

Towards real-time forest inventory using handheld LiDAR

Alexander Proudman^{a,*}, Milad Ramezani^{a,b,1}, Sundara Tejaswi Digumarti^a,
Nived Chebrolu^{a,*}, Maurice Fallon^a

^a Oxford Robotics Institute, The George Building, 23 Banbury Road, Oxford, OX2 6NN, UK

^b CSIRO Data61, Queensland Centre for Advanced Technologies, 1 Technology Court, Pullenvale QLD 4069, Australia



ARTICLE INFO

Article history:
Available online 22 August 2022

Keywords:
Forestry
SLAM
Real-time Operation
Automated forest inventory

ABSTRACT

While mobile LiDAR sensors are increasingly used to scan in ecology and forestry applications, reconstruction and characterization are typically carried out offline. Motivated by this, we present an online LiDAR system which is capable of running on a handheld device. Our system is capable of creating 3D point cloud reconstructions of large forest areas, segment and track individual trees, and create an inventory for the detected trees. Segments relating to each tree are accumulated over time, and tree models are completed as more scans are captured from different perspectives. The LiDAR scans are processed in an online fashion, and feedback can be provided to the operator via a screen mounted on the device. This allows the operator to ensure the desired area is mapped satisfactorily without any gaps or missing sections. We employ a pose-graph based SLAM system with loop closures to correct for drift errors allowing us to map large areas accurately. Our mapping system also provides multi-session capability where data captured during different runs can be automatically merged in a post-processing step. In this work, we estimate the Diameter at Breast Height (DBH) of individual trees as an example parameter for the forest inventory. The DBH is estimated online by fitting a cylinder to each tree trunk through a least-squares optimization within a RANSAC loop. We demonstrate our mapping approach operating in two different forests (both ecological and commercial) with the total travel distance spanning several kilometres. Further, we also provide experimental results comparing our DBH estimation to ground-truth measurements recorded manually in an ecological forest (Wytham Woods, Oxford). We demonstrate that our DBH estimates are within ~7 cm accuracy for 90% of individual trees detected in the dataset.

© 2022 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Researchers in ecology and forestry monitor the size and growth of individual trees within a forest to infer arboreal health. It is common to retrieve quantitative metrics such as Diameter at Breast Height (DBH) using manual methods, such as measuring tapes, to ensure consistency with previous surveys. Such metrics go towards forming the forest inventory and describes the state of the forest.

Over the last decade, 3D LiDAR scanners have been used to map forests and extract various metrics from the point cloud data. Automated methods have been developed for tree segmentation and the construction of Quantitative Structure Models (QSMs) [1].

These models can be used to estimate the above-ground biomass and carbon stock by estimating the volume of individual trees [2, 3]. This kind of analysis has traditionally been done on high-quality point cloud data captured with Terrestrial Laser Scanners (TLS), which is typically expensive to obtain. Furthermore, computation of these tree-scale metrics requires analysing a point cloud with a large number of points and is often conducted offline as a post-processing step.

In this work, we present an online forestry mapping system capable of providing real-time feedback to an operator in the field. Our mapping system uses a handheld LiDAR setup with on-board compute capability (see Fig. 2). This provides an easier and more flexible surveying experience for the operator compared to traditional TLS systems.

The design of the mapping system takes into account the challenges of typical forestry surveys. For example, we use an elastic pose-graph structure as the underlying representation, which allows us to scale the mapping to large areas. We also support multi-session mapping allowing the operator to survey the forest in multiple runs, which may be necessary due to constrained battery powers or operator fatigue. Furthermore, we

* Corresponding author.

E-mail addresses: alexander.proudman@wadham.ox.ac.uk (A. Proudman), milad.ramezani@data61.csiro.au (M. Ramezani), tejaswid@robots.ox.ac.uk (S.T. Digumarti), nived@robots.ox.ac.uk (N. Chebrolu), mfallon@robots.ox.ac.uk (M. Fallon).

¹ This presented work was done when the author was with Oxford Robotics Institute.

