Project Number: 774571
Start Date of Project: 2017/11/01
Duration: 48 months

Type of document 4.1 – V1.0

Multispectral LiDAR Point Clouds

Dissemination level	PU
Submission Date	2018-12-31
Work Package	WP4
Task	T4:2
Type	Report
Version	1.0
Author	Sebastian Lamprecht
Approved by	PMC

Executive Summary

This document comprises a detailed description of the terrestrial remote sensing pipeline (Task 4.2) to generate multi-spectral LiDAR point clouds. These point clouds serve as an input to Task 4.3 – *Tree Geometry Reconstruction* and Task 4.8 – *Fruit Detection*.

Furthermore, this document describes the monitoring concept and experimental setup to receive the required data quality for the PANTHEON project. The following sub-tasks have been identified:

1. Elaboration of a UGV data acquisition concept
2. Selection and purchase of the remote sensing sensors
3. Elaboration of methods for automation
4. Design of experimental setups to calibrate the sensors
5. Implementation of the terrestrial remote sensing pipeline
6. Collection of test data

Table of Content

Abbreviations and Acronyms

2D / 3D	two-dimensional / three-dimensional
ASPRS	American Society for Photogrammetry & Remote Sensing
DEM	Digital Elevation Model
FLS	Faro Laser Scan (file format)
GPS	Global Positioning System
ICP	Iterative Closest Point
LAS	LASer (file format)
NIR	Near InfraRed
RGB	Red-Green-Blue
RTK	Real Time Kinematic
TIFF	Tagged Image File Format
TIN	Triangulated Irregular Network

1 Introduction

The major task of Task 4.2 Terrestrial Remote Sensing Pipeline is to provide an automated processing pipeline, to derive high quality three-dimensional multispectral point clouds as input for Task 4.3 – *Tree Geometry Reconstruction* and Task 4.8 – *Fruit Detection*.

Since the pipeline processes terrestrial remote sensing data, a data acquisition concept needed to be developed, taking into account the technical limitations of the Unmanned Ground Vehicle (UGV) and its remote sensors. Thus, this task also includes the selection and purchase of suitable remote sensors to address the requirements of WP4 and WP5. By following the proposed acquisition design, the UGV should be able to collect data of sufficient quality, to address the subsequent Tasks 4.3 and 4.8, as well as workpackage W5 – *Agronomic Decision Making*. To be able to evaluate the remote sensing concept, remote sensing data—suitable for testing acquisition design and the sensors—needs to be collected in the orchard. Since the UGV is not expected be ready at this stage of the project, a manual data acquisition is required.

Based on the data acquisition concept, the processing pipeline needs to be developed and implemented. In particular, the raw camera and laser scanning data need to be post-processed, to allow for a spectral enrichment of the laser scans by the camera's spectral information, resulting in the desired multi-spectral point clouds. This task also involves the classification of the point clouds and noise filtering. To receive a multi-perspective point cloud for each tree, the scans need also to be aligned to diminish remaining GPS and compass errors. To develop, train and evaluate the processing pipeline, field data as well as laboratory data needs to be collected. Furthermore, experimental setups for the calibration of the sensors need to be elaborated, which shall be used in future as soon as the UGV is available.

To be consistent with the requirements of PANTHEONs SCARDA concept, the implementation of the processing pipeline also included the elaboration of an appropriate data storage model—compliant with the requirements of Task 3.3 – *Definition and Implementation of the Data Repository*—, as well as the consideration of concepts to increase the degree of automation.

2 UGV remote sensing concept

2.1 Sensor requirements

To address PANTHEON's objectives, the UGV need to be equipped with sensors compatible to the planned SCARDA concept. Monitoring the geometrical and spectral characteristics of the hazelnut trees is expected to provide the required information for WP4 and WP5.

2.1.1 Cameras

To detect fruits (Objective 2.4) and water stress (Objective 2.2), as well as pests and diseases (Objective 2.3), spectral analyses of the hazelnut trees is particular expedient. As hyperspectral cameras are still costly and not yet, operationally applicable, a multi-spectral camera array, as a composition of several lenses equipped with spectral filters image sensors of differing sensitivities, is able to measure the characteristic spectral features of the plants. Spectral filters are used to define the wavelength ranges of the spectral bands. Narrow filter widths focus on the spectral features, but reduce the energy received at the sensor. To still obtain a sufficient signal-to-noise-ratio, with time either the exposure time or the sensor area needs to be enlarged. Thus, in general multi-spectral cameras have a lower geometrical resolution—number of image pixels—compared to single band cameras.

To combine the advantages of a multi-spectral camera and the advantages of a high geometrical resolution, the UGV has been equipped with both, a professional multi-spectral and a custom red-green-blue (RGB)

camera. To be able to compute a virtual multi-band image as a combination of all image bands, the cameras are triggered simultaneously.

2.1.2 Laser scanner

For tree geometry reconstruction and sucker's detection (Objective 2.1), a tree-dimensional (3D) analysis of the hazelnut trees is required. To identify suckers, a 3D point density in the magnitude of the sucker diameter—approximately 5 mm—is required. To receive the 3D point clouds with the required density and quality, a laser scanner is the instrument of choice.

Although structure from motion (SIFT) algorithms are also able to generate high density point clouds using stereo images, this indirect method would require a lot of UGV movement and an enormous computational post-processing effort. In addition, laser scans are much less prone to errors than point clouds derived by SIFT.

2.2 Sensor integration

The selection and integration of the remote sensors has been planned in close cooperation with Task 3.1, to ensure a smooth system integration. The UGV should be capable of capturing a tree almost completely by placing the laser scanner and the cameras at various positions and heights.

Assuming the same sensor positions, the camera images can be projected from the two-dimensional (2D) senor plane to the 3D laser point cloud. Increasing distance between the cameras and the laser scanner positions results in increasing projection errors, especially for objects close to the sensors. Thus, the UGV should be capable of bringing the cameras into almost the same position as the laser scanner.

To achieve these goals, the UGV has been equipped with a swivel-mounted horizontal bar, attached at the top of a telescopic arm. The laser scanner has been mounted towards a gimbal on the horizontal bar, while the cameras have been integrated into the gimbal (see Fig. 1). By swiveling the horizontal arm by 180°, the cameras are capable of capturing a tree from almost the same position as the laser scanner. The gimbal stabilizes the cameras and allows for a vertical and horizontal rotation of the cameras, which results in a larger field of view.

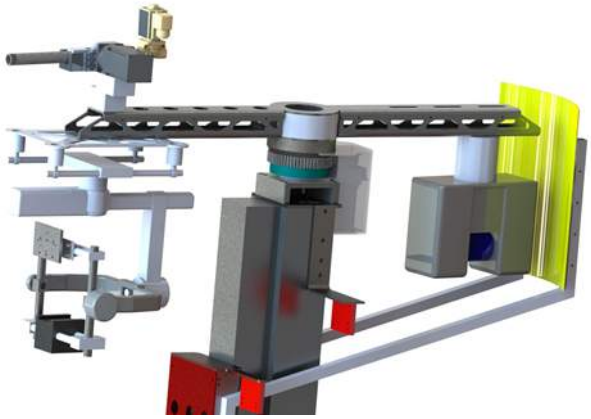


Fig. 1: Sketch of the sensor mounting on the UGV. The cameras are mounted within the gimbal (left), while the laser scanner is mounted upside-down towards the gimbal (right).

To be able to fuse the camera and laser data, the extrinsic position and orientation of the sensors is recorded during capture. This information is provided by the real-time-kinematic (RTK) GPS, gimbal and kinematic sensors of the UGV. The relative position of the sensors in relation to the GPS, as well as the orientation of

the cameras are provided for the spectral enrichment. In addition, the GPS position and UGV orientation are provided for georeferencing the scans.

2.3 Data acquisition concept

To limit the number of scans, but still capture the hazelnut trees almost completely, each tree is scanned by the UGV from four surrounding positions. Fig. 2. illustrates the UGV positions, as well as the swivel of the horizontal arm to place the sensors at similar positions. Based on the typical planting patterns, a typical sensors distance to the tree trunk of about 3m can be assumed.

With the given monitoring pattern, two scanning positions of two neighbored trees are always the same. So, if an orchard is monitored completely, the number of required scanning positions is reduced by factor four.

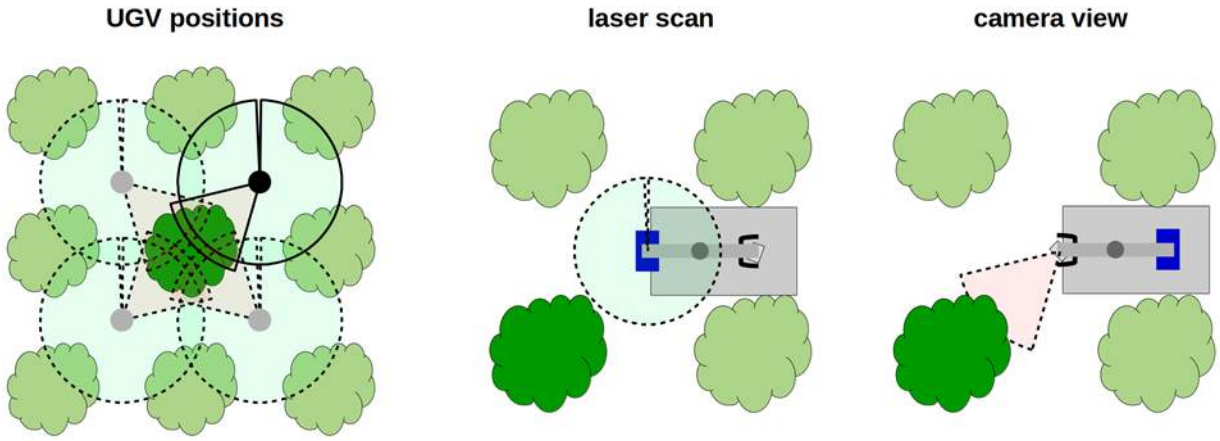


Fig. 2: Schematic illustration of the UGV capturing a specific tree from four positions (left), with the laser scanner (center) and with the cameras (right).

The number of images required to capture a tree almost completely depends on the distance to the tree, the size of the tree and the camera's field of view. With the cameras presented in section 2.4 and the given regular planting pattern, it is planned to scan young trees (field 18) with two image captures per UGV position (see Fig. 3), and mature trees (field 16) with six image captures (see Fig. 4).

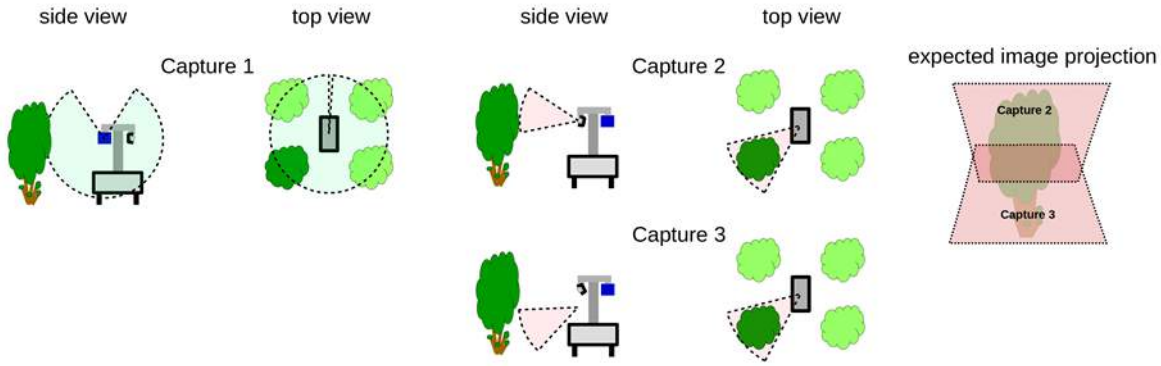


Fig. 3: Sensing concept to monitor young trees with a laser scan and two image captures.

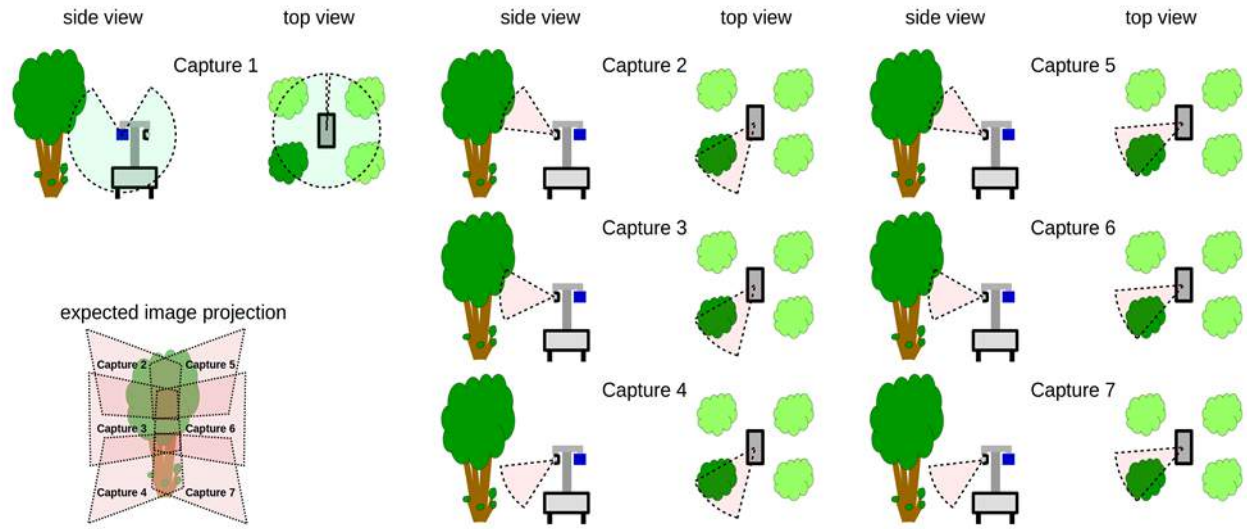


Fig. 4: Sensing concept to monitor mature trees with a laser scan and six image captures.

2.4 Sensor selection

2.4.1 Multispectral camera

Based on spectral sensor characteristics, Linux support, camera dimension, data quality and price, a *MicaSense RedEdge-M* has been purchased. "The MicaSense RedEdge-M is a professional multispectral camera capable of simultaneous capture of five discrete spectral bands to generate precise and quantitative information on the vigor and health of crops." [1]. Previous versions of the camera have been used for research projects with a focus on agricultural applications at TRIER University.

Tab. 1 summarizes the spectral characteristics of the camera. Fig. 5 illustrates the position of the spectral bands in relation to a typical plant reflectance spectrum. Due to the narrow bands, the relevant spectral characteristics of a hazelnut tree—like the green peak and red-edge—can be extracted. Disadvantages of the camera are the limited geometrical resolution of 960 x 1280 pixels and parallaxes between five sensors, with an individual lens for each band. The fixed lenses of modest quality result in significant lens distortion and vignetting effects, which need to be corrected by post-processing the images.

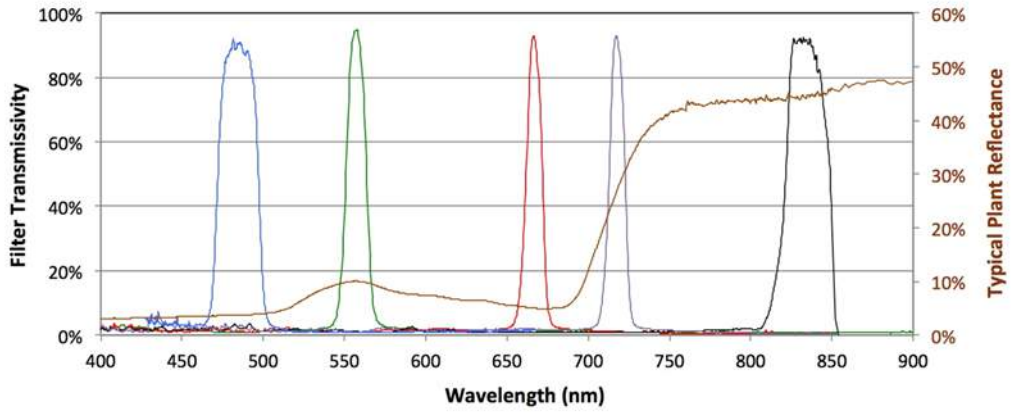


Fig. 5: Spectral characteristics of the MicaSense RedEdge-M camera **[1]**.

Tab. 1: Spectral characteristics of the MicaSense RedEdge-M camera **[1]**.

Band Name	Center Wavelength (nm)	Bandwidth FWHM (nm)
Blue	475	20
Green	560	20
Red	668	10
Near infrared	840	40
Red Edge	717	10

2.5 RGB camera

As a counterpart to the professional multi-spectral camera, a custom *Sony α5100* RGB camera has been purchased, because of its high geometrical resolution and its robust design. To achieve high-quality close-range images with a large field of view, the camera has been equipped with an optical lens (Sony SEL-28F20) of 28mm focal length and F2.0 light intensity. The spectral characteristics, with only visible and broad diversified bands (see Fig. 6), are not suitable for a meaningful spectral analysis. But the high geometrical resolution is particularly suitable to apply object recognition techniques—*e. g.* for fruit detection.

