Design and Development of Multispectral Camera

For Plant Health Monitoring-

Designed for Aerial Vehicles

Authors

Gokul Edakke Puram- 22541618

Manal Ajit Vartak- 29241614

Pravin Tulshidas Palve- 13541620

Students of Master Mechatronics

Ravensburg Weingarten University of Applied Science

Under Mentorship of

Prof. Dr. Jörg Eberhardt

Scientific Project for Summer Semester-2024

Index

1. Abstract

This project aims at the design and development of a multispectral camera setup for plant health monitoring. The setup is designed to be mounted on airborne crafts like airplanes and UAVs. The project also demonstrates how advancements in computer vision and image processing can be used for modern agriculture and environmental monitoring. The physical setup, optics, camera calibration, image registration, image fusion, and interpretation are thoroughly discussed in the report.

Daylight images of vegetation in urban areas were collected, and the final results discuss the reflective behaviour of plants in different electromagnetic radiation spectrums and the significance of this behaviour.

2. Introduction

Modern agriculture is being equipped with cutting-edge technologies to yield more product, monitor crop health, and make efficient use of resources. Plants communicate their health through a broader range of wavelengths than what our eyes can perceive. This has been repeatedly demonstrated through multispectral imaging (MSI), which is one of the new trends in modern agricultural methods. MSI is not entirely novel; it is being used since long time by Military, Medical, Remote Sensing, and Astronomical applications. Many remote sensing satellite systems, like the Copernicus Global Land service, employ MSI to monitor various aspects of vegetation, including quality and quantity. It has contributed to developments in earth science, space, and medical fields.

There are numerous commercial off-the-shelf (COTS) MSI camera systems available with significant resolution. A few of them are the Parrot Sequoia Multispectral Camera [1], DJI Enterprise Multicopter Phantom 4 Multispectral [2], and Sentera 6x Multispectral [3]. Our goal in this project is to develop a setup similar to the Parrot Sequoia. The whole process of optics configuration, filter selections, hardware design, camera calibration, and image registration will be thoroughly discussed in this report.

3. Literature Review

As hinted earlier, MSI camera setups are already developed and commercially available with sophisticated software to post-process and interpret the collected data. Many of these also have solar irradiance sensors that measure incoming solar irradiance, allowing recordings taken under different illumination conditions to be normalized for comparison.

For image registration, many studies have implemented calibration techniques involving homography matrix calibration using Scale-Invariant Feature Transform (SIFT) and Random Sample Consensus (RANSAC) [4]. Other studies have gone further, detecting crop or fruit types using MSI and Support Vector Machines (SVM) [5]. The big challenge of multispectral image registration is that the characteristics of the paired images are inconsistent in intensity, meaning some features are not sufficiently visible in all spectrums. This inconsistency makes it difficult to register images using feature matching methods such as SIFT. One study adopted gradient information from the reference image (RGB) for registration to bridge the gap between multispectral paired images and facilitate network learning with training data generation [6].

4. Physical Setup

The whole process of development of this MSI setup can be grouped into Hardware, Optics, Image registration and Image fusion.

4.1 Hardware

As the intended use of the setup is for airborne crafts like UAVs or airplanes, size and weight play important roles in hardware selection. Additionally, for easy prototype development, we needed something standard with strong community and accessory support. It was also important to have a high-performance processor, memory, and integrated GPU for image processing and data handling, so a Raspberry Pi 4 was selected. With 4 GB of RAM, it is also very useful for capturing and storing multiple images in real-time and sending them to other devices when required.

For data storage and loading the operating system, we used a standard off-the-shelf high-speed 64 GB SD card.

4.2 Camera

As we want to measure specific spectra of reflected sunlight with each camera, a simple monochromatic sensor is sufficient. These camera modules are specially designed to be integrated

Parameter	Typical Value
Sensor	Monochrome global shutter OV9281
Pixel Size	3 µm x 3 µm
Active array size	1280 x 800
Optical Size	1/4 inch
Default Lens	M27280M07S
EFL	2.8 mm
F.NO	2.8
FOV	75º Horizontal
Build-in IR cut filter	None
Focusing Range	30cm ~ infinite (manually adjustable)
Output interface	2-lane MIPI serial output
Output formats	8/10-bit BW RAW
Maximum image transfer rate	1280 x 800@120 fps
Board Size	40mm x 40mm

Figure 1- Camera Module Specifications

with the Raspberry Pi. The detailed specifications of the camera module, along with its lens, are shown in Figures 1 and 2. We needed four of such cameras to capture spectral images across four different wavelength ranges. The lenses used are standard attachments for the previously mentioned camera modules, with specifications listed below.

Optical Format	1/2.7"
EFL(mm)	2.8
35mm EFL	30.27
BFL(mm)	2.6
Construction	2G2P+IR
F/NO	2.8
FOV (D/H/V) @ 1/2.7"	125/110/85

Lens Holder Height	7mm
IR filter	650 IR filter
Mount	M12
Working Wavelength	400-700nm
MOD	0.3m
Size(mm)	14 x 15.6
Weight	4g
HFOV on 1/4" RPi Cam	75°

Figure 2- Lens Specifications

The M12 lens specifications mention that it blocks 650 nm IR light. This would have been a problem for our cameras, as we are interested in the wavelength range from 510 nm to 840 nm. Before buying new lenses, we tested the M12 lens on a spectroscope and found almost no effect on the transmissivity of light at all (refer to Figure 3). This means the IR filter mentioned in the specifications is not actually present. This was also proven later in the tests when we were able to capture images with a 650 nm bandpass filter on. This saved us from buying new lenses. The transmissivity graph of the lens against the solar spectrum is shown in Figure 3. (Measured with spectroscope from JETI Technische Instrumente GmbH).