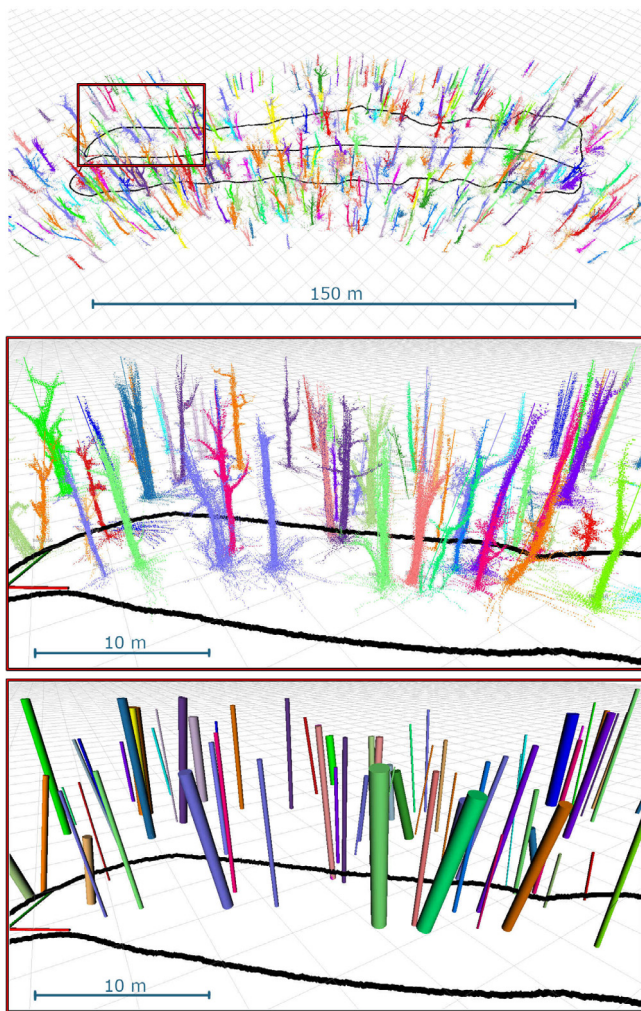


Fig. 1. Visualization of the system output. **Top:** a high level view showing a 150 m transect. The path taken by the person carrying the sensor in Fig. 9 is shown in black. **Centre and Bottom:** Detail of a section is shown with two different representations: one showing each tree segmented and individually coloured and the other shows a cylinder (primary axis and diameter) fitted at breast height.

present an online point-cloud processing tool chain which can segment and track individual trees. The ground surface can also be inferred, allowing us to estimate the DBH of each tree (at 1.4 m height) as the scanning process is carried out.

The main contributions of this work are as follows:

- An online mapping system for forest environments using a handheld LiDAR capable of real-time feedback for the operator to aid in the survey mission.
- Segmentation, tracking and estimation of DBH parameter for individual trees in the forest.
- Maintaining an accurate map of individual trees using a deformable pose-graph based SLAM system with loop closure detection to correct for odometry drift. We use a standard pose-graph based SLAM system with our main contribution being its application for large-scale forestry mapping.
- A multi-session mapping capability where data captured during multiple runs can be merged in a post-processing step.
- Demonstration of the system on challenging large scale datasets from ecological as well as commercial forests spanning several kilometres.

We also performed a quantitative evaluation of our DBH estimates against measurements taken manually at our test site for comparison against a commercial tool (using the final accumulated point cloud). We further perform some studies to characterize the behaviour of different modules in our system.

Video: <https://tinyurl.com/ecmr-forest-mapping>

2. Related work

Advances in LiDAR technology have led to it being used in non-traditional fields such as ecology, forestry and remote sensing. Here, we review some of the state-of-the-art approaches for tree segmentation from LiDAR point clouds. We first review techniques developed for high end Terrestrial LiDAR scanners followed by more recent approaches developed for mobile LiDAR scanners.

2.1. Tree segmentation using terrestrial LiDAR

Many existing tree segmentation techniques operate on a point cloud accumulated from multiple static scanning locations using a Terrestrial Laser Scanner (TLS), in post-processing. Rau-monen et al. [4] and Trochta et al. [5] clustered these scans into smaller point clouds to work around the large size of these scans and to reduce memory consumption. They then segmented tree-level clouds within each cluster by growing segments using assumptions of fixed inter-cluster distance and orientation to infer connectivity. Methods such as [6] utilize concepts from graph theory to find the connectivity between adjacent points and segment each tree. These approaches, however, rely on multiple assumptions about a tree's architecture as well as assuming minimal interconnection between their crowns.

Burt et al. [7] present a software package named *treeseq* that uses a region-growing technique to segment individual trees. Key features in *treeseq*'s design are its independence of forest type, scanning instrument and no assumptions about the tree structure.

2.2. Tree segmentation using mobile LiDAR

Most of the above approaches are intended primarily for very high quality TLS point clouds. Their performance drops when applied to noisier point clouds captured from Aerial Laser Scanners (ALS) and ground based mobile laser scanners (MLS). However, the lower cost and increased portability of mobile laser scanners in comparison to TLS makes them a desirable pick despite the measurement data being noisier. As a result several works have looked at tree segmentation and automatic inventory generation systems for these scanners.

Heo et al. [8] used mobile LiDAR to collect data in an urban area, including parks and streets, to estimate the height of trees and their DBH by calculating the height-above-ground and using a least-squares circle fit approach [9]. They emphasized the advantage of using mobile LiDAR to reduce shadow and occlusion effects, which are more prominent with terrestrial LiDAR systems, especially in an urban forest environment. They utilized a Stencil LiDAR system produced by Kaarta² which includes a Velodyne sensor.

Similarly, Zhou et al. [10] collected LiDAR data with a Velodyne VLP-16 LiDAR sensor. The authors first removed the points residing on the ground and estimated the DBH offline using Random Sample Consensus (RANSAC) algorithm on the segments produced by an Euclidean-based clustering algorithm [11].

² <https://www.kaarta.com/>



Fig. 2. The handheld device used in our system. The labelled components work together to provide online feedback on the forestry data collection.

Westling et al. [12] scanned individual avocado trees with a 5 m spacing using a GeoSLAM Zebedee 1 handheld device. They first voxelised point clouds and conducted a graph-based search over the voxels to find all paths connecting to a root voxel, which was considered to be the tree node. The tree node is segmented from the ground by comparing the height of points locally within a search radius. Aijazi et al. [13] also extract trees and estimate various attributes both from handheld Zebedee device as well as from 3D TLS point clouds.