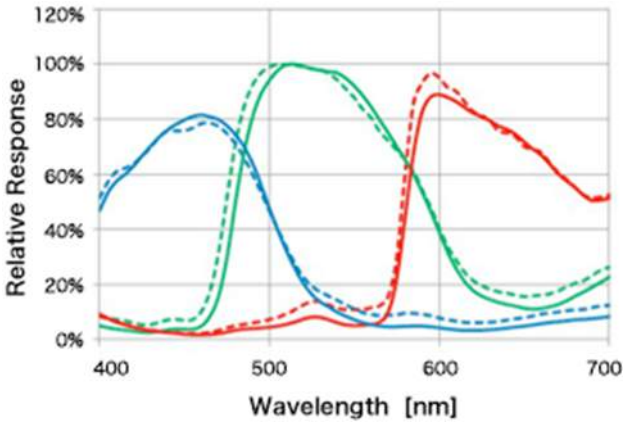


Fig. 6: Spectral characteristics of the IMX214 (dashed line) and IMX135 sensor (solid lone), which are comparable to the CMOS sensor of the Sony α5100 **[2]**.

3 Data storage

3.1 File data structure

Based on the data acquisition concept presented in section 2.3, at each UGV position a tree might be captured several times using varying sensor positions and perspectives. For each capture several sensors might be triggered, which results in multiple files per capture. To limit redundancy, the following hierarchical meta data structure has been elaborated in close cooperation with Task 3.3 - *Definition and Implementation of the Data Repository*

The actual sensor data (*content*) is stored separately from the meta data. This allows for a disc space friendly storage, as well as for subsequent query and modification of all relevant meta data, without the need to access the actual measured data. To allow for a maximal flexibility in terms of file type, the file content is stored in binary format.

Based on the given data structure, the extrinsic orientation of the UGV, image or laser scan can be reconstructed. To ease automation, each *position* is labeled with a field *purpose*, which gives information whether the measurements are intended for *e. g.* sensor calibration, or monitoring. Finally, all images or scans of a specific measurement day, UGV position, or capture can be queried on demand.

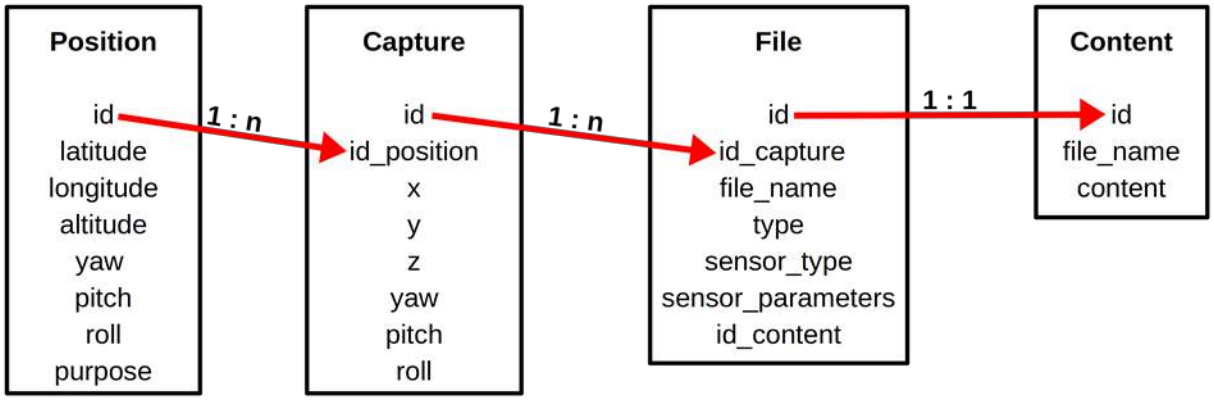


Fig. 7: Elaborated file data structure.

3.2 Other data

Next to files, also other data, like meta data, algorithm parameters or geo-spatial data, need to be stored. To start with the properties of the hazelnut trees are to be stored. Each tree is identified by its unique *id*, which is used to link specific measurements, like information on disease or sucker presence at a specific date. In addition, approximate latitude and longitude coordinates are stored to allow for the selection of neighbored scans.

3.3 Implementation

In Task 3.3 - Definition and Implementation of the Data Repository it is planned to store the laser scans and image files in a MongoDB [3] implementing the given data structure. The metadata will be stored in the widely spread JSON [4] format. This allows for a subsequent extension and adaption of the meta data on demand, without the need to redefine the data structure. Fig. 8 and Fig. 9 illustrate the JSON metadata representation of a file and a tree, respectively. If wished, the geo-data can be converted to common vector file formats like GeoJSON [5].

Position	**capture**	**File**

{
"id":"pos_4",
"timestamp":"2018-5-25 15:15:00",
"latitude":42.27982770,
"longitude":12.29821763,
"altitude":279.664,
"yaw":1.5,
"pitch":3.1,
"roll":2.1,
"purpose":"monitoring"
}
	{
	"id":"cap_20",
	"id_position":"pos_4",
	"x":0.45,
	"y":0,
	"z":0.73,
	"yaw":2.3,
	"pitch":179.4,
	"roll":-1.3
	}
		{
		"id":80,
		"id_capture":"cap_20",
		"sensor_type":"Faro S70",
		"file_type":".las";
		"file_path": "./LAS",
		"file_name":"Scan_080",
		"sensor_parameters":{
		"Resolution":"1/8",
		"Quality":"2x",
		"Distance":"near"
		}
		}

Fig. 8: Meta data of a fictional laser scan.

Tree

```
{
        "id": 1,
        "latitude":42.279809,
        "longitude":12.298184
}
```


Fig. 9: Exemplary JSON representation of a tree.

Since the database will be provided after project month 30 by D3.2: *Data Management*, the *Python* package *tinydb* [6] has been used to emulate the database for the development of the processing pipeline. The file content is stored locally using a common hierarchical file system.

4 Augmented reality markers

To increase the automation level of the UGV processing pipeline, augmented reality (AR) markers have been introduced. Using such markers, relevant meta data to decide how the remote sensing needs to be treated can be extracted automatically based on image analysis.

AR markers are characterized by a typical shape with features of a high contrast. Typically, a black frame surrounded by a white margin is used. Information is encoded binarily using alternating black and white features. The appearance allows for an easy detection of the markers within the image and a reliable extraction of information.

In the PANTHEON project several marker types are used (see Fig. 10). Depending on the application, a unique identifier, the marker's orientation or a well-known pattern is encoded.

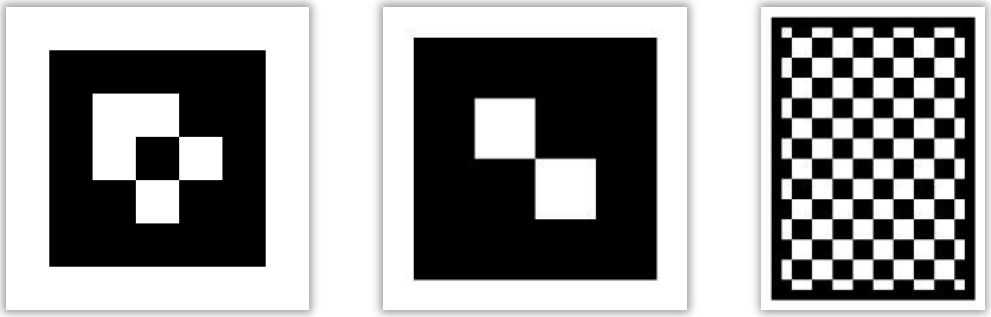


Fig. 10: Examples of markers used in PANTHEON. ID-Marker for the identification of unique objects (left), tie-point marker for the comparison of positions (center) and chessboard marker for the calibration of the sensors (right).

The AR markers are detected using the OpenCV library [7]. In general, the following steps are applied to identify markers in an image and extract the relevant meta data. Elaboration of a UGV data acquisition concept

1. Contrast enhancement of the image
2. Image blurring to remove small objects
3. Binary image creation to highlight the edges of objects
4. Contour detection [8]
5. Selection of potential markers by filtering rectangular contours

6. Validation of the marker properties to filter erroneous detections
7. Decoding of the marker's binary information to extract its type, orientation, or unique identifier

5 Implementation of the terrestrial remote sensing pipeline

5.1 Determination of the demands

The major purpose of Task 4.*2 Terrestrial Remote Sensing Pipeline* is to provide a fully automated processing pipeline to derive high quality three dimensional spectrally enriched point clouds as input for Task 4.3 – *Tree Geometry Reconstruction* and Task 4.8 – *Fruit Detection*.

The algorithms for data evaluation—like tree geometry reconstruction, sucker's detection or fruit detection—are planned to be developed and implemented later in the project. Thus, most requirements on data quality can only be found later in the project by evaluating actual monitoring data with these algorithms. In addition, in the first year of the project the UGV has not been planned to provide actual measurement data, which resulted in the need to collect remote sensing data manually. Because of the limited knowledge on the required data quality and missing actual UGV data, the major task was to design and implement a robust automated pipeline which considers all relevant processing steps but is still flexible for modifications.

To address these requirements the decision has been made to implement the pipeline modular. So, each granular processing step has been implemented either as a standalone command line script, or an importable package. The modules' implemented python packages are characterized by a high level of abstraction, so typically they only need to be fed by the model parameters provided by a configuration file and the remote sensing data. Thus, the processing pipeline has been implemented as a chain of subsequently executed modules, which can be refined, rearranged or replaced easily on demand. Relevant terms and definitions can be found in appendix 10.1.

5.1.1 File type conversion

To retain all available remote sensing information of the hazelnut trees, lossless geo-spatial files are required. Here, common free file formats, independent from the sensor type ease the implementation of the processing pipelines. Since the two UGV cameras may be seen as one virtual camera, each spectral band should be stored separately. Thus, each RGB image needs to be split up in three separate image files.

5.1.2 Vignetting correction

"Vignetting refers to the fall-off pixel intensity from the center towards the edges of the image. This effect is undesirable in image processing and analysis." [9]. This undesirable effect for image processing and analysis is mostly caused by the curvature of the camera lens, but also by sensor design. For digital cameras, also pixel vignetting, caused by the angle dependence of the sensor sensitivity, occurs. Thus, depending on the wavelength, the camera lens, and sensor characteristics, the magnitude of the vignetting effect varies within the image.

As a consequence, a vignetting function has to be calibrated for each sensor individually to compensate for the vignetting effects and to allow for the generation of homogeneous multi-spectral point clouds.

5.1.3 Point classification

To ease the subsequent analysis of the laser scans, a classification as well as a denoising of the point clouds is reasonable. Points associated with ground or noise are typically omitted for further analyses. Since most algorithms—*e. g.* for tree geometry reconstruction or sucker's detection (Task 4.3)—will refine the results, only a coarse pre-classification or outlier detection is required.

5.1.4 Spectral enrichment

To receive the desired spectrally enriched point clouds, for each UGV position, the 2D camera images need to be projected to their corresponding 3D laser scan. The task is to trace the ray of an arbitrary image pixel to find none or many laser points which have captured the same target. Since the viewing perspectives differ between the sensor systems and the ray trace is non-linear due to lens distortion, a detailed knowledge of the imaging geometry is required.

The curvature of the camera lenses leads to a curved projection path resulting in distorted images. To compensate for this effect and reconstruct the projection path, basic camera characteristics, like the focal length of the lens, need to be estimated and the distortion must be modeled. Thus, the intrinsic camera matrix as well as some distortion parameters need to be estimated for each camera band individually.

Since the viewing perspective between the camera sensors differs due to shift and twist of the cameras (inter-camera) and lenses (intra-camera), a band alignment is required to reconstruct the imaging geometry of all sensors during capture. In addition, the alignment of the camera sensors in relation to the laser scanner needs to be reconstructed.

5.2 Structure of the UGV processing pipeline

To prepare the spectral enrichment of the laser scans, the camera images as well as the laser scans need to be pre-processed. After the spectral enrichment, multiple multi-spectral point clouds need to be aligned to achieve a fused multi-spectral point cloud for each tree. Consequently, the processing pipeline illustrated in Fig. 11 has been elaborated. The following partial processing pipelines can be identified:

- Image pre-processing
- Laser scan pre-processing
- Spectral enrichment
- Scan alignment

In the following, methods and algorithms used to implement the given processing pipeline are described in detail.

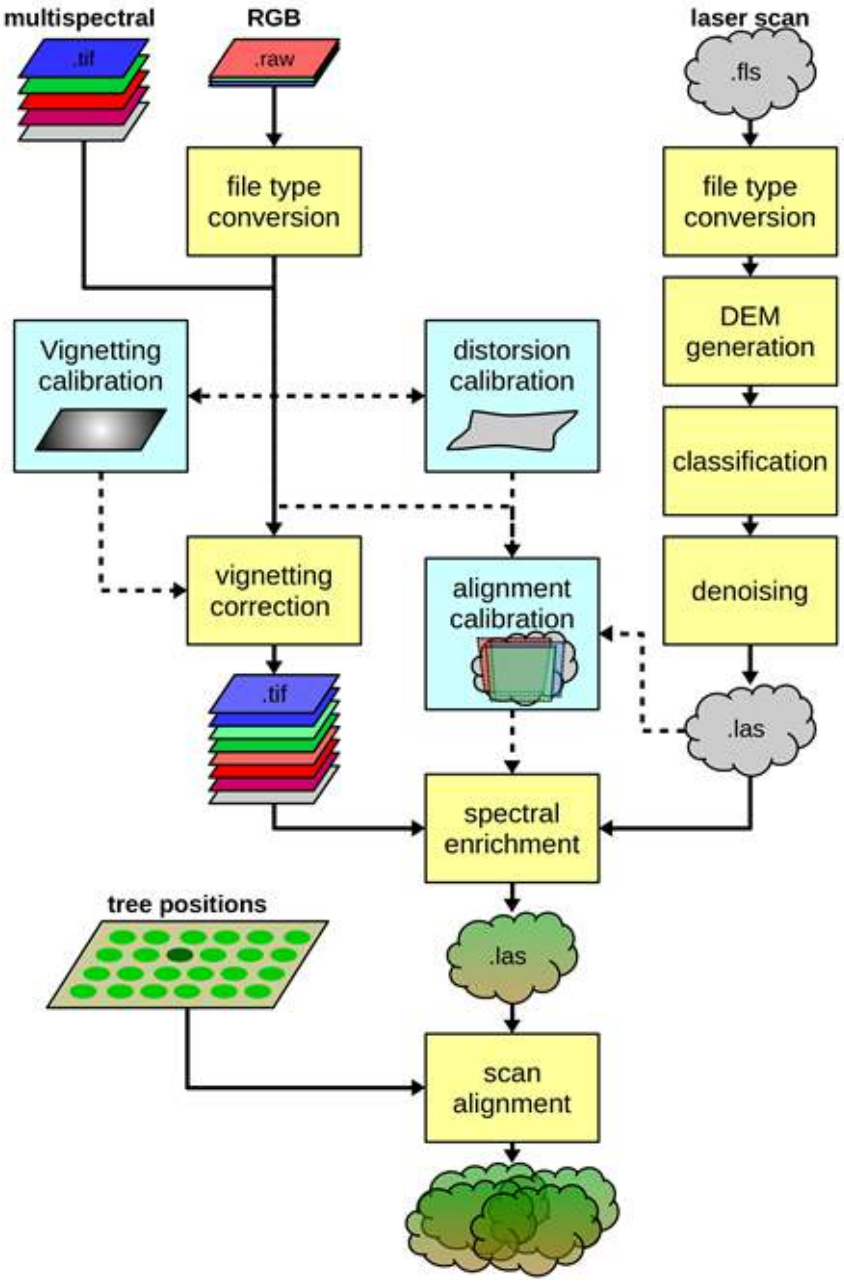


Fig. 11: Structure of the UGV processing pipeline. An image pre-processing pipeline, as well as a laser pre-processing pipeline can be identified. Steps marked in blue are associated with the calibration of the sensors to prepare the spectral enrichment.

5.3 Methods and implementation

5.3.1 Software

Based on the experience of previous research projects, the decision has been made to implement complex algorithms in *Python* using several open source libraries, to speed up software development. Since the used external python libraries typically implement time-critical operations in *C++*, a smooth processing of the pipeline is guaranteed.

5.3.2 Image pre-processing

5.3.2.1 File type conversion

To be able to enrich the camera images with geo-spatial information, all images are converted to the commonly used TIFF format. The MicaSense RedEdge-M provides five .tif images innately, while the Sony α5100 is configured to provide one .arw file per capture—resulting in three .tif images after conversion. The file type conversion has been implemented in *Python* using the packages *rawpy* [10] and *rawkit* [11]. The converted image files are added to the database.