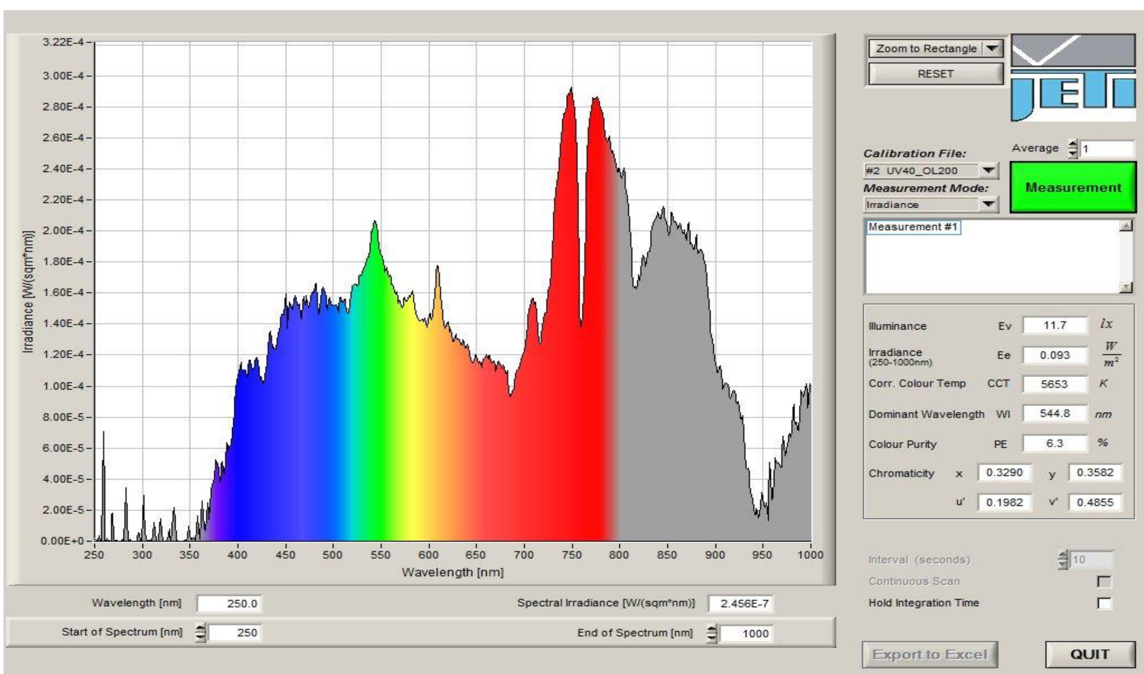

Figure 3- Transmissivity of Camera

4.3 Filters

The filter selection was primarily based on the features of the Parrot Sequoia, as the goal of the project is to design a similar setup. However, we did not find exact matches for the filters. Instead, we chose filters with similar specifications from Edmund Optics. The table below shows a comparison of the onboard filters of the Parrot Sequoia versus those we implemented in our setup.

Spectrum	Filters on Parrot Seqouia	Filters on our setup
Green	550nm CWL, 40nm bandwidth	550nm CWL, 80nm bandwidth
Red	660nm CWL, 40nm bandwidth	650nm CWL, 80nm bandwidth
Red Edge	735nm CWL, 10nm bandwidth	730nm CWL, 10nm bandwidth
Near Infra-Red	790nm CWL, 40nm bandwidth	800nm CWL, 40nm bandwidth

The filters were tested on the transmissivity spectrometer and Fig. 4 shows percentage of incident light of a specific wavelength transmitted through the filters. The significance of these wavelengths will be discussed later in the sensor fusion section. From the graph we can observe that the transmissivity of filter matches pretty accurately with the Specifications.

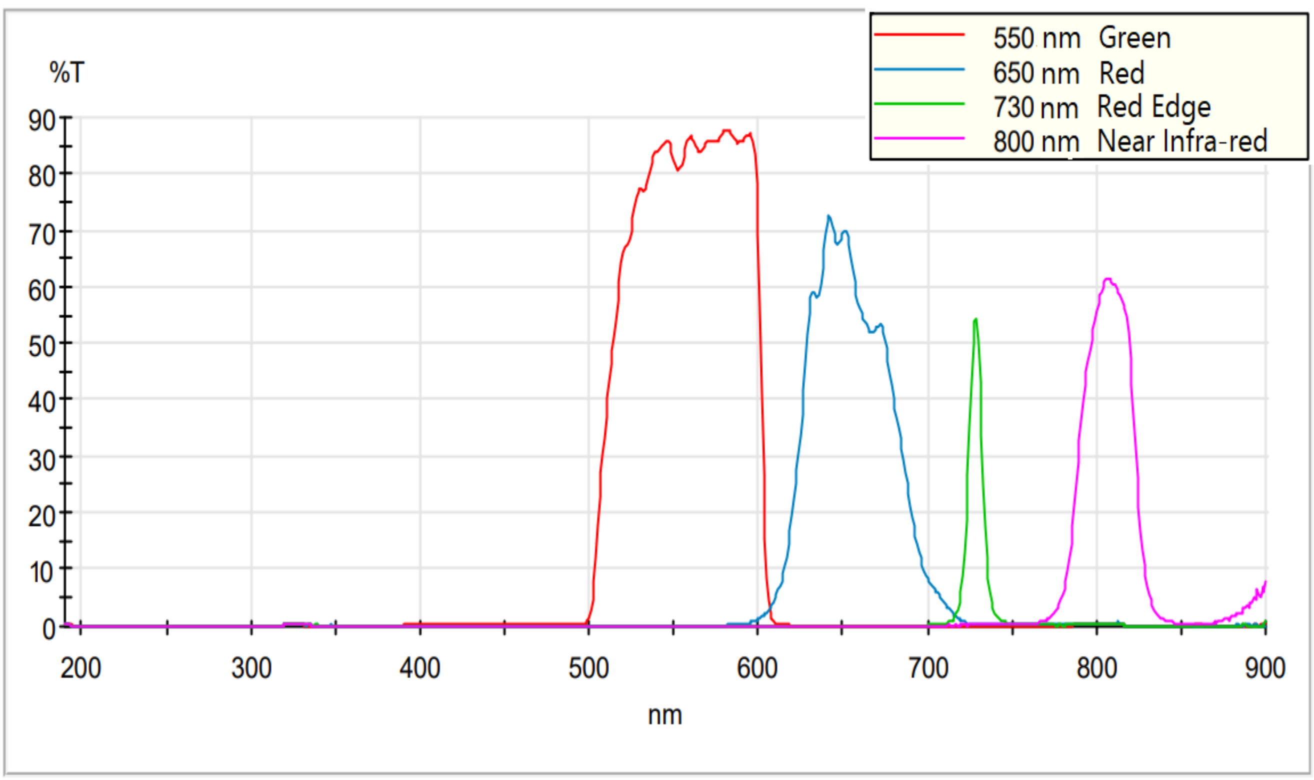

Figure 4- Transmissivity of Filters

Custom adapters were needed to mount these filters on the camera lenses. After mounting the filters on the lenses and taking some pictures, we found that the filters were blocking part of the cameras' field of view. Buying new filters was not an option, as the next available size was too large (25 mm). A sample image showing the filter partially obstructing the camera's field of view is shown in Figure 5.