More recently data driven and learning based approaches have also been used for tree segmentation. Digumarti et al. [14] train a random forest classifier to segment individual trees into their component structures. Convolutional Neural Networks (CNNs), trained on simulated tree data are used in [15] to segment trees from colour and depth images. Windrim and Bryson [16] detect trees in point clouds collected using ALS by encoding the point clouds as a 2D raster image and detecting points of high density. This is followed by applying a segmentation network, adapted from *PointNet* [17], to segment trees. Krisanski et al. [18] further extend the idea of learning based tree segmentation and present a sensor agnostic approach that works with point clouds of different densities.

The above presented approaches typically work as post-processing techniques applied after data collection. This does not allow the operator to perceive the reconstruction during scanning. Our motivation is to develop a LiDAR-driven technique which can reconstruct point clouds, extract individual trees and estimate their structural parameters in dense forests with rough terrain in real-time. A real-time mapping system can ensure full coverage by providing feedback to the operator, to help identify gaps in the scanned region, enable rescanning and ensure that the environment is fully scanned and processed at runtime.

3. Approach

Our device consists of an Ouster OS0-128 LiDAR and an Intel Realsense D435i, each with a built-in IMU (Fig. 2). However, we only use the data from the LiDAR sensor and its IMU measurements for our LiDAR odometry and tree tracking systems. The LiDAR has 128 beams and a 90° vertical field of view (FOV). While it cannot completely cover the environment vertically, the 90° vertical FOV is sufficient to identify individual trees within about 20 m of the sensor. The sensors used in systems such as Zeb-Revo

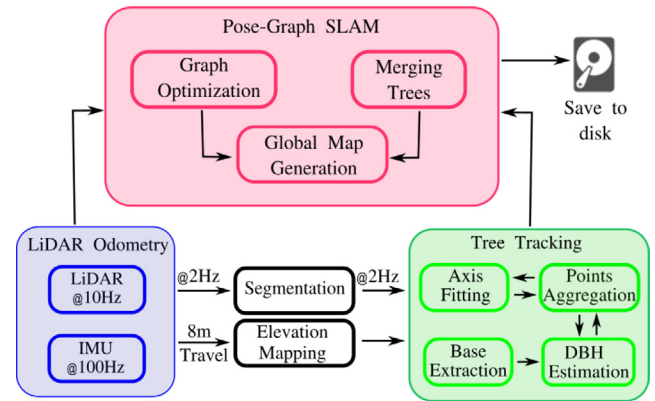


Fig. 3. An overview of the system pipeline, the frequencies of communication between blocks are given.

or Zebedee³ have a narrow field of view and require the sensor to be either rotated or oscillated to cover the environment entirely.

Our proposed system architecture is shown in Fig. 3. Due to the system's online nature, the pipeline is different from many existing methods, and alterations have been made to parallelize and speed up the overall pipeline.

In the following sections, we describe the algorithms and methods used by each functional block within the system architecture. We start with a LiDAR-inertial odometry module to estimate the pose of the sensor which is accurate over short distances (Section 4.1). The estimated odometry is fed to the SLAM module (Section 4) which builds a pose-graph structure as the sensor moves in the environment. Local maps created by accumulating laser scans are attached to the nodes of the pose graph. When a loop closure is detected on re-visiting a place, the pose estimates are corrected for drift. The pose-graph SLAM module allows us to scale the mapping to larger areas in the forest, and extend the mapping capability to multiple-sessions.

Parallel to the pose-graph SLAM module, we process the incoming LiDAR scans to segment and track trees. The *Tree Tracking* block (Section 5), not used in the typical LiDAR forest inventory systems, segments, tracks and fuses each tree detection. Finally, we also compute the DBH parameter for each of the detected trees (Section 6). Advantages of using this online fused data approach are that:

- Prior inventory data can be used to automatically identify trees.
- Success of the data collection can be evaluated online, including the sensor's coverage. This allows the operator to have real time feedback of the reconstruction and ensure that desired area is being mapped completely without any gaps/missing sections in the surroundings.
- Data collection and analysis only takes the amount of time needed to walk along a selected transect (a narrow section of land along which measurements are taken).

4. Localization and mapping

4.1. LiDAR odometry

Our LiDAR odometry system [19] is a factor-graph based windowed smoother which fuses Inertial Measurement Unit (IMU) readings with measurements from a multi-beam LiDAR scanner mounted on a handheld device (Fig. 2). The odometry system

³ <https://geoslam.com/>

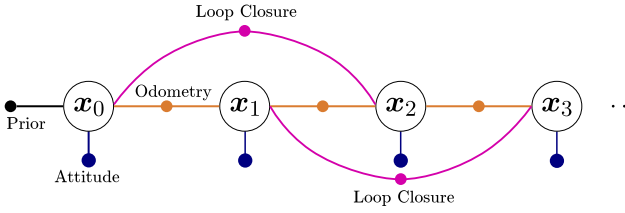


Fig. 4. Pose-graph representation of the SLAM system. Each node represents the pose of the sensor whereas the edges represent the constraints coming from multiple sources.

computes a 6 DoF pose estimate at 2 Hz. The particular configuration used in this work, uses IMU pre-integration to remove motion distortion from scans and to initialize Iterative Closest Point (ICP) registration, specifically the implementation of Pomerleau [20], to determine the relative transform for every 2 metres of distance travelled. To limit the computational time, we restrict the maximum number of iterations in ICP registration to 100. Further, a registration is considered successful only if more than 40% of points in the registered cloud have less than 20 cm of point-to-plane error. The extrinsics between the LiDAR and IMU are computed using an open-source calibration tool called Kalibr [21].

