Multispectral Imaging at Astroplant Overview

Matthias Kortleven
4290615

Industrial Internship
Imaging Physics
TU Delft

June, 2019
Auréle Adam
Sidney Niccolson

Contents

1 Production

1.1 Building the module

After some testing of small LEDs it can be concluded that there is enough light to move on with this strategy. With very limited light and exposure times of around a fifth of a second the image is still decent. The shorter this time, the better, because plants tend to move around somewhat, especially in a kit that has active air circulation, such as the Astroplant kit. Sensitivity in the NIR range is also perfectly fine and delivers well on image quality. To measure the NDVI three red LEDs (630 nm) and three NIR LEDs (850 nm) will be used initially. A test setup was made containing these LEDs, as well as two narrow angle white LEDs, two wide angle white LEDs for the regular photos, and a single amber, green and blue LED, just for further testing. All LEDs are high quality Cree LEDs producing considerably more light than the cheap LEDs used initially. The LEDs are all driven by the 5V rail of the Pi via a ULN2003 transitor array, resulting in an extra volt of play room when matching the LEDs with a resistor. All LEDs were matched with a resistor such that the current through the LEDs is around 15 mA, with the exception of the wide angle white LEDs, which were matched at 20 mA, since they have a higher continuous current rating.

Further design of the module and the boundary conditions within the kit requires some more looking into the desired behavior of the module and what we are actually trying to measure.

1.1.1 NDVI

The NDVI (Normalized Difference Vegetation Index) gives a very rudimentary value for the amount of live vegetation (and the quality of the vegetation up to some point) contained in a certain portion of an image. It is defined as follows:

$$NDVI = \frac{R_{NIR} - R_{Red}}{R_{NIR} + R_{Red}} \tag{1}$$

Where R_X is the reflectance (the ratio between incoming and reflected light) of a certain spectral band X. Values lie between -1 and 1 and typical values for vegetation are between 0.3 and 0.8. Other mass in the photos will tend to have a lower NDVI. Because it is built up out of reflection ration's, this immediately raises an important point for the lighting used to image the plant.

1.2 Diffuse light source

Since all of the calculations are done relative to an original flatfield image, we can also look into taking the flatfielding out all together. If the radiation pattern of the lights is flat enough, this step can be skipped. Now this is hard, since diffuse light sources are not that easy to come by (most LEDs have a relatively small cone of light shooting outward) and it still will not fix the problem with direct reflections. Both can be fixed at the expense of total amount of light by introducing somewhat of a trick. If the light sources are pointed upwards towards a diffuse reflective surface (the same white paper used for flatfielding before) we can simulate a diffuse light source that has some spatial size, reducing direct reflections. If the field is flat enough for both the red and the nir channels, the flatfield image used before can be reduced to a flatfield number, to which the relative calculations can be performed. This means the process of calculating the NDVI will look as follows in formula form:

$$NDVI = \frac{\frac{I_{nir}}{I_{nir,ff}} - \frac{I_{red}}{I_{red,ff}}}{\frac{I_{nir}}{I_{nir,ff}} + \frac{I_{red}}{I_{red,ff}}} \quad , \tag{2}$$

where the $I_{xxx,ff}$ terms are simply floating point numbers captured from a reference image with a white diffuse surface on the bottom plate. The gains used by the camera are recorded as well, providing enough information to obtain relative values for the amount of light emitted at the location of a certain pixel in the image. Assuming that the camera works linearly (which will be shown in another section) the apparent intensity of light of pixel P of a certain channel is proportional to:

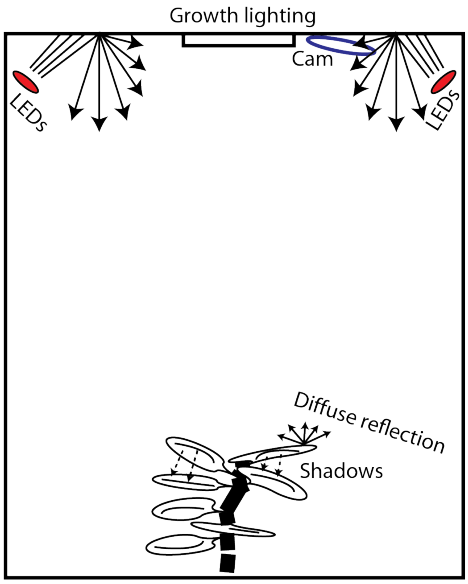
$$I_P \propto \frac{PV_P}{PV_{ff}} \frac{G_{ff}}{G_{photo}} \quad , \tag{3}$$

where PV is the pixel value of a certain pixel or part of the image, and G_x represents the gain of the camera in case x. In the case where diffuse lighting is used PV_{ff} can be taken the average of the flatfield image, which will be approximately constant. This approach will work when the pixel values remain far enough away from the clipping 0 and 255 values and if the flatfields in the infrared and red spectra are sufficiently similar in dropoff. The result is a setup as displayed in figure 1a. Note that because of the layout, reflections will start to play a larger role in the total amount of light reaching a certain part of the plant, which is shown in figure 1b.

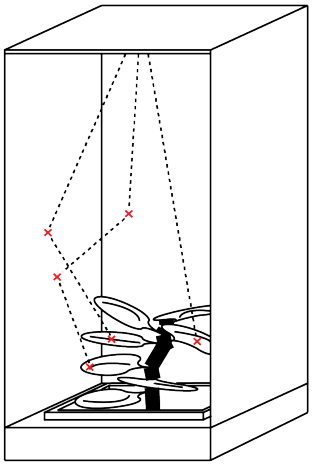
What can be observed when looking at images in the red and infrared spectra is that the reflection coefficient for the foil on the sides of the kit is different for different kinds of light. The reflection coefficient was determined by comparing the pixel values in the corners of the kit with their reflections, compensating for a black reference point in the image. This calculations resulted in a reflection coefficient for red light of 0.92 ± 0.02 and a reflection coefficient of 0.75 ± 0.02 for near infrared light. This discrepancy poses some problems for the intensity distribution of the light when plants grow taller. Since the second and third order reflections play a relatively important role in the total amount of light that reaches a target surface, near infrared light will increase in intensity faster when moving the target surface up than the red light intensity will. Luckily, we can simulate what happens to some extend to better understand this phenomenon. In order to do this we model the indirect light source as a point source that behaves according to the rules of Lambertian reflectance, i.e. we assume that the white surface at the top of the kit is a perfect matte reflective surface. This means that its intensity distribution will behave according to Lambert's cosine law. This law describes that the intensity of light that shines a certain direction depends on the angle α to the normal of the reflecting surface like this:

$$I(\alpha) = I_0 \cos(\alpha) \tag{4}$$

In our simulation, we can then assume one or more sources with a radiation pattern that behaves according to this law. This information can be used to gather data about the influence of different orders of reflection in the kit, as described above. To simulate this behavior, we can take advantage of the geometry of the Astroplant kit, which can be simplified to a square box. The reflections indicated in figure 1b can be simulated by laying out a field that consists of multiple copies of the ground plane next to each other. This grid



(a) Illustration of the lights pointing upwards and reflecting off of a white diffuse surface, creating diffuse lighting. Light bouncing off of the reflective surfaces on the sides of the kit now plays a significant role in the total amount of light reaching a target surface.



(b) Illustration of the first orders of reflection when diffuse lighting is used. In previous cases the light reflecting off of the walls was relatively minimal, since the beam was pointed directly at the bottom surface, here however, the role of indirect light becomes more serious.

Figure 1: Illustration of the indirect lighting method.