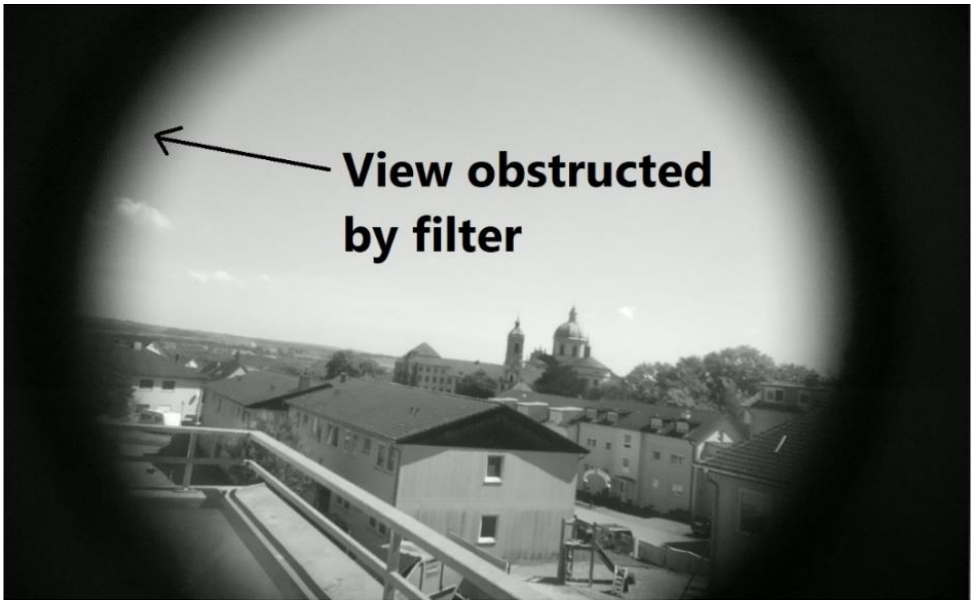

Figure 5- View obstructed by filter

The arrangement of optical components is shown in figure 6.

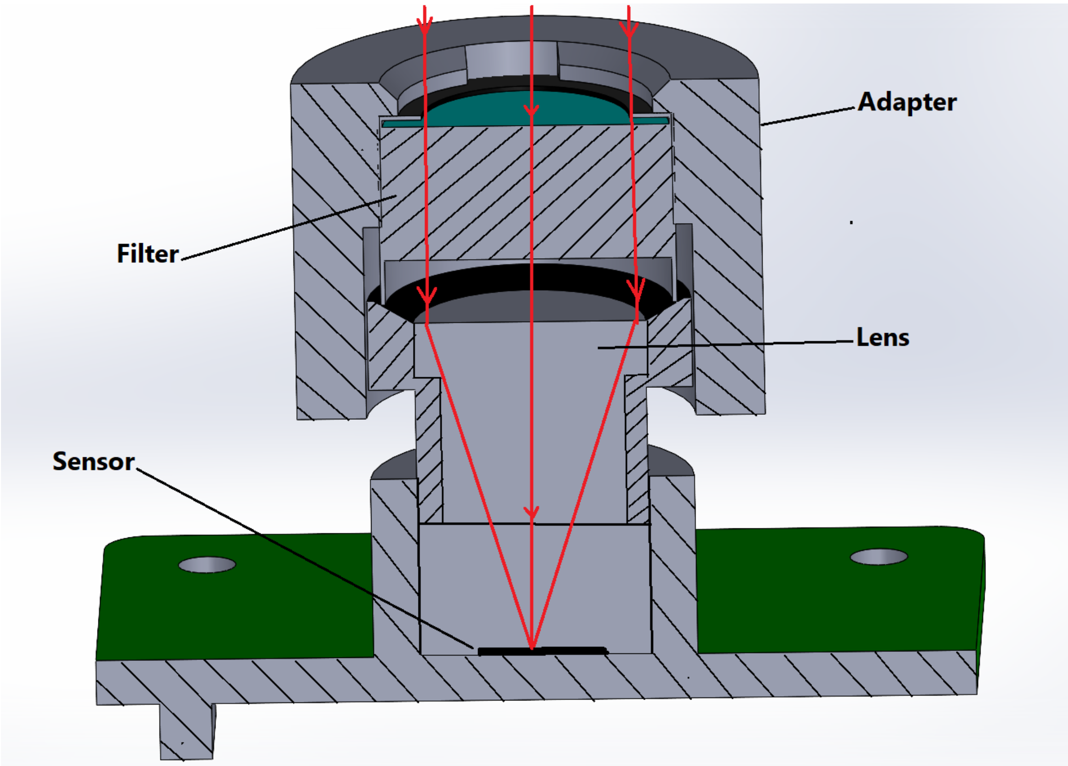

Figure 6- Optical Component Arrangement

4.4 Camera Multiplexor hat

There are 4 cameras but only one 2-lane MIPI CSI camera port on the Raspberry Pi for camera attachment. This is where the Camarray Multiplexer Hat comes in handy. This hat combines the feed from all 4 cameras side by side. The final combined feed is then sent to the Raspberry Pi. This also allows capturing all 4 camera feeds simultaneously, as we only need to capture the feed from the Multiplexer, which includes all camera feeds. This process is illustrated in Figure 7 below.

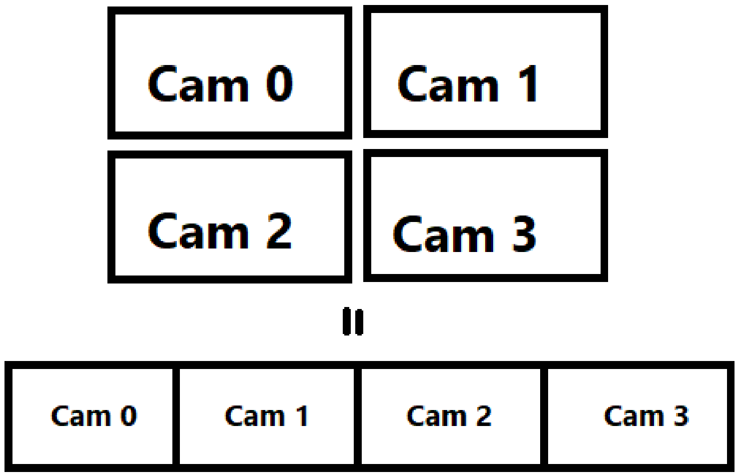

Figure 7- Multiplexor Function

However, the orientation of these cameras is not uniform due to physical limitations of the input ports on the camera Multiplexer Hat. While installing the cameras on the fixture, cable arrangement was prioritized over uniform orientation, as the images can be reoriented as needed through software post-processing. Below is an example of an image received from the camera Multiplexer Hat for reference.

Figure 8- Raw image from Camarray Hat

The physical arrangement of the Camera module, lens, filter and adapter can be realized through Figure 6.

4.5 Power Source

The plan was to use a UPS Hat designed especially for the Raspberry Pi (PiJuice). Unfortunately, due to lack of availability, we couldn't incorporate it in our setup. For now, we are powering the setup simply with a wall adapter, Power bank or laptop.

