Article

A Multisensor UAV Payload and Processing Pipeline for Generating Multispectral Point Clouds

Michiel Vlaminck [1,*], Laurens Diels [1], Wilfried Philips [1], Wouter Maes [2], René Heim [3], Bart De Wit [4] and Hiep Luong [1]

1 IPI-URC-imec, Ghent University, Sint-Pietersnieuwstraat 41, 9000 Ghent, Belgium
2 Department of Plants and Crops—URC, Ghent University, Coupure Links 653, 9000 Ghent, Belgium
3 Institut für Zuckerrübenforschung An der Universität Göttingen, Holtenser Landstraße 77, D-37079 Göttingen, Germany
4 Department of Geography, Ghent University, Krijgslaan 281 S8, 9000 Ghent, Belgium
* Correspondence: michiel.vlaminck@ugent.be

Abstract: Over the last two decades, UAVs have become an indispensable acquisition platform in the remote sensing community. Meanwhile, advanced lightweight sensors have been introduced in the market, including LiDAR scanners with multiple beams and hyperspectral cameras measuring reflectance using many different narrow-banded filters. To date, however, few fully fledged drone systems exist that combine different sensing modalities in a way that complements the strengths and weaknesses of each. In this paper, we present our multimodal drone payload and sensor fusion pipeline, which allows multispectral point clouds to be generated at subcentimeter accuracy. To that end, we combine high-frequency navigation outputs from a professional-grade GNSS with photogrammetric bundle adjustment and a dedicated point cloud registration algorithm that takes full advantage of LiDAR's specifications. We demonstrate that the latter significantly improves the quality of the reconstructed point cloud in terms of fewer ghosting effects and less noise. Finally, we thoroughly discuss the impact of the quality of the GNSS/INS system on the structure from the motion and LiDAR SLAM reconstruction process.

Keywords: LiDAR scanning; SLAM; point clouds; localization; mapping; multispectral imaging; sensor fusion

1. Introduction

Unmanned aerial vehicles (UAVs) play a vital role in remote sensing and are used as mobile measuring and monitoring devices, e.g., in precision agriculture (for observing the inter- and intra-variability of crops or for detecting pests) or in infrastructure inspection (for detecting defects or degradation such as corrosion). Many applications require both high-resolution 3D spatial and multi-band spectral data in order to infer the characteristics and behavior of various (living) materials. However, few hardware systems collect high-resolution 3D spatial and multiband spectral information simultaneously in one integrated airborne solution. In this work, we bridge the gap between LiDAR (Light Detection and Ranging) scanning and traditional imaging (such as RGB, hyperspectral, and thermal cameras), two complementary technologies. Our sensor fusion pipeline allows spectral, spatial and thermal data to be extracted and related to variables relevant to specific applications.

One of the application domains that benefit from multimodal data is infrastructure inspection or damage assessment. For example, in the case of wind farms, spatial data from LiDAR aids in the detection of concrete rot in the pedestals of the wind turbines, whereas the hyperspectral data is very well suited to detect corrosion on the blades. Similarly, in large-scale PV (photovoltaic) power plants, thermal data are used for the detection of hotspots, whereas hyperspectral images are very well suited to detect the erosion of the anti-reflectance coating. Other examples are bridges (concrete rot and coating delamination),

transmission towers (corrosion and structural damages), or high-voltage power lines (overheating and sagging). A second application domain is precision agriculture and forestry. In the case of precision agriculture, hyperspectral imaging provides insight into the spectral reflectance of the crop, which may reveal information about the plant health, e.g., to infer nutrient status or the presence of biotic or abiotic stress [1]. LiDAR data, on the other hand, provides information on the height or growth vigor of the crop. For example, parts of the crop may have stunted growth due to a lack of sunlight or overgrowing weeds. Similarly, hyperspectral imaging provides insights into the health and biodiversity of forests, their composition in terms of tree species, the presence of different types of stress (e.g., from beetles) and so on. The LiDAR data, in turn, provide information on the forest structure characteristics, including basal area, stem volume, dominant height, and biomass [2].

The combination of LiDAR and traditional imaging has not yet been fully exploited in the scientific literature because not all required techniques on calibration, registration, corrections, optimal scanning strategies, etc., are fully optimized. As such, the generation of multispectral point clouds from UAV platforms remains imprecise or suboptimal to date. In our research, we aim at making the most of both LiDAR and traditional imaging by developing a fully synchronized multimodal UAV payload and an associated processing pipeline able to produce georeferenced multispectral point clouds with subcentimeter accuracy. To that end, we have improved several state-of-the-art algorithms in the field of multimodal calibration, point cloud registration, spatial corrections, and optimal scanning strategies. More specifically, we propose an optimized self-localization and mapping algorithm that combines advanced point cloud registration with photogrammetric bundle adjustment and the output of a precision GNSS/INS (Global Navigation Satellite System/Inertial Navigation System). The latter is able to generate high-frequency navigation outputs with centimeter accuracy. However, due to the highly dynamic movements and occasional under-measurements, the accuracy can be as poor as five centimeters, causing the generated point clouds to suffer from distortions or discrepancies, especially when the UAV turns abruptly at the end of the flight line. In those cases, our LiDAR registration algorithm is able to compensate for the abrupt movements and under-measurements of the INS, thereby preventing severe distortion.

This paper is organized as follows. In Section 2, the related work is extensively described, in terms of both technology and applications. We discuss how our approach fits within their methodological frameworks, and equally importantly, where it deviates, allowing us to overcome inherent limitations. In Section 3, the hardware set-up and the pipeline are described, including all contributions. In Section 4, the experiments and results of the pipeline are discussed. Finally, Section 5 contains the main conclusions of this research.

2. Related Work

On self-localization and mapping through the fusion of inertial, LiDAR, and visual data, several papers have been published in recent years. Many of these works were conducted in the context of autonomous driving, for which computation time is crucial. In 2015, Zhang et al. [3] proposed their Visual–LiDAR odometry and mapping, better known as V-LOAM, which has been the best method for years according to the well-known KITTI odometry benchmark [4], until it was beaten by SOFT2 [5] in early 2022. Other well-performing and fast LiDAR–Visual SLAM methods are LIMO (LiDAR-monocular SLAM) [6], DVL-SLAM [7] and the method presented by Wang et al. [8].

A common criticism regarding the KITTI benchmark, which is focused on autonomous driving, is that researchers tend to over-optimize to the specifications of that particular dataset, which limits the generality of the proposed methods. For example, in the case of a moving car, there are hardly any rotations around the roll axis and the "short-term" dynamics are very limited. In this paper, we deal with scenes and circumstances that are a lot more challenging in nature compared to the KITTI dataset. Since we consider data

acquired from UAVs, all LiDAR points are captured from a relatively far distance (i.e., the height of the UAV flight trajectory), and many LiDAR points are sampled on a relatively flat ground plane (which could lead to a degenerate situation). Additionally, the drone is subject to many highly dynamic micro-movements, and often, the scene lacks structural elements or consists of large regions with a lot of repetitiveness (e.g., crops). On the other hand, since we focus on outdoor environments, we can use GNSS data to compensate for drift and to initialize self-localization prior to the actual point cloud registration. In addition, our use case does not require real-time execution since the final output (multispectral point clouds rather than the odometry itself) is mainly used for offline analysis.

The literature on the fusion of inertial, LiDAR, and visual data for UAV applications is much scarcer. Obviously, it is a lot harder to build multisensor payloads that fit under a UAV and do not surpass its maximum payload weight. Nagai et al. [9] pioneered building a multisensor UAV platform, consisting of two regular RGB cameras, two infrared cameras, a SICK laser scanner, a GNSS receiver and an inexpensive IMU. They integrated the GNSS and IMU data using Kalman filtering to obtain position and attitude estimates that guide the search for tie points used to register the image data through bundle block adjustment (BBA). The output of the BBA, in turn, aids the Kalman filtering by initializing the position and attitude in the next step in order to acquire a much more accurate trajectory. The final trajectory is eventually used to register the laser range data and to obtain a digital surface model, for which the authors report an average error of approximately 10 to 30 cm. A major difference with our work is that we increase the accuracy of the position and attitude data from the INS using a dedicated point cloud registration technique, which leads to fewer (visual) artifacts but also contributes to the overall accuracy. Consequently, we obtain point clouds or digital surface models that are one order of magnitude more accurate.