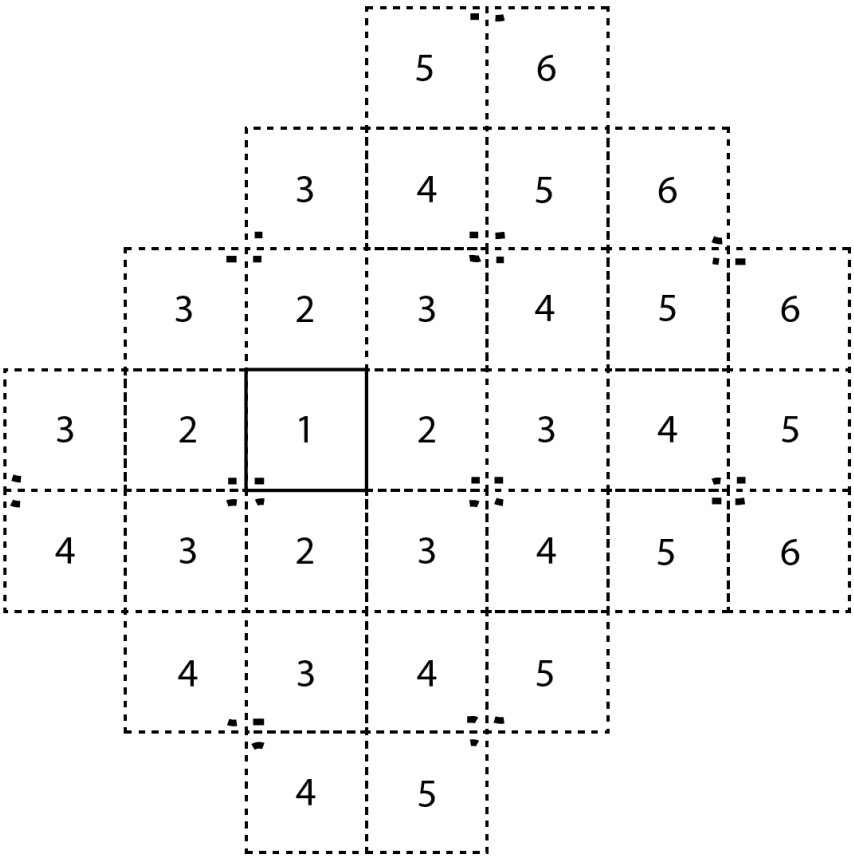


Figure 2: Overview of the simulation grid used to simulate light intensity captured from different orders of reflection in the Astroplant kit. The square with the 1 inside is the direct light received from the reflection off of the top panel. All neighboring squares are second order reflections (light rays that hit the reflective walls once), and so on. A small black square in the corner indicates in which direction a certain plane is mirrored when mapped back to the original ground plane.

is displayed in figure 2. Each neigboring square maps back to the original ground plane via the reflective surfaces. This creates a simple plane in which calculations can be done easily with the help of the formula above. Because the diffuse reflection of the light on the bottom plane also behaves according to the cosine law, we can multiply by this term again to obtain the light intensity captured by the camera.

To gain insight in the different orders of reflection, we can split them out into their separate contributions. These contributions can be seen in figure 3. Note that in this simulation four perfectly diffuse light sources were placed halfway between the middle and each of the four corners. Clear is that the second and third orders contribute significantly to the total intensity in the final picture. Also the radiation pattern changes significantly after the second order reflection. When summing all contributions we get the final image displayed in figure 4. As can be seen there, the result is a decently flat field, which strokes with the fields we see when taking actual pictures in the red spectrum. It is nearly flat, but falls off a little near the corners of the ground plane.

Taking the average intensity of the ground plane when moving up the target plane inside the kit, we can determine what happens to the ratio between near infrared and red light for different distances to the light

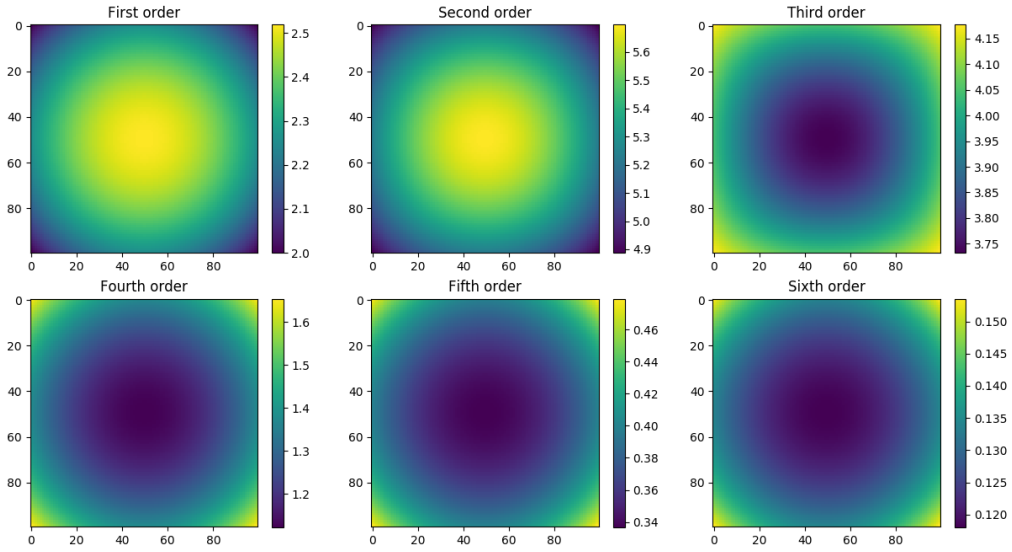


Figure 3: Result of the simulation of the intensity at the ground plane for different orders of reflection for a reflection coefficient of 0.92 and reference intensity $I_0 = 1$. Note that the second order reflections contribute significantly more to the final image than the first order reflections. Contributions only lower from the fifth order onwards.

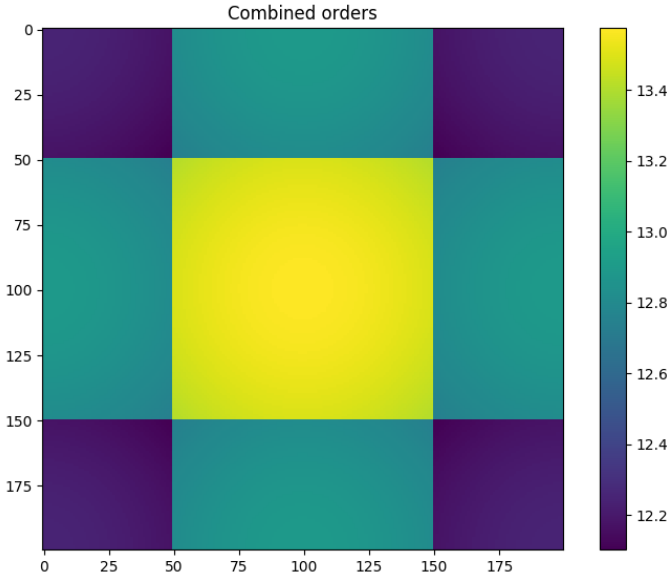


Figure 4: Total intensity mask when summing the results from figure 3. The field is nearly flat, with a falloff from 13.6 in the center to 13.3 at the sides. This is acceptable as flat enough to take the average of the ground plane as the flatfield value.

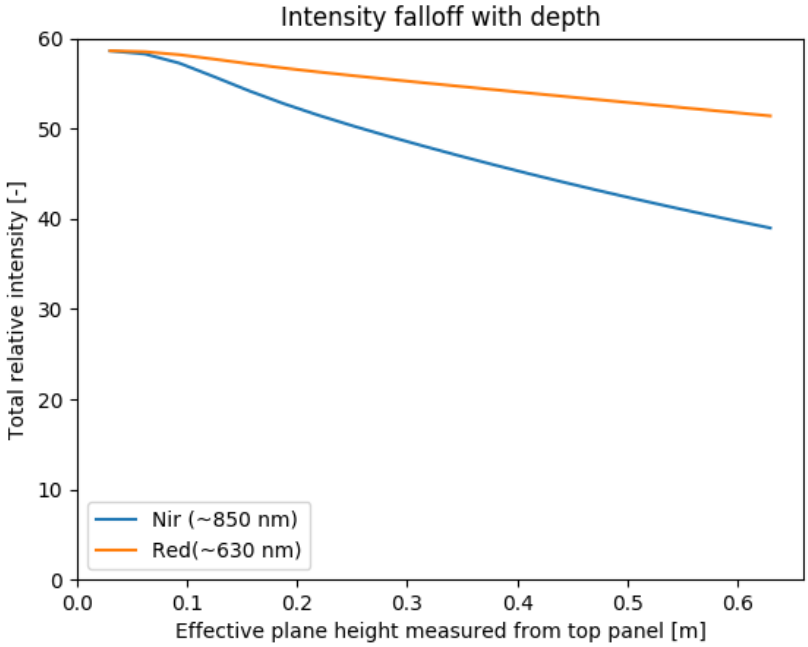


Figure 5: Intensity falloff with depth in the kit. Clear is that red does not gain much intensity when moving the target plane up to the lights, but near infrared does. This significantly impacts NDVI calculations, as the result almost differs by a factor two between the bottom of the kit and the top. This characteristic can be fixed by using foil that reflects both channels in a similar fashion.

sources. The result of this analysis can be seen in figure 5. Clear is that the NDVI will suffer from this if the kit is calibrated to the bottom plane. When moving up higher NDVI will increase as the near infrared channel will gain intensity faster than the red channel. This can be fixed by having a depth map of the plant (but as discussed before, this is hard to achieve, and impractical), by replacing the reflective surfaces on the sides of the kit with material that reflects both channels equally well (the exact reflection coefficient doesn't matter, as long as it is the same for both channels) or by estimating an error based on the average height of the plant. The second option is the best here, since it fixes the problems at the source, and does not try to patch problems later in the process.