4.6 Housing

Enclosing all these components in a compact and lightweight way is also important to ensure suitability for aerial applications. The whole process is divided into the following three steps:

1. **Create a digital twin of physical components**: The first step was to create an exact 3D replica of all the hardware components being used. We found an accurate 3D model of the Raspberry Pi 4 on the 3D model-sharing platform GrabCAD. For the remaining components, such as camera modules, the multiplexer hat, filters, and lenses, we had to measure them physically and design parts with similar dimensions.
2. **Assembly in modelling software**: Once all components were modelled, we tried assembling them in different orientations to minimize space occupancy. The housing was then developed around this assembly to securely contain all components while also providing structural strength. An exploded view of the whole assembly can be seen in Figure 10.
3. **Prototyping**: The final step was to print these housing components with a 3D printer and check the fitting of each component. A few iterations were needed to ensure all components fit together properly. For the setup to work correctly, it was crucial that all cameras were rigidly mounted in their seats with no relative motion concerning each other. Figure 9 shows the actual final assembly.

All parts were modelled in SolidWorks and 3D printed with PETG using Prusa MK4 and XL.

Figure 9- Full Assembly

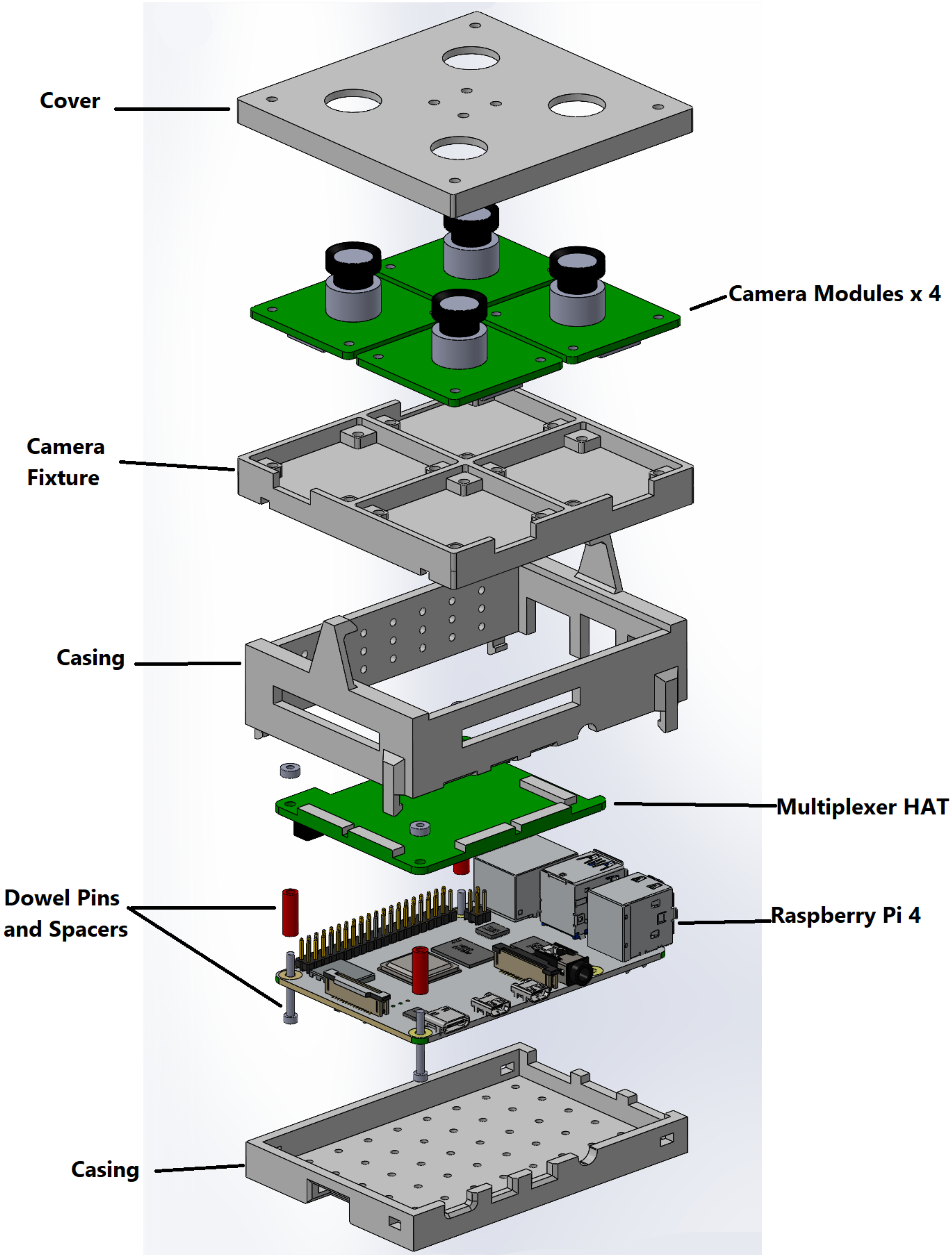

Figure 10 – Exploded view of the Setup

4.7 Preparing Raspberry Pi

To interact with the data received from cameras, we needed an Operating System on Rpi. We used a Raspberry Pi OS 64 bit which has great community support and documentation. Camera drivers were installed following the official documentation of the camera modules used from ArduCam. Python 3.11 was installed with libraries such as OpenCV and Numpy for image and data processing.

4.8 Accessing Image Data from Rpi

The images were captured using the *libcamera* driver installed and setup in the Rpi. *libcamera* is a software library aimed at supporting complex camera systems directly from the Linux operating system. In the case of the Raspberry Pi it enables us to drive the camera system directly from open source code running on ARM processors.

As there are no physical trigger buttons on the setup like a normal camera to capture images, we used Pitunnel to access the Rpi and then remotely executed scripts through bash to capture images. The downside of this is that we always needed the Rpi to be connected to the internet.

The captured images are then transferred from the Rpi to the local system (in this case, our personal Laptop) where we process the images further using SFTP protocol. A SSH connection was established through internet to the Rpi from our local system via *Pitunnel*. It is a free to use (with some restrictions) online tool that reserves a communication tunnel through internet between the Rpi and any system logged into the *Pitunnel* account where the device is added. More information is available on its official website.

4.9 Intrinsic Camera Calibration

Now that we have captured and saved the images on our local system, it's time to prepare the images for the further processing. The first step would be to slice the source image into 4 separate images. This was done by slicing the image array using numpy array slicing operation. The calibration was done to find out the intrinsic camera matrix. This is a standard process for any kind computer vision applications as it describes how the 3D world is mapped on the 2D camera coordinate. We used a chequered board to check and correct any distortions present in the image. Standard OpenCV libraries are available for this task which find the closest approximation to correct any distortion available in the captured image. Refer Figure 11 for the reference.