5.3.2.2 Vignetting

5.3.2.2.1 Methods

A commonly used approach to correct vignetting effects is to capture a uniform bright plane several times ensuring a homogeneous illumination. Without vignetting effects, the images are expected to have uniform pixel values with no systematic gradients. Thus, first the pixel values of the images are averaged to remove noise effects. Using the resulting average image A, a vignetting function is fitted and a vignette image V is created. The vignette image is intended to compensate an image I for vignetting effects using equation (1).

$$\check{I}_{i,j} = \frac{I_{i,j}}{V_{i,j}} \tag{1}$$

Two variants of how to calibrate a vignetting function have been implemented. The first variant scales the pixel values by the averaged image to compensate for pixel-wise effects using equation (2). The second variant fits a two dimensional polynomial of degree four to randomly selected pixels of the image using the function *sklearn.processing.PolynomialFeatures* of the Python package *Scikit-learn* [12]. This variant calibrates a continuous vignetting function, which does not model pixel vignetting effects, but is universal.

$$V_{ij} = \frac{A_{ij}}{\bar{A}} \tag{2}$$

5.3.2.2.2 Implementation

To identify images intended for vignetting calibration automatically, the related UGV position meta data is labeled with the *purpose* "vignetting" (see section 3.1). In addition, to identify erroneous calibration images—*e. g.* by missing the white reference target—the brightness and uniformity of the images checked.

An image is classified as bright if its pixel mean value is larger than 20% of the sensor value range and at most 30% of the pixels are overexposed (equal to the maximum sensor value). To check whether an image is almost uniform, the IQR method [13] is used to identify outlying pixels. An image is used to calibrate the vignetting function only if at most 1.5% of its pixels exceed 1.5 times the inter quantile range.

Finally, all images intended for vignetting correction after having passed these tests are used to calibrate a vignetting function for each band individually. After an initial calibration of the vignetting functions, all images are compensated for vignetting effects.

5.3.2.3 Lens Distortion Correction

5.3.2.3.1 Method

A common method to estimate the intrinsic camera orientation and distortion parameters is to capture a real-world object with well-known coordinates—*e. g.* a chessboard—from several perspectives (see section 6.2). By linking the 3D coordinates of the reference target to their corresponding 2D pixel coordinates, the intrinsic camera matrix as well as the distortion parameters are estimated with the method of Zhang [14].

5.3.2.3.2 Implementation

To identify images intended for lens distortion calibration automatically, the related UGV *position* metadata is labeled with the *purpose* "calibration" (see section 3.1). The chessboard markers are detected automatically in each image band using the approach presented in section 4. Since the chessboard pattern is well known, the 2D intersection points of the fields can be identified easily within the image and linked to their corresponding 3D world coordinates, while the z-coordinates are assumed to be zero.

Based on these corresponding point sets, each camera sensor is calibrated individually using the *OpenCV* [7] function *calibrateCamera* which implements the method of Zhang [14]. The calibration is abstracted by a camera object, which handles the detection of the markers as well as the calibration. It can be used to undistort the images on demand. The resulting intrinsic matrices and distortion parameters are stored in the database in JSON format for later reuse or refinement.

5.3.3 Laser scan pre-processing

To prepare the laser scans for spectral enrichment and to enhance the quality of information for thematical analyses, the following laser scan pre-processing pipeline has been elaborated:

- File type conversion and geo-referencing
- DEM generation
- Pre-classification
- Denoising
- In the following the processing steps are explained in detail.

5.3.3.1 File type conversion and georeferencing

The laser scanner utilized in the PANTHEON project provides the point clouds in the proprietary *FLS* format. Since the manufacturer has not been able to provide *Linux* binaries to process the laser data directly by now, the .fls files are converted manually to the more popular *LAS* format using *CloudCompare* [15] with the *FARO LS plugin*. It is planned to stream the raw laser data directly to LAS in the future.

The LAS standard [16] is defined by the American Society for Photogrammetry & Remote Sensing (ASPRS). Next to the storage of laser points, the LAS standard allows for the definition of a spatial reference, the classification of points and the definition of custom attributes.

For D4.1, the classification scheme illustrated in Tab. 2 has been elaborated. If required, the scheme will be refined in the future. Next to the point classes, the classification bit (see Tab. 3) is used to characterize specific point types.

Tab. 2: LAS classification scheme used for D4.1.

LAS class ID	Meaning	Notes
0	not classified (default value)	ASPRS standard
1	not assigned	ASPRS standard
2	ground	ASPRS standard
3	low vegetation	ASPRS standard
4	mean vegetation	ASPRS standard
5	high vegetation	ASPRS standard
7	low point (noise)	ASPRS standard
18	high noise	
19	vehicle	

Tab. 3: Las classification bit according to ASPRS **[16]**.

Bit	Field name	Notes
0 to 4	classification	
5	synthetic	planned to be used in future for artificial points generated by a model
6	key-point	set for characteristic points
7	withheld	

Since most geo-spatial algorithms rely on metric coordinates, the GPS latitude and longitude coordinates—provided by the *position* meta data—are projected to the metric RDN2008 / UTM zone 33N (EPSG: 7792) coordinate reference system. The sensor position also takes into account the relative position of the scanner in relation to the GPS sensor, which is provided by capture meta data. Finally, the spatial reference and the origin of the LAS file are updated. Due to remaining GPS errors, a subsequent alignment of the scans is required (see section 7.8 scan alignment).

5.3.3.2 DEM generation

The height above ground provides useful information to classify the point clouds, to align the scans or to extract the trees. Thus, a digital elevation model (DEM) is generated for each point cloud.

To create a DEM, key points Q^3 that adequately represent the ground are extracted from the original point cloud P^3. The ground filtering is done in two steps. First, a set of key points $K^3 \subseteq P^3$—probably representing ground points—is filtered. Second, the set of key points K^3 is further filtered to ensure a smooth and flat surface. Implementation details can be found in appendix 10.2.2 and at REF *pyoints pyoints.filters.radial_dem_filter*.

Based on the final set of ground key points, a Delaunay triangulation is generated to derive the height above ground for all points of the point cloud. The result is stored in a custom LAS field height. In addition, the LAS classification field of P^3 is initialized. The classification of all points in K^3 is set to 2 (*ground point*) while the key point flag is also set. Points outside the Delaunay triangulation are pre-classified as 1 (*not assigned*).

5.3.3.3 Classification

To ease and speed up subsequent processing steps, a preliminary classification is performed. The classification results are stored by updating the LAS classification field. The point cloud is sliced vertically, to apply the classification scheme of Tab. 4.

Tab. 4: Classification scheme used for point type classification based on height.

height above ground (m)	class number	meaning
up to -0.05	7	low noise
-0.05 to 0.01	2	ground
0.01 to 0.15	3	low vegetation
0.15 to 0.75	4	intermediate vegetation
0.75 to 10.0	5	high vegetation
larger than 10.0	18	high noise

Due to the scanner characteristics, the ground points are typically the most frequent class of points, but ground points are usually not considered for further analyses. The class *low vegetation* is associated with grass, stems and suckers. The class *intermediate vegetation* should be mostly characterized by stems, branches, and large suckers. Most branches and leaves will fall in class *high vegetation*. Unexpectedly low or high points fall in the class *low noise* or *high noise*, respectively.

Since the laser scanner is attached upside-down to the robot, also large parts of the UGV are recorded by the laser scanner. Thus, points in the surrounding of the scanner are classified as *vehicle* (class number 19).

5.3.3.4 Denoising

The noise of the laser scans is mostly characterized by isolated points, but also by sparse linear traces of points at edges of objects, as an effect of partially hitting the object with the laser beam (see Fig. 27). Due to the scanning geometry, the distance between neighbored points increases linearly with increasing distance to the scanner. Thus, a distance adaptive approach is used to decide if a point is isolated or not.

A point p is classified as noise, if within radius r_p no neighbors ($|P^3_{r_p}(p)^2| \leq 1$), or within radius $2 \cdot r_p$ at most one neighbor can be found ($|P^3_{2r_p}(p)^2| \leq 2$). By defining $r_p = \sin(\alpha) \cdot \|p\|_2$, with angle α depending on the scanner resolution, the noise filter is adaptive to the scanner settings. The noise information of the filtered points is stored by setting the LAS classification field to 7.

5.3.4 Sensor alignment

5.3.4.1 Method

To align the camera sensors with the laser scanner, the chessboard marker already used for distortion correction (see section 5.3.2.3) are used. Again, for calibration, the chessboard—defining a well-known local coordinate system—is scanned from several perspectives, which allows for a back projection of the 2D image to 3D world coordinates.

In detail, the laser scanner serves as a fixed reference. A synthetic 2D intensity image is extracted from the point cloud in front of the laser scanner (see Fig. 12). To compensate the image for directional reflectance effects, a polynomial of degree four is fitted to receive homogeneous intensity values. If a chessboard marker is detected in such an image, the extrinsic orientation of the image in relation to the chessboard is calculated by estimating its pose using the 3D chessboard coordinates and the corresponding 2D image pixels. Based on this pose the 3D chessboard coordinates in relation to the laser scanner are calculated.

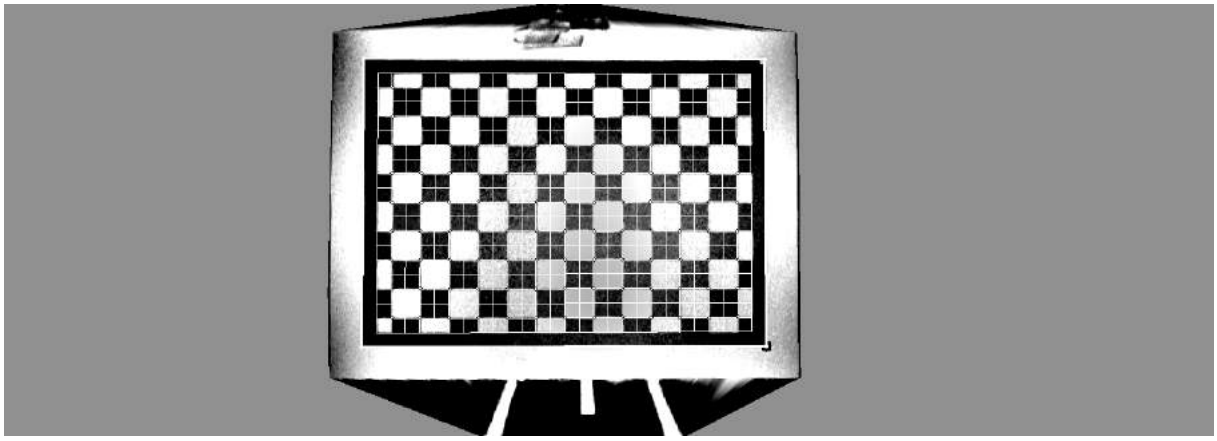


Fig. 12: Synthetic intensity image extracted in front of the laser scanner. The image has been augmented with the detected chessboard (white lines and black circles).

In a second step, the camera sensors are aligned by detecting the chessboard in the images and calculating the pose of the chessboard for each band individually. By using the 3D chessboard coordinates in the reference system of the scanner, the sensors are aligned to the laser scanner.

Since the accurate positions of the camera sensors are hard to measure, an arbitrary point within the gimbal serves as a fixed reference. By calculating the deviation between this reference and the extrinsic orientation derived from the chessboard, a corrective roto-translation matrix is calibrated for each band. In consequence, the calibration results in a roto-translation matrix for each camera band to refine the provided approximate extrinsic orientation of the gimbal in relation to the laser scanner.

5.3.4.2 Implementation

To calibrate the sensor alignment, a virtual camera object—gathering functions required for sensor alignment calibration—is initialized. It provides the roto-translation matrices for the refinement of the extrinsic orientation of each camera sensor. Since the band alignment and the camera calibration are related, it also holds an object for lens distortion calibration (see section 5.3.2.3) for each band.

For calibration, all UGV positions labeled with *purpose* "calibration" are inspected. The chessboards are detected in both the camera images and the artificial laser scan image. If the chessboard is not detected in at least one of the images, it is not used for the sensor alignment, but for distortion correction. The required sensor positions and orientation are provided by the UGV *capture* meta data. To estimate the pose of a sensor in relation to the other sensors, the chessboard marker, while the *OpenCV* function *stereoCalibrate* calibrates the camera band alignment.

After the calibration the camera object is stored in the database for later reuse and spectral enrichment of the laser scans.

5.3.5 Spectral enrichment

To receive multi-spectral point clouds, each laser scan is enriched with all images taken from the same UGV position. Each vignetting corrected single band image (see section 5.3.2.2) is projected to its corresponding point cloud using its extrinsic matrix, intrinsic matrix and lens distortion parameters. This 2D to 3D projection corresponds to a ray-tracing of the image pixels and has been implemented in python using the *OpenCV* function *projectPoints*. The extrinsic matrices are derived from the meta data of the UGV and are refined by the sensor alignment calibration (see section 5.3.4). The intrinsic matrix as well as the distortion parameters of the cameras are provided by the lens distortion calibration (see section 5.3.2.3).

To store the spectral information, a custom field is added to the LAS files for each camera band. In case of multiple captures per position and camera, some laser points might be assigned by several image pixels of the same band. If so, the pixel values are averaged.

5.3.6 Laser scan alignment

5.3.6.1 Methods

The scan alignment problem is solved locally for each tree individually. Thus, for each tree, date, and scanner parameters, the closest surrounding scans are aligned. To avoid redundancy, the alignment information is used to create a joint point cloud of the tree on demand only.

Assuming infinitesimal rotations, two or more laser scans can be aligned using the least squares method if a sufficient number of matching points is known (see appendix 10.2.3 for details). To find matching points and to align the scans, a variant of the iterative closest point (ICP) algorithm [17], [18], [19] has been developed

and implemented. Besides taking into account the point coordinates it also uses the point normals (see appendix 10.2.4 for details).

Compared to the vertical alignment of terrestrial laser scans, the horizontal alignment is much prone to errors. To achieve a robust method, the decision was made to split the alignment in three sequential steps:

1. horizontal alignment
2. vertical alignment
3. refinement

5.3.6.1.1 Horizontal alignment

In the hazelnut orchard, points close to the root of the hazelnut trees bear much information on the horizontal displacement of the scans. Thus, points associated with intermediate vegetation (see section 5.3.3.3) are selected. To limit the computational effort, the points are filtered by a duplicate point filter (see appendix 10.2.1) of radius ϵ, while points with less than two neighbors are omitted. To compensate for a vertical displacement the height values derived in section 5.3.3.2 are used as the z-coordinates.

Finally, our ICP variant (see appendix 10.2.4) is applied to achieve an approximation of the roto-translation parameters. To avoid the effect of erroneous point assignments and limit the degree of freedom, the ICP variant is applied several times with decreasing values of the resolution vector δ. For example by beginning with $\delta = \left(1.2,1.2,0.1, \sqrt[3]{\sin(15°)}, \sqrt[3]{\sin(15°)}, \sqrt[3]{\sin(15°)}\right)$—associated with GPS and compass errors up to 1.2m, respectively 15°—and finally with $\delta = \left(0.05,0.05,0.1, \sqrt[3]{\sin(5°)}, \sqrt[3]{\sin(5°)}, \sqrt[3]{\sin(5°)}\right)$.

5.3.6.1.2 Vertical alignment

To align the scans vertically, for each point cloud a random subset of the points already used for DEM generation is selected. Then, the closest ground point in each neighboring scan is selected. By applying this procedure for all neighbored scans, for each point cloud a subset particularly suitable to refine the vertical alignment is filtered. Finally, our ICP variant is applied once, to refine the previously derived horizontal alignment.

5.3.6.1.3 Refinement

To diminish remaining alignment errors—caused by the separate horizontal and vertical processing—the point subsets derived for horizontal and vertical alignment are merged to joint subsets. Finally, the ICP algorithm is applied a last time, using the previously derived vertical alignment as an initial point, to provide the final roto-translation matrix for each input point cloud.

5.3.6.2 Implementation

For each tree, the four surrounding laser scans of the same measurement day are aligned, which results in a roto-translation matrix for each of the input point clouds. The alignment matrices are stored with supplementary information in JSON format. Since the alignment meta data is linked to a specific tree, as well as to the neighbored scans, the alignment can be performed on demand. Thus, no laser scan file needs to be duplicated. An example of potential alignment meta data is illustrated in Fig. 13:

Alignment

```
{
        "id": "tree_field16_1_2018-05-25",
        "id_tree": "tree_field16_1",
        "date": "2018-05-25",
        "sensor_parameters": { … },
        "capture_alignment": {
                "cap_49": [
                        [  0.999,  0.012, 0.004, -59816.234],
                        [ -0.012,  0.999, 0.003,   3952.623],
                        [ -0.004, -0.003, 0.999,  15806.961],
                        [       0,       0,       0,       1]
                ],
                "cap_51": [ … ],
                "cap_53": [ … ],
                "cap_55": [ … ]
        },
        "report": { … }
}
```

Fig. 13: Exemplary alignment meta data of a tree in JSON format.