When taking a photo with the indirect light approach, we can immediately spot the differences between this approach and the two direct approaches. A summary of the different images is displayed in figure 6. Results are definately better than the two previous tries, and the results above are reflected in a way that would be expected. The wooden bar in the right of the image lights up in the NDVI image, as would be expected from a relatively high-reflective object placed at that height in the kit. All unwanted reflections present in the previous images are gone here, which is good. Contrast is also better, but again, NDVI will be a little too high for the leafs that are higher up, which capture relatively more near infrared light than red light.

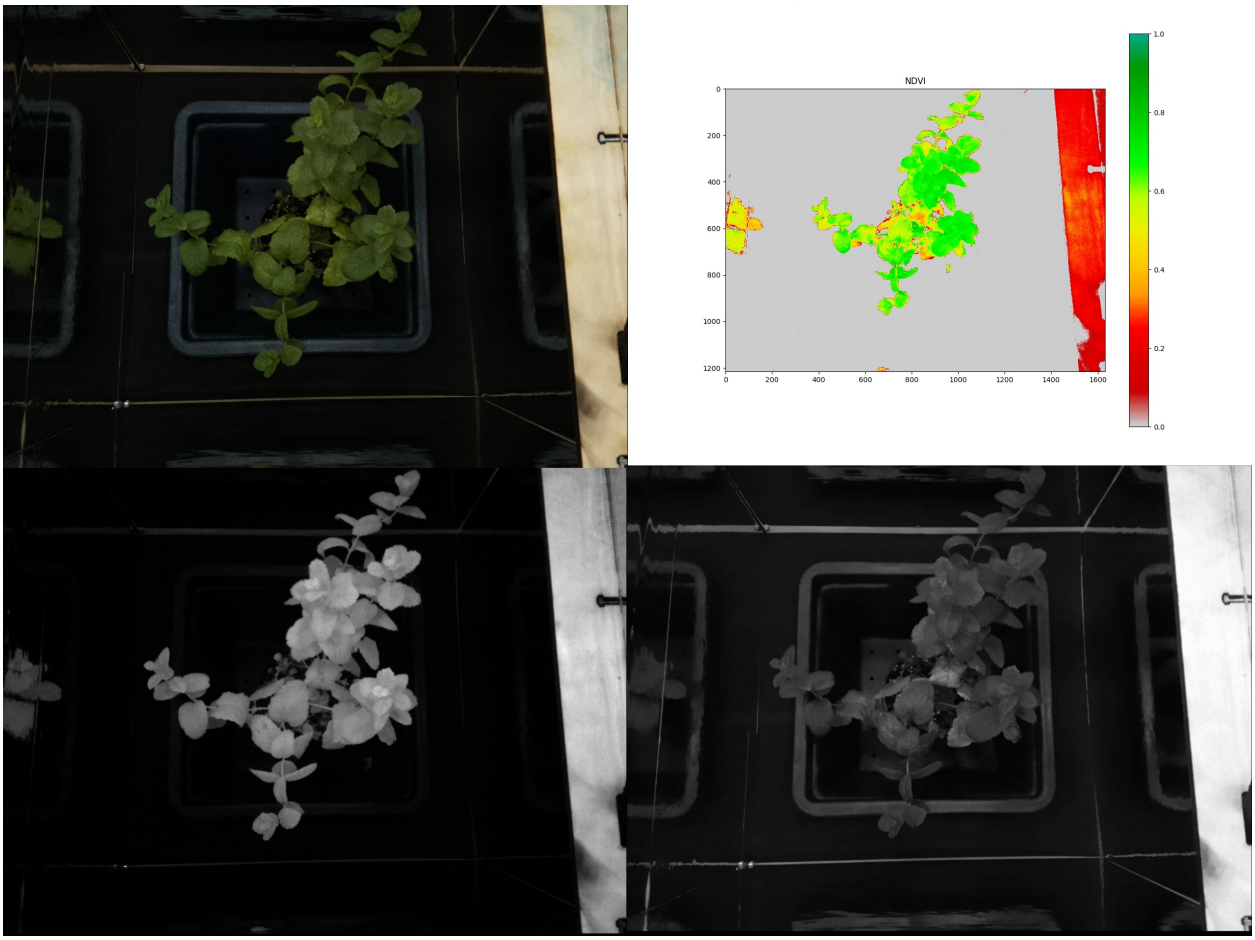


Figure 6: Representation of the first results using indirect lighting, as depicted in figure 1a. (clockwise: white, NDVI, red, nir.) Results are better than with the two direct approaches, giving better contrast between different leafs and getting rid of the reflections on the plant container. The wooden bar on the right is visible in the NDVI image due to the effect described above, where the reflective material on the sides reflects red light better than near infrared light. The bar is just above the halfway mark inside the kit, so it isn't strange that it becomes visible in the NDVI image.

1.3 Coding

The source code for this project can be found in a git repository at `https://github.com/Liveshort/astroplant-camera-module`. It contains the source code, a README and some results from testing. A few choices that were made deserve to be highlighted, as there are other approaches available, or the specific implementation required some creative problem solving. To keep things organized, the structure of the code will be highlighted first.

1.3.1 Code Structure

The code within the Astroplant camera module Python module is organized as follows:

```
+ astroplant-camera-module
|
+--+ cameras
|  |
|  +--+ pi_cam_noir_v21.py
|  +--+ pi_cam_v21.py
|
+--+ core
|  |
|  +--+ camera.py
|  +--+ ndvi.py
|
+--+ misc
|  |
|  +--+ debug_print.py
|  +--+ helper.py
|
+--+ __init__.py
+--+ setup.py
+--+ typedef.py
```

The `cameras` folder contains the different implementations for specific camera's. If users want to use their own camera, this is the place to add another file with implementations specific to their camera. During the project focus was laid on the Pi Camera modules available commercially for around 30 euros. An implementation is given for the regular camera, capable of shooting regular pictures with white light, or with the growth light. The NoIR camera implementation allows for NDVI photos as well, if the user installed appropriate light sources within the kit. Note that these different camera files contain child objects of the actual camera objects. Requirements to functions supplied to the module are given in the README.

The `core` folder contains the actual brains of the module. The `camera.py` file contains routines for making pictures in a single channel, calibration and the logic behind the command comprehension. The `ndvi.py` file contains routines to calculate the average NDVI, but also take NDVI pictures. Other quantities of interest acquired in similar fashion (two or three sequential photos in different bands) could be programmed in a similar manner, with minimal changes to the existing code.

The `misc` folder contains some helper functions and a print function that can be disabled at will for debug purposes.

The other files are related to the setup of the module and the indication that it is actually a Python

module. The typedef.py file contains the allowed commands that a user can send to the module. This approach was chosen to minimize errors due to wrong spelling. Also, because these are separate objects, Python can perform a type check when functions get called that use a command.