The IMU measurements and the relative transforms then form constraints in a factor graph to estimate the poses in a sliding window optimization utilizing iSAM2 [22] (as part of the GTSAM library). Additionally, since IMU provides gravity information, we can align the clouds with the gravity direction. This alignment helps determine the base of trees using the z direction. Evaluation of our LiDAR odometry system is discussed in Section 7.1.

4.2. Pose-graph SLAM with loop closures

The LiDAR odometry module provides accurate local tracking over short distances (~ 100 m). However, as we travel further, the pose estimated by the odometry system accumulates drift. This drift will in turn affect other parts of the pipeline, and can potentially result in incorrect association of the previously generated maps and the trees segmented in them.

We overcome this problem by implementing a pose-graph SLAM system which corrects for odometry drift using loop closure constraints. Fig. 4 shows the pose-graph structure where nodes represent the pose of the sensor, and the edges represent constraints from multiple sources. New nodes are added to the pose-graph as the device moves through the environment. We initialize a new node after travelling a distance of 5 m. With each node \mathbf{x}_i in the pose-graph, we associate a local map $\mathcal{M}_{\mathbf{x}_i}$ obtained by accumulating the LiDAR scans using the odometry estimate computed in Section 4.1. Here, we assume that the drift accumulated within a local map is small, and does not affect the tree segmentation and tracking procedures. The point cloud of each local map is transformed into a co-ordinate system whose origin corresponds to a pose from the SLAM node.

Consecutive nodes in the graph are constrained with LiDAR odometry (shown in orange), where as loop closure constraints (shown in pink) are obtained on re-visiting a place within the existing map. The entire pose-graph is optimized at regular intervals, which ensures that the current pose estimates are computed with all the constraints in the graph. During optimization, the loop closure constraints are responsible for correcting the drift error.

In our online implementation, we detect the loop closures between non-consecutive nodes in the graph based on their distances in the global coordinate frame. We consider nodes within a distance of 3 m as loop closure candidates. We then retrieve the

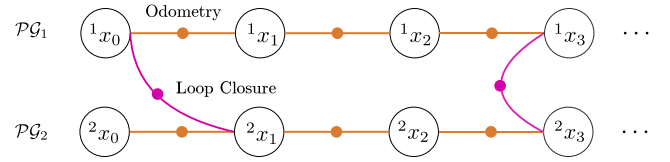


Fig. 5. Mapping over multiple sessions as a post-processing step. Loop closures between the two sessions are computed (in purple) and jointly optimized. A global map is then computed using the jointly optimized poses. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

local maps associated with the corresponding nodes, and perform a scan matching step using ICP. If a successful registration is obtained between the two local map point clouds, we add a loop closure constraint in the pose-graph given by the transformation estimated by ICP procedure. Through the regular addition of loop closure constraints and reoptimization, the drift error is corrected.

Finally, we obtain the combined map of the environment \mathcal{M} by transforming the local maps $\mathcal{M}_{\mathbf{x}_i}$ into the global frame using the pose of the corresponding node in the pose-graph.

The pose-graph SLAM system with loop closure integration allows us to extend our tree mapping approach to larger areas than was possible using LiDAR odometry only.

4.3. Multi-session mapping

While the pose-graph SLAM system in Section 4.2 allows us to map large areas, forestry survey missions can require multiple scanning sessions. This may be due to the limited battery capacity of the handheld device, or to provide a break for the operator to limit fatigue. In certain cases, multiple operators would scan the forests separately to cover larger areas. In all these scenarios, it becomes critical for an effective mapping system to be able to merge maps generated from multiple sessions.

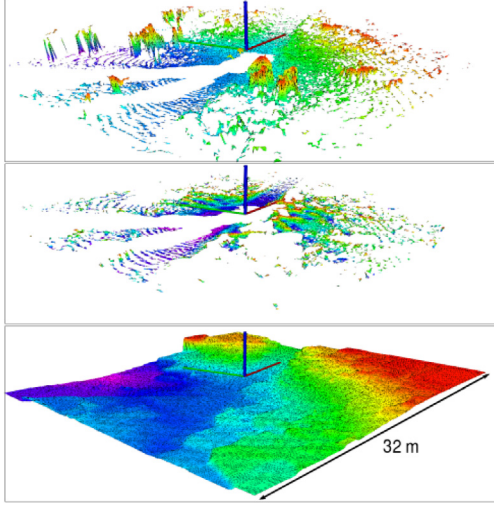
To achieve multi-session mapping capability, we propose an approach where we exploit the pose-graph structures built during individual missions, and merge them in a post-processing step. At the end of each session, we save the pose-graph as well as the corresponding local maps to the disk. Once all the sessions are completed, we use a place recognition system by Giseop et al. named *ScanContext* [23] to propose loop closure constraints between the different sessions. *ScanContext* computes a compact descriptor for each local map in our pose-graph. The descriptor captures the distribution of the points in 3D space and creates a discriminative signature for the point cloud. We compute these descriptors for each local map in all of the sessions, and compare them to retrieve a set of loop-closure candidates. The loop-closure candidates correspond to the places where the local maps created during different sessions overlap with each other. Note that our approach assumes that the individual sessions have an overlap with at least one other session in the entire mission.