6 Collection of training data

6.1 Vignetting calibration

Data intended for vignetting calibration of the Sony a5100 and the MicaSense RedEdge-M have been collected at the end of November 2018 in a laboratory of UNIROMA3. To ensure a sufficient illumination intensity, the white reference plane has been measured in the proximity of large windows at midday, while all lamps of the lab have been turned on. The illumination has been almost constant during measurement, due to a closed cloud cover. The cameras have been targeted towards the windows, to ensure a uniform diffuse illumination of the target without shadowing effects. The viewing perspective has been slightly varied between the captures using the gimbal to compensate for remaining illumination effects. The meta data has been created manually in JSON format.

6.2 Sensor calibration

To calibrate the cameras—intrinsic orientation and lens distortion—and to align the sensors, experiments intended for the sensor calibration have been performed at the end of November and beginning of December 2018 in a laboratory of UNIROMA3.

As indicated in sections 5.3.2.3 and 5.3.4, the lens distortion correction, as well as the sensor alignment requires capturing the chessboard marker from several perspectives. This can be achieved by swiveling the horizontal arm of the UGV, as well as rotating the gimbal. The experimental setup is illustrated in Fig. 14 and Fig. 15. In general, the chessboard is measured from four UGV positions with both the laser scanner and the cameras.

The first position is intended to take a fully covered image of the marker. The second to fourth positions shall provide variability by capturing the marker five times with the cameras, four in the corners and one in the center of the images. By providing this variance, the calibration is expected to be more robust and accurate

than by just using one position. For simplification, the chessboard marker is attached to a tripod, which enables a repositioning of the marker instead of the UGV.

Since the calibration procedure requires for the extrinsic orientation of an arbitrary fixed point within the gimbal and the extrinsic orientation of the laser scanner as described in section 5.3.4, either the UGV kinematics or similar data need to be added to the meta data.

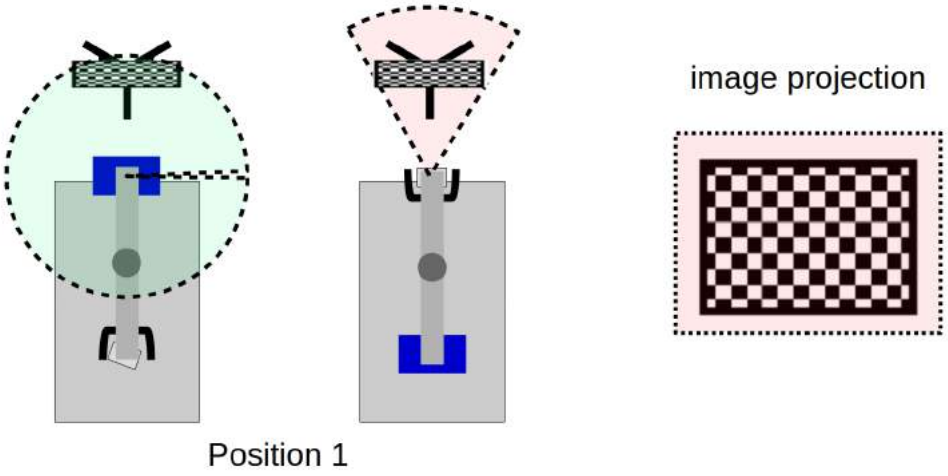


Fig. 14: Experimental setup for the UGV sensor calibration with low distance to the chessboard marker. The reference marker is captured once with the laser scanner and once with both cameras. The marker should cover the images almost completely.

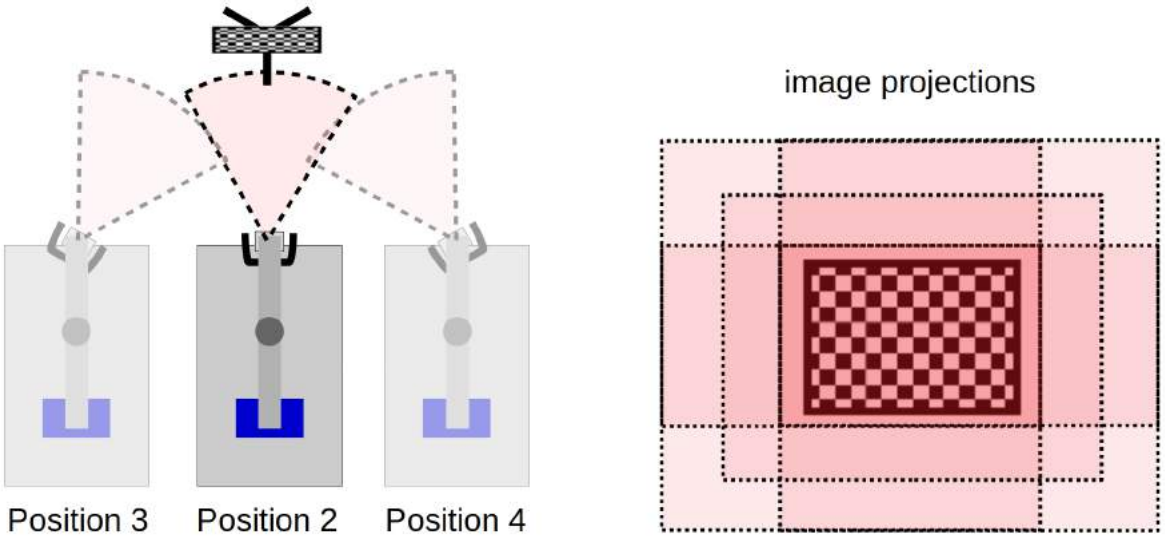


Fig. 15: Experimental setup for the UGV sensor calibration with high distance to the chessboard marker. The reference marker is captured once with the laser scanner and once with both cameras at each of the tree positions. The marker should cover about one third to one half of the images.

Since at the time of the experiments the UGV has not been functioning yet, a UGV dummy—able to simulate the swiveling of the horizontal arm of the UGV by attaching the laser scanner and the gimbal to a table—has been built. The exact positions and orientations of a point within the gimbal as well as the chessboard have been measured using the optical motion capturing system *OptiTrack* [20]. For this reason, optical markers have been attached to the MicaSense camera and the chessboard (see Fig. 16).

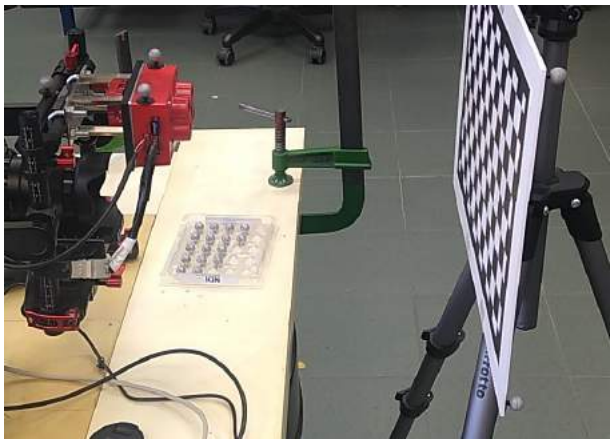


Fig. 16: Experimental setup with *OptiTrack* markers attached.

Several tests have shown that the automatic exposure settings, with a focus of the Sony camera of about 0.4m, should be used. In addition, the marker has always been scanned two times with the two reasonable laser scanner resolutions “1/4” and “1/8”.

6.3 Field campaigns

To collect test and training data, several measurement campaigns have been performed in the hazelnut orchard. To save resources and time, field data has been collected to address multiple purposes at once. So, it is planned to reuse some of the data for training the tree geometry reconstruction and sucker detection (Task 4.3).

6.3.1 May 2018

To get a first impression of the actual UGV sensor characteristics and to collect training data for algorithm development, a field campaign was conducted from 23th to 25th May 2018. The campaign had the following objectives:

a) Finding optimal laser scanner settings
b) Testing different camera settings
c) Verifying monitoring design
d) Collecting training data for laser scan alignment
e) Collecting training data for distortion correction

Since at the date of measurement the UGV was not available yet, the sensor data was collected manually. To still be able to use the data for subsequent analyses, the laser scanning data was linked to GPS RTK measurements. By using a tripod, a consistent height above ground of about 1.7m has been ensured.

To prepare future measurement campaigns, several laser scanner settings were tested. For operational reasons the scanning time had to be restricted to at most two minutes. Thus, the point cloud quality had to be balanced against scanning time. In consequence, combinations of the following parameters were tested systematically to get an impression of their effect on the point cloud quality.

- Distance: [“near”, “normal”]
- Resolution: [“1/8”, “1/10”, “1/16”, “1/20”, “1/32”]
- Quality: [“2x”, “3x”, “4x”, “6x”]

To verify the suitability of the striven monitoring design and to have test data for scan alignment, four middle-aged (field 18) as well as four young-aged hazelnut trees (field 16) were measured with the cameras (Sony α5100 and the MicaSense RedEdge-M) as well as with the laser scanner (Faro S70) following the monitoring design presented in section 2.3. To minimize the measurement effort and to receive scans suitable for scan alignment, only neighbored trees were selected. This resulted in ten positions at field 16 and ten positions at field 18 (see Figure Fig. 17).

Fig. 17: Map of the laser scanner positions located in field 16 and field 18.

Next to the measurements of the trees, additional reference data was recorded. A chessboard marker was captured with both cameras several times for training distortion correction and camera band alignment. Additionally, a reference panel and a black reference (covered lens) was captured to find suitable exposure settings and to get an impression of the signal to noise ratio. Due to constantly changing illumination conditions, caused by clouds, the exposure time of the MicaSense camera was set to automatic.

6.3.2 September 2018

On 10th September twelve mature trees of field 20—six without suckers and six with suckers—were measured with the laser scanner according to the monitoring scheme presented in section 2.3. The data is intended for a later training of the suckers’ detection algorithm (Task 4.3) and to verify the scanner settings determined in the previous campaign. For practical reasons the trees were scanned manually using a tripod without the GPS.

7 Results and Discussion

To evaluate the success of Task 4.3, the elaborated monitoring design has had to be checked for its general suitability and the results of the processing have been checked for correctness. Wherever reasonable, the accuracy of the algorithms has been assessed statistically.

7.1 Monitoring design

To verify the monitoring design, the camera images taken in May 2018 (see section 6.3.1) were inspected. As originally planned, the young trees can be captured with one image per camera (see Fig. 18). Due to technical limitations of the gimbal, in future the UGV will capture young trees with two shots instead. Although the viewing perspective differs for each band—caused by different lens positions and aperture angles—the tree can be captured by all bands completely (see Fig. 19).

As expected, taller trees need to be captured six times at each position to cover them almost completely (see Fig. 20). A later image fusion of images with fixed exposure time seems reasonable, since the illumination of this preliminary pseudo-panorama image seems consistent.

Based on these results the proposed monitoring design has been confirmed to be generally suitable for the PANTHON project. A refinement of the number of images taken for tall trees does not seem to be necessary.

Fig. 18: RGB images of a young hazelnut tree taken from north-east (left), north-west (center-left), south-east (center-right) and south-west (right) position.

Fig. 19: Images of RGB and multispectral camera taken form north-west position (RGB, blue, green, red, red-edge, NIR).

Fig. 20: Pseudo-panorama RGB image composition of a middle-aged hazelnut tree. The images have been taken from the same position, but with different viewing directions.

7.2 Laser scanner settings

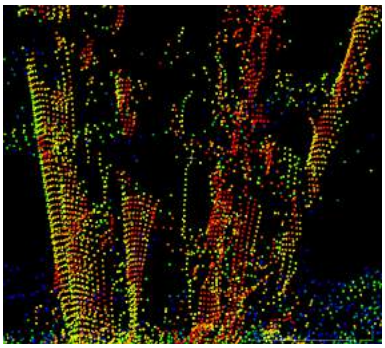
To select proper laser scanner settings for future measurement campaigns, the parameters *Distance*, *Resolution* and *Quality* have been analyzed using the field data collected in May 2018 (see section 6.3.1). The focus has been set on the effect of the parameter on the point cloud quality under the restriction of a reasonable scanning time.

7.2.1 Resolution

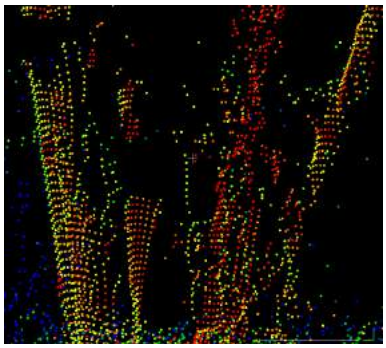
To identify suckers and branches automatically (Objective 2.1), they need to be represented by the point cloud. A sucker or branch has to be hit several times by the laser beam to allow for a visual identification of the sucker or to model the branching structure and branch diameter.

The scanner parameter *Resolution* defines the step width of the laser beam between two pulses. With highest resolution “1/1” a step width of 0.009° is achieved. The step width is divided by the resolution factor, which results in increased point distances with decreasing resolution factor. For example, a *Resolution* of “1/8” results in a minimal point distance of about 0.38mm at 3m distance to the scanner.

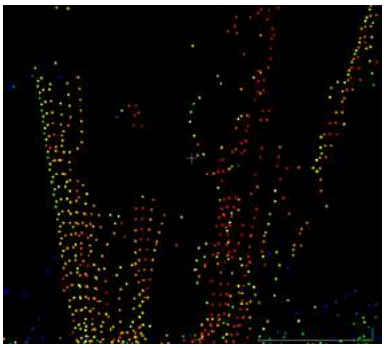
Fig. 21 displays three laser scans of the root area of a hazelnut tree, scanned from the same perspective. Suckers are represented by linearly arranged points (left image). With decreasing resolution, the sucker is hit more rarely by the laser beam (center image). For low resolutions, the suckers are no longer identifiable and even the branches are barely represented (right image). Based on these results, resolutions of at least “1/8” are expected to be sufficient to identify suckers with a high certainty, while lower resolutions might result in increased omission errors.



Resolution: **1/8**; Quality: 3x; Distance: near



Resolution: **1/10**; Quality: 3x; Distance: near



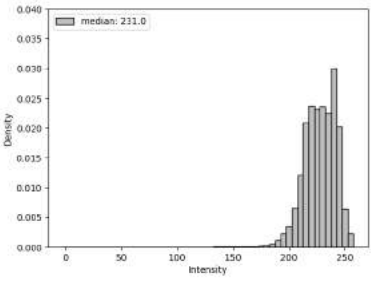
Resolution: **1/16**; Quality: 3x; Distance: near

Fig. 21: Effect of the scanner parameter *Resolution* on the representation of suckers and branches. The resolution decreases from "1/8" (left) over "1/10" (center) and "1/16" (right), while the *Quality* and *Resolution* settings are fixed. The intensity values have been color coded from blue (low intensity) over green (intermediate intensity) to red (high intensity).

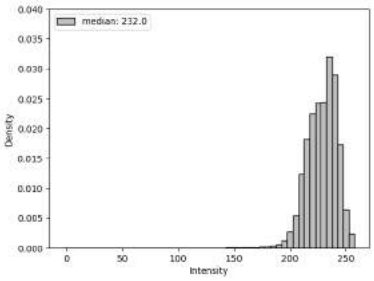
7.2.2 Distance

Fig. 22 illustrates that the measured intensity values tend to be close to the upper value range of the laser scanner of 255 in combination with a slight right skewing of the value distribution, as an indicator of overexposure. The overexposure is most likely caused by the close distance of the scanner to the target, which results in a high amount of energy reflected to the scanner. This effect is strengthened by the high reflectance of green vegetation at the laser scanners' wavelength of 1550nm (see Fig. 22).

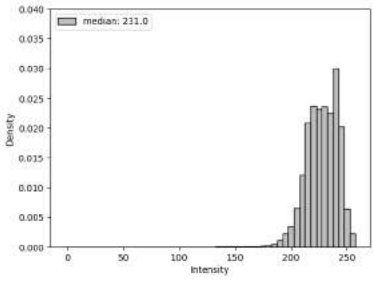
To compensate this effect, the scanner parameter *Distance* should be set to "near", which results in slightly lower intensity values and a more symmetrical histogram. Especially for bright targets, like branches, better results are achieved. Filially, the *Distance* parameter should also improve the 3D precision of the points.



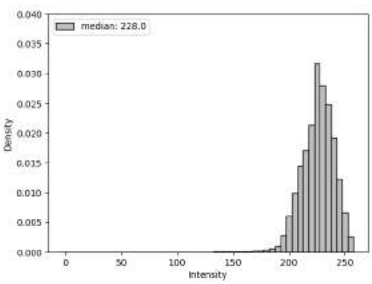
Resolution: 1/8; Quality: 3x; Distance: **normal**



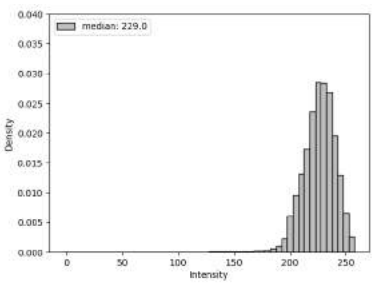
Resolution: 1/10; Quality: 3x; Distance: **normal**



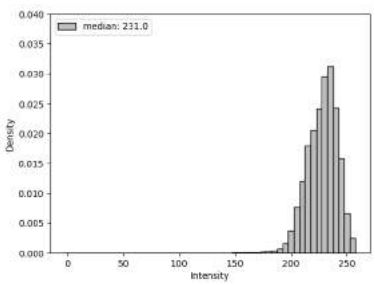
Resolution: 1/16; Quality: 3x; Distance: **normal**



Resolution: 1/8; Quality: 3x; Distance: **near**



Resolution: 1/10; Quality: 3x; Distance: **near**



Resolution: 1/16; Quality: 3x; Distance: **near**

Fig. 22: Effect of the scanner parameter *Distance* on the measured intensity values. In particular, scans with *Distance* "normal" tend to be overexposed.

