The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLVIII-1/W4-2025
EuroCOW 2025 – European Workshop on Calibration and Orientation Remote Sensing, 16–18 June 2025, Warsaw, Poland

A generic multi-lidar data batching strategy on the sensor driver level

Tamás Faitli[1], Heikki Hyyti[1], Juha Hyyppä[1], Antero Kukko[1], Harri Kaartinen[1]

[1] Dept. of Remote Sensing and Photogrammetry, Finnish Geospatial Research Institute,
FI-02150 Espoo, Finland - (tamas.faitli, heikki.hyyti, juha.hyyppa, antero.kukko, harri.kaartinen)@nls.fi

Commission I

Keywords: Multi-lidar, Sensor fusion, Lidar synchronization, Lidar batching, forest harvester, SLAM

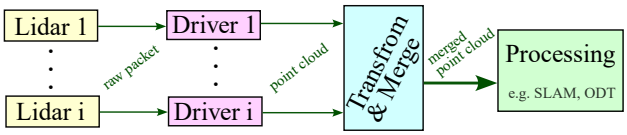
Abstract

This paper addresses how to utilize multiple spinning lidar sensors for real-time applications. Especially how to derive back the problem to having only a single lidar input, to which there are countless available algorithms solving odometry, mapping, object detection and tracking and many other tasks. We provide a strategy that can be implemented to most if not all spinning lidars on the market. Instead of traditional data batching that accumulates data packets based on the spinning angle, we propose batching based on the sampling time, which also enable us to ensure strict time alignment within the multiple lidar sources. In order to demonstrate our batching strategy, we provide a case study where we evaluated a SLAM algorithm with a single and a dual-lidar setup. Our batching algorithm enabled us to use the SLAM algorithm that was previously designed for a single spinning lidar without any additional change, while it showcased benefits, especially in stability due to the larger field of view and reduced occlusion.

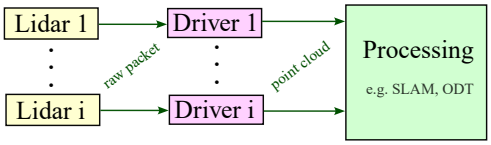
1. Introduction

Spinning lidars with fast and accurate range measurements has become essential in many modern real-time automation applications. As the cost of these sensors is decreasing, today's question is more about how to utilize multiple units on the same platform rather than concerns related to its adoption. Combining multiple units can increase field of view (FOV), reduce occlusion problems posed by the platform or the environment, and provide redundancy in case a sensor fails. However, it is not perfectly clear and explicitly discussed how to utilize multiple sensor units at the same time. Most state-of-the-art algorithms reliant on lidar data including odometry, positioning, mapping (e.g. SLAM) and object detection and tracking (ODT) algorithms assume only a single lidar. While efforts on sensor fusion is immense, the focus has primarily been to fuse lidar with other type of sensors, including inertial measurement units (IMUs) and cameras.

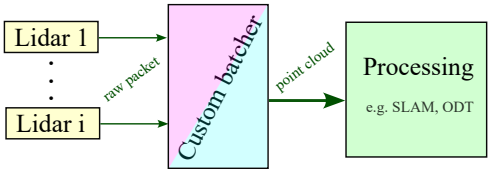
Oftentimes, the manufacturer provides a driver that batches and assembles raw data packets provided by the sensor into a point cloud or single scan. Algorithms that utilize multiple lidars either: (a) combine these point clouds right after the driver and consider them as a single point cloud (from now on referred to as Approach A), or (b) they process each of these point clouds individually and fuse the measurements derived from them (see e.g Jiao et al. 2022; Tasdelen and Sezer 2020 and Figure 1). The first approach can be cumbersome, as ideally, we want to ensure that points are batched within the same time window synchronously for all lidar. This requires additional support from the hardware and/or the driver on top of sensor clock synchronization. In certain cases, synchronous batching might not be guaranteed even if the sensor supports phase locking, due to considerations in mounting angles or inference issues. On the other hand, the second approach requires special care during the algorithm design (e.g. how to estimate and propagate uncertainties of the observations derived from the different lidars), often introducing a lot of complexity. The survey Lee et al. (2024) further discusses efforts taken towards using multiple lidars for odometry purposes.