We retrieve these loop closure candidates for each session pair, and finally create a combined pose graph consisting of individual pose-graphs along with the inter-session loop closures. An illustrative figure with two pose-graphs \mathcal{PG}_1 and \mathcal{PG}_2 along with the loop closures between them is shown in Fig. 5. We then optimize the combined pose-graph to obtain the final poses of the missions in a common co-ordinate frame. Finally, we obtain the combined map over all sessions by transforming each local map into a common coordinate frame. This is similar to the final map generation step performed for a single pose-graph.

Table 1

Values of the parameters used in the experiments.

Tree segmentation		Tree tracking		Base segmentation	
d_{voxel}	0.2 m	θ_{vertical}	20°	r_{base}	2 m
d_{cluster}	0.37 m	h_{min}	3.5 m		
		d_{merge}	0.7 m		

**Fig. 6.** Filtering process on the elevation map: Original elevation map (top) is filtered to remove spikes (middle). Finally, holes are filled using a morphological closing filter on the grid map (bottom).

5. Tree segmentation and tracking

In parallel to the pose-graph SLAM process, we also segment and track trees from the LiDAR scans. The tree segmentation and tracking module uses the point clouds accumulated with LiDAR odometry. The segmentation results are then integrated into pose-graph SLAM system as illustrated in Fig. 3.

To roughly extract the tree trunks, branches, canopy and shrubbery in the original point cloud, Euclidean segmentation is carried out. Using the PCL library,⁴ the cloud is first downsampled using a voxel filter (with voxel size d_{voxel}) to lessen the radial variation of the point density and to speed up the segmentation process as also done by [7].

After voxel filtering the point cloud, it is reformulated into a k -d tree to quickly find the nearest neighbours of every point. These points are clustered together into groups of points which are within some distance threshold (d_{cluster}) of each other.

Note that the parameters of the voxel filtering and Euclidean segmentation methods are tuned based on the LiDAR sensor's characteristics to maximize the chances that the point clusters contain individual trees. The parameter values used are presented in Table 1.

5.1. Elevation mapping

To perform *Base Segmentation* of the trees being tracked, we employ an open-source sensor-centric elevation mapping framework⁵ which is built upon a universal Grid Map library [24]. The algorithm, presented in [25], generates consistent elevation maps of the environment surrounding the sensor, at the same frequency of the LiDAR output (10 Hz). In this work, we use a resolution of 16 cm for the grid of 32 m × 32 m.

Nevertheless, due to the presence of foliage and branches, the terrain created by the elevation mapping software can contain spikes, making it difficult to detect the precise base of certain trees. To solve this problem, for each cell we compute the normal direction. The normals can be estimated efficiently using the difference between the heights of adjacent cells in the grid map [24]. If the angle subtended by this normal vector with the upwards direction (positive Z axis in this paper) is greater than a threshold (0.5 radians), these cells are removed. This removes the majority of spikes, resulting in smooth terrain with some remaining holes. Finally, we apply a morphological closing filter on the grid map to fill the holes.

Applying the chain of filters is relatively time consuming so we update the elevation map every 8 m of travel distance based on the state estimate from our LiDAR odometry. This travel distance allows online operation. We analyse the computation time of each component including elevation mapping in the next section. Fig. 6 shows this process.

5.2. Tree tracking

We define a minimal ‘tree descriptor’ t with the following characteristics:

- A unique id,
- Major axis of the tree incline (\mathbf{I} , a vector),
- Diameter at Breast Height (DBH) (D),
- 3D position of the tree base (\mathbf{b} , a vector),
- The minimum and maximum height of the point cloud defining the tree (l_{min} , l_{max}).

Every tree descriptor is derived from a corresponding point cloud of the tree points P ($t = f(P)$). These points are accumulated over time, gathering information from different angles as well as reducing the effect of occlusion and improving the estimate of the tree descriptor as Heo et al. [8] indicated.

Most of the characteristics are easily evaluated with the exception of the major axis of the tree. We used fitting of a line, to the set of tree points (P), to find this axis. The algorithm is designed to ignore branches and any potentially segmented crown or ground points. This is described in Section 5.2.1.

Internally, the *Tree Tracker* holds an inventory of n distinct trees T , each defined by a descriptor and set of tree points $T_i \sim t_i, P_i$ for $0 \leq i \leq n$.

Referring to the system diagram, Fig. 3, incoming clusters C from the *Segmentation* block are either discarded or assigned to a new/existing tree descriptor. If assigned to an existing descriptor, the cluster of points is merged with the corresponding set of tree points and the descriptor is re-evaluated $P_{\text{new}} = P_{\text{old}} \cup C$, $t_{\text{new}} = f(P_{\text{new}})$.

An assign/discard decision is made by evaluating a tree descriptor for each input cluster. A cluster is confirmed and subsequently assigned only if it has characteristics within the tolerances of given parameters. The parameter tolerances that resulted in the most consistent tree assignment are:

- Requiring the major axis of the tree to be close to vertical, i.e. $|\mathbf{I} \cdot [0, 0, 1]| < \theta_{\text{vertical}}$
- Requiring a minimum height, i.e. $l_{\text{max}} - l_{\text{min}} > h_{\text{min}}$

A cluster is considered to be a tree if it satisfies these properties. This cluster is then compared to the existing set of confirmed trees T and merged if a match is found. Two clusters are assumed to belong to the same tree if the distance (d) between the intercepts of their major axes with a horizontal plane is less than a threshold (d_{merge}). If there is no overlap along the vertical direction (Z direction), the horizontal plane bisects the