1.3.2 Code workings

The first step in using this code is creating a camera object in the user code. Settings specific to each version of the kit are supplied with the camera objects and need to be supplied when creating a camera object. This will set up the work space for the camera (generate configuration folders and a place to save the images). If the work space is already set up the code will try to load previous configuration files so that calibration is not necessary every time the camera is called. For first time use a calibration cycle has to be run, in which the white balance is determined and the flatfield values are calibrated for NDVI measurements. The code assumes that the user has closed the kit and put a white diffuse surface on the bottom plate. This routine will take about 5 minutes to complete, and does not have to be repeated until significant changes are made to the dimensions or the insides of the kit. Notable here is that the white balance calibration uses the `picamera` Python module, and determines the right balance by gradually increasing or decreasing the red and blue gains until the image is white for the white, growth and nir channel.

Taking a picture loads up the camera and executes a `raspistill` command in a separate subprocess. This was done because the gains can be set manually using this command, whereas this is not possible with the current version of the `picamera` module. Workings are fairly simple, the code turns on the appropriate light using a `light_control` function supplied by the user, then proceeds to take a bright and a dark frame (where the light is off again) and performs dark frame subtraction to eliminate light leaking in form the outside. The image is saved to disk and a dictionary with a small report is returned to the user. This `dict` is standardized for all commands and is built up as follows:

```
{
  'photo_kind':
    [
      'raw NDVI',
      'processed NDVI'
    ],
  'encountered_error': False,
  'value_kind': ['NDVI'],
  'value_error': [0.0],
  'timestamp': '20190606-140728',
  'contains_value': True,
  'contains_photo': True,
  'value': [0.3598392680470938],
  'photo_path':
    [
      '.../astroplant-camera-module/tests/cam/img/ndvi1_20190606-140728.tif',
      '.../astroplant-camera-module/tests/cam/img/ndvi2_20190606-140728.jpg'
    ]
}
```

Because of this structure, the user can always check whether the operation executed successfully, and see whether a value or photo is present in the result. After a photo is taken, the code will check whether the current gain settings are still up to date. If the last update was more than a day ago, an `update` function will be called that will determine the correct gain settings. This will take around a minute per active channel. The reason this is necessary is that plants grow over time, and when they get taller, light gets significantly more intense. This requires the camera to be adjusted accordingly. An important note here is that brighter pixels in the photos for high analog and digital gains do not scale perfectly linear. To overcome this behavior,

gains are limited between 0.67 and 1.5 times the flatfield gain for analog, and limited to maximum 1.7 for digital gain.

For the NDVI photos, the code gets a little more interesting. In essence it is not much more than taking two pictures in different color channels and laying them on top of each other, but in practice some difficulties arise. First a red and nir photo are taken and translated to a float numpy array. As the photos are recorded in 8 bit by the camera, the original matrix contains values from 0 to 255 in all color channels. Values are translated to reflectivity values by dividing by 255 and comparing their values to the flatfield values obtained during the calibration process. Values are corrected for differences in gain according to equation 3. Then the NDVI matrix can be calculated using the regular NDVI formula. Since we assume that nir and red intensities rise similarly when the plant grows, this formula is not corrected for different plant heights. We know that this is in fact not the case for the reflective foil, but since there is no depth map, appropriate corrections can not be made. It is left to user to interpret the data appropriately and realize that higher plant parts will have slightly lower NDVI than depicted in the image. Two images are delivered to the user, a raw `tiff` file (0 maps to -1, 255 to 1) that users can use to create their own NDVI maps, and a jpg containing a processed NDVI image with the Polariks color map and a scale on the side. `matplotlib` is used to create this image, which introduces a few programming difficulties. `matplotlib` is not built to be run in an endless loop, and for some reason produces a memory leak when used to simply create and save an image. This was overcome by placing the `matplotlib` routine in a separate subprocess that frees its memory when it gets destroyed after it is finished. Furthermore, `matplotlib`'s regular backend requires an active X server to be running. Since most of the Raspberry Pi's are running headless (without connected peripherals that is) another backend has to be used. This can be achieved by calling `matplotlib.use('Agg')` when the module is imported.

2 Results and Discussion

Tests were performed with LED panel as described in the previous chapters. A schematic reference of this setup is given in figure 7.

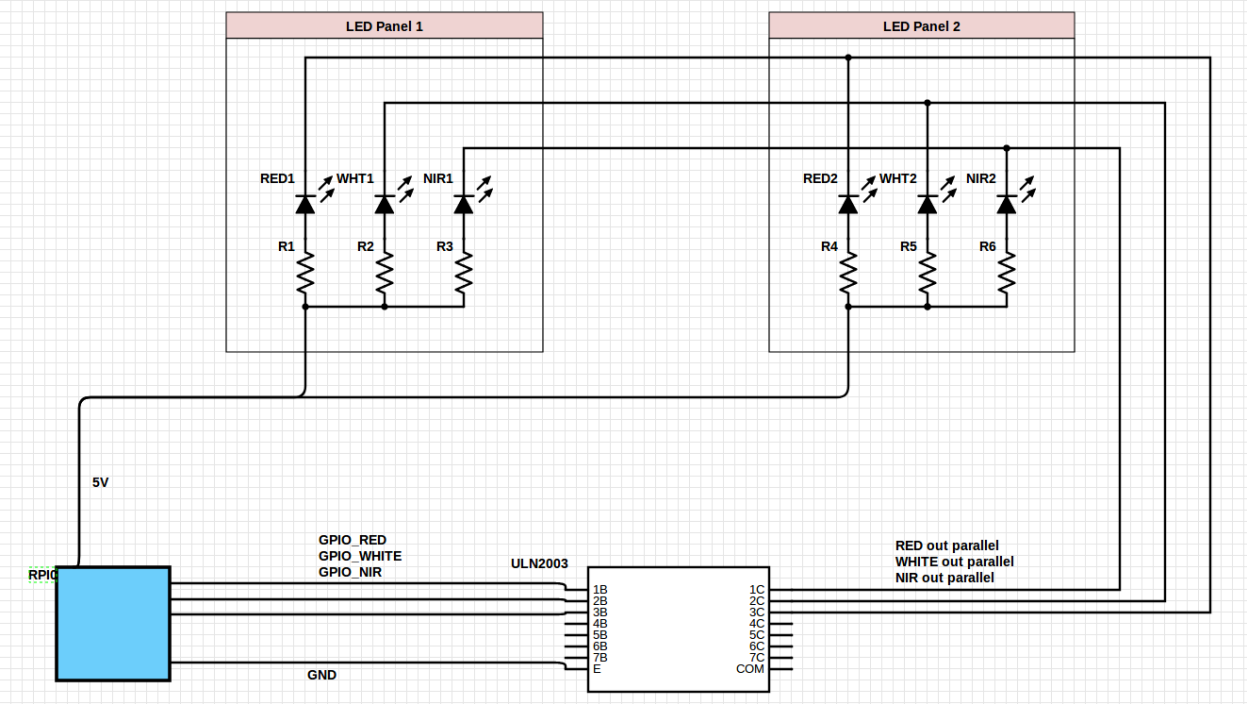


Figure 7: Schematic representation of the test setup used for the following tests. Two boards of three LEDs are present in the kit, mounted in such a way that they radiate upwards on a white diffuse surface. Light from this surface lights the scene with the plant. Resistors in this schematic are chosen to limit the current through the LEDs to 15 mA, and are (for red, nir and white respectively) 150, 200 and 100 Ohm's. A driver circuit is used in order to be able to use the full 5V rail of the Raspberry Pi.

2.1 Linearity Test

It was shown in previous research that the raw mode of the Pi Camera V2 behaves linearly. [11] To check if this behavior is also accurate for regular pictures taken with the camera, a small test was performed. Sheets of paper with different shades of gray were put at the bottom of the kit, and were lit by two white LEDs. White balance is set up such that the image is white (red, green and blue channels similar values for white parts). Then a set of different settings was run to determine linearity in exposure time, analog gain and digital gain. Default settings for both gains are 1, default setting for the exposure is 40 ms, corresponding to a frame rate of 25. Results are given in figures 8, 9 and 10.

It becomes clear from the graphs that the camera mainly struggles with brighter pixels with higher gains. Exposure scales almost linearly, until clipping starts to play a role. This is not the case for the analog and the digital gain, where values flatten out for higher gains. This is a problem that can actually occur when using the camera module. Mainly the red channel has a large deficit between gains during calibration and operation. Since plants have low red reflectance (when they're healthy at least), but the white calibration paper has high reflectance, the difference in gain is significant. The way this is solved in the code is limiting

the maximum gain the camera can use. This will result in slightly darker pictures that lose some color resolution, but linearity is preserved.