(a) Combines point clouds into a single unit and handles it similarly to a single lidar system (see e.g., Tasdelen and Sezer 2020).



(b) Develops algorithms that derive measurements from each lidar separately and fuses them on the observation level (e.g., Jiao et al. 2022).



(c) The proposed batching strategy that eliminates potential issues arising in approach (a)

Figure 1. Common (a and b) and proposed (c) strategies to utilize multiple lidar sensors.

We propose a generic strategy to combine data from any number of lidars. Up to the authors knowledge, the only similar solution has been implemented in Nguyen et al. (2023). However, we believe our strategy further generalizes, optimizes, and emphasizes the concept, and it could provide value for multi-lidar data consumers focused not only on state estimation and mapping, but any lidar related tasks. Similar to the Approach A, our goal is to combine data from multiple lidars before it is fed to any other subsystem. Hence easier utilization of the

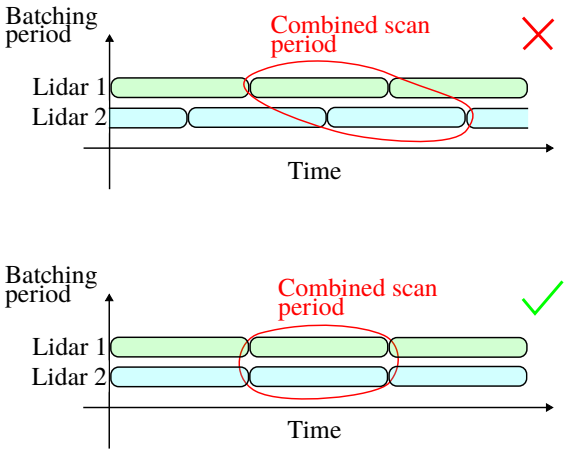


Figure 2. Illustration of time misaligned combination of multiple lidar data.

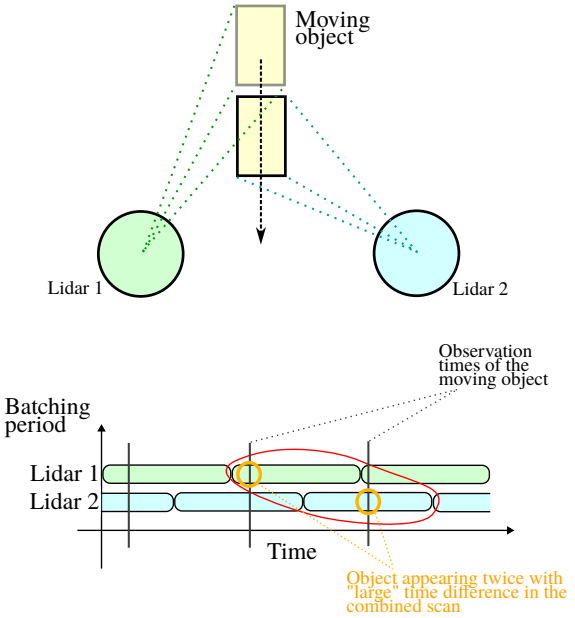


Figure 3. Additional potential issues when considering external dynamic objects in time misaligned scans.

wide range of available algorithms already developed considering only a single lidar, but with the benefits of larger FOV and reduced occlusion. Our multi-lidar batching strategy is implemented directly in the sensor driver, and it ensures time consistency within the batched point cloud data across all sensors. It relies on time information included in the raw data packets (available for most commonly used, if not all, spinning lidars) ensuring time consistency also throughout all ignition cycle even without phase locking mechanisms. Furthermore, it provides flexibility to output partial scans in case faster update times are required, similary as in Qu et al. (2021). On the downside, this approach requires custom implementation of the sensor drivers, including the computation of transforming the raw sensor readings into cartesian coordinates.

1.1 Potential issues with Approach A in real-time applications

Figure 2 illustrates what might happen if we are batching the data based on spinning angle, independently for each lidar. When we would combine those scans afterward relying on Approach A, we would get a scan that covers a significantly larger time period than what we would get from a single lidar. This can cause a lot of issues when we try to integrate this kind of sensor data to real-time algorithms, especially where timely updates are required and synchronization is essential. There might be situations where we want to use different lidars, and they might have different spinning frequencies. Approach A would be inconsistent which scans to actually combine, and the time period at which we want to update our e.g. state estimation algorithm might end up varying causing inconsistencies and difficulties.