More recently, Qian et al. [10] presented a multisensor UAV payload and SLAM algorithm that is based on the fusion of Velodyne LiDAR data and RGB imagery. More specifically, they extract line and plane features from the LiDAR data and use them to compute the relative pose between consecutive frames in an ICP-based manner. Afterward, the relative pose is refined by combining it with visual odometry computed from the images. For that purpose, they use the relative pose error as a prior and subsequently minimize a photometric error. The work of Qian et al. is similar to ours in the sense that we also use the poses computed by LiDAR-based odometry (along with the output of the GNSS/IMU) as a prior for image-based reconstruction. However, we conduct a full 3D reconstruction using the image data (based on bundle adjustment) instead of limiting ourselves to visual odometry. The latter is justified, as we focus more on the overall mapping accuracy instead of the computation time. Note that our LiDAR and GNSS/INS-based self-localization itself is running in real-time. Finally, compared to [10], our approach allows fusing data from different imaging modalities such as a multispectral, hyperspectral or thermal camera, which usually have a limited and different spatial resolution.

Haala et al. [11,12] presented another multisensor UAV platform consisting of a Riegl VUX-1 LiDAR Scanner, Applanix AP 20 GNSS/IMU-unit and Phase One iXM-RS150F camera. Based on the foundations laid in the previous work of Cramer et al. [13], the authors apply hybrid georeferencing, integrating photogrammetric bundle block adjustment with direct georeferencing of LiDAR point clouds. They demonstrate that 3D point cloud accuracies of subcentimeter level can be achieved. This is realized by a joint orientation of laser scans and images in a hybrid adjustment framework, which enables accuracies corresponding to the ground sampling distance (GSD) of the captured imagery. Although the work of [11] has some similarities with ours, the main difference is that it does not use a dedicated point-cloud registration algorithm to improve the position and attitude estimates of the GNSS, nor does it integrate data from multispectral or thermal cameras.

Bultman et al. [14] propose a UAV system for real-time semantic fusion and segmentation of multiple sensor modalities. For visual perception, their UAV carries two Intel RealSense D455 RGB-D cameras mounted on top of each other to increase the vertical field-of-view and a FLIR thermal camera to alleviate person detection in search-and-rescue

scenarios. For 3D perception and odometry, they additionally integrated an Ouster LiDAR scanner on their UAV. Unlike our work, the authors do not focus on the alignment of the scans and images, nor do they focus on the SLAM algorithm. They mention the multi-view aggregation of point clouds, but for that purpose, they use voxel hashing to save memory, as their main use case is real-time semantic segmentation. As a result, the resolution of the obtained map is low.

Besides the actual mapping, other problems need to be solved, one of them being the calibration between the different sensors. In this work, we adopt late fusion by precisely merging the different orthophotos from the cameras with the point cloud obtained by the LiDAR scanner. In the future, we plan to integrate a more tight coupling between the point cloud registration and photometric bundle adjustment, for which an accurate calibration between the LiDAR and cameras would be required. One such method is CalibRCNN, presented by Shi et al. [15], which uses a Recurrent Convolutional Neural Network (RCNN) to infer the six-degrees-of-freedom rigid body transformation between the two modalities. For our current approach, only the calibration of the GNSS/INS device and the other sensors has to be estimated, which is described in detail in Section 3.3.

Plenty of other research relies on data captured from a multi-sensor UAV payload. Most of those publications focus more on the specific application rather than on the technical challenges related to the mapping and fusion of multimodal data. As mentioned in the introduction, a lot of this research is conducted in the context of agriculture or forestry. For example, Zhou et al. [16] used a DJI M600 UAV carrying a Pika L hyperspectral camera and an LR1601-IRIS LiDAR scanner for the purpose of detecting emerald ash borer (EAB) stress in ash. In [17], the authors used the same setup to detect pine wilt disease at the tree level. Finally, the authors of [18] fused UAV-based hyperspectral images and LiDAR data to detect pine shoot beetle. For more examples, we refer to the literature review by Maes and Steppe [1].

3. Materials and Methods

3.1. Sensors

As shown in Figure 1, our multisensor payload consists of an Ouster OS1-128 LiDAR scanner, a Micasense RedEdge-MX Dual camera system, a Teax ThermalCapture 2.0 thermal camera with an integrated FLIR Tau 2 sensor, and an inertial navigation system (INS) aided by an SBG Quanta GNSS (with a dual antenna system). The payload is controlled by an NVIDIA Jetson Xavier NX.
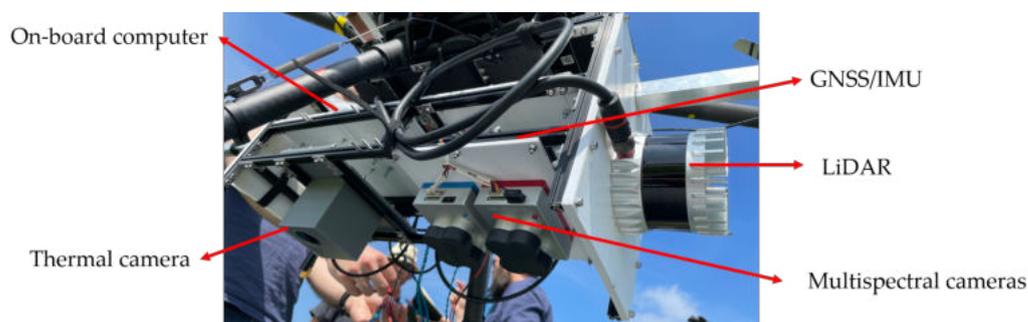

Figure 1. Our multisensor rig consisting of an Ouster LiDAR scanner, a Micasense RedEdge dual multispectral camera system, a Teax ThermalCapture 2.0 thermal camera, and an SBG Quanta GNSS system. It is controlled by an NVIDIA Jetson Xavier NX.

The Quanta has a roll/pitch accuracy of 0.015°, a heading accuracy of 0.035°, and a position accuracy of around 1 cm in post-processing (PPK). The Micasense cameras have an image resolution of 1280 × 960 and a field of view of 47.2° horizontally and 35.4° vertically. Consequently, they have a ground sampling distance (GSD) of 8 cm per pixel at a height of 120 m or 1.33 cm at a height of 20 m. Together, they cover 10 different wavelengths in the range between 444 nm and 842 nm. Every second, an image is captured for each band.

The Teax ThermalCapture 2.0 has a 640 × 480 resolution and operates at 30 or 60 Hz. With a 13 mm lens and 17 μm pixel pitch, it has a GSD of 2.6 cm/pixel at a height of 20 m.

Finally, the Ouster LiDAR scanner has 128 scan lines, a horizontal FOV of 360°, and a vertical FOV of 45°. It has a precision of 1 to 3 cm and operates in the 865 nm wavelength. Furthermore, it supports three different scanning configurations (horizontal resolution × rotation rate): 1024 × 10, 1024 × 20, and 2048 × 10. The GSD depends on the rotation rate and horizontal resolution and is different in the vertical and horizontal directions. For the optimal setup of the LiDAR sensor for UAV swath surveys, we refer the interested reader to [19]. The main specifications of the sensors are summarized in Table 1.

Table 1. Summary of the main specifications of the sensors on our acquisition payload.

Sensor	Specifications
Ouster OS1-128	- 128 lasers - 1024 or 2048 horizontal resolution - 10 or 20 Hz rotation frequency - 45° vertical FOV, 0.35° vertical angular resolution - 1–3 cm precision - 865 nm wavelength - 447 g weight
Micasense Rededge MX dual	- 444, 475, 531, 560, 650, 668, 705, 717, 740, 842 nm bands - 1280 × 960 resolution per band - 1 capture per second - 508.8 g weight
Teax ThermalCapture 2.0	- thermal camera (LWIR) - 640 × 480 resolution - 13 mm lens - 17 μm pixel pitch - 30/60 Hz framerate - 72 g weight
SBG Quanta UAV	- GNSS aided INS - roll/pitch avg. accuracy 0.015° (PPK) - heading avg. accuracy 0.035° (PPK) - position avg. accuracy 0.01 m - 38 g weight

3.2. Synchronization

The Ouster scanner is synchronized with the Quanta INS by time-stamping every firing of laser beams. The scanner supports three clock sources: (1) internal clock, (2) external PPS (Pulse Per Second) sync pulse from a GNSS/INS, and (3) external Precision Time Protocol (PTP). In our case, we adopt the PPS time synchronization to query the GNSS time from the Quanta and write it in the data frame of the LiDAR scan. In other words, every time the Ouster scanner fires 128 laser beams, a GNSS timestamp is saved in its data frame. Depending on the configuration, either 1024 (possible at a rotation rate of 10 Hz or 20 Hz) or 2048 (in case of 10 Hz rotation rate) timestamps are saved. The time precision, i.e., the minimum increment for the time, is 10 ns. The Micasense and the ThermalCapture cameras each have their dedicated GNSS receiver, and the GNSS timestamp is written in the metadata of the acquired images. By matching the timestamps of the cameras with the INS output from the Quanta, high-precision position and attitude information can be fetched for every image.