7.2.3 Quality

To reduce noise and increase point accuracy, the parameter *Quality* can be set to higher values at the expense of increased scanning times. Thus, we have investigated the effect of the parameter Q*uality* to decide if a high scanning quality is required, or if a post-processing is sufficient.

To investigate the effect of the *quality* parameter on the proportion of noise, the denoising algorithm (see section 5.3.3.4) has been applied to all points associated with vegetation (see section 5.3.3.3) using a *Resolution* of "1/8" and reasonable *Quality* values—in terms of scanning time—of *"2x", "3x" and "4x"*. About 9% to 10% of the points have been classified as noise, while no significant effect of the *Quality* parameter could be observed.

7.2.4 Reasonable parameter sets

Based on the previously presented findings, the following parameter sets have been identified for future field campaigns:

- *Resolution*: "1/4", *Quality*: "1x", *Distance*: "near"
- *Resolution*: "1/8", *Quality*: "2x", *Distance*: "near"

Due to the decreased scanning time of 1:27 minutes, the second parameter set is preferred over the first set with a scanning time of 1:54 minutes.

7.3 Camera settings

The images captured during the field campaign in May 2018 (see section May 20186.3.1) have been investigated to find suitable camera settings. The most important camera parameters are exposure time and focus.

The focus of the Sony α5100 camera should be adapted to the expected average target distance of about 3. The focus of the MicaSense RedEdge-M camera is infinity due to camera design.

Changing illumination conditions—caused by season, time of day, weather or shadows—require a continuous adaption of the exposure time. On the other hand, to compare several images, fixed exposure settings should be used. To balance these contrasting requirements, future field campaigns should use the same exposure time during a whole measurement day, although this could result in some underexposed or overexposed images. Thus, at the beginning of the measurement day a sufficient exposure time will be received from images taken with automatic exposure settings.

7.4 Image pre-processing chain

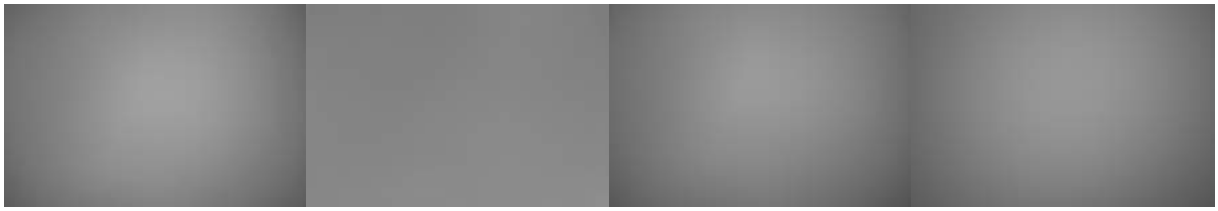
7.4.1 Vignetting correction

To assess the correctness and accuracy of the vignetting correction algorithms, the calibration data taken at the end of November 2018 (see section 6.2) has been used for calibration.

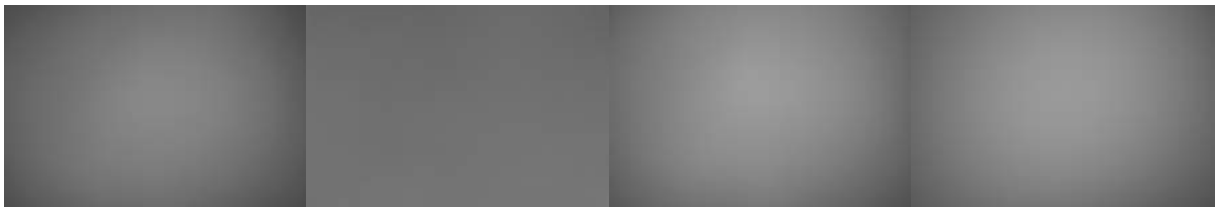
Depending on the band, the algorithm used a selection of five to seven suitable captures for vignetting calibration. After the calibration of the vignetting functions all images have been corrected for vignetting effects. Fig. 23 illustrates the vignetting correction for all image bands: the vignetting effects are removed almost completely, indicated by uniformly gray corrected images.

Band 1

Band 2



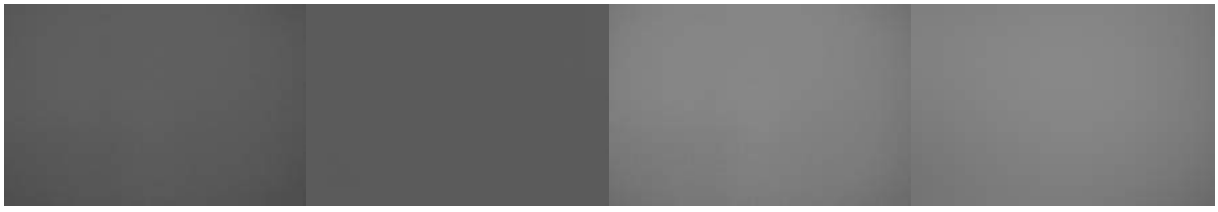
Band 3



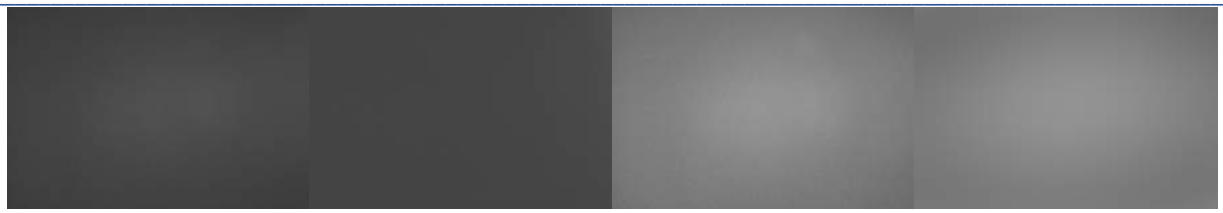
Band 4

Band 5

Band 6



Band 7



Band 8

Fig. 23: Illustration of the vignetting correction for all eight camera bands using a white reference target. Image before vignetting correction (left) and after vignetting correction (centre-left) using the average vignetting function (centre-right). For completeness, also the polynomial vignetting function is displayed (right).

To assess the accuracy of the correction, the proportion of variance explained by the vignette R^2, as well as the standard deviation before and after the vignetting correction have been calculated for all images used for the calibration. Tab. 5 summarizes the findings for all bands using the pixel average vignetting method. The standard deviations of the images before vignetting correction range from 7.1% to 11.9%. The standard deviations after vignetting correction range from 2.2% to 6.4%. This intended decrease indicates more homogeneous pixel values. The R^2 values range of the RGB camera are in a range of 0.91 to 0.97. Thus, the vignetting function explains most of the variance of the image. In contrast, the R^2 values of the multi-spectral camera are in a range of 0.46 to 0.71, indicating slighter vignetting effects in relation to the image noise.

Tab. 5: Results of the vignetting calibration.

band	sensor	wavelength region	standard deviation before correction (%)	mean standard deviation after correction (%)	mean R^2
1	Sony α5100	red	11.35 ± 0.45	3.00 ± 0.35	0.94 ± 0.01
2	Sony α5100	green	11.67 ± 0.30	2.15 ± 0.26	0.97 ± 0.01
3	Sony α5100	blue	13.09 ± 0.45	4.13 ± 0.44	0.91 ± 0.01
4	MicaSense RedEdge-M	blue	7.09 ± 0.05	4.02 ± 0.03	0.68 ± 0.01
5	MicaSense RedEdge-M	green	7.74 ± 0.04	4.38 ± 0.11	0.68 ± 0.01
6	MicaSense RedEdge-M	red	8.35 ± 0.04	5.28 ± 0.03	0.60 ± 0.00
7	MicaSense RedEdge-M	near infrared	11.90 ± 0.07	6.37 ± 0.08	0.71 ± 0.01
8	MicaSense RedEdge-M	red-edge	7.16 ± 0.02	5.28 ± 0.03	0.46 ± 0.00

7.4.2 Distortion correction

To assess the correctness and accuracy of the camera calibration and distortion correction, the calibration data taken at begin of December 2018 (see section 6.2) was used for calibration. Five distortion coefficients

were calibrated. Fig. 24 illustrates the distortion of a random image of the chessboard marker. Table Tab. 6 summarizes the results.

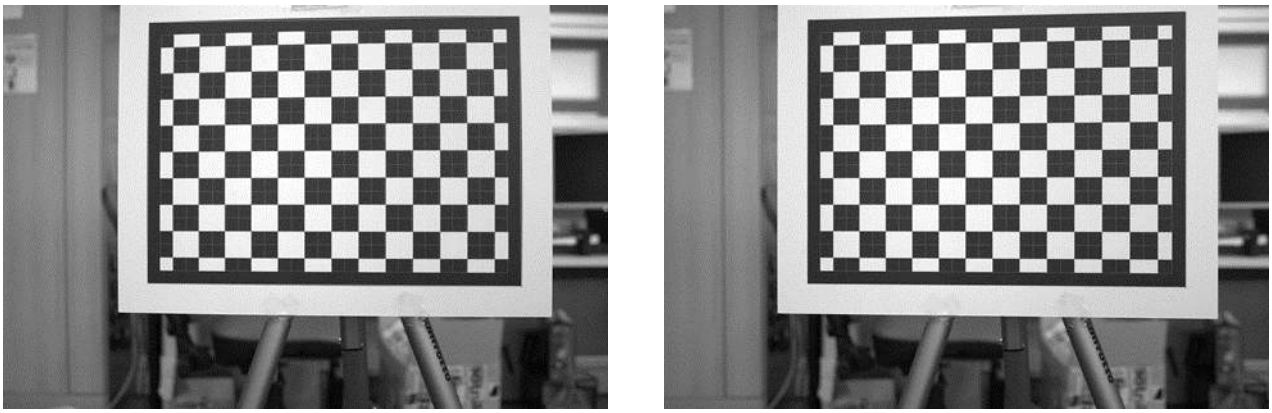


Fig. 24: Images of the chessboard pattern before (left) and after (right) distortion correction. The images have been augmented with the detected chessboard pattern (fine white lines and black dots). A slight Barrel distortion of the uncorrected image is recognizable.

Tab. 6: Summary of the camera calibration results for each camera band.

Band	Sensor	Focal length (mm)	Distortion parameters k_1, k_2, p_1, p_2 and k_3 (-)	Re-projection error (pixels)
1	Sony α5100	43.1	-0.15, 0.201, 0.001, -0.0, -0.555	0.92
2	Sony α5100	43.1	-0.146, 0.162, 0.001, -0.0, -0.391	0.44
3	Sony α5100	43.1	-0.147, 0.199, 0.001, -0.0, -0.547	1.00
4	MicaSense RedEdge-M	35.7	-0.113, 0.307, 0.001, 0.001, -0.437	0.22
5	MicaSense RedEdge-M	35.7	-0.099, 0.161, 0.001, 0.0, -0.06	0.13
6	MicaSense RedEdge-M	35.6	-0.11, 0.251, 0.001, -0.0, -0.27	0.38
7	MicaSense RedEdge-M	35.7	-0.104, 0.209, 0.001, -0.001, -0.245	0.22
8	MicaSense RedEdge-M	35.8	-0.108, 0.201, 0.001, 0.001, -0.125	0.22

With a focal length of 43.1mm for all three bands, the camera calibration of the RGB camera provides consistent results. Also, the calibrated focal lengths of multispectral camera of about 35.7mm are in line with the expectations. The variation of the focal lengths could be caused by slightly differing lens characteristics. The re-projection errors of the RGB camera are with values up to one pixel higher than the re-projection error of the multispectral camera. This seems to be caused by the roughly four times higher resolution of the RGB camera in combination with larger distortion effects.

7.5 Laser scan pre-processing

7.5.1 File type conversion and geo-referencing

Since the manufacturer of the laser scanner could not provide the Linux binaries until submitting this deliverable, the conversion from the proprietary FLS format to the common LAS formats still needs to be done manually. Thus, only the georeferencing step could be checked for correctness successfully.

7.5.2 DEM generation

To evaluate the correctness of the DEM generation algorithm, the plausibility has been checked visually. The visual plausibility check has not revealed any apparent malfunction of the algorithm. But, especially in regions with low point densities—e. g. caused by shades or far distant areas—the filtering algorithm cannot calibrate a proper DEM, due to a lack of information.

To assess the quality, the DEMs of different scans at the same position have been compared. If the DEM filtering algorithm generates meaningful and robust results for different scanner resolutions, vertical differences between the DEMs close to zero can be expected. Ten positions at field 16 and ten positions at field 18 of the campaign in May 2018 have served for evaluation. At each position two scans—one with a *Resolution* of "1/4" and one with "1/8"—have been investigated. For both point clouds, the DEM points have been filtered (see section 5.3.3.2) and a Delaunay triangulated DEM has been generated. For both scans, the residuals (vertical distances) between the DEM points and the triangulated DEM of the other scan have been calculated. To avoid boundary effects, only DEM points with a distance less than 6m to the scanner center have been considered.

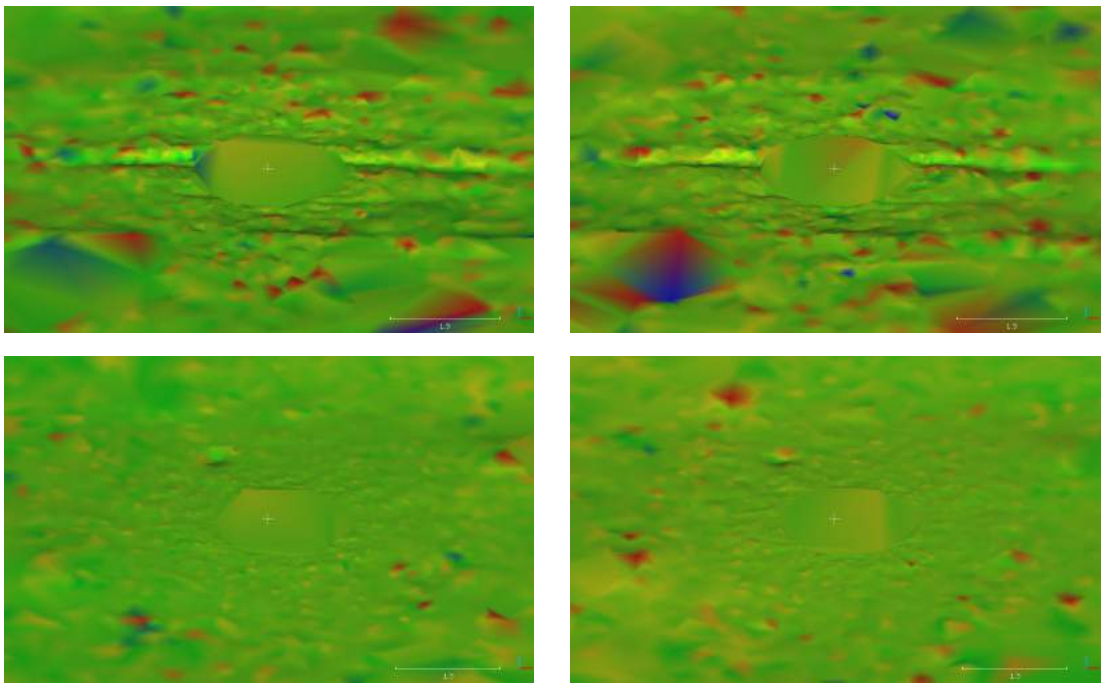


Fig. 25: Visualization of the DEM differences of two point clouds, scanned at the same position in field 16 (top) and field 18 (bottom), with *Resolution* "1/4" (left) and "1/8" (right). The residuals have been color-coded ranging from blue (-3 cm) over green to red (+3cm).

The results of the residual analysis are summarized in Tab. 7. In general, slight vertical differences between the DEMs occur. With a standard deviation of about 5.3mm, 99% of the differences are in a range of 4.2cm. Some of the differences between the DEMs are caused by slight offsets between the point clouds caused by compass errors.

The larger residuals of field 16 compared to field 18—with standard deviations of 7.0mm, respectively 3.7 mm—are caused by a differing surface roughness. Field 18 is characterized by a float compressed clayey ground, while field 16 is characterized by a sandy, but rough surface. Thus, to model the surface of field 18 more accurately the resolution of the DEM would have to be increased.