Additional issues might appear in scenarios with dynamic surrounding. Especially when it is not the lidar that is moving but external objects whose trajectories we cannot estimate from other sources to apply corrections for. We illustrate in Figure 3 what might happen with highly misaligned batching. As our batching might cover one and a half times more time than what a single lidar scan would, it is possible that the same object appears in the combined scan a full scan period apart. To provide an example, if our lidars spin at 10 Hz, and two lidars both see the same object going at 10 km/h, it might appear in the combined scan twice with around 28 cm offset. An offset we would have no means to correct.

Our approach solves these potential issues, and it makes it easy and straightforward to integrate multi-lidar setups with any real-time algorithm that was designed with a single spinning lidar in mind (e.g. Shan et al. (2020); Xu et al. (2021); Vizzo et al. (2023)).

2. Materials and Methods

2.1 Multi-lidar batching strategy

For our batching strategy, we consider measurement blocks as the smallest building elements. A single measurement block contains a number of lidar reading (usually depending on the number of channels in the given sensor), and they all correspond to the same time stamp. Spinning lidars usually organize their data broadcast into packets containing multiple measurement blocks. Our goal is to batch measurement blocks into single scans with a 2D structure, where each column corresponds to the same time stamp, and the columns are increasing in time. Furthermore, the output rate of batched scans should be consistent.

First we need to initialize our system. An arbitrary output frequency f_{scan} can be selected for the batcher, from which the nominal scan period can be computed as $t_{scan} = 1/f_{scan}$. Then, after caching at least one packet from each lidar, we identify the largest starting timestamp $t_{0,max}$ within all sensor. From which we compute the starting time for the first full scan as:

$$t_{init} = t_{0,max} - (t_{0,max} \bmod t_{scan}) + t_{scan} \ . \qquad (1)$$

Finally, we discard all measurement blocks with stamps smaller than t_{init}, concluding the initialization phase.

The main batching strategy is illustrated in Figure 4. We maintain a custom packet cache, implementing a first in, first out

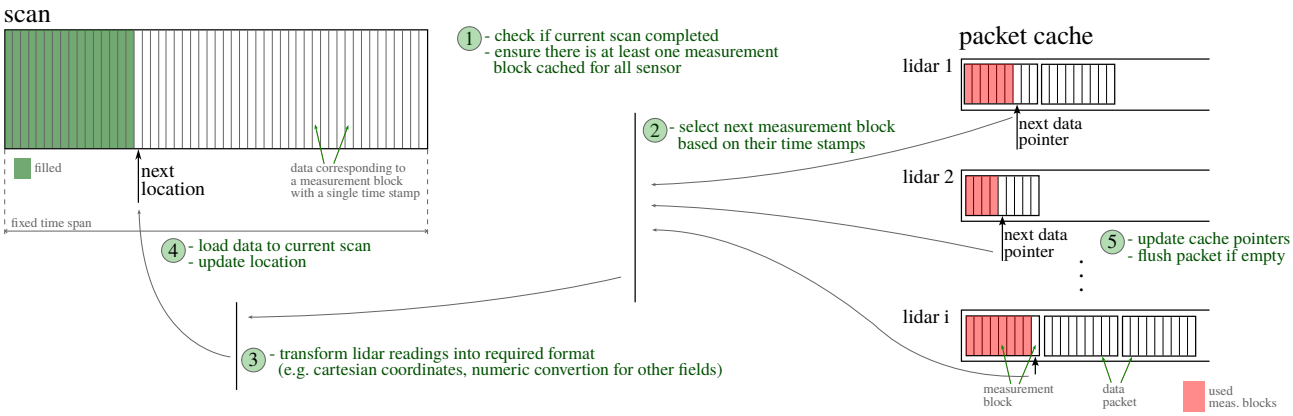


Figure 4. Illustration of the main batching strategy. Two main components are the scan under construction having columns with monotone increasing timestamps, and a packet cache, maintaining a FIFO-like queue for each lidar sensor.