3.3. Calibration of the Sensors

For the extrinsic calibration between the different sensors, we look for the rigid body transformations mapping each sensor's coordinate frame to the Quanta GNSS's frame. Due

to the right-angled nature of our sensor rig and mounting configurations, specifying the rotational parts is straightforward. For example, in Figure 2, the LiDAR's $z^{(O)}$-direction coincides with the UAV's forward direction, which is, in our mounting configuration, the Quanta's negative $y^{(Q)}$-direction. Similarly, all other sensor axis directions coincide with one of the Quanta's axis directions. If we intentionally mount a sensor at an oblique angle, we can specify the rotation via Euler angles in a convenient order so that two Euler angles are a multiple of 90° and the third is our intended oblique angle (which we usually know in advance, or we can simply measure if not). The translational parts of the transformation matrices, i.e., the location of the sensor frame's origin in local Quanta coordinates, can also be measured using a simple ruler.

Figure 2. DJI M600 drone and sensor rig mounted to it. The weight of the payload is just below 5 kg. The axes with superscript (O) refer to the Ouster (sensor) reference frame, the ones with (Q) refer to the Quanta frame, and finally, those with (W) refer to the world frame, where $x^{(W)}$ points east and $y^{(W)}$ north.

However, unknown deviations from the right-angled nature of the mounting can occur, for example, if some screws are fixed tighter than others. Therefore, we will use the calibrations described above as the initialization for an optimization method, which will refine the calibration parameters. For this purpose, we make use of a 97 cm by 96 cm calibration plate, shown in Figure 3. As can be seen, it has a 2 × 2 black-and-white checkerboard pattern, allowing us to use standard (grayscale) techniques to detect it in the multispectral images. Since we actively cool the white parts with ice (and the black parts should also become warmer than the surrounding through reflecting sunlight less), the same can be said for the thermal images. Together with a calibration flight, where we fly over the plate at different heights and at different angles, and with ground measurement of the plate center's location, we can optimize the external camera calibration parameters.


**Figure 3.** Our calibration plate placed on a table.

To detect the calibration plate in the LiDAR data, we found intensity/reflectivity information to be insufficient, and we therefore placed the plate on a table approximately 1.5 m high. This allowed us to separate the plate from the ground directly from the 3D LiDAR information, at least if the calibration error remains within reasonable margins. This would not be the case if we used all scans from the entire calibration flight. However, if we only take scans on the same flight line, where the UAV's orientation is approximately constant, we are indeed able to segment out the calibration plate. For this purpose, we found a simple $z^{(W)}$-threshold (made somewhat adaptive by declaring it in terms of a

$z^{(W)}$-quantile) to work sufficiently well. To find the plate center in the data, we used a componentwise median to be more robust to noise. Using the distance between the data-detected plate center and its ground truth location as a loss function, we can improve the calibration parameters using gradient-descent-based techniques by going through the different flight lines multiple times.

The in-field location was measured using Belgian Lambert 72 coordinates, whereas the Quanta GNSS and hence the LiDAR data use WGS coordinates, which we can easily convert to UTM 31N. Unfortunately, the conversion between Lambert 72 and UTM is not well defined and has only a low accuracy, meaning we cannot actually use this in-field measurement as ground truth. Instead, we also estimated the location of the calibration plate, simply by adding three extra learnable parameters, which we initialized at the converted measured coordinates. Thus, instead of training for consistency with the ground truth, we now trained for internal consistency across all flight lines. We found this approach to work reasonably well, reducing the average error from 1.07 m to 0.186 m. In Figure 4, we show qualitative results. One can observe that our approach already cleans up most of the calibration noise. Of course, there is still some room for improvement. We plan to employ further consistency measures, such as entropy (for minimization) and consistency in the overlap region between multiple flight lines (i.e., strip adjustment, for maximization). However, this is only important if for the LiDAR registration we would rely solely on the GNSS/IMU data and extrinsic calibration. Instead (see Section 3.5), we use a dedicated LiDAR point cloud registration algorithm, for which we now have a very good initialization.
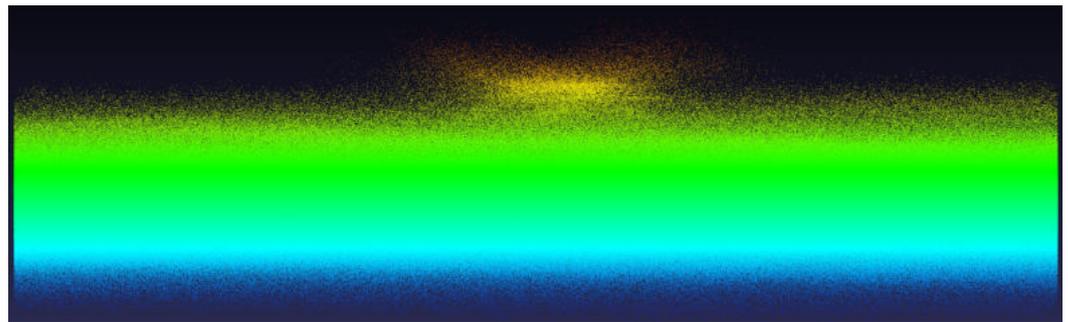
(**a**) Using the initial calibration with theoretical rotation matrix and measured translation.
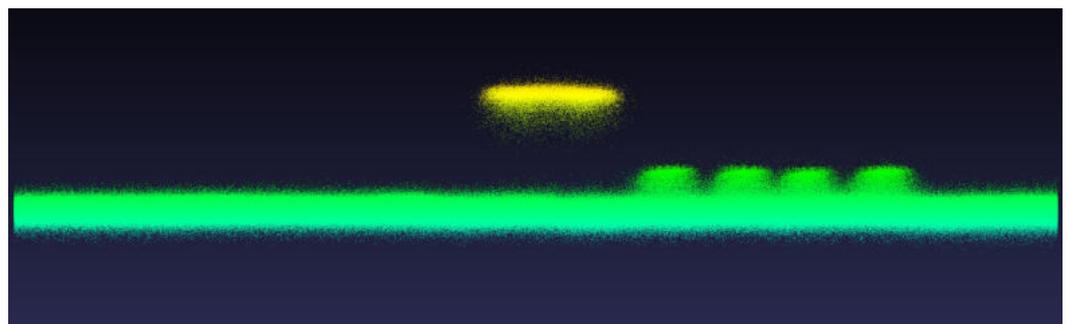
(**b**) Using the calibration method based on the calibration plate with a learned center.

Figure 4. The effect of our extrinsic LiDAR-GNSS/IMU calibration optimization approach on the registration of a LiDAR point cloud of a scene containing a plate on a table and some boxes on the ground. Parallel projections from a side view.

3.4. Post-Processing of INS Data

The attitude and position information provided by the Quanta GNSS-aided INS is used to initiate a preliminary alignment of the point clouds captured by the LiDAR scanner. However, GNSS satellite signals are subject to atmospheric effects and to reflection and diffraction caused by interference between radio waves, which in turn leads to multipath

errors. To deal with this, the GNSS data are first corrected through post-processed kinematic (PPK), which uses two receivers (one on a nearby base station and one on the Quanta device) instead of one.