Tab. 7: Aggregated results of the DEM accuracy assessment. At each position the DEMs received from two different point clouds have been compared.

area	mean (mm)	standard deviation (mm)	99% range (mm)
overall	-0.79	5.28	-26.4 – 13.8
field 16	-1.15	7.00	-30.9 – 17.9
field 18	-0.47	3.68	-16.8 – 10.2

The given results indicate adequare DEM qualities for subsequent analyses. Significant differences between the DEMs mainly occur in shaded areas behind the hazelnut tree, which are of low interest for the subsequent tasks. A further refinement might be required only if the identification of small suckers would be simplified by more accurate elevation models.

7.5.3 Classification

Since the classification of the laser scans is based on the DEM and extent of the UGV only, a plausibility check is sufficient to evaluate the algorithm. In particular, the classification is intended for a simplified selection of points with similar characteristics only—*e. g.* to ease algorithm development—only approximate results are required. Fig. 26 illustrates the classification results of laser scans captured in May 2018 (see section 6.3.1).

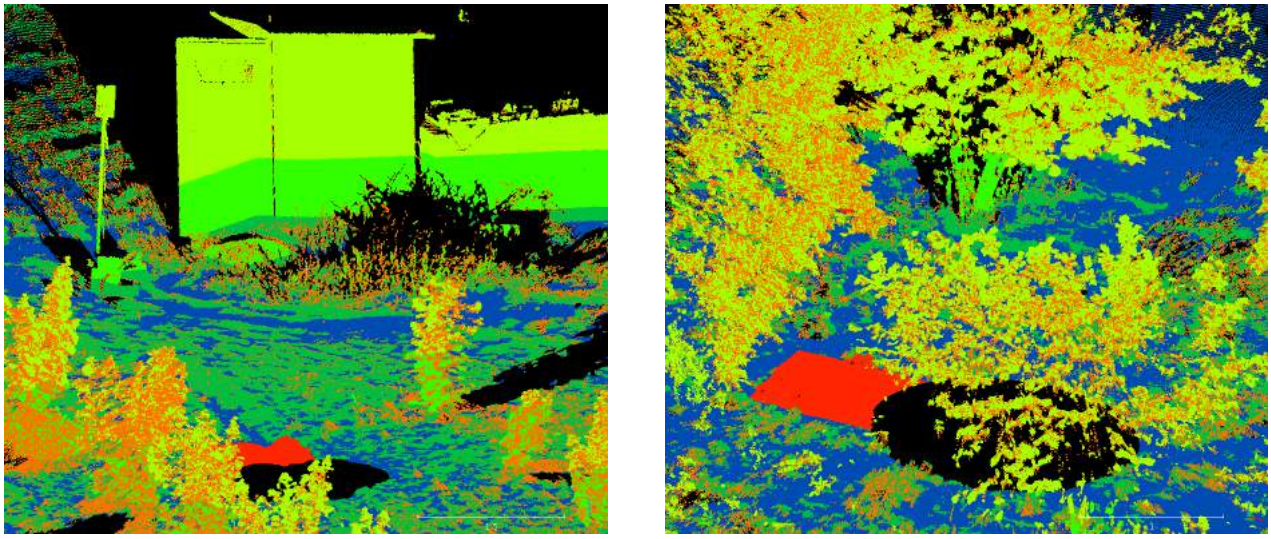


Fig. 26: Classified laser scans of field 16 (left) and field 18 (right) displaying points associated with ground (blue), low vegetation (dark green), intermediate vegetation (green), high vegetation (light green), noise (orange) and the UGV (red). The class assignment by height results in reasonable classes for vegetation, but in a miss-classification of non-vegetative objects (background of left image).

7.5.4 Denoising

To check the correctness of the denoising algorithm, the scans of the campaign in May 2018 (see section 6.3.1) have been classified (see section 5.3.3.3) and denoised (see section 5.3.3.4). For scans with a scanner *Resolution* of "1/4", respectively "1/8", the filter resolution has been set to 0.06° and 0.12°, respectively. Objects with characteristic features required for subsequent tasks have been investigated visually for unexpected results.

The half-cylindrical shape in the center of Fig. 27 (left) is a result of scanning a hazelnut stem. Linear trails of points at the edges of the shape are a result of partially hitting the stem with the laser beam. Scanning of leaves results in curved surfaces of close points, as shown in Fig. 27 (right). Between the leaves many isolated points can be identified, as a result of the laser beam partially hitting the leaves. Points primarily associated with connected shapes remain after noise removal.

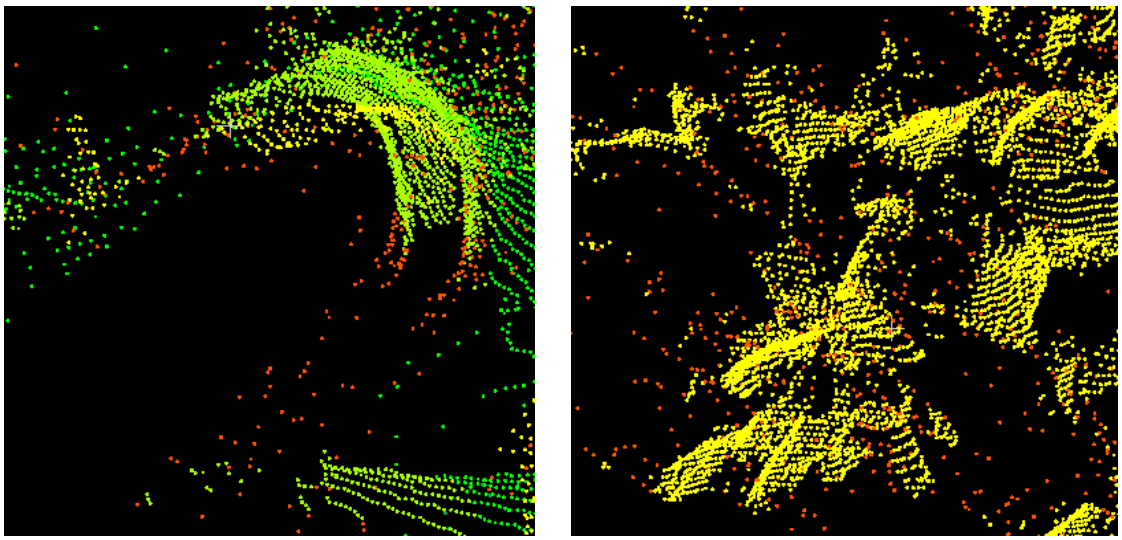


Fig. 27: Laser scans of a hazelnut stem (left) and hazelnut leaves (right). Points identified as noise are labeled in red.

Based on these results and the resolution adaptive approach, a further refinement of the algorithm does not seem necessary. Nevertheless, when enough monitoring data is available, the *Resolution* parameter should be optimized with care, since very low values might result in an omission of small objects, like suckers, while too high values would result in a still noisy result.

7.6 Sensor alignment

To test the sensor alignment, the training data already used for distortion calibration has been used. To assess the accuracy of the alignment, the re-projection error has been extracted for each band combination. The results have been summarized in The alignment is plausible, since it is similar to the true sensor alignment, with the characteristic five lenses of the MicaSense camera and the Sony camera located behind. The re-projection errors, with values below 1.5 pixels are promising, since these slight errors ensure a smooth spectral enrichment of the point clouds.

Tab. 8. The diagonal of the table corresponds to the re-projection error caused by lens distortion only. Fig. 28 illustrates the derived 3D alignment of the camera sensors.

The alignment is plausible, since it is similar to the true sensor alignment, with the characteristic five lenses of the MicaSense camera and the Sony camera located behind. The re-projection errors, with values below 1.5 pixels are promising, since these slight errors ensure a smooth spectral enrichment of the point clouds.

Tab. 8: Pairwise re-projection error in pixels.

	Band 1	Band 2	Band 3	Band 4	Band 5	Band 6	Band 7	Band 8
Band 1	0.92	0.73	0.97	1.46	1.40	1.44	1.47	1.44
Band 2	0.73	0.44	0.79	1.34	1.28	1.33	1.36	1.33
Band 3	0.97	0.79	1.00	1.49	1.43	1.47	1.51	1.47
Band 4	1.46	1.34	1.49	0.22	0.19	0.31	0.23	0.24
Band 5	1.40	1.28	1.43	0.19	0.13	0.28	0.20	0.20
Band 6	1.44	1.33	1.47	0.31	0.28	0.38	0.31	0.32
Band 7	1.47	1.36	1.51	0.23	0.20	0.31	0.22	0.22
Band 8	1.44	1.33	1.47	0.24	0.20	0.32	0.22	0.22

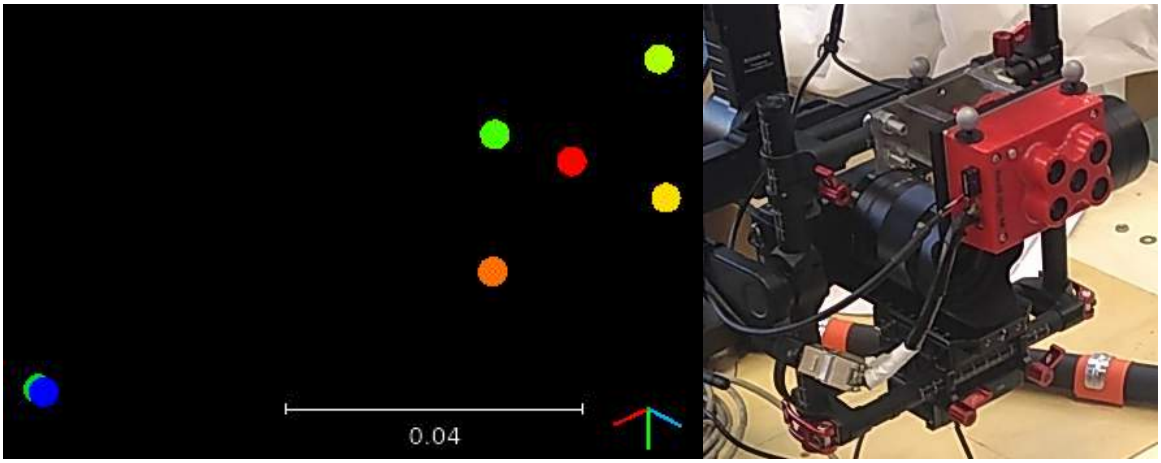


Fig. 28: Calibrated 3D alignment of the camera sensors (left) and true sensor positions within the gimbal (right).

7.7 Spectral enrichment

Since during development of the spectral enrichment algorithms, no actual UGV data was available, the extrinsic matrices had to be estimated differently. Thus, the data intended for sensor alignment and camera calibration has been used, since the chessboard markers provide the required information on the extrinsic orientation of all sensors.

Fig. 29 illustrates screenshots of a laser scan used for the sensor calibration. Fig. 30 to Fig. 32 illustrate the results of the spectral enrichment. Although the camera alignment is correct, is obvious, that the projection of the images to the point cloud is erroneous. This is most likely caused by a bug in the calculation of the extrinsic matrices. Unfortunately, this bug could not be fixed before submitting this deliverable. Nevertheless, it does not compromise or affect the validity of the analysis carried out so far. The accuracy of the final spectral enrichment mostly depends on the sensor alignment and camera calibration, so the expected accuracy measures can be found section 7.6 and section 7.4.2.

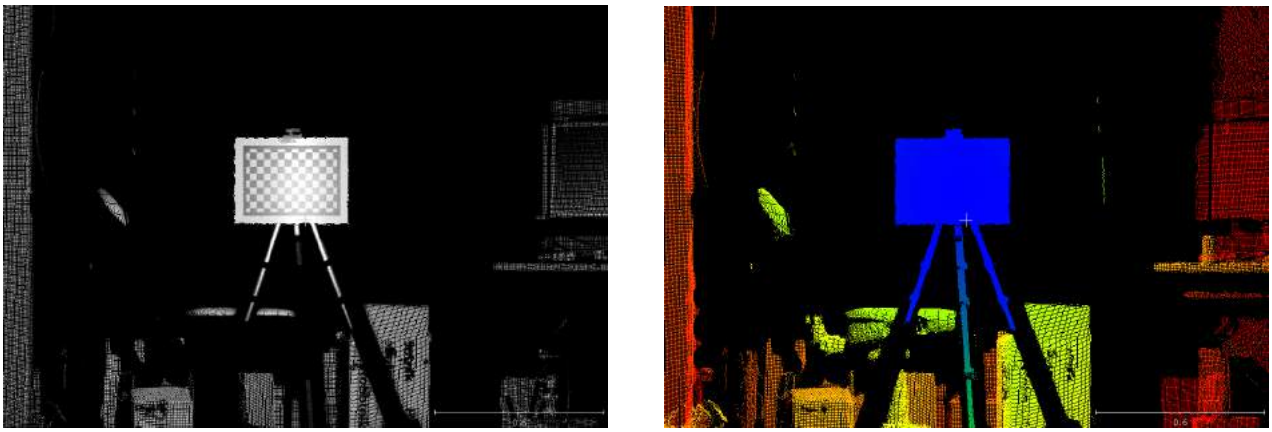


Fig. 29: Intensity image (left) and corresponding distance image (right) of a laser scan taken to calibrate the sensor alignment.

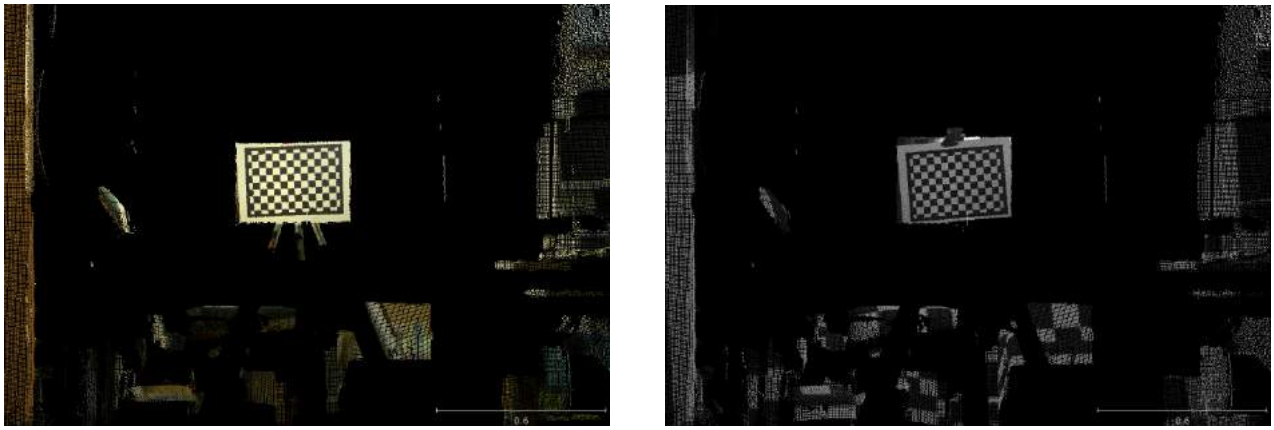


Fig. 30: Screenshot of a laser scan, which has been spectrally enriched with the first three RGB bands (left) and band 4 (right).

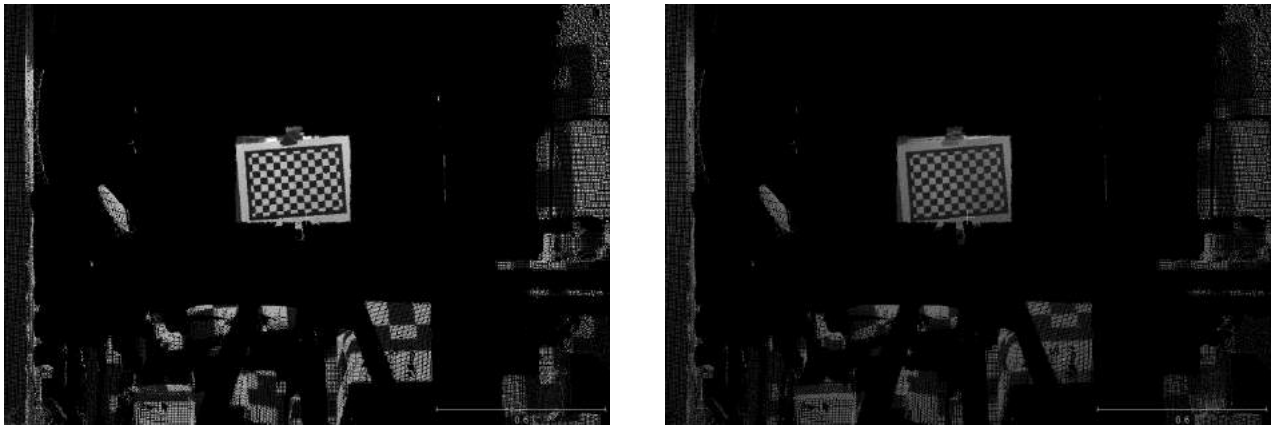


Fig. 31: Screenshot of a laser scan, which has been spectrally enriched with band 5 (left) and band 6 (right).