(FIFO) type queue for each sensor. Furthermore, we maintain the location of the next available measurement block within the front (head) packet for all lidar. The algorithm is triggered every time after a new packet is cached, and it repeatedly executes 5 main steps until a stopping condition in the first step yields false. This condition checks whether there is available data for all lidar sensors, in order to ensure that the measurement blocks are selected in a strictly increasing manner. The second step is the selection process itself, where we look through the front elements in the packet cache and identify the measurement block with the smallest timestamp. In the third step we convert the selected measurement block into the desired format. This contains conversion of range values and all other fields from how they are encoded in the raw packets into formats desired. This is also the step where we convert from polar coordinates into cartesian coordinates. In order to improve efficiency, sensor drivers often compute a lookup table once in the initialization phase that already contains extrinsic calibration information on top of the conversion from polar to cartesian coordinates. We do the same, applying known transformation matrices for each lidar, directly converting the raw data into a common body frame, saving additional computations transforming the point clouds once again as it would be in Approach A. The forth step is to load the converted data into the scan under construction at the current location pointer and shift this pointer by one column. Finally, mark the selected measurement block used in the packet cache by incrementing the packet's data pointer. In case it was the last block in the packet, we flush the whole packet.

During the first step we also check whether the currently batched scan has been completed. We check this by maintaining the start time of the next scan by adding t_{scan} to the previous starting time, initialized by t_{init} as discussed previously. If the lowest timestamp in the packet cache is larger than the start time of the next scan, than the batching of the current scan is concluded. We publish it and empty the scan buffer in order to start batching the next one.

2.2 Case study - Forest harvester positioning

In this case study we illustrate the benefits of having multiple lidars on top of a forest harvester for a lidar-inertial based SLAM solution, furthermore illustrate how our custom batcher enables us to run our algorithm previously designed and function only with a single lidar.

We collected data with a dual Ouster 0S0-128 lidar setup, mounted at angles on both side of the boom on a Ponsse Scorpion King CTL forest harvester (see Figure 5). Besides the lidars, we also mounted a Lord Microstrain 3DM-GQ7 GNSS/IMU to support our lidar-inertial positioning algorithm. We recorded data in two scenarios, in which both the forest harvester performed commercial thinning operation inside the forest. In one case, the machine was in a sparse pine forest, providing a scenario considered as an easier environment for lidar based real-time positioning. The other case was a dense and young forest, that is considered as a difficult area including a lot of occlusion and irregular shapes. An image to illustrate both case can be seen in Figure 6. In both cases the machine was continue thinning and go deeper into the forest following a relatively straight trajectory until the total station was able to track the machine, at which point it turned back and drove back to the starting location. The easier scenario took about 15 minutes, while the difficult one took 60 minutes of machine operation. In order to evaluate the positioning of the forest harvester, we recorded reference trajectory throughout the measurements using a Leica TS60 robotic total station, that was tracking a prism also mounted on the machine. The reference trajectories can be seen in Figure 7.

The actual positioning algorithm for this case study was based on our earlier forest harvester related work from Faitli et al. (2023, 2024), with certain changes and improvements. The algorithm expected complete lidar scans and IMU measurements. We first performed a preprocessing step on the scan, where we removed the machine points and corrected for motion distortion. The machine points were removed using a large