Figure 11- Images of Chequered board for calibration.

4.10 Extrinsic Camera Calibration

Extrinsic calibration is used to determine the position and orientation (pose) of a camera relative to another sensor. It establishes how the camera is positioned in a global reference frame or relative to master Camera. In our case, camera with 550 nm (Green) bandpass filter is selected as our master coordinate system. Extrinsic calibration often uses a known pattern or object (in our case, Chequered board) with known dimensions. Multiple images of this pattern are captured from different angles.

The orientation of cameras is not same and limited by physical placement of camera modules as explained in figure 11. So, we had to rotate the feed from other cameras by multiples of 90 degrees to align with the master frame.

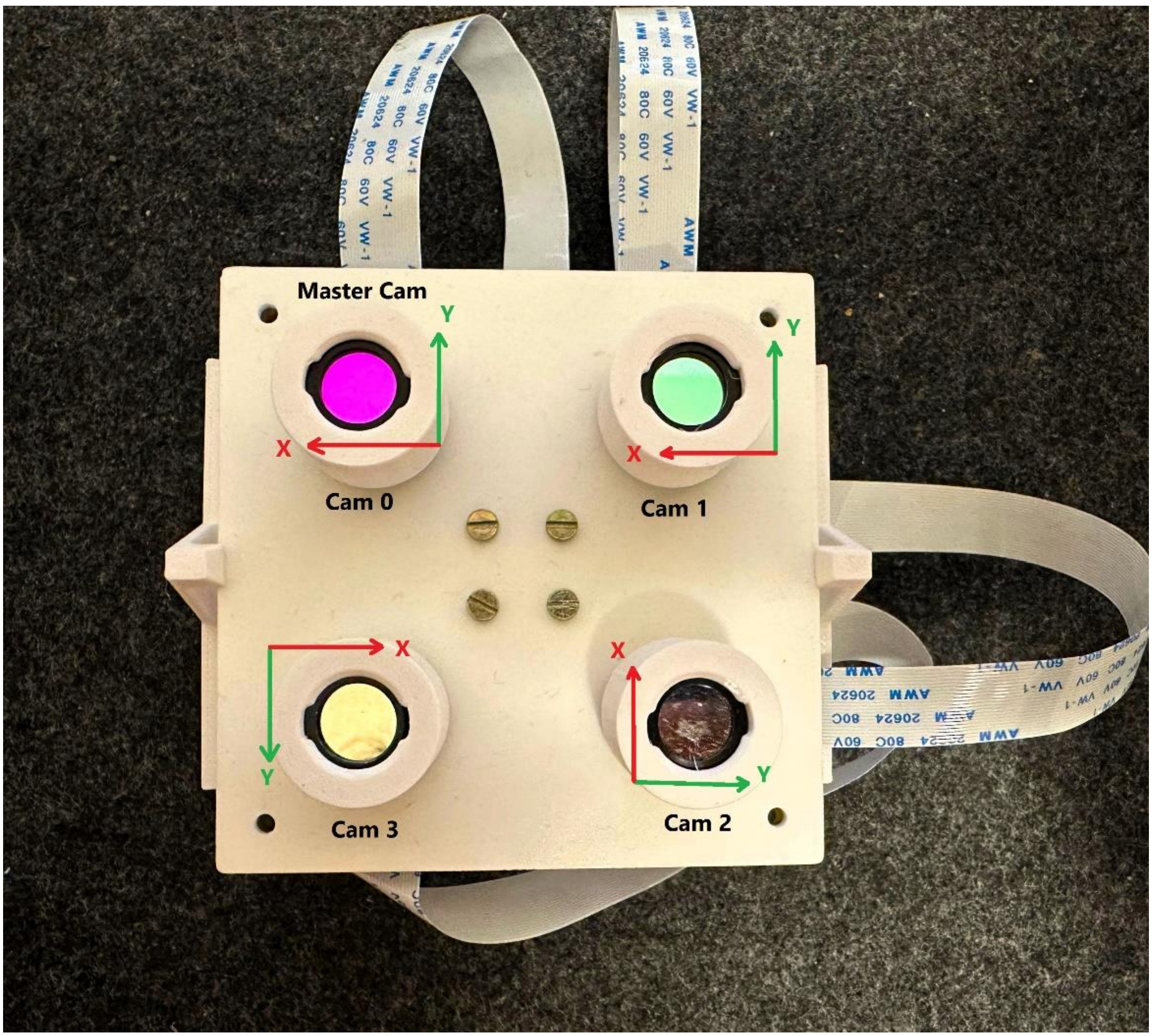

Figure 11- Camera Orientation

4.11 Image Registration

Image registration in the context of multispectral cameras refers to the process of aligning multiple images taken at different wavelengths or spectral bands so that they perfectly overlay with one another. This alignment is crucial because each image captures different information about the same scene, and combining this information accurately is necessary for effective analysis. The ideal process of image registration includes following steps:

- **Feature Detection**: Identifiable features (like edges, corners, or distinct points) are detected in each image.
- **Feature Matching**: The detected features are matched across the different spectral images to establish corresponding points.
- **Transformation Estimation**: A mathematical model is used to estimate the transformation (e.g., translation, rotation, scaling) needed to align the images based on the matched features.
- **Image Resampling and Alignment**: The images are then resampled (adjusted) according to the estimated transformation to achieve alignment. The output is a set of aligned images where each pixel across different bands corresponds to the same point in the scene.

It is important to mention that the images for image registration were taken before mounting the filters. If the images would have been captured with filters on, some features would not have appeared consistently across images. This inconsistency makes it challenging for RANSAC to find matching features and align images, which is why multispectral images often struggle with alignment.