2.2 Stability Test

During the last part of the internship, a two week stability test was run to check code operation during longer periods of time. To this end photos were taken in the white channel, as well as NDVI photos at regular two hour intervals. During the first three days problems arose surrounding the memory leak in the `matplotlib` module, which were subsequently fixed after which the test was resumed. Two identical plants were placed inside the kit, one of which received water, while the other did not. At the end, both plants did not receive water for a while. This test had to be run with an incomplete kit and natural light, which means that some of the photos taken during daytime are unusable due to light leakage from outside. Results still show a good time lapse of the plant health in the photos that are usable, which means the test in general was a success. Note that for this test the inside of the kit was padded with white paper, to also check whether the problem with different reflectance values for nir and red light could be solved in this way. Test was started on May 20th in the morning, with both plants healthy and watered. Some results along the way are shown in figures 11 through 16.

A few things can be taken away from this test. First off, the goal of making NDVI pictures in order to be able to say a little more about the plant health seems to be achieved. In both cases the NDVI pictures show a significant change before the regular photo with the white lighting does. This shows that plant stress can be detected earlier with this method than with regular camera checking. Secondly, results seem fairly consistent over time, there are no huge spikes in NDVI over time. Thirdly, even though white paper in principle should give better results, it does hurt the image quality. Using reflective foil makes the area surrounding the plant dark (because that is what is directly reflected from the bottom plate), which in turn makes the camera make better decisions with respect to gains. The images with reflective foil depicted in the previous chapter have considerably better contrast between the plant and the background. Because of this, I conclude it is better to have the error margin from the reflective material and the better photos, than the theoretically more accurate white paper. When it is calibrated to the middle of the kit the error should not exceed ~ 10 percent at both maxima (the bottom and top of the kit).

2.3 Intensity Falloff Test

As described in a previous chapter, intensity falls off faster for nir light than for red light, due to the difference in reflectance coefficient between red and nir light. To quantify this difference, a simulation has been described above that gives some insight in this phenomenon. To verify the correctness of the simulation, a test was performed with the test set up. In order to do this the white diffuse surface used for calibration was placed at various heights inside the kit, with intervals of 10 cm's. The average corrected values for nir and red intensity with respect to the calibration values were calculated both for the case with reflective foil on the walls of the kit and the case with diffuse white paper on the walls of the kit. Results are listed in figures 17, 18 and 19.

From these figures, it becomes clear that the simulation is fairly accurate for the actual real life situation. Nir light does becomes stronger faster than red light in the case of reflective foil, and the difference between white paper and reflective foil seems to match the simulation as well. A point of discussion for this test is the hard to determine error margin. The white paper was mounted as good as possible inside the kit, but differences of 0.5-1 cm could have occurred. This would not have had a dramatic influence on the results, but even so. The error of the relative intensity values is harder to determine, since it depends on the camera settings, lighting setup and light conditions during the test. Photo to photo differences were not that large under the same circumstances, but differences of a few percent could have arisen.

2.4 Memory/CPU Utilization

During the stability test, resources of the Raspberry Pi were monitored. The system reported 443,868 kB of memory present, where the rest of the 512 MB in the system were dedicated to the GPU. When idling, resource usage was minimal [0% CPU usage, only idle Python 3 interpreter usage (which is still 15-16%, but this will be the case in the Astroplant software too)]. When taking photos and performing calculations CPU usage rose to $\sim$ 95%, indicating efficient use of resources by `numpy`. Taking a white picture takes just under minute (actual photo is taken after about 20 seconds), while taking an NDVI picture takes up to 2 minutes. Most of this time is in the plotting with `matplotlib`, which takes up over a minute in this process. Memory usage is also highest during this operation. The `matplotlib` process takes up between 22% and 27% of the total memory available, next to the already running test process that takes up to 25% memory. Taking a regular picture in a single channel opens up a new process that uses only 10% to 15% of memory. Both numbers are not problematic, but might require some background stuff from the kit to be temporarily postponed in order to have enough memory. If this is impossible, it is also possible to cut out the `matplotlib` part and rely on the raw `tiff` file that can be converted to a nicer view on another pc (or the server, for that matter). The CPU should not be the problem, since it will just calculate as fast as it is allowed to. If another process is running in the background, it will take a little longer, but it should not stop. This is also demonstrated by the `pigpio` module, which can sometimes take up 5% to 10% of memory, while the plotting subroutine is running. Both will still finish, and there does not appear to be a problem. An entire cycle of updating the gains, taking a white photo and taking a NDVI photo takes about 4-5 minutes. This should be ok to do twice a day without the plant caring much.

References

[1] Computer Vision/LED Plant Measurement System. https://publiclab.org/notes/MaggPi/03-15-2018/computer-vision-led-plant-measurement-system. Accessed: 2019-02-06.

[2] IMX219 spectral response curve. https://github.com/hyperspy/hyperspy/issues/1775, mail response from Sony. Accessed: 2019-02-07.

[3] NDVI pictures generated with RPi regular and NoIR camera. https://publiclab.org/notes/khufkens/05-10-2015/multispectral-raspberry-pi-first-light-ndvi-images. Accessed: 2019-02-06.

[4] Open Source Computer Vision Library. https://opencv.org/. Accessed: 2019-02-11.

[5] PlantCV. https://plantcv.readthedocs.io/en/latest/. Accessed: 2019-02-11.

[6] H. Dong, S. Yin, W. Xu, Z. Zhang, R. Shi, L. Liu, and S. Wei. An automatic depth map generation for 2d-to-3d conversion. *IEEE ISCE*, 2014.

[7] M. S. Kim, J. E. McMurtrey, C. L. Mulchi, C. S. T. Daughtry, E. W. Chappelle, and Y.-R. Chen. Steady-state multispectral fluorescence imaging system for plant leaves. *Appl. Opt.*, 40(1):157–166, Jan 2001.

[8] L. Mendonca, C. Chaves, E. De Lima, and L. E. Vicente. Low-Cost Multi-Spectral Camera Platform for In-Flight Near Real-Time Vegetation Index Computation and Delivery. 05 2017.

[9] A. Morgand and M. Tamaazousti. Generic and real-time detection of specular reflections in images. *IEEE*, 2014.

[10] E. Murchie and T. Lawson. Chlorophyll fluorescence analysis: a guide to good practice and understanding some new applications. *Journal of Experimental Botany*, 64(13):3983–3998, 08 2013.

[11] M. A. Pagnutti, R. E. Ryan, G. J. Cazenavette, M. J. Gold, R. Harlan, E. Leggett, and J. F. Pagnutti. Laying the foundation to use Raspberry Pi 3 V2 camera module imagery for scientific and engineering purposes. *Journal of Electronic Imaging*, 26:26 – 26 – 13, 2017.

[12] T. C. Wilkes, A. J. S. McGonigle, J. R. Willmott, T. D. Pering, and J. M. Cook. Low-cost 3D printed 1 nm resolution smartphone sensor-based spectrometer: instrument design and application in ultraviolet spectroscopy. *Opt. Lett.*, 42(21):4323–4326, Nov 2017.

[13] H. Xiang and L. Tian. Development of a low-cost agricultural remote sensing system based on an autonomous unmanned aerial vehicle (uav). *Biosystems Engineering*, 108(2):174 – 190, 2011.