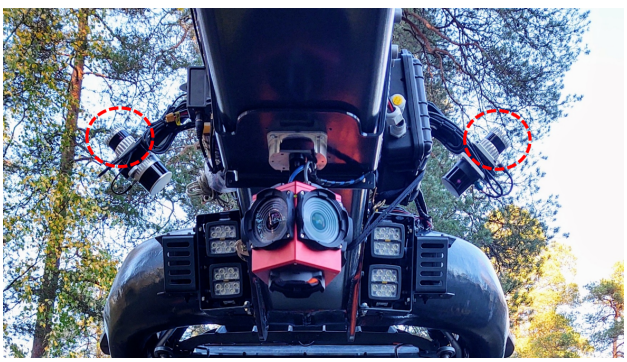
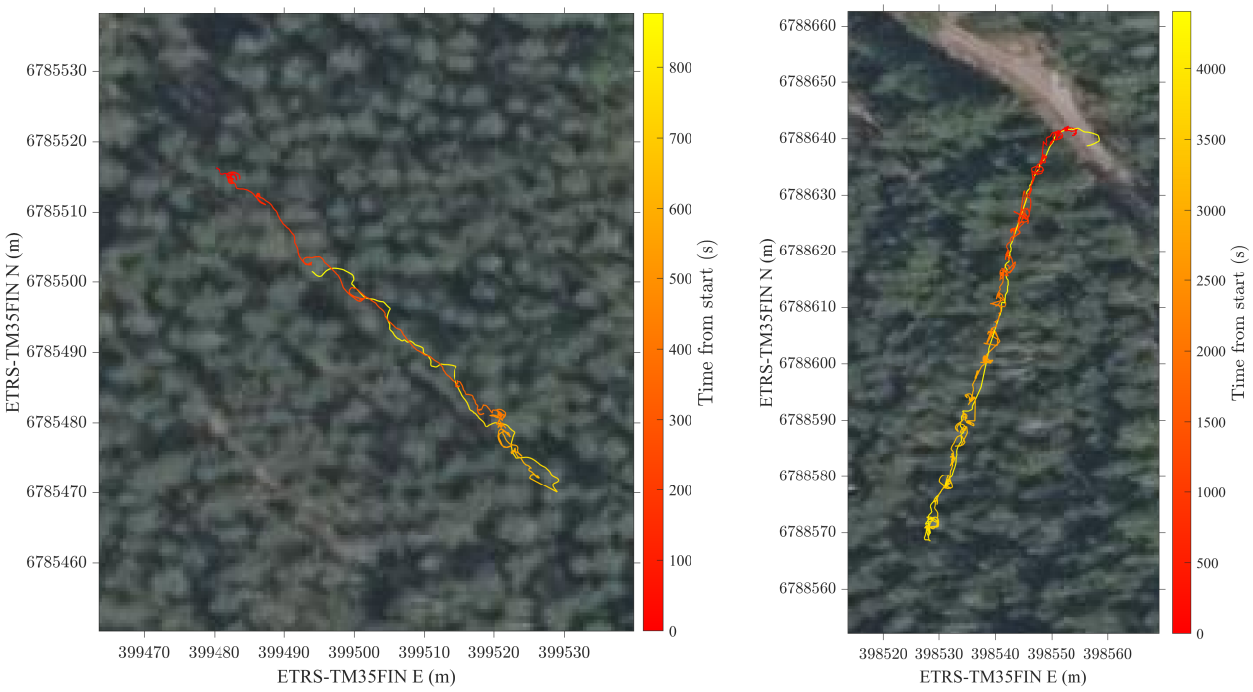


Figure 5. Forest harvester equipped with multiple lidars in different orientations. The topmost sensors on both sides (circled with red) are the Ouster OS0-128 lidars used in this work.

(a) a thinning of a pine forest

(b) a pre-commercial thinning of a mixed-species forest

Figure 6. The two forest environments for a comparative study. Subfigure (a) shows a sparser pine forest that is considered as an easier case, while (b) depicts a younger mixed-species forest that provided the harder use case.

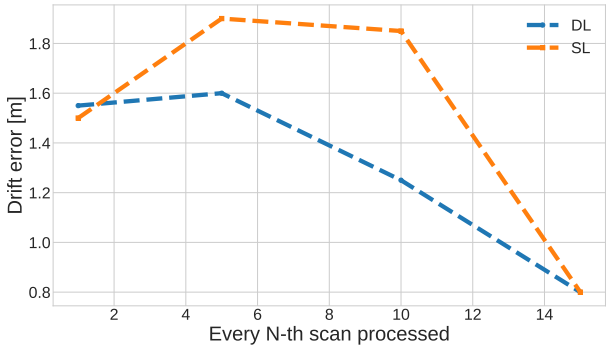


(a) a thinning of a pine forest

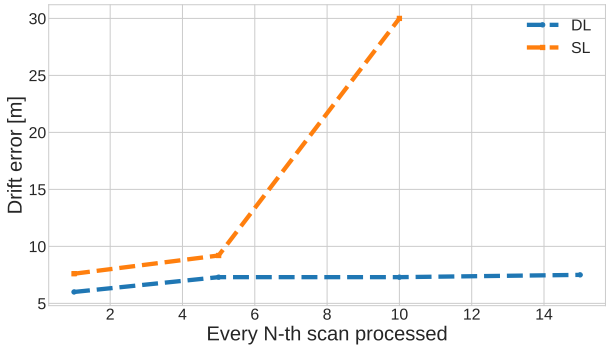
(b) a pre-commercial thinning of a mixed-species forest