⁴ <https://pointclouds.org/>

⁵ https://github.com/ANYbotics/elevation_mapping

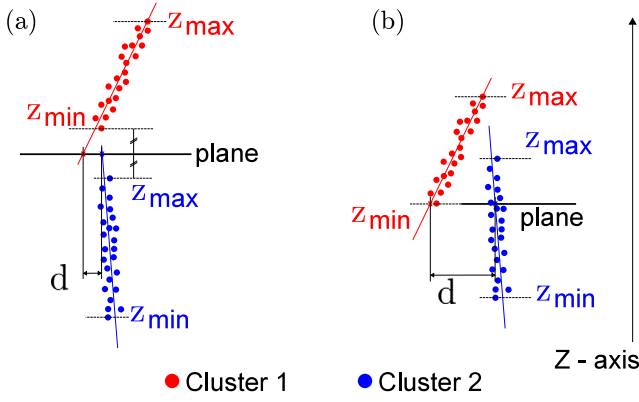


Fig. 7. An illustration of the different cases of *Tree Merging*, showing the distribution of points of two clusters along the Z direction. (a) There is no overlap of points of the clusters along the Z direction. The horizontal plane that bisects the region between two clusters is the plane of separation. (b) There is an overlap of points along the Z direction and the horizontal plane passing through the base of the higher cluster is used for merging. Distance between the intercepts of the major axes with the horizontal plane (d) is checked against a threshold (d_{merge}) to decide if the two clusters should be merged.

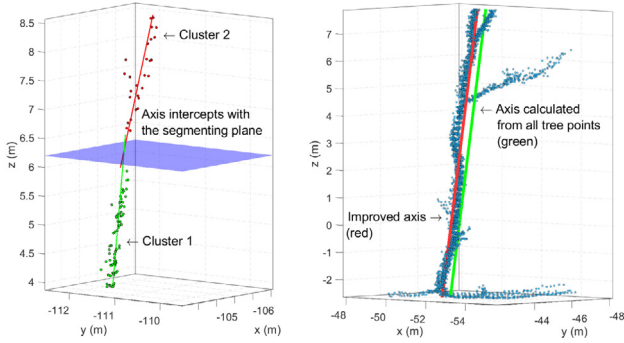


Fig. 8. An example of *Tree Tracking*. **Left:** The two clusters C with their major axis shown and the plane between them. **Right:** Line of best fit on all tree points (green) and on the points excluding branch and ground points (red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

region between the two clusters, see Fig. 7(a). Otherwise it passes through the base of the higher cluster, see Fig. 7(b). Fig. 8 (left) demonstrates an example of the same in 3D.

Alg. 1 summarizes the *Tree Merging* pipeline. The evaluation of a match and the subsequent merge process is assessed until no more matches are found for an input cluster. This is important to ensure all potential clusters belonging to a tree are merged.

5.2.1. Robust estimation of the tree's major axis

As mentioned earlier, the calculation of the tree's major axis, or a cluster's major axis, is done using regression on a subset of the tree's points, to find the line of best fit. Beginning with regression utilizing every point in the cluster gives an initial estimate of the incline (\mathbf{I}) and its intercept.

However, since the objective is to find the axis of the tree's trunk, least squares regression can be affected when the branches, ground or crown points are included in the segmentation. Therefore, points that outside the central 90% of the tree's height are ignored. This removes points close to the ground and the crown of the tree. We compute the distances between all the points from the previously computed axis and ignore points that are farther than 1.3 times the mean distance from the axis. This helps to remove points lying on branches spanning out from the trunk.

The regression is re-calculated with the subset of points. This is iterated multiple times with successively smaller tolerances on the inclusion of points. This result is shown in Fig. 8 (right).

5.3. Base segmentation

As the ground has not been removed prior to segmentation, points that are not necessarily portions of the tree have the potential to be segmented. As a result of this the lowest point in P (minimum z value) may not be an accurate estimate of the tree's base.

The two most recent overlapping elevation maps, from Section 5.1, are converted into a point cloud so that a search over these 'ground points' can be achieved. Neighbouring points within a radius (r_{base}) of the lowest point in P are identified. From this set, the closest point to the tree's axis is found and set as the tree's base \mathbf{b} .

Values of parameters used in the experiments are summarized in Table 1.

6. Estimation of Diameter at Breast Height (DBH)

To estimate the DBH of each individual tree, we employ a procedure similar to that used in the literature for general cylinder fitting [9]. This procedure is broken down into three parts: firstly, segmenting the points at breast height, secondly, projecting these points onto a plane, and lastly, fitting a circle to the projected points.

Employing the method described by Zhou et al. [10] and Heo et al. [8], our system segments the LiDAR points, for recently updated trees in T . Points (S), from the accumulated point cloud (P), that are located within a 10 cm height range centred 1.4 m above the tree's base are segmented. These points are projected onto a plane with normal ($\hat{\mathbf{n}}$) equal to the tree's incline (\mathbf{I}) to ensure the set of points can be modelled closely to a circle. The alternative is to use elliptical fitting for trees angled from the vertical. In practice the sensor only captures data from a single side of the tree, which may cause overfitting when using an elliptical distribution, especially for trees which have irregularly curved surfaces.