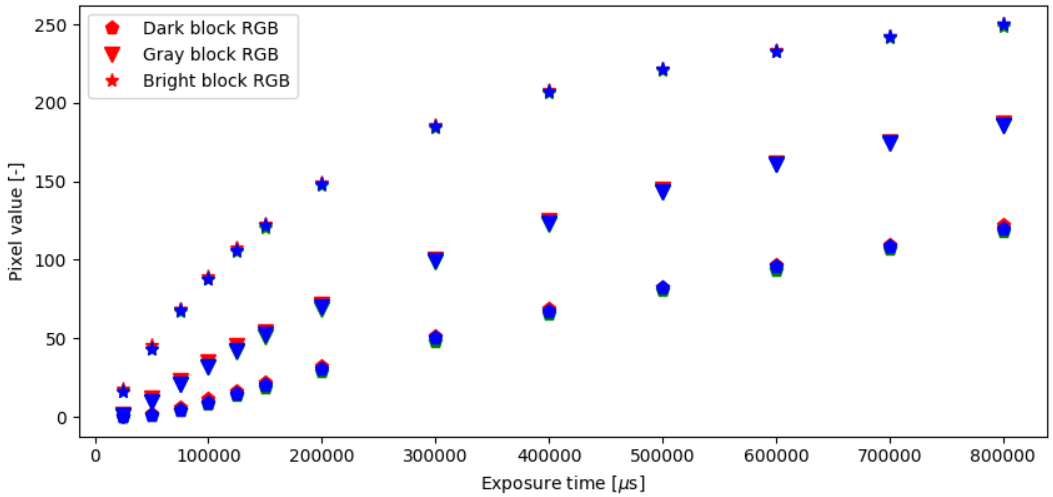


Figure 8: Results for the linearity test for the exposure of the Pi Camera V2. Behavior in the lower exposure times is linear, but for larger exposure times behavior becomes less linear. This is largely due to clipping of the pixels (the darker blocks perform better). Since exposure time is locked in the implementation, this non-linearity does not matter for the results obtained with our code.

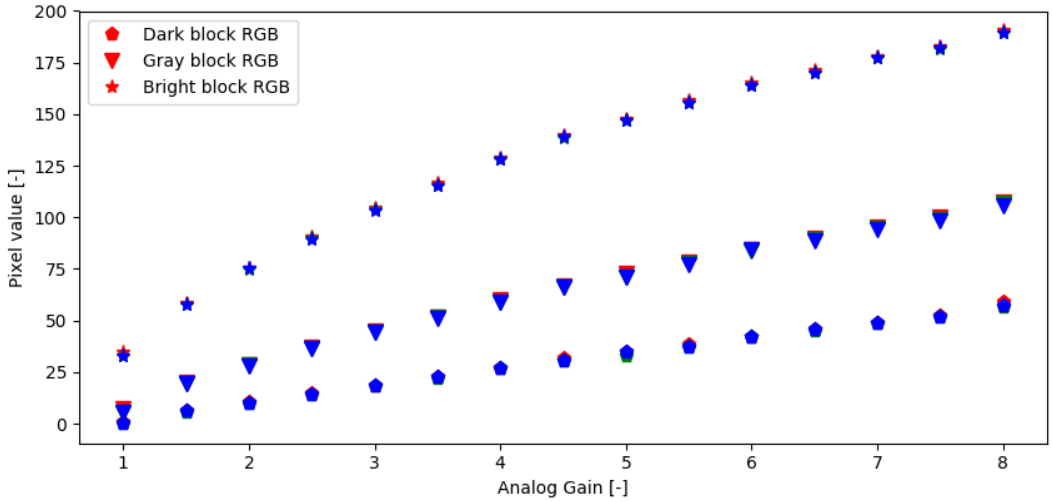


Figure 9: Results for the linearity test for the analog gain of the Pi Camera V2. Behavior in the lower gains is linear, but for larger gains behavior becomes less linear. Clipping should not be a problem yet around pixel values of 150-200, but values increase slower than they should. This is circumvented by limiting the maximum achievable analog gain to 3 in the working code.

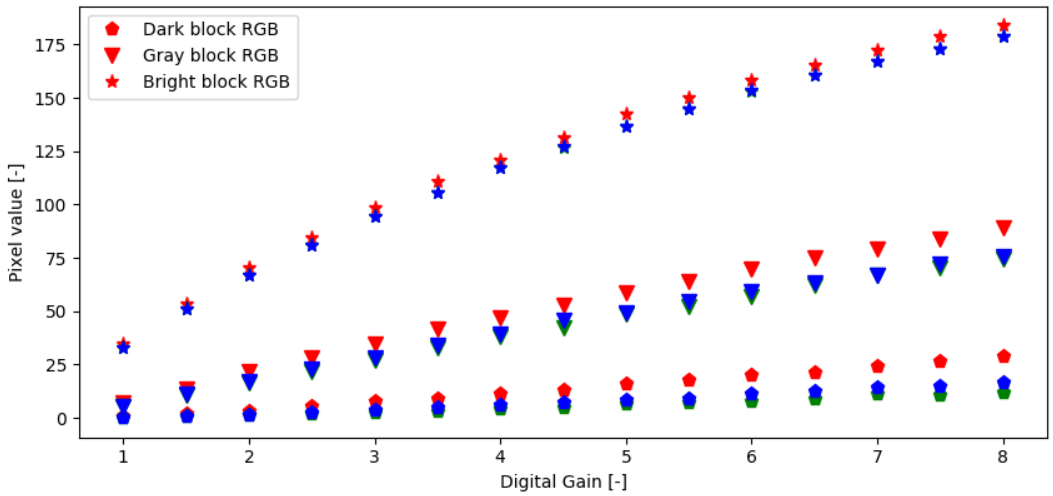


Figure 10: Results for the linearity test for the digital gain of the Pi Camera V2. Behavior in the lower gains is linear, but for larger gains behavior becomes less linear. Clipping should not be a problem yet around pixel values of 150-200, but values increase slower than they should. This is circumvented by limiting the maximum achievable digital gain to 2 in the working code.

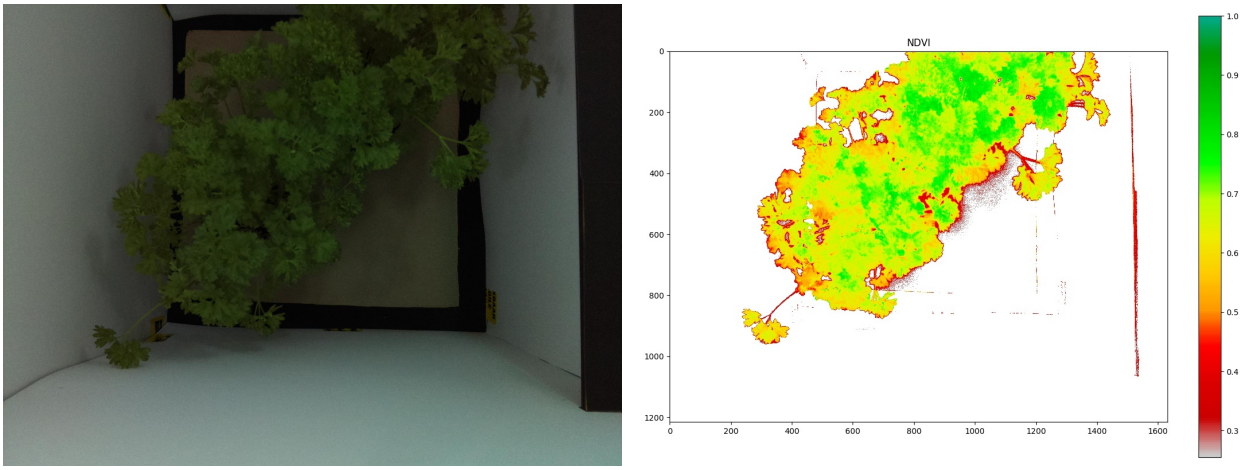


Figure 11: Situation on 20/05 at 14:57. Both plants are healthy, which can also be concluded from the NDVI image. Note that there is some noise in the image, as light is diffusely reflected from the plant on the white paper. In this regard, reflective foil tends to perform better.

