continue further. SIFT features are calculated, and homography is calculated using RANSAC method in map generation before stitching each image with respect to the first image plane. Graph generation is achieved through a bottom-up approach by graph-based segmentation technique proposed by Felzenszwalb et. al.

We believe that after integrating each part, this system will create an inflection point in the tea industry, solving many burning problems that exist in the industry at present.

5 REFERENCES

[1] L. Jolliffe and M. S. Aslam, "Tea heritage tourism: evidence from Sri Lanka," *Journal of Heritage Tourism,* vol. 4, no. 4, pp. 331-344, 2009.

[2] A. Khalid, "Development of a Decision Support System to increase the Tea Crops yield," vol. 8, 2015.

[3] R. D. Baruah, R. M. Bhagat, S. Roy and L. N. Sethi, "Decision Support System for Climate Smartening of Tea Landscapes for Future Sustainability in North East India," *Indian Journal of Science and Technology,* pp. 1-8, 2017.

[4] B. McCullagh, "Real-time disparity map computation using the cell broadband engine," *Journal of Real-Time Image Processing,* vol. volume 7, p. 87–93, 2012.

[5] S. Larabi, N. Baha and H. Touzene, "FPGA implementation for stereo matching algorithm," in *2013 Science and Information Conference,* London, UK, 2013.

[6] D. Toratani, T. Higuchi and S. Ueno, "Terrain Following Flight of UAV using Information Amount Feedback," in *SICE Annual Conference 2013*, Nagoya, Japan, 2013.

[7] F. Fuentes-Peailillo, S. Ortega-Farías, M. Rivera, M. Bardeen and M. Moreno, "Comparison of vegetation indices acquired from RGB and Multispectral sensors placed on UAV," in *2018 IEEE International Conference on Automation/XXIII Congress of the Chilean Association of Automatic Control (ICA-ACCA)*, Concepcion, Chile, 2018.

[8] A. Cartelat, Z. Cerovic, Y. Goulas, S. Meyer, C. Lelarge and J. Prioul, "Optically assessed contents of leaf polyphenolics and chlorophyll as indicators of nitrogen deficiency in wheat (Triticum aestivum L.)," *Field Crops Research,* vol. 91, no. 1, pp. 35-49, 2005.

[9] C. Johnston, K. Gribbon and D. Bailey, "Implementing Image Processing Algorithms on FPGAs," in *Proceedings of the Eleventh Electronics New Zealand Conference, ENZCon'04*, Palmerston North, 2004.

[10] Y. Huang and F. Lee, "An automatic machine vision-guided grasping system for phalaenopsis tissue culture plantlets.," in *Comput Electron Agric*, 2010.

[11] M. Brochier, A. Vacavant, G. Cerutti, C. Kurtz, J. Weber and L. Tougne, “Tree leaves extraction in natural images: comparative study of preprocessing tools and segmentation methods.,” in *IEEE Trans Image Process*, 2015.

[12] X. Tang, M. Liu, H. Zhao and W. Tao, “Leaf extraction from complicated background.,” in *Proc CISP ’09 Proceedings of the 2009 International Congress on Image and Signal Processing.*, Tianjin, china, 2009.

[13] S. Peidi, W. Minghui, W. Xianwei, Z. Jun and L. Sheng, “Research on the tea bud recognition based on improved k means algorithm,” in *MATEC Web of Conferences*, 2018.

[14] N. Xiaojing, “Image Segmentation Algorithm for Disease Detection of Wheat Leaves,” in *IEEE Proceedings of the 2014 International Conference on Advanced Mechatronic Systems*, Japan, 2014.

[15] M. Brown and D. Lowe, “Automatic Panoramic Image Stitching using Invariant Features,” *International Journal of Computer Vision,* vol. 74, pp. 59-73, 2007.

[16] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision,* vol. v.60, no. n.2, pp. p.91-110, 2004.

[17] P. Felzenszwalb and D. Huttenlocher, “Efficient Graph-Based Image Segmentation,” *International Journal of Computer Vision,* vol. 59, pp. 167-181, 2004.

[18] Z. Ze, L. Minzan and L. Xin, “The development of a hand-held multi-spectral camera based on FPGA,” in *2010 World Automation Congress,* Kobe, Japan, 2010.

[19] H. Oleynikova, D. Honegger and M. Pollefeys, “Reactive avoidance using embedded stereo vision for MAV flight,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA, 2015.