A potential problem with this method is that these points are taken from the downsampled point cloud so are likely to be

Algorithm 1: Tree Tracking Algorithm.

```

input: Set of point clouds  $C$ 
output: Set of updated trees  $T$ 
begin
  for Cluster  $C_i \in C$  do
    Calculate the tree descriptor  $t_i = f(C_i)$ 
    if  $t_i$  is classified as being a tree then
      if a first match for  $t_i$  is found in  $T$  ( $T_j$ ) then
        merge cluster into the set of tree points  $P_j = P_j \cup C_i$  and
        re-calculate the tree descriptor  $t_j = f(P_j)$ 
        while a subsequent match for the updated  $t_j$  is found in  $T$ 
        ( $T_{\text{tmp}}$ ) do
          merge the two sets of tree points  $P_j = P_j \cup P_{\text{tmp}}$  and
          re-calculate the tree descriptor  $t_j = f(P_j)$ 
          delete  $T_{\text{tmp}}$  from  $T$ 
        end
      else
        Push back the  $t_i, C_i$  into the set  $T$ 
      end
    end
  end
end

```

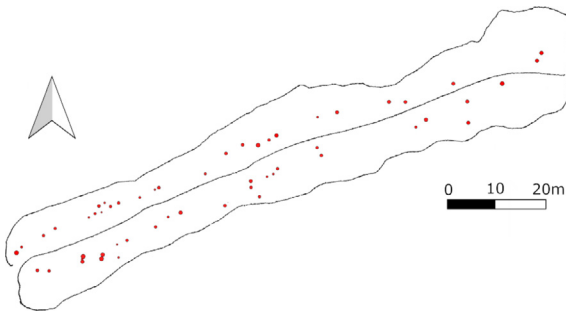



Fig. 9. Visualization of the trees selected for measurement within Wytham Woods, Oxford, as well as the path taken using the handheld device. The location of the chosen transect is 51°46'60.59"N, 001°20'21.47"W.

inaccurate. However, an advantage of this method is that points should cover a greater proportion of the tree's surface if the LiDAR sensor measured it from a set of different angles.

RANSAC circle fitting is then used on the set of resultant 2D points, which is robust to potential outliers from errors in segmentation [26].

7. Experiments and evaluation

The described system was implemented in C++ and the Robot Operating System (ROS) was used for communication between the handheld device and the software modules. We recorded the raw data as rosbag files to the onboard NUC computer kit (Model: NUC8v7PNB) shown in Fig. 2. The computations are performed on a Dell Precision 7550 laptop with Intel Core i7 processor by playing back the recorded rosbag files and processing data in an online fashion. The processing capability of the laptop is equivalent to that of the onboard NUC.

We evaluated our system at two locations. The first one is an ecological forest called Wytham Woods in Oxford, UK. The woods are spread over 400 ha in area and has been used for ecology research for over 80 years. It is a Smithsonian ForestGEO site; which is a collection of forestry plots located world-wide used for collective ecology research. Within this scheme, it has taken part in 3 mass censuses and about 16200 trees have been manually measured. This forest consists of a variety of broadleaf trees throughout the woodland. The second location for our evaluation is a commercial conifer forest in one hour north of Helsinki, Finland spread over several kilometres. These two locations differ in the type and densities of trees as well as different profiles of the underlying terrain.

We also perform a quantitative analysis of our DBH estimations by collecting ground truth data for a particular transect at Wytham Woods. We selected 55 trees along a ~150 m transect in the forest. The selected trees had various diameters, ranging from ~10 cm to ~80 cm, which could be reasonably measured manually at breast height and had clearly visible bases.

The transect of the forest used for the evaluation is shown in Fig. 9. The DBH of every tree was measured using a tape measure at a height of 1.4 m from the ground prior to the LiDAR data collection. Two types of LiDAR sensors were used: The mobile LiDAR device specified earlier and a Leica BLK terrestrial LiDAR⁶ (for comparison). The path taken by the mobile LiDAR can be seen in Figs. 9 and 1, while a terrestrial LiDAR map was made along the central path to generate the ground truth for evaluation of our LiDAR odometry system and to provide a performance baseline.

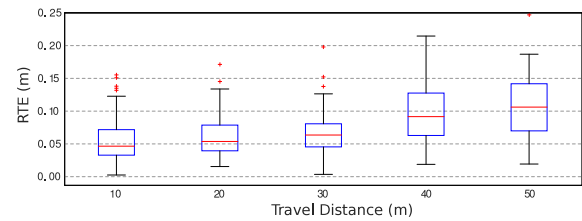


Fig. 10. Relative translation error of the LiDAR odometry at different subtrajectory lengths presented as a series of boxplots. Each box indicates the 25 and 75 percentiles of the estimation errors, the horizontal line through the box the median, and the whiskers are the maximum and minimum excluding outliers.

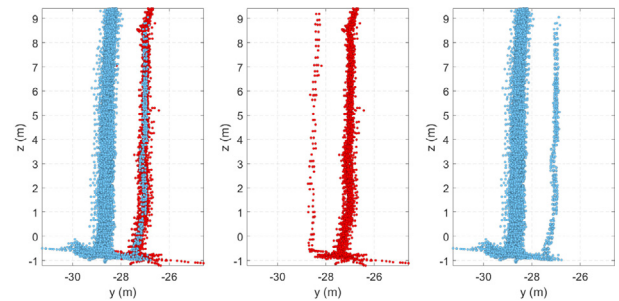


Fig. 11. An example of tree tracking failing for trees in close proximity, the ground points clearly interconnect the two trees. **Left:** The two trees shown entirely. **Centre and Right:** Segmented tree with incorrectly assigned points.