Figure 7. Reference trajectories recorded by the total station, overlaid on aerial images by the National Land Survey of Finland from the easier (a) and more difficult (b) forest environment. The trajectories are colored by the time of the measurement. Both sites are located at Evo region in southern Finland.

bounding box with a constant size and location in lidar frame. The box was sized with the preference on removing all machine points even if it sacrificed a lot of non-harvester points. Then the preprocessed scan was roughly aligned to a local map using the Distribution-to-Distribution (D2D) Normal Distributions Transform (NDT) proposed by Stoyanov (2012), and further developed by Kivioja (2022). For this rough alignment, we used an initial guess computed by integrating the IMU measurements, which was already computed earlier during the motion correction. We applied a larger cell size of 2.5 m for the preprocessed scan, while we maintained the local map's NDT representation with 1 m cell size. The result from the rough alignment was fed into a factor graph (Dellaert and Kaess (2017)) as prior factors containing measurements about the absolute pose with respect to the local map frame. Additionally, we generated and added preintegrated IMU factors (Forster et al. (2017)) to this graph. We then optimized the graph using the iSAM optimizer (Kaess et al. (2012)) and updated our current state including pose and IMU biases. When the machine travelled far enough from the previous mapped scan (4.0 m in our experiments), we updated the local map using the most recent scan. Before updating the local map, we registered the scan again but using a smaller cell size of 0.5 m. This result was used then to transform the current scan into the map frame. Then we recomputed the NDT representation with the local map's cell size of 1 m, and merged it into the map itself. To keep the local map reas-



(a) a thinning of a pine forest



(b) a pre-commercial thinning of a mixed-species forest

Figure 8. Drift error for the (a) easy and (b) difficult scenario. DL stands for dual-lidar and SL stands for single lidar system.

onably sized, we only maintained the last 3 scans this way, and always removed the oldest one after exceeding this size. At this point of experiments, we applied no further logic to correct for drifts, such as loop closure mechanisms or additional pose graph optimization.

To understand the benefits of the dual lidar setup, and to showcase that the batching strategy is functional, we measured the maximum drift occurred throughout the measurements. To measure the drift, we first resampled the output trajectory to match the timestamps of the total station trajectory. Then, we identified an initial segment based on the first 10% of accumulated travelled distance. We then used this segment to compute the alignment between the SLAM and the total station trajectories. As it is an initial alignment, the two trajectories were drifting away from each other as time passed. We computed this drift error between the corresponding trajectory points, and observed and noted the largest error that occurred, usually before the machine started driving backwards.

We performed the drift evaluation for trajectories generated by the SLAM algorithm processed using only one of the lidars as input, and then the dual-lidar provided by our batcher. In addition, we wanted to see the stability of the SLAM algorithm with both single and dual lidar configurations. Therefore we reprocessed the trajectories but using only every N-th scans to update the algorithm. The complete list including processing every scan were N = 1, 5, 10 and 15.

2.3 Results and discussion

First of all, we had no issue integrating the multi-lidar data after applying our custom batching strategy. The outcome of the batcher (see Fig. 9) was directly compatible with the original algorithm that was designed for a single spinning lidar.

The observed drift from the forest harvester positioning study can be seen in Figure 8. We found that in the easier (more sparse) forest, our SLAM functions relatively similar whether we used single or dual lidar setup. Both setups remained stable even with updating as little as only every 15th scans. On the more difficult forest environment we however observed a big difference in the two setups. Having two lidars enabled us to update the SLAM algorithm only with every 10th or 15th scans, while having only one lidar the algorithm did not manage to stay stable. Updating with every 10th scan already caused a huge drift, while it just exploded when we tried updating with every 15th scan.