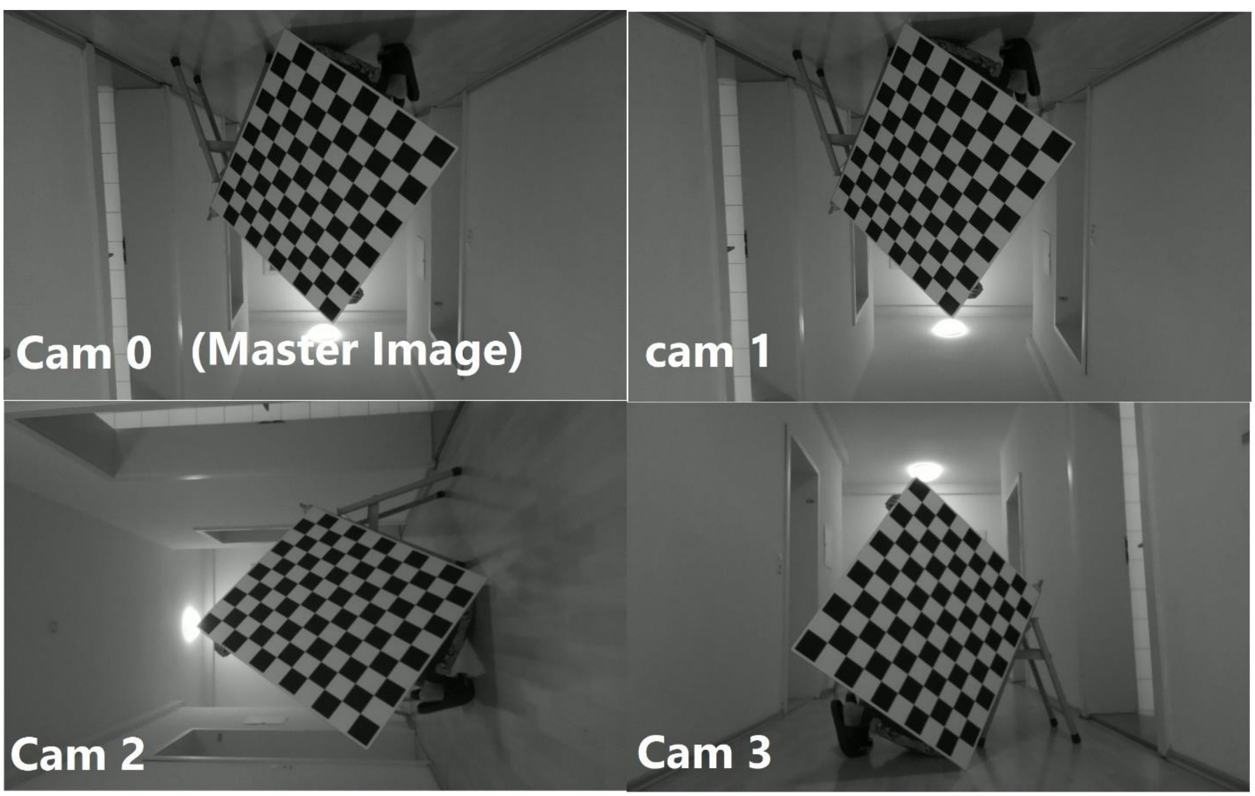

Figure 12-Input Images

Figure 13 below shows the output images after registration. The black regions around edges tell us that the images were translated and rotated to match the features.

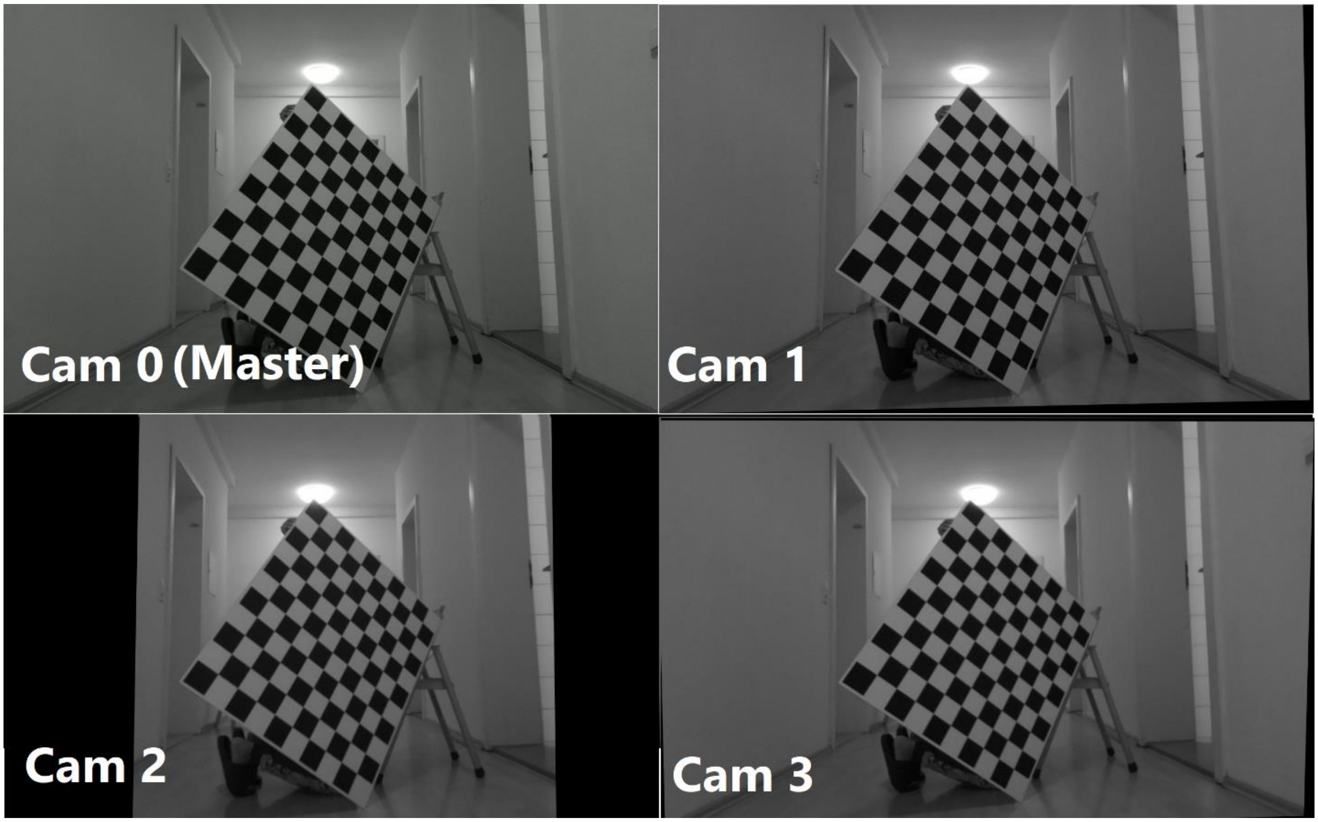

Figure 13- Output Images

The registration matrix is generated by ORB (Oriented FAST and Rotated BRIEF) function in OpenCV. The matrix is generated after matching features with brute force matching. An example of such matrix can be seen below along with the significance of each entry.

$$H= \begin{bmatrix} 9.91779902e-01 & 5.39118083e-02 & -7.74326368e+00 \\ -2.10139637e-02 & 1.04986360e+00 & 1.59618108e+01 \\ -3.40665248e-05 & 9.11349880e-05 & 1.00000000e+00 \end{bmatrix}$$

Where,

$$H= \begin{bmatrix} h11(scaling\ and\ rotation) & h12(shearing\ and\ rotation) & h13(translation\ in\ x)\ in\ pixels \\ h21(shearing\ and\ rotation) & h22(scaling\ and\ rotation) & h23(translation\ in\ y)\ in\ pixels \\ h31(perspective) & h32(perspective) & h33(perspective\ scaling) \end{bmatrix}$$

4.12 Image fusion

Now that all the images are aligned, we need to figure out how to extract meaningful information from the available multispectral images. The 4 spectral reflectance maps are fused together using low level channel fusion method into 4 separate channels after the image registration. The different channels are then extracted to calculate the Vegetative indices.