Eventually, the raw Quanta data are processed using the Qinertia software from SBG [20], which applies an extended Kalman Filter (EKF) to couple the GNSS and inertial sensors, allowing the former to correct inertial drift while keeping high frequency (200 Hz) navigation outputs. In Figure 5, an example of Qinertia's post-processing is shown. The yellow dots denote the raw GPS point at 1 Hz, whereas the green line denotes the 200 Hz EKF-improved trajectory of the sensor and the drone.
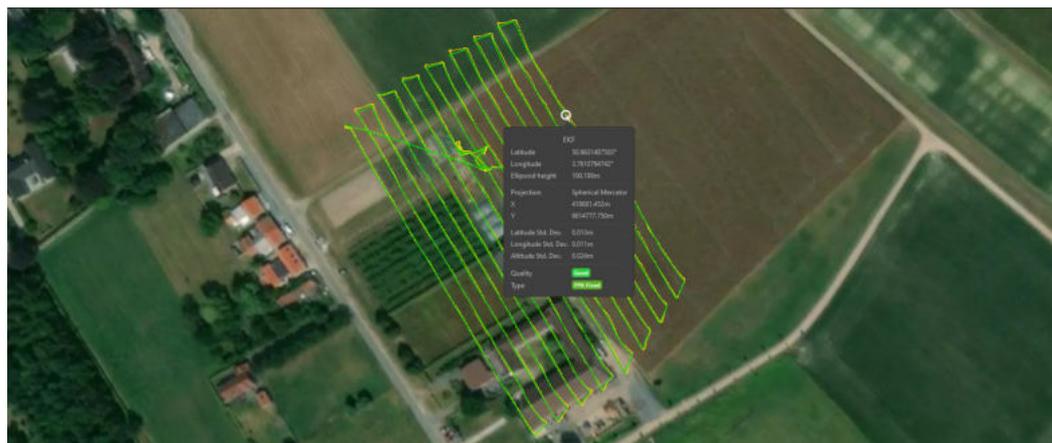

Figure 5. The trajectory of the drone using the raw data (yellow dots) and after post-processing using SBG's Qinertia software (green). The standard deviation on the latitude and longitude after post-processing is slightly more than 1 cm. The standard deviation on the altitude is around 2.4 cm. The actual position can be further improved by incorporating point cloud registration (as part of SLAM).

After Qinertia's post-processing, the standard deviation on the latitude and longitude is about 1 cm. The standard deviation on the altitude is slightly higher and averages around 2.4 cm. While this is reasonably good, minor synchronization errors, time quantization errors (although rather small as the navigation output from the Quanta is 200 Hz), and occasional under-measurements can further affect the accuracy of the resulting point cloud. In addition, the measurements of the Ouster LiDAR scanner have a limited precision of approximately 3 cm for points at a far distance. All this means that the point cloud generated by aligning the scans using the post-processed Quanta data still suffers from significant discrepancies. For that reason, we further improved the alignment by applying a proper point cloud registration technique. As the LiDAR data are particularly challenging (sparse measurements, inhomogeneous point density, etc.) and the circumstances are quite challenging (all points captured from a far distance, few structural elements, and many points sampled on a rather flat ground plane), we developed a novel dedicated point cloud registration algorithm, which is described in detail in the next section.

3.5. Dedicated Point Cloud Registration

Our dedicated point cloud registration consists of two main parts. First, two consecutive point clouds are aligned (scan-to-scan alignment), known in the literature as scan matching. The result is an initial estimate of the relative pose of the sensor. Subsequently, the second point cloud is transformed using the initial estimate combined with the previously computed pose (so-called dead reckoning) to put it in the world coordinate reference system. Then, a second matching is performed between the transformed point cloud and the aggregated point cloud built so far (scan-to-map alignment). The result from the second scan matching increases the accuracy of the final pose estimation since the aggregated point cloud is less sparse compared to the one acquired during one single scan (which often leads

to wrong point correspondences). The outline of this point cloud registration work flow is depicted in Figure 6. Note that, since the aggregation of all point clouds would eventually inflate memory usage, we use a downsampled version derived from an underlying octree data structure.
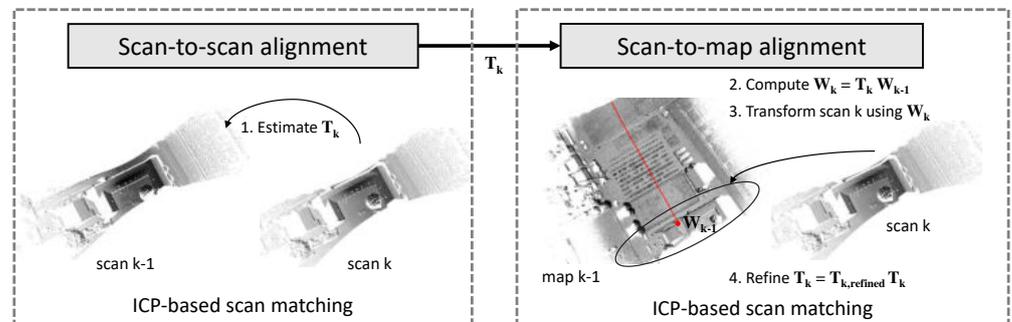

Figure 6. Schematic drawing of our dedicated point cloud registration algorithm which consists of two parts: (1) scan-to-scan alignment and (2) scan-to-map alignment. In this drawing, $\mathbf{W}_k$ denotes the pose of the lidar scanner at time k in the world coordinate space, $\mathbf{T}_k$ denotes the transformation that aligns the scan acquired at time k with the scan acquired at time $k-1$, and $\mathbf{T}_{k,refined}$ is a transformation matrix that refines $\mathbf{T}_k$ to obtain a transformation that better aligns scan k with the map at time k − 1.

3.5.1. Scan Matching

The scan matching is computed using a variant of ICP ("Iterative Closest Point"), an algorithm that iteratively determines corresponding point pairs between two point clouds—the source and the target—and uses these to estimate the transformation between the two that maximizes their overlap. Several cost functions to estimate the transformation have been proposed in the literature, such as point-to-point, point-to-plane, and plane-to-plane. The plane-to-plane variant rapidly declines when the scene lacks planar surfaces, which is often the case for outdoor environments such as agricultural fields or vast open areas. The point-to-point cost function lacks overall robustness, making the point-to-plane distance the cost function of choice.

The point-to-plane distance still requires point normals to be calculated for each point in the point cloud. The latter should be performed with care since the accuracy of the estimated pose will largely depend on it. In addition, a proper selection of candidate points is required to achieve a good pose estimation. Points sampled on stable areas, not subject to, for instance, wind (such as the leaves of trees), are preferred as surface normals can be more accurately computed in those regions. Among stable areas are solid surfaces, which are oftentimes planar in the case of constructed environments. In outdoor environments these are less common, but when present (for example facades or rooftops of houses, they should be exploited.

Points that are part of under-sampled or unstructured areas on the other hand should be avoided as they can compromise an accurate pose estimation. However, we should make sure that—at all times—the selected points provide enough '*observability*' of the translation and orientation components along all of the three axes $x^{(O)}$, $y^{(O)}$ and $z^{(O)}$. This observability denotes the extent to which the sampled points are able to explain the translation and orientation of the sensor. For example, when the drone is in stable hovering orientation, ground points provide observability of the translation along the $y^{(O)}$-axis (perpendicular to the ground plane; recall Figure 2). Facades, in turn, provide observability of the translation along the $x^{(O)}$- and $z^{(O)}$-axis (aligned to the ground plane).

We therefore propose two improvements over the original ICP method. First, we proportionate the rotational and translational observability by selecting a proper set of points to make sure that the transformation is not biased towards one particular direction. Second, we incorporate low-level features to add more importance to planar areas. Note that the lack of planar areas in outdoor environments such as agricultural fields does not

imply that the scan matching will necessarily fail in those cases. It just will not be able to exploit planarity to increase accuracy.

3.5.2. Proportioning Rotational and Translational Observability

Points sampled on an "empty" flat ground plane will always result in the same pattern in the point cloud, making it impossible to estimate the translation in the directions parallel to the $x^{(O)}$- and $z^{(O)}$-axes. Likewise, if the number of points sampled on the ground plane is too dominant compared to points sampled on objects perpendicular to the ground plane, the standard ICP-based scan matching will fail. Indeed, in that case, ICP will converge to the identity transformation since the way the points on the ground are sampled is the same along consecutive scans. Hence, it is important to select an equal amount of points on structures that are perpendicular to the ground plane; otherwise, ICP will be biased towards the identity transformation (of course, we assume that there are such structures present in the point cloud of a single scan; if not, this would be a degenerate case).