Currently the implementation used only two similar Ouster OS0-128 sensors. The setup has not yet been tested with sensors from other brands nor with more than two similar sensors. In the future work, we will test more sensors from different brands that also contain different number of lidar channels.

Furthermore, it is important to note that some of these lidars might have different modes (e.g. single or dual return), which include different raw data package structures. In cases when it is desired to test different modes, our batcher algorithm is less flexible and requires more implementation effort to use. However, this work has to be experimented and done once for a sensor setup, which enables then the use of most available algorithms designed for a single spinning lidar without additional implementation and design efforts. The one exception, and as limitations might arise with algorithms using the lidar scans as 2D images (e.g. with range or intensity values), feeding them into deep learning based models commonly trained on camera images. As data packages might arrive randomly from the different sensors, and the packages are anyway mixed from the multiple sources, the 2D image representation of the combined scans most likely looks very far from an actual camera image.

3. Conclusions

We have shown an alternative approach to combine data obtained from multiple lidars for real-time applications. Our approach guarantees by design that the merged point clouds have no temporal issues, making multi-lidar systems more robust and compatible with algorithms originally designed for a single lidar system. Our strategy is generic and in theory works with most, if not all, spinning lidar sensors on the market, however it requires additional effort as our implementation must replace existing drivers commonly provided by the manufacturer. Our case study showed that our batching strategy is functional. Furthermore it hinted some of the benefits of having multiple lidars, especially the significant improvement in the the stability of our positioning and mapping algorithm inside difficult forest area. Currently the implementation used only two similar Ouster OS0-128 sensors. In the future work, we will compare sensors from different manufacturers and different amount of lidar beams in them using the proposed framework.

4. Acknowledgements

We gratefully acknowledge Academy of Finland that funded this research through grants “Forest-Human-Machine Interplay

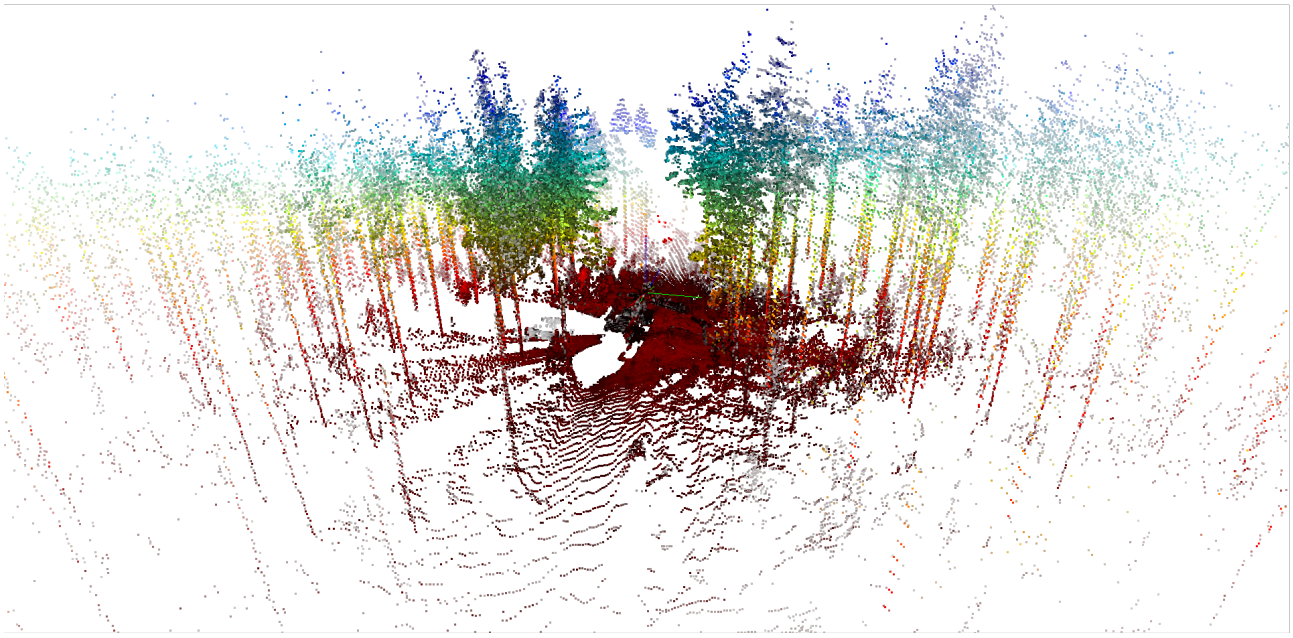


Figure 9. A combined scan batched using our custom strategy.