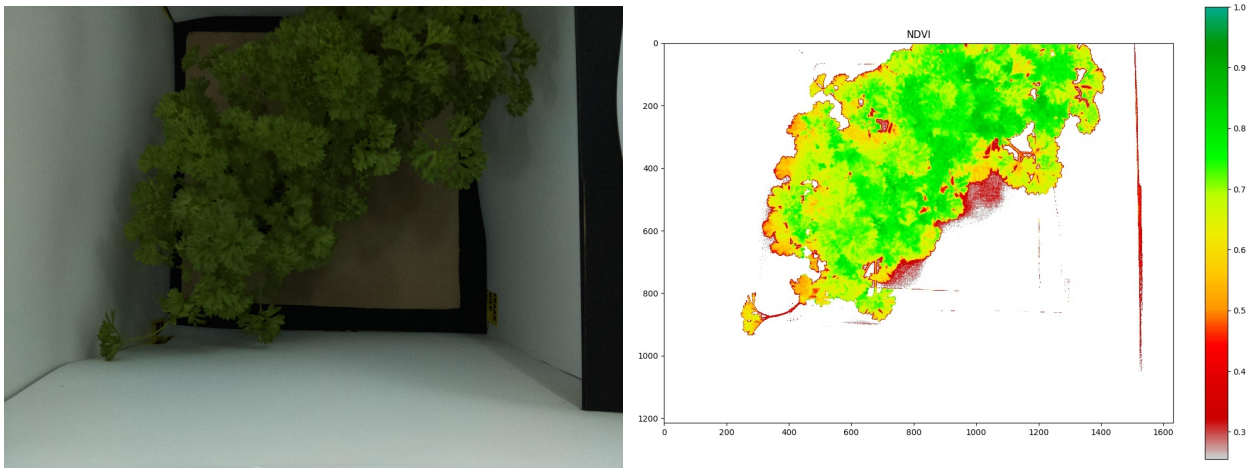


Figure 12: Situation on 20/05 at 23:08. Both plants are healthy and have taken up the water given to them earlier on the day.

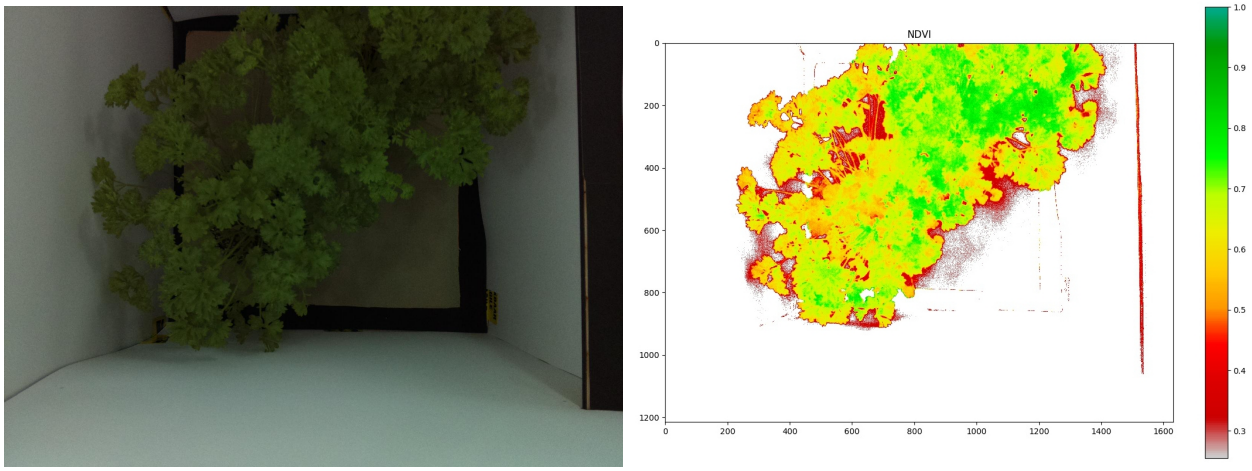


Figure 13: Situation on 23/05 at 14:20. The upper plant has received water, while the other has not. Decay starts to show in the NDVI image, but is still hard to spot in the regular image.

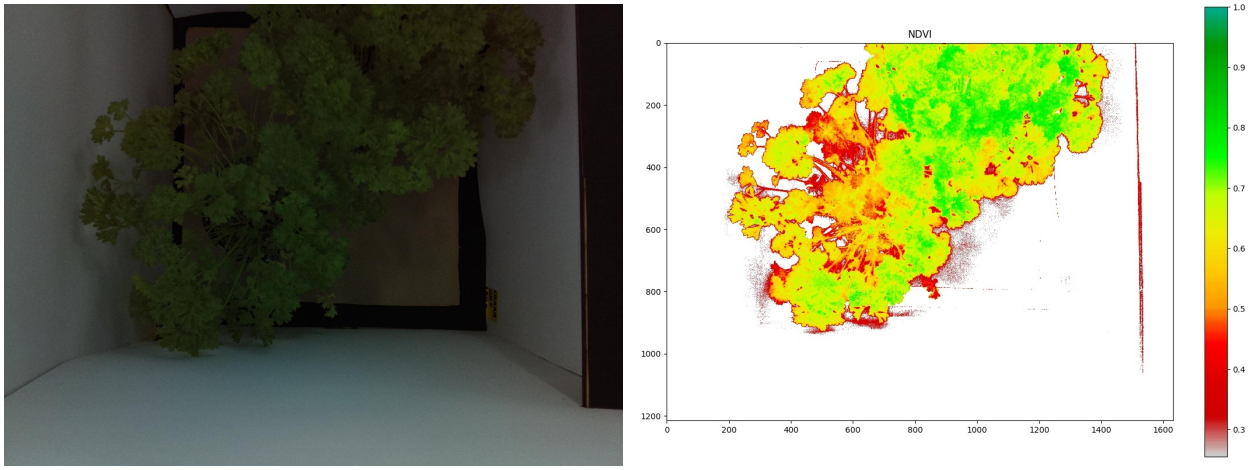


Figure 14: Situation on 24/05 at 22:06. The upper plant is doing pretty well. The lower plant is now clearly dying in the NDVI image, and it starts to show in the regular image as well.

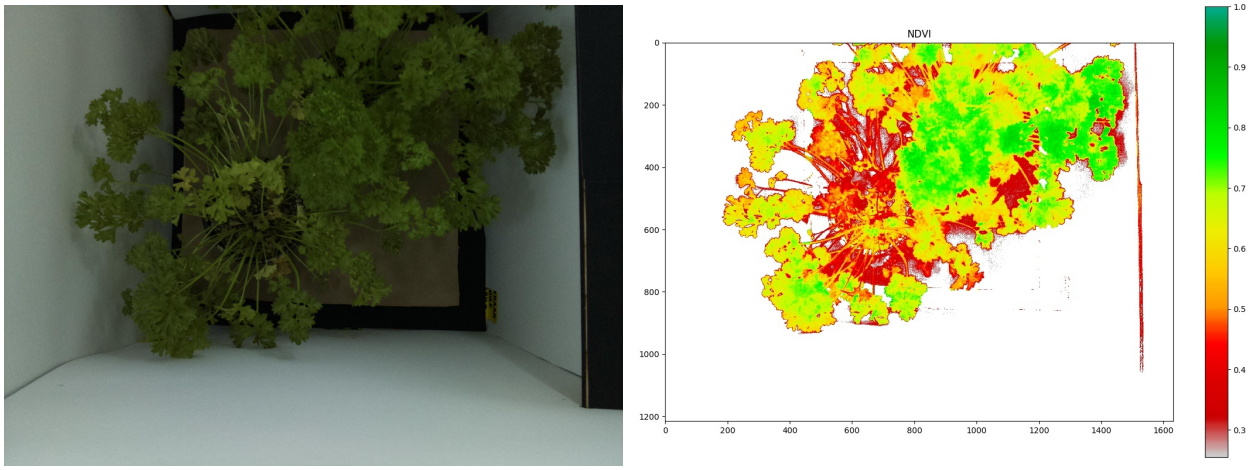


Figure 15: Situation on 26/05 at 21:52. The upper plant is still doing pretty well, but starts to suffer from presumably a little lack of daylight (as it starts to hang slightly). The lower plant is is still dying, with leaves turning notably yellow in the white image. These leaves correspond with a low NDVI, as shown in the right picture.