In order to fulfill this constraint, we start by defining the following six *"observability"* values for each point $\mathbf{p}_i$ in the point cloud: $|\mathbf{n}_i \cdot \mathbf{x}^{(O)}|$, $|\mathbf{n}_i \cdot \mathbf{y}^{(O)}|$, $|\mathbf{n}_i \cdot \mathbf{z}^{(O)}|$, $(\mathbf{p}_i \times \mathbf{n}_i) \cdot \mathbf{x}^{(O)}$, $(\mathbf{p}_i \times \mathbf{n}_i) \cdot \mathbf{y}^{(O)}$, $(\mathbf{p}_i \times \mathbf{n}_i) \cdot \mathbf{z}^{(O)}$. In all these definitions, $\mathbf{x}^{(O)} = (1, 0, 0)$, $\mathbf{y}^{(O)} = (0, 1, 0)$, and $\mathbf{z}^{(O)} = (0, 0, 1)$ (in Ouster coordinates), and $\mathbf{n}_i$ represents the normal vector of the point $\mathbf{p}_i$. The former three values indicate the contribution of a point $\mathbf{p}_i$ to the observability of the three unknown components of the translation vector. The last three values indicate the contribution of a point $\mathbf{p}_i$ to the observability of the different unknown angles (pitch, yaw, and roll) of the UAV. Note that points located further away from the sensor contribute more. The idea is to sample points equally distributed among those that score high for each of the six observability values.

3.5.3. Incorporating Low-Level Features

In addition to maximizing the rotational and translational observability, it is also beneficial to select points that are sampled on solid objects such as the facades of houses. One way to deal with this is to scale the observability values from the previous section with an additional value denoting the planarity of the point's neighborhood, which is defined as $\psi_i = \frac{\lambda_{2,i} - \lambda_{3,i}}{\lambda_{1,i}}$, in which $\lambda_{1,i}$, $\lambda_{2,i}$, and $\lambda_{3,i}$ are the eigenvalues corresponding to the eigenvectors $\mathbf{v}_{1,i}$, $\mathbf{v}_{2,i}$, and $\mathbf{v}_{3,i}$ computed from the covariance matrix of a set of neighboring points of $\mathbf{p}_i$. We thus propose updating the values as follows: $\psi_i|\mathbf{n}_i \cdot \mathbf{x}|$, $\psi_i|\mathbf{n}_i \cdot \mathbf{y}|$, $\psi_i|\mathbf{n}_i \cdot \mathbf{z}|$, $\psi_i(\mathbf{p}_i \times \mathbf{n}_i) \cdot \mathbf{x}$, $\psi_i(\mathbf{p}_i \times \mathbf{n}_i) \cdot \mathbf{y}$, $\psi_i(\mathbf{p}_i \times \mathbf{n}_i) \cdot \mathbf{z}$, in which ψ_i denotes the planarity of point $\mathbf{p}_i$. In this way, we give a higher weight to those points that are sampled on "robust" planar regions.

Eventually, we sort the points based on all six values, generating six ordered lists. For every sorted list, we select the N highest-valued points. Note that this implies that fewer than $6N$ points can be selected in total, as the same point can be part of multiple selected shortlists. The exact value of N is determined through experimental evaluation.

3.6. Stitching of Camera Data

To create georeferenced orthomosaics for the multispectral and thermal image data, we apply structure from motion (SfM) and multiview stereo using the Metashape software from Agisoft [21]. First of all, we calibrate the reflectance via panels we placed in the field and via irradiance sensor data that measure incoming sunlight. We also disable all images obtained while turning the UAV between flight lines or during takeoff and landing. Since all images can be georeferenced with the Quanta GNSS data, this filtering of images is rather easy. Additionally, the GNSS+IMU data, in combination with basic extrinsic calibration between the cameras and the Quanta, allow for a good initialization when aligning the cameras (i.e., computing their poses). After this alignment, we obtain a sparse point cloud, which we filter both using the criteria provided by Agisoft Metashape and by manually removing outliers. Next, we optimize the camera poses and disable those cameras that still have a pixel error greater than 1, followed again by pose optimization.

Given the camera poses, which are now of high quality, we build the dense point cloud, digital elevation model (DEM), and orthomosaic. Using this orthomosaic, we can easily locate our GCP markers in the field and fill in the coordinates we measured on the ground, where we leave the conversion from Belgian Lambert 72 coordinates to Agisoft Metashape. Finally, we optimize the camera alignment again with this new information and rebuild the dense point cloud, DEM, and orthomosaic, which is now properly and fully georeferenced.

3.7. Georeferencing LiDAR Point Cloud and Data Fusion

For the georeferencing of the LiDAR point cloud, we again use the GNSS data from the Quanta (recall that it outputs 200 Hz attitude and position information), albeit in a different way than the image data. For every "firing" of the laser beams, we know the absolute position of the LiDAR sensor (of course after combining it with the Quanta–Ouster calibration), as explained in Section 3.2, from which we can compute the absolute position of each point. Obviously, since we improved the point cloud registration (and thus the location of the drone), the geolocation of each point was updated accordingly.

To eventually merge the image data from the multispectral and thermal cameras with the LiDAR point cloud, we currently adopt late fusion. The point clouds derived from the structure of the motion process (cfr. Section 3.6) are inherently aligned with the point cloud generated using LiDAR SLAM through georeferencing. Although we aim for an early fusion approach in the future, late fusion between the multispectral, thermal, and lidar point cloud through georeferencing is quite straightforward. More specifically, we use the accurate (and dense) LiDAR point cloud (or its 2D orthoprojection) as the basis and attribute the spectral information from the multispectral and thermal orthomosaic to each corresponding point. In this manner, we obtain a multispectral point cloud.

4. Experiments and Results

4.1. Impact of Point Cloud Registration on the Accuracy

In order to indicate the difference in accuracy between a standard LiDAR SLAM approach and our own improved processing pipeline (combining GNSS/INS data with proper point cloud registration), we depict two resulting point clouds and drone trajectories for a dataset acquired in Bottelare (Belgium) in Figure 7. The left one is computed using poses estimated by a standard LiDAR SLAM algorithm, whereas the right one is computed using the Quanta INS data combined with our point cloud registration algorithm. One can clearly notice the drift due to localization errors while looking at the outlines of the buildings. The accumulated errors give rise to ghosting effects, such as the duplicated walls of the homestead indicated by the yellow ellipses in Figure 7.