- Building Resilience, Redefining Value Networks and Enabling Meaningful Experiences" (357908), "Collecting Accurate Individual Tree Information for Harvester Operation Decision Making" (359554) and Ministry of Agriculture, and Forestry of Finland and European Union NextGenerationEU through the project "IlmoStar" (VN/27353/2022).

References

Dellaert, F., Kaess, M., 2017. Factor Graphs for Robot Perception. *Foundations and Trends in Robotics*, 6, 1-139. http://www.nowpublishers.com/article/Details/ROB-043.

Faitli, T., Hakala, T., Kaartinen, H., Hyyppä, J., Kukko, A., 2023. Real-time lidar-inertial positioning and mapping for forestry automation. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48, 145–150.

Faitli, T., Hyyppä, E., Hyyti, H., Hakala, T., Kaartinen, H., Kukko, A., Muhojoki, J., Hyyppä, J., 2024. Integration of a Mobile Laser Scanning System with a Forest Harvester for Accurate Localization and Tree Stem Measurements. *Remote Sensing*, 16(17). https://www.mdpi.com/2072-4292/16/17/3292.

Forster, C., Carlone, L., Dellaert, F., Scaramuzza, D., 2017. On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. *IEEE Transactions on Robotics*, 33, 1-21.

Jiao, J., Ye, H., Zhu, Y., Liu, M., 2022. Robust Odometry and Mapping for Multi-LiDAR Systems with Online Extrinsic Calibration. *IEEE Transactions on Robotics*, 38, 351-371.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J. J., Dellaert, F., 2012. iSAM2: Incremental smoothing and mapping using the Bayes tree. *The International Journal of Robotics Research*, 31, 216-235. http://journals.sagepub.com/doi/10.1177/0278364911430419.

Kivioja, T., 2022. Estimating the covariance of point set registration based on the distribution-to-distribution normal distributions transform. Master's thesis, Aalto University, Espoo, Finland.

Lee, D., Jung, M., Yang, W., Kim, A., 2024. LiDAR odometry survey: recent advancements and remaining challenges. *Intelligent Service Robotics*, 17(2), 95–118. https://doi.org/10.1007/s11370-024-00515-8.

Nguyen, T.-M., Duberg, D., Jensfelt, P., Yuan, S., Xie, L., 2023. SLICT: Multi-Input Multi-Scale Surfel-Based Lidar-Inertial Continuous-Time Odometry and Mapping. *IEEE Robotics and Automation Letters*, 8(4), 2102-2109.

Qu, C., Shivakumar, S. S., Liu, W., Taylor, C. J., 2021. LLOL: Low-Latency Odometry for Spinning Lidars. http://arxiv.org/abs/2110.01725.

Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., Rus, D., 2020. LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. *arXiv*. http://arxiv.org/abs/2007.00258.

Stoyanov, T. D., 2012. Reliable autonomous navigation in semi-structured environments using the three-dimensional normal distributions transform (3D-NDT). PhD thesis, Örebro University, Örebro, Sweden.

Tasdelen, E. A., Sezer, V., 2020. Comparison and Application of Multiple 3D LIDAR Fusion Methods for Object Detection and Tracking. *2020 5th International Conference on Robotics and Automation Engineering, ICRAE 2020*, 64-69.

Vizzo, I., Guadagnino, T., Mersch, B., Wiesmann, L., Behley, J., Stachniss, C., 2023. KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way. *IEEE Robotics and Automation Letters*, 8(2), 1029-1036.

Xu, W., Cai, Y., He, D., Lin, J., Zhang, F., 2021. FAST-LIO2: Fast Direct LiDAR-inertial Odometry. http://arxiv.org/abs/2107.06829.