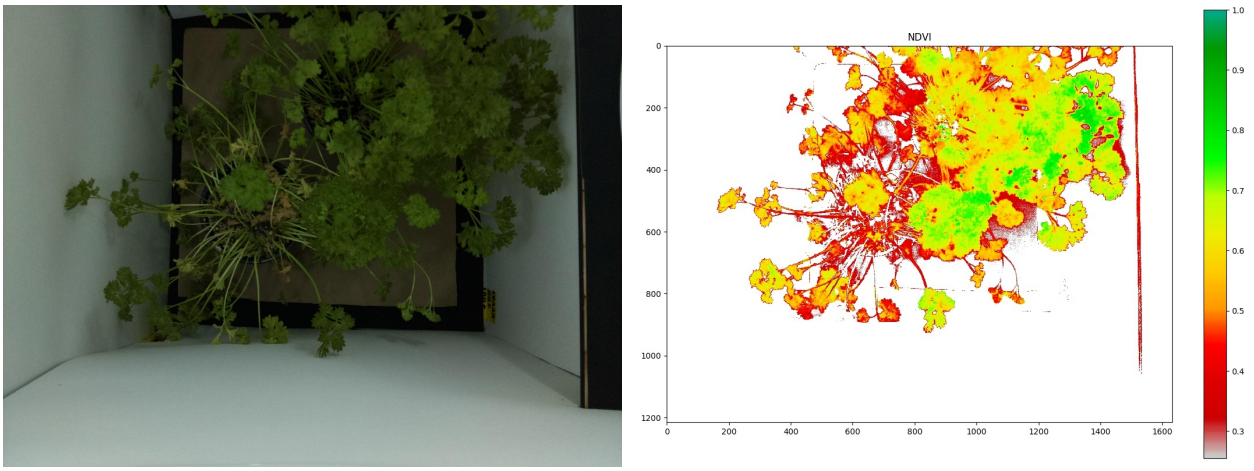


Figure 16: Situation on 30/05 at 04:37. The upper plant now also has not received water for a few days. Clearly the NDVI starts going down with respect to the previous set of photos. Again, nothing seems too bad on the regular image, but the NDVI image shows a less healthy plant. The lower plant is close to dead by now, as can be seen in both photos.

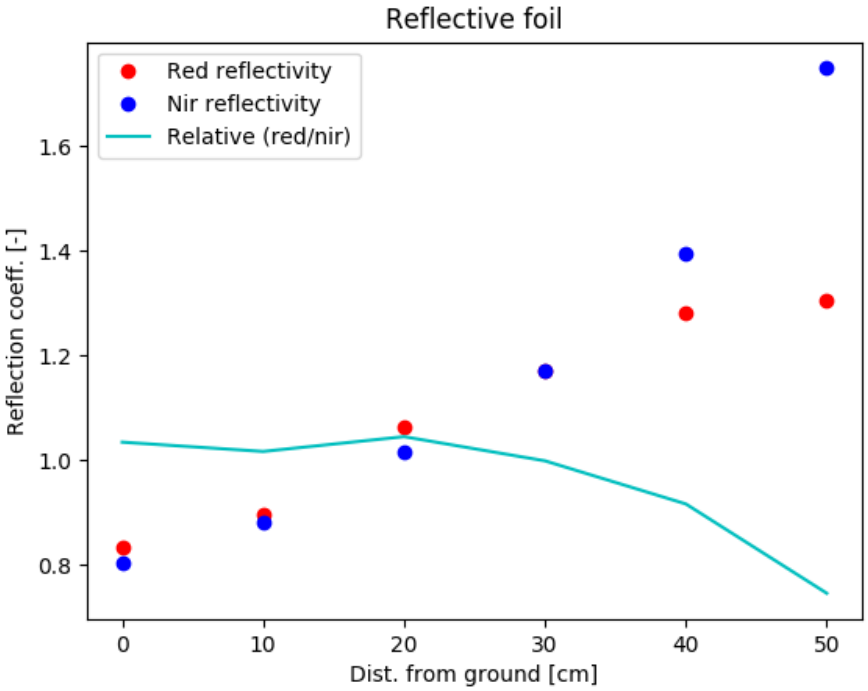


Figure 17: Intensity falloff when the reflective foil is put on the walls of the kit. The first few data points seem to keep the difference between red and nir light fairly constant, but after the 30 cm point, nir light starts to get relatively stronger and stronger, as expected based on the simulation.

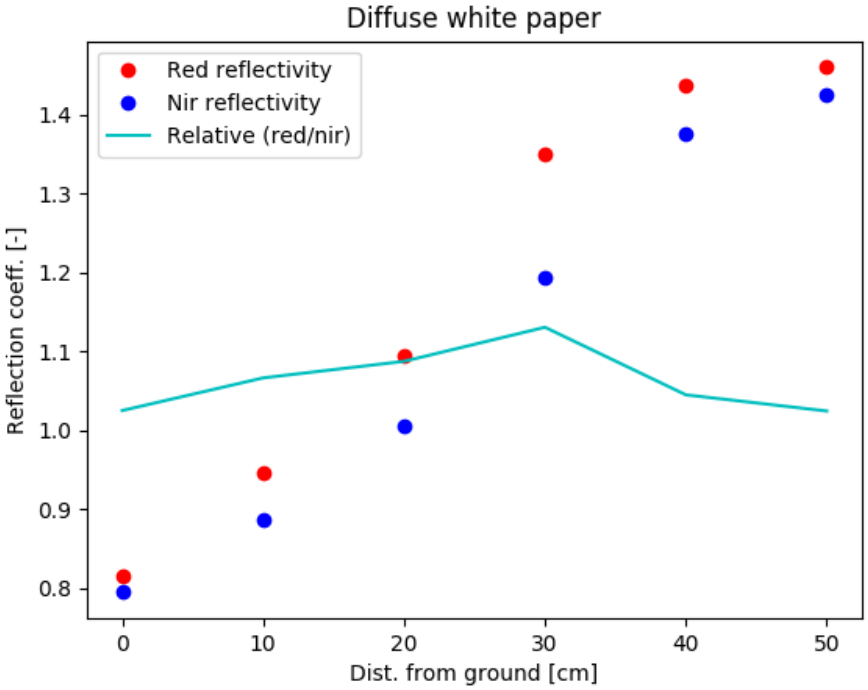


Figure 18: Intensity falloff when white paper is put on the walls of the kit. The difference between nir and red light stays fairly constant, as expected. There is a bit of a strange peak at 30 cm's, which could be due to sudden light leakage from outside.

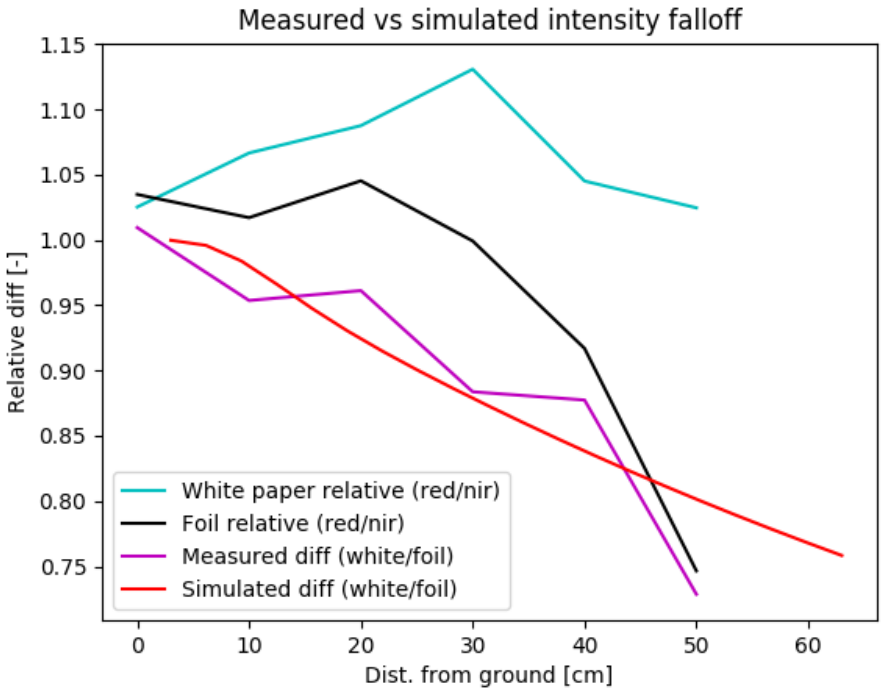


Figure 19: Comparison between the simulated falloff and the measured falloff. The two upper lines represent the the relative intensity falloffs from the previous figures, calculated as the red reflectance over the nir reflectance. The red line represents the simulated difference between white paper and reflective foil, while the purple line represents the measured difference between the red and the nir channel. The measured and simulated difference match fairly well, except for the last point at 50 cm. This is to be expected, since the conditions in the kit at the uppermost 10 cm's are not as ideal as the simulation (material does not extent up perfectly, cables and fans might interfere etc.)