To demonstrate the increase in accuracy between the point cloud reconstructed using only the Quanta GNSS/INS data and the one using our improved processing pipeline, Figures 8 and 9 depict two point clouds. The left one is the result of using only Quanta poses, whereas the right is the result of integrating proper point cloud registration. It is clear that the "Quanta" point cloud has some severe artifacts, such as the ghosting effect of the crane in Figure 8 or the benches in the courtyard in Figure 9. Overall, the improved point cloud has a lot more detail and contains much less noise. The windows in the houses and the homestead are clearly distinguishable in the improved point cloud, whereas for the "Quanta" point cloud, some of the windows are "absorbed" in the noise from points sampled on the walls.
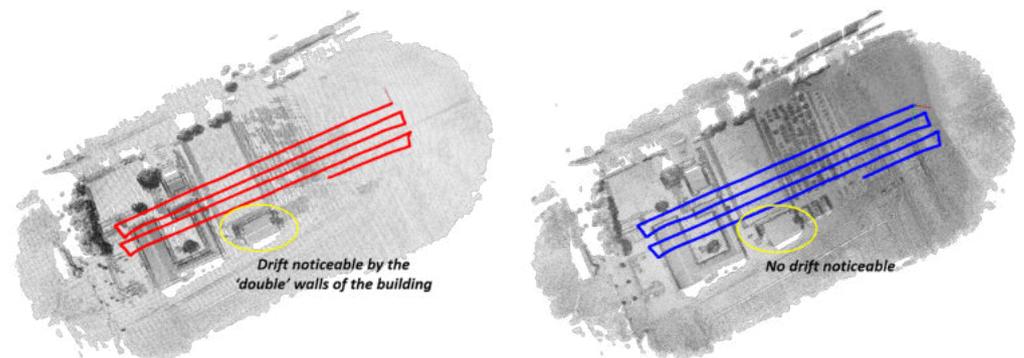
Figure 7. Two resulting point clouds and drone trajectories for the dataset acquired at Bottelare (Belgium) computed using standard LiDAR SLAM (**left**) and our improved SLAM algorithm combining the GNSS and IMU data from the Quanta with proper point cloud registration (**right**).
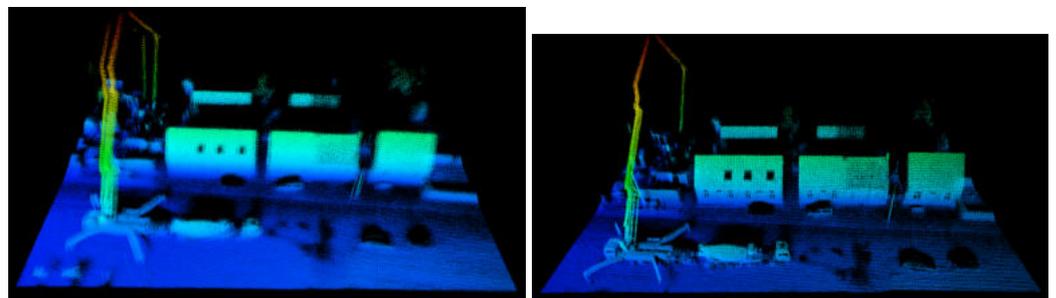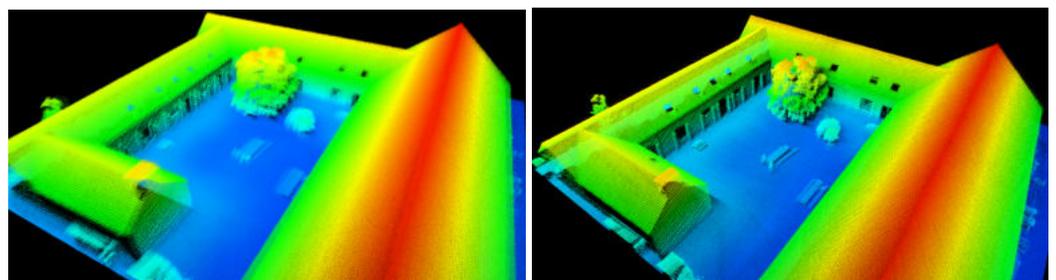

Figure 8. Zoomed-in part of the Bottelare point cloud computed using only the Quanta INS data (**left**) and incorporating proper point cloud registration (**right**). For the crane, ghosting artifacts are present in the "Quanta" point cloud, whereas it is nearly perfectly reconstructed in the improved point cloud. Furthermore, the overall noise is largely reduced (cf. the windows and shape of the truck).

Figure 9. Zoomed-in part of the Bottelare point cloud computed using only the Quanta INS data (**left**) and incorporating proper point cloud registration (**right**). The amount of noise is clearly reduced, as can be noticed near the windows and the edges of the benches on the courtyard.

4.2. Impact of GNSS/IMU System on Structure from Motion

In this section, we investigate the importance of having a good GNSS+IMU system. In our approach, we made use of the SBG Quanta GNSS+IMU in post-processed kinematics mode, but more basic systems could also have been used. After all, we need it mostly for the initialization of LiDAR SLAM or SfM. Above, we qualitatively showed the benefits in the case of LiDAR SLAM. Below, we give quantitative results for SfM. We consider the scene from Figure 15 and regard camera poses, produced from the full Metashape pipeline, initialized using Quanta poses, as ground truth. By adding zero-mean Gaussian noise, we simulate new poses that could be reported by a GNSS+IMU sensor. We then test to what extent Metashape's camera alignment step can fix the inaccuracies in the initial poses and if worse initialization leads to longer alignment times. In all tests, we used Metashape's default SfM parameters. More specifically, we used five GNSS+IMU set-ups:

1. A perfect system producing the ground truth poses; i.e., we use a standard deviation of 0 to produce the zero-mean noise.
2. The SBG Quanta in post-processed kinematics (PPK) mode. This is the same mode that we have used throughout the paper. According to the Quanta hardware manual [22], for UAV applications, this has a position accuracy of 1 cm in the horizontal direction ($x^{(W)}$ and $y^{(W)}$), 2 cm in the vertical direction ($z^{(W)}$), and an attitude accuracy of 0.025° for roll and pitch and 0.08° for heading (yaw). We used these values as standard deviations for our Gaussian noise.
3. The SBG Quanta in single-point (SP) mode for a single antenna. The accuracies are 1.2 m and 1.5 m for the horizontal and vertical positions, respectively, 0.05° for roll and pitch, and 0.5° for heading.
4. A basic GNSS without IMU. As the noise standard deviations for the camera positions, we take the same values as for the SP Quanta, but we keep the attitudes fixed at nadir.
5. No GNSS or IMU.

In the latter case, we have no georeferencing, and Metashape will work in a local reference frame. To compare to the world reference frame used in the ground-truth (GT) poses, we employed global PCA (Principal Component Analysis) alignment, ensuring that the camera locations have the same mean and principal directions in both reference systems. We also fixed the scale indeterminacy by aligning the eigenvalues for the first principal component. This PCA alignment step could also be useful for the other GNSS+IMU systems, e.g., in case the effective mean of our sampled noise is not actually zero.

When we do not initialize camera poses, i.e., in the situation without GNSS or IMU, Metashape fails to align all cameras. The number of aligned cameras varies by run, showing that Metashape's algorithm has some stochasticity. As shown in Figure 10, this leads to a wide range of alignment times, which is positively correlated with the number of cameras Metashape actually manages to align. Note that this alignment time does not include image matching. Further note that each of the 10 different wavelengths the Micasenses capture at are counted as different images ("cameras"). The sensor for each wavelength is indeed at a slightly different position, and of course there are also two different cameras capturing each at only five wavelengths. However, the software takes into account that the relative poses should remain fixed. Not only are the alignment times extremely variable in case of no GNSS or IMU, but the same can be said of the produced pose quality (even when only taking aligned cameras into account). For this reason, we exclude this GNSS+IMU setup from the following figures.

For the actual GNSS (and IMU) set-ups, we see an increase in alignment time for decreasing GNSS+IMU quality, although the differences are modest. This is presented in Figure 11. For each system, we repeat the noisy initialization and alignment 25 times to account for the noise's randomness. To evaluate the poses after camera alignment we compare to the ground truth via Euclidean distance for the locations, and by computing the absolute angle of the relative rotation matrix $R^{-1}R_{\mathrm{gt}}$, when converted to axis–angle format. Here, R and R_{gt} denote the rotation matrices of an output camera pose and of the corresponding ground truth pose, respectively. These distances and angles are then averaged over all camera poses in a run. Boxplots of the results are presented in Figure 12 for position and in Figure 13 for rotation.

The results without a GNSS or IMU for comparison ranged from 20.8 cm to 33.6 m (i.e., more than a factor of 15 higher) for position, and from 7.01° to 35.6° for rotation. The median errors were closer to the lower values though: 32.2 cm for position and 12.5° for rotation. However, we need to emphasize that this is when only taking aligned cameras into account; the fact that we do not have any pose information for a number of cameras makes the option of excluding a GNSS significantly worse. From Figures 12 and 13, it can be observed that the PPK mode reduces the errors by an order of magnitude compared to the SP mode, both for position and for rotation. On the other hand, recall that the GNSS accuracy was about 100 times better for PPK and SP. Therefore, there are clearly diminishing returns. Finally,

the inclusion of an IMU makes little difference in this situation. In particular, Metashape's SfM can retrieve the attitude information regardless of any initialization.
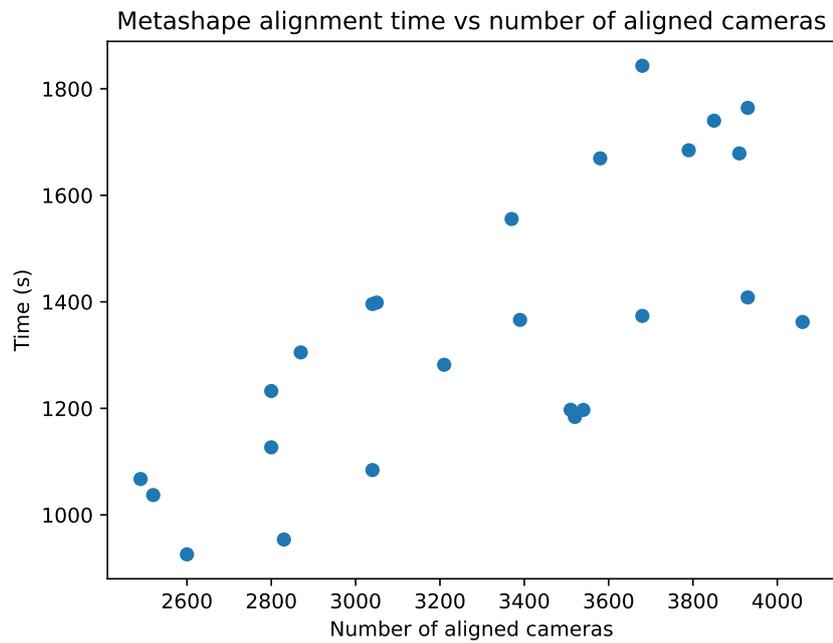