[20] L. Matthies, R. Brockers, Y. Kuwata and S. Weiss, “Stereo vision-based obstacle avoidance for micro air vehicles using disparity space,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, 2014.

[21] S. Kumar, “Binocular Stereo Vision Based Obstacle Avoidance Algorithm for Autonomous Mobile Robots,” in *2009 IEEE International Advance Computing Conference,* Patiala, India, 2009.

[22] J. Yi, J. Kim, L. Li, J. Morris, G. Lee and P. Leclercq, “Real-Time three dimensional vision,” *Lect. Notes Comput. Sci.,* vol. 3189, pp. 309-320, 2004.

[23] Y. Maruyama and T. Miyajima, “A Real-Time Stereo Vision System with FPGA,” *Lect. Notes Computer Science 2778,* pp. 448-457, 2003.

[24] B. Khaleghi, S. Ahuja and J. Q. M. Wu, “A New Miniaturized Embedded Stereo-Vision System (MESVS_I),” *”In proceedings of Canadian Conference on Computer and Robot Vision,* pp. 26-33, 2008.

[25] D. Scharstein and R. Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” *Int. Journal on Computer Vision,* vol. 47, p. 7–42, 2002.

[26] R. Yang and M. Pollefeys, “Multi-resolution real-time stereo on commodity graphics hardware,” in *proc. Conf.Comput.Vis. Pattern Recognt,* 2003.

[27] T. Maruyama and H. Niitsuma, “Real-time detection of moving objects,” *Lect. Notes Comput.Sci,* vol. 3203, p. 1155–1157, 2004.

[28] D. Hwang and D. Han, “A novel stereo matching method for wide disparity range detection,” *Lect. Notes Comput. Sci,* vol. 3656, p. 643–650, 2005.

[29] J. I. Woodill, R. Buck, D. Jurasek, G. Gordon and T. Brown, “3D Vision: Developing an Embedded Stereo-Vision System,” *IEEE Computer 40(5),* pp. 106-108, 2007.

[30] S. Hadjitheophanous, C. Ttofis, A. Georghides and Theocharides.T, “Towads Hardware Stereoscopique 3D Reconstitution: A Real time FPGA Computation of the Disparity Map,” *Design automation & Test in Europe Conference & Exhibition, IEEE,* pp. 1743-1748, 2010.

[31] K. Ambrosch and W. Kubinger, “Accurate hardware-based stereo vision,” *Computer Vision and Image Understanding journal,* vol. 114, pp. 1303-1316, 2010.

[32] S. Perri, D. Colonna, P. Zicari and P. Corsonello, “SAD-based stereo matching circuit for FPGAs,” in *Proceedings of the 13th IEEE International Conference on Electronics, Circuits and Systems,* Nice, France, 2006.

[33] C. Murphy, D. Lindquist, A. Rynning, T. Cecil, S. Leavitt and M. Chang, “Low Cost Stereo Vision on an FPGA,” in *15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM 2007),* Napa, CA, USA, 2007.

[34] D. Masrani and W. MacLean, “A Real-Time Large Disparity Range Stereo-System using FPGAs,” in *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06),* New York, NY, USA, USA, 2006.

[35] A. Naoulou, J. Boizard, J. Fourniols and M. Devy, “An alternative to sequential architectures to improve the processing time of passive stereovision algorithms,” in *Proceedings of the International Conference on Field Programmable Logic and Applications*, Madrid, Spain, 2006.

[36] L. Zhang, K. Zhang, T. Chang, G. Lafruit, G. Kuzmanov and D. Verkest, “Real-time high-definition stereo matching on fpga,” *Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays*, p. 55–64, 2011.

[37] B. Fisher, “Epipolar Geometry,” 16 4 1997. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/EPSRC_SSAZ/node18.html. [Accessed 14 7 2019].

[38] R. Samar and A. Rehman, “Autonomous terrain-following for unmanned air vehicles,” *Mechatronics*, vol. 21, no. 5, p. 844–860, 2011.

[39] A. Kosari, H. Maghsoudi, A. Lavaei and R. Ahmadi, “Optimal online trajectory generation for a flying robot for terrain following purposes using neural network,” *Proceedings of the Institution of Mechanical Engineers, Part G*, vol. 229, no. 6, p. 1124–1141, 2015.