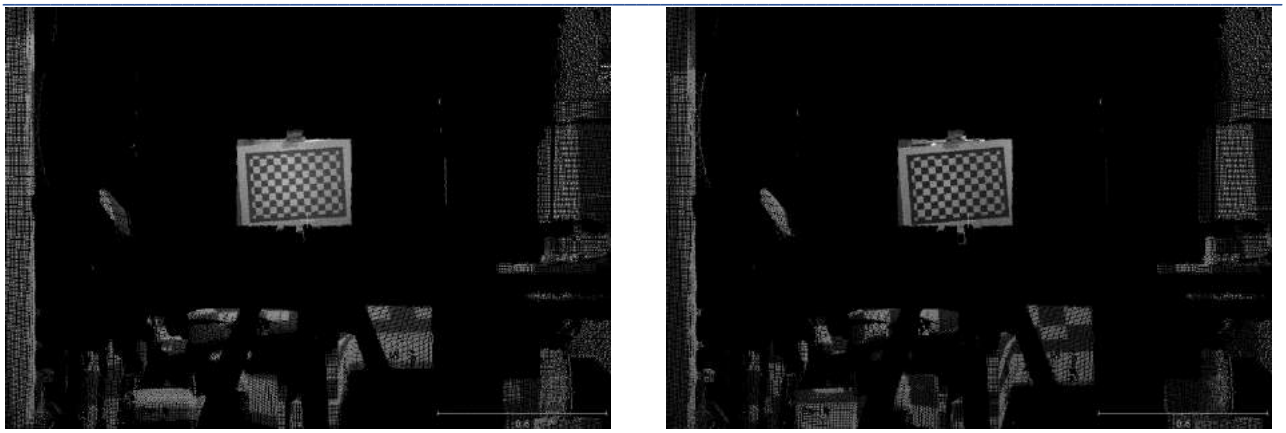


Fig. 32: Screenshot of a laser scan, which has been spectrally enriched with band 7 (left) and band 8 (right).

7.8 Laser scan alignment

To evaluate the scan alignment algorithm, the laser scans of the campaign in May 2018 (see section 6.3.1) have been used. Thus, for each of the eight trees—four young-aged (field 16) and four middle-aged (field 18)—four neighbored scans have been selected and aligned using the algorithm presented in section 7.8. The alignment results presented here have been derived after a coarse manual optimization of the algorithm parameters by selecting reasonable values based on expert knowledge and a visual check for gross registration errors.

For practical reasons no reference points, suitable to address the alignment accuracy have been defined in the field. To get still an impression of the registration accuracy and precision, the horizontal and the vertical alignment errors have been inspected separately for points with well-known characteristics.

7.8.1 Methods for horizontal accuracy assessment

To evaluate accuracy of the horizontal alignment, a subset of the outer wall of a cabin close to the scanner positions at field 16 has been investigated (see Fig. 33). Since the outer wall of the cabin is almost planar, a plane could be fitted automatically using the least squares method. Assuming no systematic horizontal alignment errors, for each original scan the average residuum (distance of the points to the plane) should be close to zero. Thus, the average residual of a specific scan represents the horizontal alignment error of the scan.

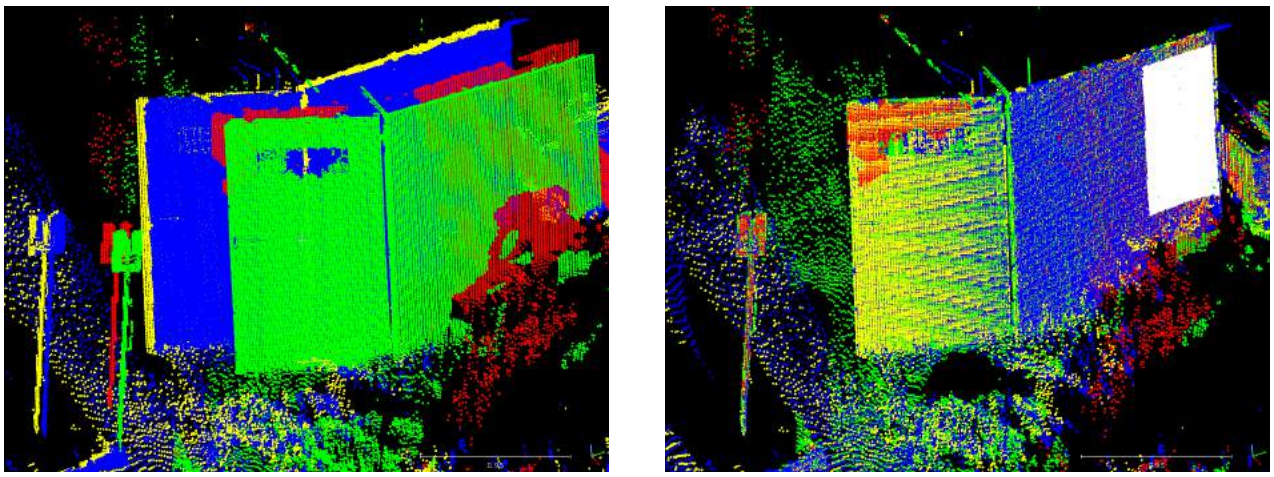


Fig. 33: Laser scans of the cabin located at field 16 before alignment (left) and after alignment (right). The area marked in white has been used for the accuracy assessment.

7.8.2 Methods for vertical accuracy assessment

To assess the vertical alignment accuracy, points classified as ground have been investigated. Assuming no systematic vertical registration errors, the vertical difference between neighbored points of different source point clouds should be close to zero. So, from the ground points 1000 points have been selected randomly from each of the original point clouds. For each of these points, the closest point in 2D has been selected from each of the source point clouds. Then, the residuals (vertical difference between the point and the average coordinates of these neighbored points) have been calculated. The average residuum of a specific scan thus represents the systematic vertical registration error of this scan.

7.8.3 Results and discussion

The results of the horizontal and vertical registration are illustrated in Fig. 34 and Fig. 35 for selected trees. For well aligned scans normally distributed residuals with an average value close to zero can be expected. Tab. 9 summarizes the derived alignment accuracies for all scans.

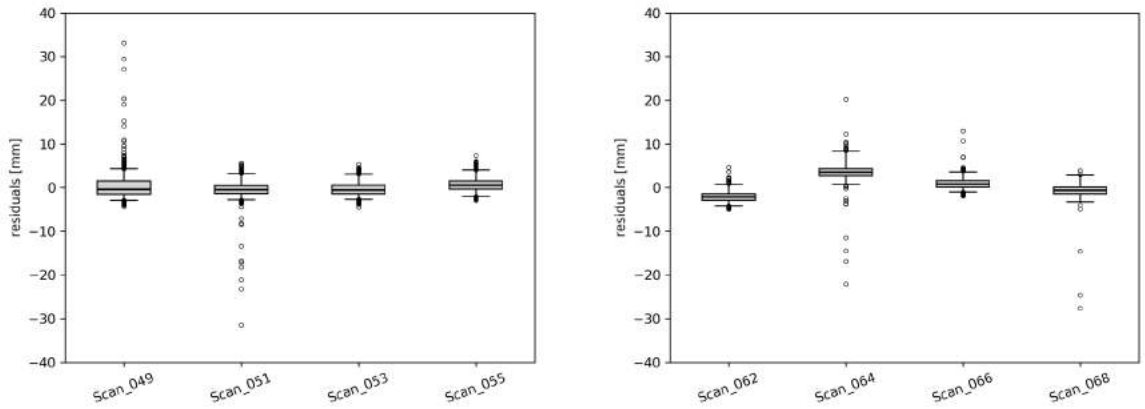


Fig. 34: Boxplots of the distances of the laser points to the cabin plane grouped by point source. The variance of the boxplots is an indicator for the precision of the laser scanner.

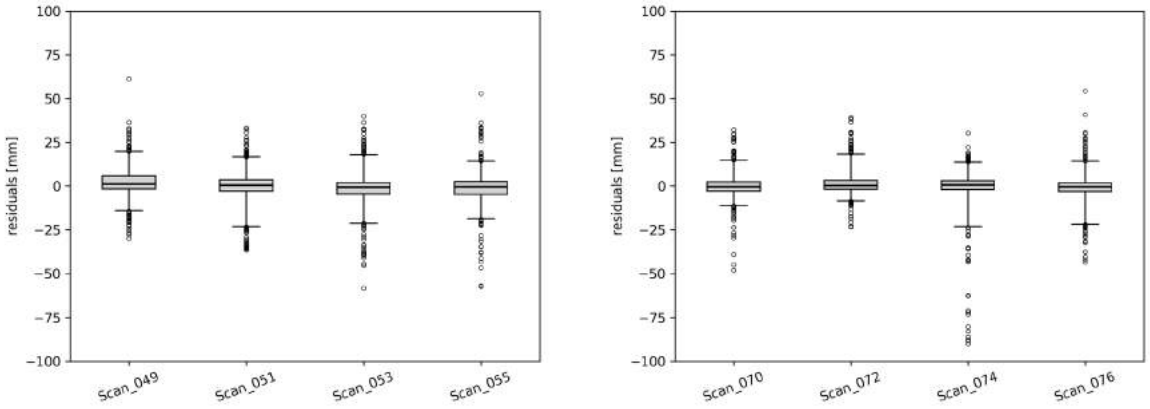


Fig. 35: Boxplots of the vertical differences between the randomly selected ground points and the average z-coordinate of their neighbors, grouped by point source.

Tab. 9: Laser scan alignment accuracies of the campaign in Mai 2018. The precision is expressed in one standard deviation. The star indicates that the average value differs significantly from zero.

tree	horizontal alignment		vertical alignment	
	mean error (mm)	standard deviation (mm)	mean error (mm)	standard deviation (mm)
field16, tree 1	0.01 ± 0.22	0.44	0.00 ± 0.71	1.41
field16, tree 2	-0.07 ± 0.66	1.32	0.00 ± 0.32	0.63
field16, tree 3	-0.03 ± 1.37	2.73	0.00 ± 0.71	1.42
field16, tree 4	0.39 ± 1.05	2.11	0.00 ± 0.39	0.77
field18, tree 1			0.00 ± 0.42	0.84
field18, tree 2			0.00 ± 0.28	0.57
field18, tree 3			0.00 ± 1.62	3.24
field18, tree 4			0.00 ± 0.44	0.88

The results—with residuals different from zero—indicate systematic alignment errors. Although this indicates a non-optimal solution of the alignment problem, the magnitudes of the differences are below 3mm, which is clearly below the benchmark of 2cm defined in deliverable D2.1. The achieved accuracy is more than sufficient for subsequent analyses.

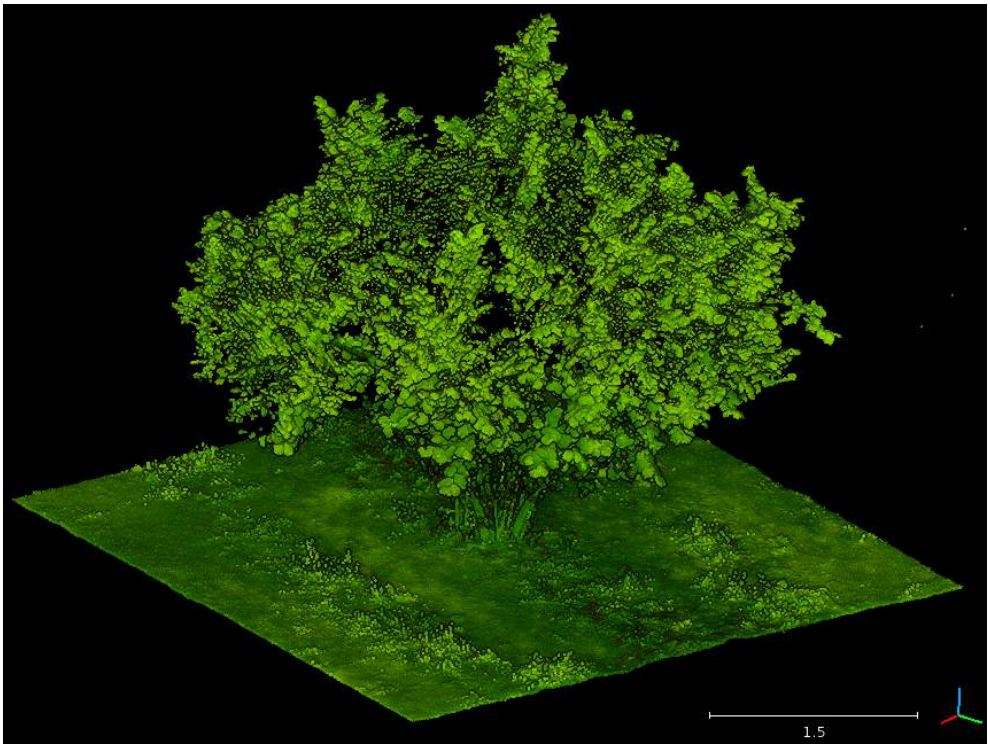


Fig. 36: Pseudo-color 3D representation of a hazelnut tree after automatic scan alignment.

Although the alignment algorithm provides accurate results for the given data and parameter set, during the development of the algorithm some unexpected results with coarse registration errors have occurred. This could be caused by the heuristic assumption, that the ICP algorithm converges to an optimal alignment. Since in some cases this assumption is not true, a continuous automatic fault detection will be implemented in the future.

7.8.4 Method alternatives

Theoretically all laser scans could be aligned at once to create just one point cloud for each measurement day. But this approach would have several drawbacks:

- Creating a joined point cloud results in an enormous need of memory. To process such a point cloud would require for splitting it into smaller samples.
- To analyze an individual tree only a small subset of the point cloud is required.
- The measurement accuracy of the laser scanner decreases with increasing distance to the scanner. Thus, far off points should be omitted.
- To calculate the roto-translation parameters of all scans of the orchard at once, millions of equations would need to be solved. This procedure is not only time consuming, but also any miss assignment would bear the risk of compromising the result for the whole orchard.

Because of these drawbacks the decision to solve the alignment problem for each tree locally seems to be suited best.

8 Conclusions

8.1 Completed tasks

The mayor objective of Task 4.2 – *Terrestrial Remote Sensing Pipeline* was to provide an automated processing pipeline to derive high quality 3D multispectral point clouds as input for Task 4.3 – *Tree Geometry Reconstruction* and Task 4.8 – *Fruit Detection*.

To achieve this goal, a terrestrial data acquisition design, compliant with the technical limitations of the UGV, has been developed. In particular, the remote sensors—a laser scanner, a RGB camera and a multispectral camera—have been selected and purchased in close cooperation with Task 3.1 – *Selection and Customization of the Unmanned Vehicles.* By evaluating camera and laser scanner data manually collected in the hazelnut orchard, the concept has shown to be suitable to collect remote sensing data of prospectively sufficient quality as an input for sufficient Tasks 4.3 and 4.8 and WP5. Next to the data acquisition design, several sensor parameters have been tested, to receive suitable parameter sets for future measurement campaigns.

Based on the data acquisition concept, a processing pipeline has been developed. This pipeline consists of two independent pipelines for image and laser scan pre-processing. In detail, the laser scans are georeferenced, a DEM is fitted to receive height values, points are classified, and a noise filter is applied. The images are converted to a common geo-data format and corrected for vignetting effects. After pre-processing the data sets are fused by spectrally enriching the laser scans. Since this requires exact knowledge of the intrinsic and extrinsic orientation of the sensors, the sensors need to be calibrated first. For this reason, experimental setups for vignetting calibration, camera calibration and sensor alignment have been

developed. To increase the degree of automation, augmented reality markers have been introduced. After the spectral enrichment, the scans are aligned automatically for each tree individually, to diminish remaining GPS and compass errors.

To be consistent with the requirements of PANTHEONs SCARDA concept, a file storage design has been elaborated in close cooperation with Task 3.3 – *Definition and Implementation of the Data Repository*. To simulate PANTHEONs database, a local file data base has been used. Thus, the integration of the algorithms should be possible, without modifications of the core code.

To test the processing pipeline, calibration data and field data have been collected manually. The algorithms have been checked for correctness and a statistical accuracy assessment has been performed whenever reasonable. Based on the given laboratory and filed data, the algorithms seem to operate successfully and within the demanded accuracies for the subsequent tasks. Only the spectral enrichment does not provide the expected results, due to a remaining bug in the calculation of the extrinsic matrixes. The actual calibration of the sensors and refinement of the algorithm parameters will be carried out as soon as the UGV is able to collect monitoring data.

8.2 Ongoing tasks

Since the manufacturer of the laser scanner could not provide *Linux* binaries for the laser scanner until the submission of this deliverable, the FLS files have to be converted manually to the LAS format using software depending on *Windows*. In the future, the Linux binaries shall provide the point clouds from the scanner directly, which allows for an instant geo-referencing and conversion to the LAS format.

Since the calibration data collected at mid of December 2018 could not be analyzed completely due to a bug in the calculation of the extrinsic matrices, the analysis will be pursued after the submission. Then, the algorithms for the calibration of extrinsic matrices as well as the spectral enrichment can be tested again. Since D4.2 uses the same calibration procedure, the code will be revised as soon as possible. A revised version of the deliverable D4.1 will be submitted and made available in the project repository once the bug is fixed.