There are more than 30 different indices which describe Vegetation cover and health [7]. Some of the commonly used ones for crop health monitoring are NDVI (Normalized Difference Vegetation Index), GNDVI (Green Normalized Difference Vegetation Index), NDRE (Normalized Difference Red Edge). All of these mentioned indices can be easily calculated with available spectral bands of our setup.

Each spectral band corresponds to unique molecular interactions that reveal specific compounds, like pigments, and structures, such as cellulose fibers. Furthermore, these bands are believed to be linked with the photosynthetic elements of leaves, including proteins, carbohydrates, structural lipids, and energy-producing organelles like mitochondria and chloroplasts [3]. Let us briefly discuss the main reason of choosing a specific wavelength for crop health monitoring.

Green- 550 nm

- Plants get their green colour from Chlorophyl. Generally, a plant with higher chlorophyll content is healthier because it can efficiently capture sunlight for photosynthesis. This leads to better growth, higher energy production, and overall vitality.
- When plants are under stress (drought, pest attack, nutrient deficiency), chlorophyll production often declines. This reduction is a protective mechanism, conserving resources when photosynthesis might not be as efficient or necessary.
- Monitoring chlorophyll content helps farmers make informed decisions about fertilization, irrigation, and pest control, optimizing crop yields and health. [3]

Red 650 nm

- Chlorophyll, the pigment crucial for photosynthesis, strongly absorbs light in the red region (around 650 nm) of the spectrum. Healthy plants with high chlorophyll content will absorb most of the red light, resulting in low reflectance at 650 nm.
- NDVI, a common index used in remote sensing, relies on the contrast between red light reflectance (650 nm) and near-infrared reflectance (around 800 nm) to assess plant health.
- Chlorosis, the yellowing of leaves due to insufficient chlorophyll, often results in increased reflectance in the red region, as the plant's ability to absorb red light diminishes. This can be detected with Red light reflectance. [5]

Red Edge 730 nm

- The red edge refers to a sharp increase in reflectance that occurs in the spectral region between the red (around 680 nm) and near-infrared (NIR, around 750 nm) wavelengths. This transition marks the boundary where chlorophyll absorption in the red region declines and reflectance in the NIR region increases significantly.
- If a plant is stressed, diseased, or suffering from nutrient deficiencies, the chlorophyll content decreases. This shifts the red edge towards shorter wavelengths and often makes it less steep, indicating reduced photosynthetic efficiency.

- The red edge is highly sensitive to early signs of stress in plants, often before visible symptoms like yellowing appear. This makes it a valuable tool for proactive management in agriculture and environmental conservation [8].

Near Infrared

- In plants, NIR light is strongly reflected rather than absorbed. The cell structure of healthy leaves reflects a significant amount of NIR light, which is why it is often used in remote sensing and agricultural monitoring. High NIR reflectance typically indicates healthy, dense vegetation.
- Plants under stress, such as drought or disease, tend to reflect less NIR light. Monitoring changes in NIR reflectance helps in early detection of stress before visible symptoms appear.
- NIR is a key component in various vegetation indices, such as the Normalized Difference Vegetation Index (NDVI). NDVI uses the contrast between NIR reflectance and red-light absorption to assess plant health, biomass, and chlorophyll content [3].

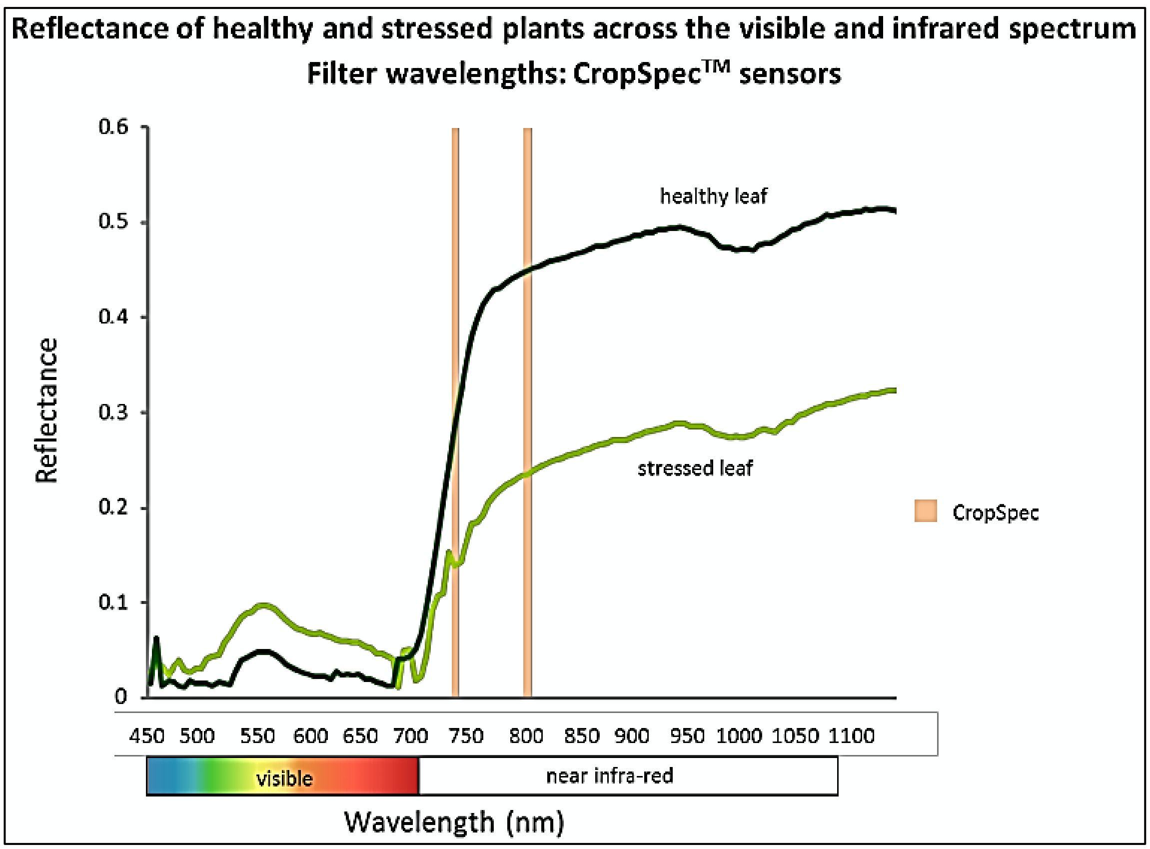

Figure 14- Reflectance of Healthy vs Stressed Crops in different spectrums.