Figure 10. Metashape's camera alignment times, when starting without initialization, compared to the number of cameras Metashape is able to align. We used a computer equipped with an Intel i9-10900X processor, 64 GB of RAM, and an NVIDIA RTX 3080 Ti GPU.
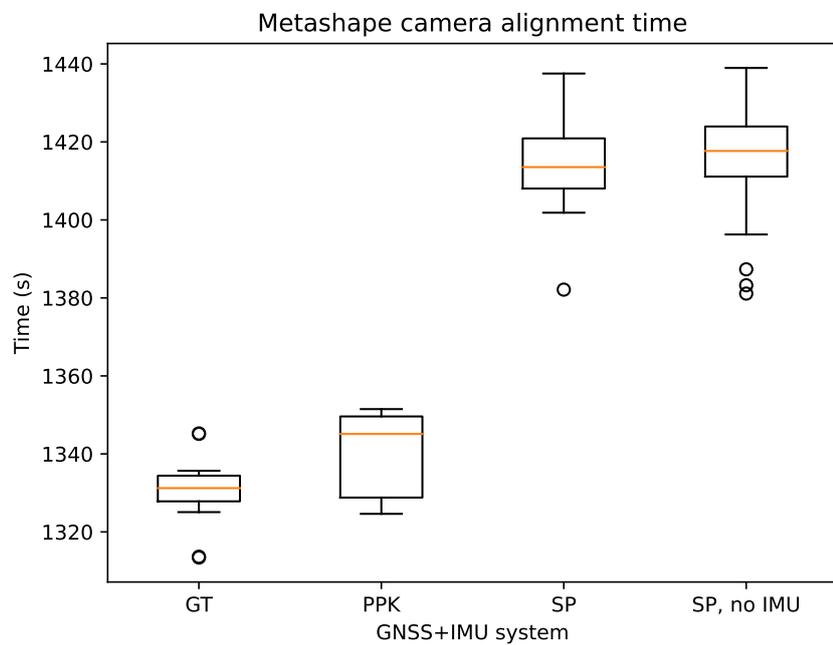

Figure 11. Boxplots of Metashape's camera alignment times for the different GNSS+IMU systems.
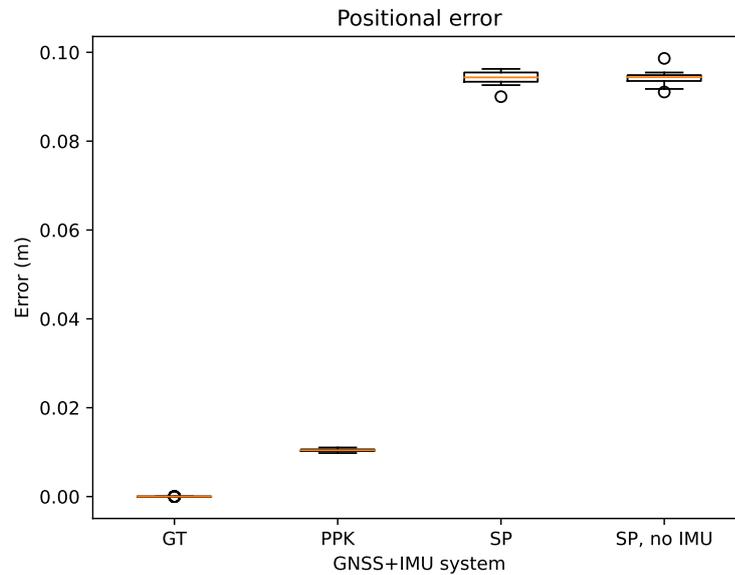
Figure 12. Boxplots of the Euclidean positional error between camera poses obtained by Metashape's camera alignment for initializations corresponding to different GNSS+IMU systems.
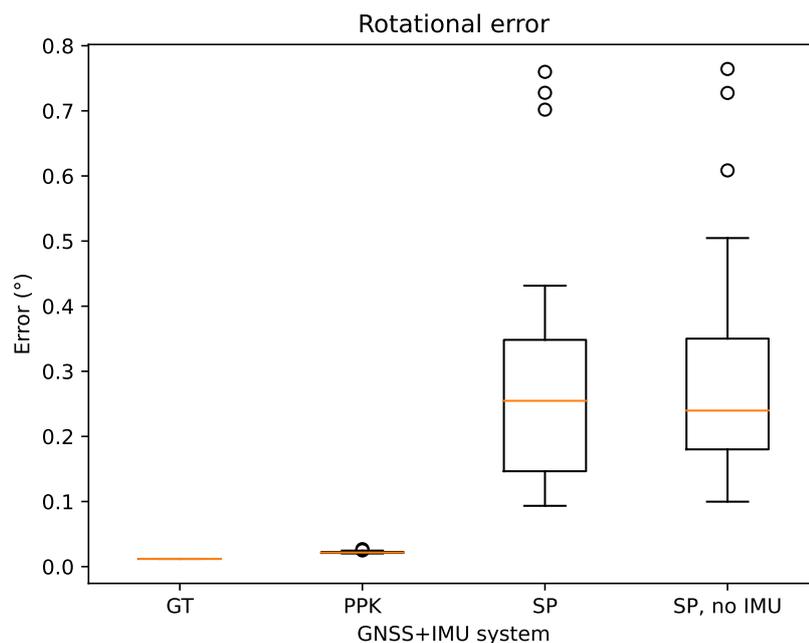
Figure 13. Boxplots of the rotational error between camera poses obtained by Metashape's camera alignment for initializations corresponding to different GNSS+IMU systems.

4.3. Final Results of Multi-Spectral Point Clouds

In Figure 14, the resulting multispectral point cloud from the Bottelare dataset is shown using three bands corresponding to the red, green and blue wavelengths. In Figure 15, we present some results from the late fusion of the LiDAR, multispectral, and thermal data acquired at a corn field in Sint-Truiden. The resulting point cloud is 14-dimensional, of which 3 are spatial, 1 is thermal, and 10 are multispectral. Figure 16 similarly shows late fusion results for the LiDAR and multispectral data we captured at Bottelare. Note how the RGB-colored image is dominated by shades of green, whereas in the higher wavelengths, we see more variation. By taking the multispectral data into account we can therefore more

easily distinguish between different types of (green) grass, crops, leaves, etc., compared to using only RGB information.