8.3 Ongoing research

Although the proposed algorithms have shown to provide the expected results and the accuracies are within expectations, future modifications might be necessary. Especially the robustness of the algorithms under real world conditions can only be tested with the actual UGV data. Thus, the accuracy assessment will be refined and pursued during the project as base data for fault detection. In case of unexpected behavior of the algorithms, modifications of the processing pipeline will be done on demand. Although the algorithms have already be designed to in line with PANTHEONs SCARDA concept and the data base design (Task 3.3), some adaptations are expected, to fit the requirements for system integration (Task 6.2).

Ensuring a uniform constant illumination of the reference target for vignetting calibration has shown to be hard due to shadowing effects. For the future, an extended experimental setup with spotlights is considered. Also, alternative methods to calibrate the vignetting function from image mosaics [21] will be considered, if the spectral information shows to be inconsistent.

During development, the accuracy of the scan alignment could not be derived, since no absolute references were recorded in the orchard. For future measurement campaigns it is planned to place tie-point markers randomly on the ground. These markers can be detected automatically in the laser scans using image analysis. Thus, the alignment of the marker centers of neighbored scans can be checked automatically and an automated accuracy assessment can be performed.

9 References

[1] MicaSense, „MicaSense RedEdge-M Multispectral Camera,“ 2017.

[2] Sony Semiconductor Solutions Corporation, *IMX214 Sony Semiconductor Solutions,* 2014.

[3] I. MongoDB, *Open Source Document Database,* 2018.

[4] I. Ecma, „The JSON Data Interchange Syntax,“ 2017.

[5] H. Butler, M. Daly, A. Doyle, S. Gillies, S. Hagen und T. Schaub, „The GeoJSON Format,“ RFC Editor, 2016.

[6] M. Siemens, *TinyDB 3.12.2 documentation,* 2018.

[7] G. Bradski, „The OpenCV Library,“ *Dr. Dobb's Journal of Software Tools,* 2000.

[8] S. Suzuki und K. Abe, „Topological structural analysis of digitized binary images by border following,“ *Computer vision, graphics, and image processing,* Bd. 30, pp. 32-46, 1985.

[9] A. Kordecki, H. Palus und A. Bal, „Practical vignetting correction method for digital camera with measurement of surface luminance distribution,“ *Signal, Image and Video Processing,* Bd. 10, pp. 1417-1424, 7 2016.

[10] M. Riechert, *letmaik/rawpy,* 2018.

[11] C. Paul und S. Whited, *rawkit 0.6.0 documentation,* 2018.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot und E. Duchesnay, „Scikit-learn: Machine Learning in Python,“ *Journal of Machine Learning Research,* Bd. 12, pp. 2825-2830, 2011.

[13] F. Mosteller und J. W. Tukey, „Data analysis and regression: a second course in statistics.,“ *Addison-Wesley Series in Behavioral Science: Quantitative Methods,* 1977.

[14] Z. Zhang, „A Flexible New Technique for Camera Calibration,“ *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Bd. 22, pp. 1330-1334, 12 2000.

[15] D. Girardeau-Montaut, *CloudCompare - Open Source project,* 2016.

[16] ASPRS, „LAS SPECIFICATION VERSION 1.4 -- R13,“ 5410 Grosvenor Lane, Suite 210 Bethesda, Maryland 20814-2160, 2013.

[17] Y. Chen und G. Medioni, „Object modelling by registration of multiple range images,“ *Image and Vision Computing,* Bd. 10, pp. 145-155, 4 1992.

[18] P. J. Besl und N. D. McKay, „A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Bd. 14, p. 239–256, 2 1992.

[19] Z. Zhang, „Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision,* Bd. 13, pp. 119-152, 10 1994.

[20] I. NaturalPoint, *Motion Capture Systems,* 2018.

[21] S. J. Kim und M. Pollefeys, „Robust Radiometric Calibration and Vignetting Correction," *IEEE Trans. Pattern Anal. Mach. Intell.,* Bd. 30, pp. 562-576, 4 2008.

[22] R. Hartley und A. Zisserman, Multiple view geometry in computer vision, Cambridge university press, 2003.

[23] G. Venuti, *Roto-translations,* 2009.

[24] J. Serafin und G. Grisetti, „Using augmented measurements to improve the convergence of icp," in *International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, 2014.

10 Appendix

10.1 Terms and definitions

10.1.1 Point clouds

10.1.1.1 Points

A k dimensional point p shall be defined by its coordinates $(p_{1,}p_{2,}\dots,p_k)$ with $p_{i\in\{1,\dots,k\}} \in \mathbb{R}$ and an arbitrary number of attributes. For better readability, we will note $p \in \mathbb{R}^k$ and $p = (p_{1,}p_{2,}\dots,p_k)$ although additional attributes might be linked to p. In addition, the coordinate axes of a 3D point $p \in \mathbb{R}^3$ are called x, y and z, which results in the notation $p = (p_x, p_y, p_z)$. Two dimensional points are noted accordingly by omitting the z axis. A set or point cloud P^k with $|P^k| = n$ represents a collection of n points of k dimension.

10.1.1.2 Neighboring points

The weighted Euclidean length of p is defined by $\|p\|_w = \sqrt{\sum_{i=1}^{k} w_i \cdot {p_i}^2}$ with $w = (w_{1,}w_{2,}\dots w_k)$ and $w_{i\in\{1,\dots,k\}} \in \mathbb{R}$. The Euclidean length $\|p\|$ corresponds to the weighted Euclidean length with all weights set to one.

We define the Euclidean topology of P^k as $P_\epsilon^k(q) = \{p \in P^k : \|p - q\| \leq \epsilon\}$ with $q \in \mathbb{R}^k$ and $\epsilon \in \mathbb{R}_{>0}$. So, a point $p \in P^k$ is assumed to be a neighbor of q, if $p \in P_\epsilon^k(q)$. If only the first $m \leq k$ coordinate dimensions are used to calculate the euclidean length, the notation $P_\epsilon^k(q)^m$ is used. Thus, only the first m dimensions are considered to define the neighboring region.

10.1.2 Images

An image I of m rows and n columns represents a 2D matrix with pixels $I_{ij} \in \mathbb{R}^{m\times n}$. Typically, image sensors provide digital numbers, with discrete values of fixed value range instead. The average value $\overline{I}$ of the image is defined by:

$$\overline{I} = \sum_{i=1}^{m}\sum_{j=1}^{n}\frac{I_{ij}}{m \cdot n} \tag{3}$$

Cameras

10.1.2.1 Camera sensors

A camera is a compilation of one to several single band sensors. A camera captures a one- to multi-band image as compilation of all single band images captured by its sensors. Thus, the geometric and radiometric characteristics of each image band depends on the intrinsic orientation of its sensor.

10.1.2.2 Intrinsic camera orientation

The intrinsic orientation of a camera sensor is characterized by the focal length f of its lens (in world units), location of the principal point $(x_{0,}\, y_0)$ within the image (in pixels), sensor width (w_x, w_y) (in world units) and image skew s. The intrinsic information is compiled in the intrinsic matrix K [22]:

$$K = \begin{bmatrix} f/w_x & s & x_0 \\ 0 & f/w_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

The lens curvature results in a characteristic lens distortion. Typically, the radial and tangential distortion is modeled during the calibration of the intrinsic camera orientation.

10.1.2.3 Virtual camera

In the PANTHEON project several cameras are mounted on the platforms (UAV and UGV) at once within a gimbal. Due to the fixed relative orientation of the cameras towards each other they can be pooled to a *virtual camera* as a compilation of all single band sensors. Following this concept, by triggering the cameras at once, a multi-band *virtual image*—with several image resolutions—is captured.

10.1.3 Extrinsic orientation

The extrinsic matrix E defines the position and orientation of a sensor in world coordinates. With a given position $t = (t_x, t_y, t_z)$ and rotation angles $\omega = (\omega_x, \omega_y, \omega_z)$ this roto-translation matrix is defined by:

$$K = E = T_t \times R_\omega \tag{5}$$

with the translation matrix:

$$T_t = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

and the rotation matrix:

$$R_\omega = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\omega_x & -\sin\omega_x & 0 \\ 0 & \sin\omega_x & \cos\omega_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos\omega_y & 0 & \sin\omega_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\omega_y & 0 & \cos\omega_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos\omega_z & -\sin\omega_z & 0 & 0 \\ \sin\omega_z & \cos\omega_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

10.2 Implementation details

10.2.1 Point filters

A point filter creates a subset Q^k of a point set P^k with magnitude $|Q^k| < |P^k|$. If the filtered subset defines characteristic features of the point cloud, a point $q^i \in Q_m^k$ is assumed to be a key point of P^k. Key points might represent the structure of a tree or might be used to create a digital elevation model.

A duplicate point filter sub-samples a point cloud by omitting similar points. For a given point $q \in P^k$, all neighboring points $p \in P_\epsilon^k(q)$ within radius ϵ are excluded from P^k, while p is added to Q^k. By applying this procedure iteratively for all points, Q^k is a subset of P^k with pairwise point distances of at least ϵ.

10.2.2 DEM filtering algorithm

10.2.2.1 Parametrization

(1) Define the desired resolution α of the DEM in degrees.

(2) Define the maximal slope of the DEM by setting a maximum zenith angle $\theta_{max} \in]0,90]$.

10.2.2.2 Filtering of points probably representing the ground

(3) Define the set of key points $Q^3 = \{\ \}$ and the set of excluded points $E = \{\ \}$

(4) Sort P^3 in ascending order of z-dimension.

(5) For $p \in P$:

(6) if $p \notin E$:

(7) Find the set of neighbors $P_\epsilon^3(p)^2$ to p, with $\epsilon = \sin(\alpha) \cdot \|p - \mu\|$ and the center of the laser scanner $\mu \in \mathbb{R}^3$.

(8) Add the points of $P_\epsilon^3(p)^2$ to the set of excluded points E.

(9) else:

(10) Add p to the set of key points Q^3.

(11) Remove isolated points by defining $K^3 = \{\, q \in Q^3 : |Q_\epsilon^3(q)^2| \geq 6 \text{ with } \epsilon = 2 \cdot \sin(\alpha) \cdot \|p - \mu\| \,\}$.

10.2.2.3 Ensuring flatness of the DEM

(12) While $|K^3|$ has not changed during last iteration:

(13) Create a Delaunay triangulation T of K^3. Thus, a triangle $t \in T$ is defined by its corners $t = \{u, v, w\}$ with $u, v, w \in K^3$.

(14) For t in T:

(15) Calculate the zenith angle θ of the triangle t to identify steep triangles.

(16) if $\theta > \theta_{max}$:

(17) Remove the triangle corner largest z value from K.

10.2.3 Estimation of roto-translation parameters

Following the idea of Venuti [23] two or more laser scans can be aligned if a sufficient number of matching points is known. In the following, an equation system, to estimate the roto-translation parameters using the least squares method is deduced.

Having a point cloud $A \subset \mathbb{R}^3$ derived by the laser scanner, we can assume a positional error— e. g. caused by a GPS—of $t_A = \left(t_{A_x}, t_{A_y}, t_{A_z}\right)$ and a slight twist—e. g. caused by compass errors—of $\omega_A = \left(\omega_{A_x}, \omega_{A_y}, \omega_{A_z}\right)$. Thus, the true world coordinates $a' = \left(a_x', a_y', a_z'\right)$ of an arbitrary point $a \in A$ can be derived by equation (8):

$$a' = t_A + R_{\omega_A} \cdot a \tag{8}$$

Due to the high accuracy of the kinematics, we can assume almost infinitesimal small rotations. Thus, the rotation matrix can be approximated satisfactorily by equation (9):

$$R_{\omega_P} = \begin{bmatrix} 1 & -\omega_{P_z} & \omega_{P_y} \\ \omega_{P_z} & 1 & -\omega_{P_x} \\ -\omega_{P_y} & \omega_{P_x} & 1 \end{bmatrix} \tag{9}$$

If an object has been captured twice by two laser scans A and B, one or many matching point pairs (a, b) with $a' = b'$ are created. Using equations (8) and (9), we receive equation (10):

$$t_A + R_{\omega_A} \cdot a = t_B + R_{\omega_B} \cdot b \tag{10}$$

Again, by assuming infinitesimal rotations ω_A and ω_B, we receive the following equation system by compiling equations (9) and (10):

$$\begin{bmatrix} t_{A_x} \\ t_{A_y} \\ t_{A_z} \end{bmatrix} + \begin{bmatrix} 1 & -\omega_{A_z} & \omega_{A_y} \\ \omega_{A_z} & 1 & -\omega_{A_x} \\ -\omega_{A_y} & \omega_{A_x} & 1 \end{bmatrix} \cdot \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} t_{B_x} \\ t_{B_y} \\ t_{B_z} \end{bmatrix} + \begin{bmatrix} 1 & -\omega_{B_z} & \omega_{B_y} \\ \omega_{B_z} & 1 & -\omega_{B_x} \\ -\omega_{B_y} & \omega_{B_x} & 1 \end{bmatrix} \cdot \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \tag{11}$$

equation (11) is rearranged to equation (12) and finally to equation (13):

$$\begin{bmatrix} 1 & 0 & 0 & 0 & -a_z & a_y \\ 0 & 1 & 0 & a_z & 0 & -a_x \\ 0 & 0 & 1 & -a_y & a_x & 0 \end{bmatrix} \cdot \begin{bmatrix} t_{A_x} \\ t_{A_y} \\ t_{A_z} \\ \omega_{A_x} \\ \omega_{A_y} \\ \omega_{A_z} \end{bmatrix} + \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & -b_z & b_y \\ 0 & 1 & 0 & b_z & 0 & -b_x \\ 0 & 0 & 1 & -b_y & b_x & 0 \end{bmatrix} \cdot \begin{bmatrix} t_{B_x} \\ t_{B_y} \\ t_{B_z} \\ \omega_{B_x} \\ \omega_{B_y} \\ \omega_{B_z} \end{bmatrix} - \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \quad (12)$$

$$\begin{bmatrix} b_x - a_x \\ b_y - a_y \\ b_z - a_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & -a_z & a_y & -1 & 0 & 0 & 0 & b_z & -b_y \\ 0 & 1 & 0 & a_z & 0 & -a_x & 0 & -1 & 0 & -b_z & 0 & b_x \\ 0 & 0 & 1 & -a_y & a_x & 0 & 0 & 0 & -1 & b_y & -b_x & 0 \end{bmatrix} \cdot \begin{bmatrix} t_{A_x} \\ t_{A_y} \\ t_{A_z} \\ \omega_{A_x} \\ \omega_{A_y} \\ \omega_{A_z} \\ t_{B_x} \\ t_{B_y} \\ t_{B_z} \\ \omega_{B_x} \\ \omega_{B_y} \\ \omega_{B_z} \end{bmatrix} \quad (13)$$

Obviously, the given equation system is under-determined. By adding at least two additional matching point pairs, the resulting equation system can be solved using the least squares method. Additional point clouds can be added by extending the equation system accordingly. In this way an arability number of point clouds can be aligned if a sufficient number of matching points is available.

10.2.4 ICP variant

The ICP algorithm refines an initial relative roto-translation of two similar point clouds A and Biteratively, by sub-sampling the point clouds and assuming close point pairs. In general, the ICP loop works as follows:

- while the accuracy is not satisfying:
  - transformation of A and B using the latest roto-translation parameters
  - search for probably matching point pairs
  - refinement of the roto-translation parameters using the matches

Our modification of the ICP algorithm uses the initial idea of the normal iterative closest point (NICP) algorithm [24] to not only take into account the point distance, but also the orientation of a point. This promises to find probably matching point pairs efficiently. For simplification, no objective function is defined like in standard NICP, but point normals are derived to extend the tree dimensional point coordinates by the three normal dimensions, resulting in six dimensional coordinates.

To calculate the normal of a point, the three principal vectors of the k closest points are calculated. If a planar distribution of the points is assumed, the principal vector explaining the smallest proportion of variance represents the point normal. Since the laser scanner captures the front of an object only, all point normals facing backwards to the scanner are inverted.

Before entering the ICP loop, the six-dimensional points—coordinates and normals—are scaled by a resolution vector $\delta = (\delta_1, \ldots \delta_6)$, which results in the point sets $\bar{A} \subset \mathbb{R}^6$ and $\bar{B} \subset \mathbb{R}^6$. By applying these weights, we achieve the property:

$$\bar{a} \in \bar{A}_\delta^6(\bar{b}) \Leftrightarrow \bar{a}_i - \bar{b}_i \leq \epsilon \cdot \delta_i \; \forall i \in \{1, \ldots, 6\} \text{ and } \bar{b} \in \bar{B} \quad (14)$$

Thus, the resolution δ defines the maximum difference between two points $\bar{a}$ and $\bar{b}$ to let the ICP algorithm assume a match. For example, by setting δ_1 and δ_2 to $\sqrt{0.5}$m and δ_3 to 0.1m, as well as δ_4, δ_5 and δ_6 to $\sqrt[3]{\sin(10°)}$, two points $\bar{p}$ and $\bar{q}$ must not exceed a horizontal distance of 0.5m, a vertical difference of 0.1m and a normal difference of 10° to let $\bar{a}$ fall into $\bar{A}_1^6(\bar{b})$.