Figure 14 compares the reflectance graph of all above discussed wavelengths for healthy and stressed plants [6]. It is evident from the graph that healthy plants absorb visible radiation more efficiently and hence (if one has observed in their lifetime) look dark in color while they reflect the Near infrared spectrum very efficiently.

5. Testing and Results

To test out the setup and our methodology, we took some daylight images of urban area with some vegetation. We tried to calculate the Most commonly used vegetation index which is Normalized Difference Vegetation Index (NDVI). NDVI is calculated as followed…

$$NDVI = \frac{(NIR + Red)}{(NIR - Red)}$$

As shown in figure 14, healthy vegetation absorbs red light very efficiently and strongly reflects NIR light. The combination of low red reflectance (due to chlorophyll absorption in healthy plants) and high NIR reflectance (due to the leaf's internal structure) creates a strong contrast between vegetated and non-vegetated areas. This contrast allows NDVI to effectively highlight areas with healthy vegetation, as healthy plants distinctly reflect light differently compared to non-vegetated surfaces. As an ideal result, the vegetation should look bright and other areas should look darker giving strong contrast and highlighting the vegetation. Below are the images taken and then the NDVI calculated. The results conform well with the prediction and implies that setup works as planned. Refer figure 15 and 16 comparing raw image and calculated NDVI index.

Figure 15- Left: Image from master camera, right: NDVI

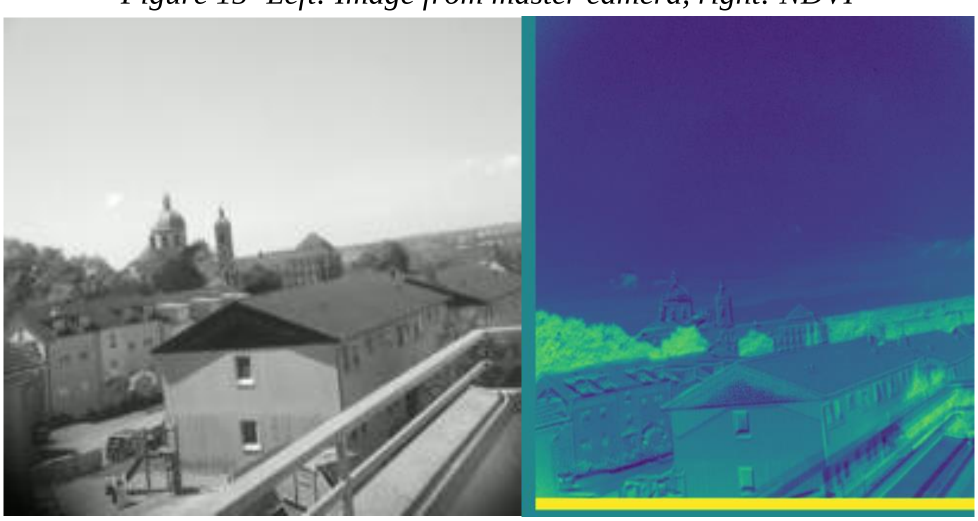

Figure 16- Left: Image from master camera, right: NDVI

6. Scope for Improvement

Even though the setup is functional and works as expected, it is far from being a reliable tool for obtaining actionable data. The following are areas that can be improved further:

1. **Camera Resolution**
   The current setup uses only 1-megapixel sensors. For the system to be effective in aerial applications, it needs a higher resolution to capture detailed data from heights.
2. **Filter Size**
   As discussed earlier, the filters obstruct the view. A new solution is needed to achieve a fully unobstructed view of the scene.
3. **Heat Dissipation**
   Even without onboard processing, the Raspberry Pi heats up significantly just from capturing images. To increase the image capture speed per minute and potentially perform some onboard processing (like cropping and rotating to the correct orientation), we need cooling solutions such as a vapor chamber.
4. **Trigger Button**
   The setup currently has no way to interact with it without using Pitunnel. To make the arrangement more independent, buttons can be integrated using the general I/O pins available onboard.
5. **RGB Camera**
   In addition to the existing sensors, it would be useful to have a reference image of the scene in the standard RGB channel, so the user can better understand the scene.
6. **Image Registration**
   The current methodology involves manually tweaking the homography matrix to align the two channels of spectral images. One study has used CNNs and image gradients for image registration. This solution could be explored as well.
7. **Preview Window**
   A small, low-resolution, and even monochromatic display could be mounted on the setup to provide a preview of the scene being captured. This would be helpful for handheld use.
8. **GPS Service**
   Having records of where images were taken is an important feature, as it allows the user to know which images correspond to specific sections of the farm.

Bibliography

[1] DJI. [Online]. Available: https://store.dji.com/de/product/dji-care-enterprise-basic-p4m?vid=147921.

[2] S. S. a. Drones. [Online]. Available: https://senterasensors.com/hardware/sensors/6x/.

[3] R. Falcioni, W. C. Antunes, R. B. d. Oliveira and M. L. Chicati, "Assessment of Combined Reflectance, Transmittance, and Absorbance Hyperspectral Sensors for Prediction of Chlorophyll a Fluroscence Parameters," 2023.

[4] J. Feng, L. Zeng and L. He, "Apple Fruit Recognition Algorithm Based on Multi-Spectral Dynamic Image Analysis," *MDPI,* 2019.

[5] M. Goebel and D. Iwaszczuk, "Spectral analysis of images of plants under stress using a close range camera," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences,* 2023.

[6] P. &. Y. I. &. G. M. McVeagh, "PASTURE YIELD MAPPING FROM YOUR GROUNDSPREAD TRUCK.," 2012.

[7] A. B. a, D. M. a, F. B. a and A. R. H. b, "A review of vegetation indices," *Remote Sensing Reviews ,* 1995.

[8] S. Rubo and J. Zinkernagel, " Exploring hyperspectral reflectance indices for the estimation of water and nitrogen status of spinach," *ELSEVIER,* 2021.

[9] Parrot, Parrot Drone, [Online]. Available: https://www.parrot.com/en/support/documentation/sequoia.

[10] S. Rana, S. Gerbino, C. M, V. Cirillo, P. Carillo, F. Sarghini and A. Maggio, "Comprehensive Evaluation of Multispectral Image Registration Strategies in Heterogenous Agriculture Environment," *MDPI,* 2024.

[11] Z. Wei, C. Jung and C. Su, "RegiNet: Gradient guided multispectral image registration using convolutional neural networks," *ELSEVIER,* 2020.