[40] S. J. Asseo, “Terrain following/terrain avoidance path optimization using the method of steepest descent,”,” in *Aerospace and Electronics Conference*, NAECON, 1998.

[41] T. Templeton, D. H. Shim, C. Geyer and S. S. Sastry, “Autonomous vision-based landing and terrain mapping using an MPC-controlled unmanned rotorcraft,” *Robotics and automation, 2007 IEEE international conference*, p. 1349–1356, 2007.

[42] F. Ruffier and N. Franceschini, “Visually guided micro-aerial vehicle: automatic take off, terrain following, landing and wind reaction,” *Robotics and Automation ICRA’04*, vol. 3, p. 2339–2346, 2004.

[43] B. Herisse, T. Hamel, R. Mahony and F. Russotto, “A terrain-following control approach for a VTOL Unmanned Aerial Vehicle using average optical flow,” *Autonomous robots*, vol. 29, no. 3, 2010.

[44] Y. Lin, J. Hyyppa and A. Jaakkola, “Mini-uav-borne lidar for fine-scale mapping,” *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 3, p. 426–430, 2011.

[45] C. Carreon-Limones, A. Rashid, P. Chung and S. Bhandari, “3-d mapping using lidar and autonomous unmanned aerial vehicle,” *AIAA Information Systems-AIAA Infotech@ Aerospace*, p. 1155, 2017.

[46] M. Clark and R. C. Roberts, “Autonomous Quadrotor Terrain-Following with a Laser Rangefinder and Gimbal System,” in *2017 IEEE SENSORS*, Glasgow, UK, 2017.

[47] N. A. AlQahtani, B. J. Emran and H. Najjaran, “Adaptive Motion Planning for Terrain Following Quadrotors,” in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Banff, Canada, 2017.

[48] I. Khademi, B. Maleki and A. N. Mood, “Optimal Three Dimensional Terrain Following/Terrain Avoidance for Aircraft Using Direct Transcription Method,” in *19th Mediterranean Conference on Control and Automation*, Corfu, Greece, 2011.

[49] V. Dworak, J. Selbeck, K.-H. Dammer, M. Hoffmann, A. Zarezadeh and C. Bobda, “Strategy for the Development of a Smart NDVI Camera System for Outdoor Plant Detection and Agricultural Embedded Systems,” *Sensors*, vol. 13, no. 2, pp. 1523-1538, 2013.

[50] K.-H. Dammer, H. Thöle, T. Volk and B. Hau, “Variable-rate fungicide spraying in real time by combining a plant cover sensor and a decision support system,” *Precision Agriculture,* vol. 10, no. 5, pp. 431-442, 2008.

[51] J. Selbeck, V. Dworak and D. Ehlert, “Testing a vehicle-based scanning lidar sensor for crop detection,” *Canadian Journal of Remote Sensing,* vol. 36, no. 1, pp. 24-35, 2010.

[52] G. Rabatel, N. Gorretta and S. Labbé, “Getting NDVI Spectral Bands from a Single Standard RGB Digital Camera: A Methodological Approach,” *Advances in Artificial Intelligence,* pp. 333-342, 2011.

[53] W. Lee and S. W. Searcy, “Multispectral sensor for detecting nitrogen in crop plants,” in *ASAE Annual International Meeting*, Wisconsin, 2000.

[54] A. Dawood, S. Visser and J. Williams, “Reconfigurable FPGAs for real time image processing in space,” in *14th International Conference on Digital Signal Processing Proceedings.*, Santorini, Greece, 2002.

[55] I. Chiuchisan, M. Cerlinca, A.-D. Potorac and A. Graur, “Image Enhancement Methods Approach using,” in *11th International Conference on DEVELOPMENT AND APPLICATION SYSTEMS*, Suceava, Romania, 2012.

[56] W. Shen, C. Zhang and Z. Chen, “Research on automatic counting soybean leaf aphids system based on computer vision technology.,” in *ICMLC ‘07 Proceedings of the 2007 International Conference on Machine Learning and Cybernetics.*, Hong Kong, China, 2007, p. 35–38.

[57] G. Cerutti, L. Tougne, A. Vacavant and D. Coquin, “A parametric active polygon for leaf segmentation and shape estimation.,” in *ISVC ‘11 Proceedings of the 2011 International Symposium on Visual Computing.*, Las Vegas, USA, 2011., p. 202–13.