Figure 14. The resulting multispectral point cloud from the Bottelare dataset, here colored using the bands corresponding with the red, green, and blue wavelengths.
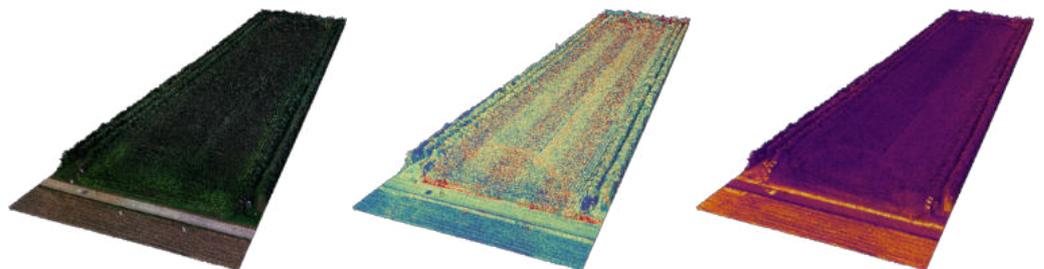

Figure 15. Visualizations of the multispectral+thermal point cloud captured at a corn field in Sint-Truiden. The left image shows the point cloud colored by the red green and blue color bands. The middle uses the spectral color map for the 842 nm near-infrared multispectral band. The colors in the image to the right represent temperature, using the Inferno color map.
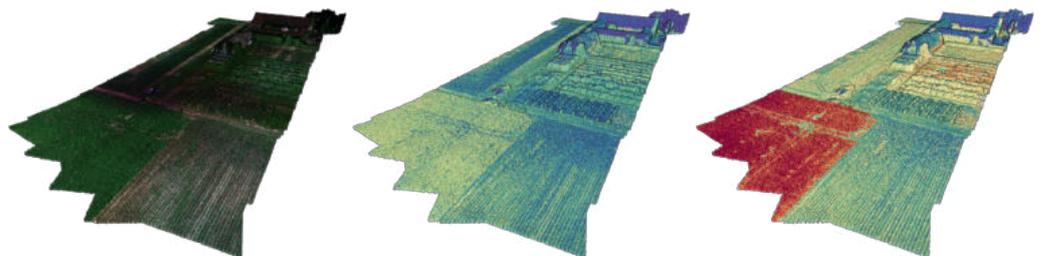

Figure 16. Visualizations of the multispectral point cloud captured at Bottelare. We show the RGB-coloring of the point cloud and two spectral color map coded colorings for the 717 nm (red edge) and 842 nm (near infrared) multispectral bands.

5. Conclusions

In this paper, we presented our multimodal UAV payload and processing pipeline. The payload consists of an Ouster LiDAR scanner, a Micasense RedEdge dual multispectral camera system, a Teax ThermalCapture 2.0 thermal camera, and a professional-grade GNSS system from SBG. Thanks to the combination of high-frequency navigation outputs from the GNSS and a dedicated point cloud registration algorithm, we obtain highly accurate position and attitude information, which is used to precisely align the data from the different modalities. In addition, for the structure-from-motion process, having access to a high-quality GNSS yields significant improvements, but at least for aerial photography, we found that an IMU is rather optional.

At this moment, late fusion is used to combine the data from the different modalities into one single data structure, being a 14-dimensional point cloud (3 spatial, 1 thermal, and 10 multispectral dimensions) which has subcentimeter accuracy. In the future, we plan to integrate early fusion of the multimodal data in our pipeline in order to more tightly couple the point cloud registration and photometric bundle adjustment, thereby resolving the remaining minor artefacts.

Author Contributions: Methodology, M.V., L.D., W.M., B.D.W. and H.L.; software, M.V. and L.D.; validation, B.D.W. and R.H.; investigation, M.V., L.D. and R.H.; resources, R.H. and B.D.W.; data curation, M.V. and L.D.; writing—original draft preparation, M.V. and L.D.; writing—review and editing, W.M. and H.L.; supervision, W.P., W.M. and H.L.; funding acquisition, W.P., W.M. and H.L. All authors have read and agreed to the published version of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Maes, W.; Steppe, K. Perspectives for remote sensing with unmanned aerial vehicles in precision agriculture. *Trends Plant Sci.* **2019**, *24*, 152–164. [CrossRef] [PubMed]
2. Demol, M.; Verbeeck, H.; Gielen, B.; Armston, J.; Burt, A.; Disney, M.; Duncanson, L.; Hackenberg, J.; Kukenbrink, D.; Lau, A.; et al. Estimating forest above-ground biomass with terrestrial laser scanning: Current status and future directions. *Methods Ecol. Evol.* **2022**, *13*, 1628–1639. [CrossRef]
3. Zhang, J.; Singh, S. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 2174–2181.
4. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets Robotics: The KITTI Dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
5. Cvišić, I.; Marković, I.; Ivan, P. SOFT2: Stereo Visual Odometry for Road Vehicles Based on a Point-to-Epipolar-Line Metric. *IEEE Trans. Robot.* **2022**, *39*, 273–288. [CrossRef]
6. Graeter, J.; Wilczynski, A.; Lauer, M. LIMO: Lidar-Monocular Visual Odometry. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 7872–7879. [CrossRef]
7. Shin, Y.-S.; Park, Y.S.; Kim, A. DVL-SLAM: Sparse depth enhanced direct visual-LiDAR SLAM. *Auton. Robot.* **2020**, *44*, 115–130. [CrossRef]
8. Wang, Z.; Zhang, J.; Chen, S.; Yuan, C.; Zhang, J.; Zhang, J. Robust High Accuracy Visual-Inertial-Laser SLAM System. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 6636–6641. [CrossRef]
9. Nagai, M.; Chen, T.; Shibasaki, R.; Kumagai, H.; Ahmed, A. UAV-Borne 3-D Mapping System by Multisensor Integration. *IEEE Trans. Geosci. Remote Sens.* **2009**, *47*, 701–708. [CrossRef]
10. Qian, J.; Chen, K.; Chen, Q.; Yang, Y.; Zhang, J.; Chen, S. Robust Visual-Lidar Simultaneous Localization and Mapping System for UAV. *IEEE Geosci. Remote Sens. Lett.* **2022**, *19*, 1–5. [CrossRef]
11. Haala, N.; Kölle, M.; Cramer, M.; Laupheimer, D.; Zimmermann, F. Hybrid georeferencing of images and LiDAR data for UAV-based point cloud collection at millimetre accuracy. *ISPRS Open J. Photogramm. Remote Sens.* **2022**, *4*, 100014. [CrossRef]
12. Haala, N.; Kölle, M.; Cramer, M.; Laupheimer, D.; Mandlburger, G.; Glira, P. Hybrid Georeferencing, Enhancement and Classification of Ultra-High Resolution UAV LiDAR and Image Point Clouds for Monitoring Applications. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2020**, *V-2-2020*, 727–734. [CrossRef]
13. Cramer, M.; Haala, N.; Laupheimer, D.; Mandlburger, G.; Havel, P. Ultra-High Precision UAV-based LiDAR and Dense Image Matching. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2018**, *XLII-1*, 115–120. [CrossRef]
14. Bultmann, S.; Quenzel, J.; Behnke, S. Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles. In Proceedings of the European Conference on Mobile Robots ECMR, Bonn, Germany, 1 August–3 September 2021; pp. 1–8.
15. Shi, J.; Zhu, Z.; Zhang, J.; Liu, R.; Wang, Z.; Chen, S.; Liu, H. CalibRCNN: Calibrating Camera and LiDAR by Recurrent Convolutional Neural Network and Geometric Constraints. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 10197–10202. [CrossRef]

16. Zhou, Q.; Yu, L.; Zhang, X.; Liu, Y.; Zhan, Z.; Ren, L.; Luo, Y. Fusion of UAV Hyperspectral Imaging and LiDAR for the Early Detection of EAB Stress in Ash and a New EAB Detection Index—NDVI(776,678). *Remote Sens.* **2022**, *14*, 2428. [CrossRef]
17. Yu, R.; Luo, Y.; Zhou, Q.; Zhang, X.; Wu, D.; Ren, L. A machine learning algorithm to detect pine wilt disease using UAV-based hyperspectral imagery and LiDAR data at the tree level. *Int. J. Appl. Earth Obs. Geoinf.* **2021**, *101*, 102363. [CrossRef]
18. Lin, Q.; Huang, H.; Wang, J.; Huang, K.; Liu, Y. Detection of Pine Shoot Beetle (PSB) Stress on Pine Forests at Individual Tree Level using UAV-Based Hyperspectral Imagery and Lidar. *Remote Sens.* **2019**, *11*, 2540. [CrossRef]
19. Diels, L.; Vlaminck, M.; De Wit, B.; Philips, W.; Luong, H. On the optimal mounting angle for a spinning LiDAR on a UAV. *IEEE Sens. J.* **2022**, *22*, 21240–21247. [CrossRef]
20. SBG Systems. Qinertia UAV. Available online: https://www.sbg-systems.com/products/qinertia-ins-gnss-post-processing-software/#qinertia-uav (accessed on 2 February 2023).
21. Agisoft Metashape Professional (Version 1.7.6). 2021. Available online: https://www.agisoft.com/downloads/installer/ (accessed on 2 February 2023).
22. SBG Systems. Quanta Series Hardware Manual. 2020. Available online: https://support.sbg-systems.com/sc/hp/files/latest/44466391/44466379/1/1652716028703/Quanta+-+Hardware+Manual.pdf (accessed on 2 February 2023).