[58] A. Janwale, “Plant Leaves Image Segmentation Techniques: A Review.,” *INTERNATIONAL JOURNAL OF COMPUTER SCIENCES AND ENGINEERING.*, vol. 5, pp. 147-150, 2017.

[59] G. Wenshuo, Z. Xiaoguang, Y. Lei and L. Huizhong, “An improved Sobel edge detection,” in *3rd International Conference on Computer Science and Information Technology*, Chengdu, 2010.

[60] W. Rong, Z. Li, W. Zhang and L. Sun, “An improved Canny edge detection algorithm,” in *IEEE International Conference on Mechatronics and Automation*, Tianjin, 2014.

[61] Vala, J. Hetal and B. Astha, “A review on Otsu image segmentation algorithm,” *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)2*, pp. 387-389, 2013.

[62] L. Wen-Nung, “Automatic target segmentation by locally adaptive image thresholding,,” *IEEE Transactions on Image Processing*, vol. 4, pp. 1036-1041, 1995.

[63] J. C. Tilton, “Image segmentation by iterative parallel region growing and splitting.,” in *NASA Goddard Space Flight Center*, 1989.

[64] H. P. Ng, S. H. Ong, K. W. C. Foong, P. S. Goh and W. L. Nowinski, “Medical Image Segmentation Using K-Means Clustering and Improved Watershed Algorithm,,” in *IEEE Southwest Symposium on Image Analysis and Interpretation*, Denver, 2006.

[65] A. Darshana, M. Jharna and A. Shilpa, “Segmentation Method for Automatic Leaf Disease Detection,” *IJIRCCE*, vol. 3, pp. 1-7, 2015.

[66] P. Kumar and S. Domnic, “Image based leaf segmentation and counting in rosette plants,” *INFORMATION PROCESSING IN AGRICULTURE*, 2018.

[67] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *International Conference on Computer Vision*, Barcelona, 2011.

[68] Denisova, Anna & Kuznetsov, Andrey & Glumov and Nikolay, “Multichannel Image Segmentation Algorithm for Agricultural Crop Detection,” in *6th International Conference on Analysis of Images, Social Networks and Texts (AIST-2017)*, Moscow, 2018.

[69] Shi, Jianbo & Malik and Jitendra, “Normalized Cuts and Image Segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* no. 22, 2002.

[70] Arbelaez, Pablo & Maire, Michael & Fowlkes, Charless & Malik and Jitendra, “Contour Detection and Hierarchical Image Segmentation,” *IEEE transactions on pattern analysis and machine intelligence,* vol. 33, pp. 898-916, 2011.

[71] L. Grady, “Random walks for image segmentation,” *Pattern Anal Mach Intell IEEE Transact,* vol. 28, 2006.

[72] S. &. L. Y. A. Zhu, “Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation,” *Pattern Analysis and Machine Intelligence, IEEE Transactions,* vol. 18, pp. 884 - 900, 1996.

[73] Y. &. M. B. Deng, “Unsupervised Segmentation of Color-Texture Regions in Images and video,” *Pattern Analysis and Machine Intelligence,* vol. 13, pp. 800-810, 2001.

[74] K. Takaya and Z. Qian, “FPGA Based Stereo Vision System to Display Disparity Map in Realtime,” in *2012 International Conference on Information Science and Applications,* Suwon, South Korea, 2012.

[75] Y. Chene, D. Rousseau, P. Lucidarme, J. Bertheloot, V. Caffier, l. P. More and a. et, “On the use of depth camera for 3D phenotyping of entire plants,” in *Comput Electron Agric*, 2012.

[76] X. Yin, X. Liu, J. Chen and D. Kramer, “Multi-leaf tracking from fluorescence plant videos.,” in *Proc ICIP ’14 Proceedings of the 2014 IEEE international conference on image processing*, Paris, France, 2014.

[77] B. Dellen, H. Scharr and C. Torras, “Growth signatures of rosette plants from time-lapse video.,” in *IEEE/ACM Trans Comput Biol Bioinf*, 2015.

[78] J. DeVylder, D. Ochoa, W. Philips, L. Chaerle and D. VanDerStraeten, “Leaf segmentation and tracking using probabilistic parametric active contours.,” in *MIRAGE ’11 Proceedings of the 2011 International Conference on Computer Vision/ Computer Graphics Collaboration Techniques*, Rocquencourt, France, 2011, p. 75–85.