7.1. LiDAR odometry

To evaluate the odometry component of the system, we used the reconstruction created by the Leica BLK as a prior map. We then registered the individual point clouds from the handheld LiDAR against the prior map to generate an accurate 6 degree-of-freedom ground truth trajectory at 10 Hz. The estimated trajectory from the LiDAR odometry system (without access to the Leica map) was then compared against the ground truth trajectory using Relative Translation Error (RTE) at travel distances of between 10 m and 50 m to produce Fig. 10.

We aim to accumulate and process LiDAR scans for individual trees as we walk up and pass them at ranges up to 20 m, thus having a rate of just a few centimetres on this scale is satisfactory.

7.2. Tree tracking

The *Tree Tracking* system performed satisfactorily within the constraints of online performance and the density of foliage and trees on the test transect. Occasionally, trees were clustered together when they should not have been, this is most often due to low hanging interconnecting branches or foliage close to a tree's base. Out of the 55 measured trees 3 of them contained points which belonged to other objects. It was observed that this was also partially due to a transient effect in *Tree Tracking* in which assignment between nearby clusters is uncertain when the tree is first detected. This problem is shown in Fig. 11.

On the other hand, from the 55 measured trees, one failed to be automatically segmented due to its small size, having a DBH ~ 0.11 m. This indicates that our current hardware setup and software configuration limits the performance of the system detecting small trees with DBH below about 10 cm.

7.3. DBH estimation with LiDAR360

To provide a baseline, we used the commercial package LiDAR360, in particular its TLS forestry module, which is designed

⁶ <https://leica-geosystems.com/en-gb/products/laser-scanners/scanners/blk360>

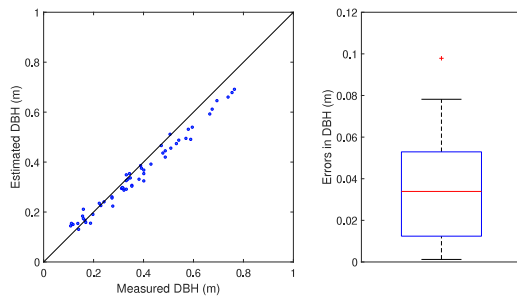


Fig. 12. Estimation of DBH using the LiDAR360 Forestry module (using the mobile LiDAR data). **Left:** Plot of the 55 measured trees vs estimated values as extrapolated from the package. **Right:** Boxplot of the errors between the estimated and measured values, RMSE = 0.056 m.

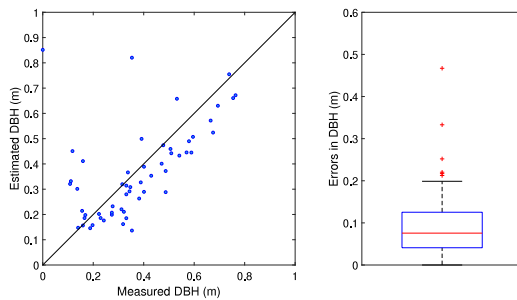


Fig. 13. Estimation of DBH using the designed system. **Left:** Plot of 54 measured trees vs estimated values, one tree was not segmented. **Right:** Boxplot of the errors between the estimated and measured values, RMSE = 0.07 m excluding the outliers and 0.14 m including them.

for tree segmentation and DBH estimation, on both our handheld LiDAR and Leica BLK datasets.

To evaluate the difference between the estimated values of DBH and the manual ground truth measurements, we use the Root Mean Square Error (RMSE) metric.

LiDAR360 successfully identified 53 of the 55 trees in the Leica BLK data. The remaining two, unidentified trees, were manually selected, then DBH estimation was done, across all 55 trees, resulting in an overall RMSE of 0.046 m. The mobile LiDAR point clouds were accumulated, after odometry, and also tested using the same software. Every tree was successfully segmented in the accumulated point cloud and the resulting RMSE was 0.056 m.

The results for LiDAR360's forestry module are shown in Fig. 12. The graphical results for the terrestrial LiDAR was similar to that of the mobile LiDAR so the graph is omitted.

7.4. DBH comparison with proposed system

The results for our system are shown in Fig. 13, these results contain data points for every segmented tree (54/55). Three of the trees were poorly segmented which resulted in outlier measurements in the figure. The RMSE of these estimates, not including the outliers is 0.07 m. The RMSE is slightly higher than that of the commercial software, indicating reasonable performance of the system but some room for improvement. Table 2 summarizes the DBH estimates and segmentation results for the proposed system and LiDAR360 using Leica BLK and accumulated mobile LiDAR data.

7.5. Timing analysis

To demonstrate the key feature of our presented system, that it can run online, we carried out an evaluation of the computation

Table 2

Summary of the DBH results for the systems evaluated.

Algorithm	Detections	RMSE (m)	Mode	Data used
LiDAR360	53/55	0.046	Offline	Leica BLK
LiDAR360	55/55	0.056	Offline	Ouster (accumulated)
Proposed	54/55	0.07/0.14	Online	Ouster (live)

Table 3

Computation times for the *Tree Tracking* modules.

Process	Tree Tracking	Circle Fitting	Euclidean Clustering	Base Segmentation	Total Time
Time (ms)	88	33	66	20	232
std (ms)	±30	±8	±33	±6	±44

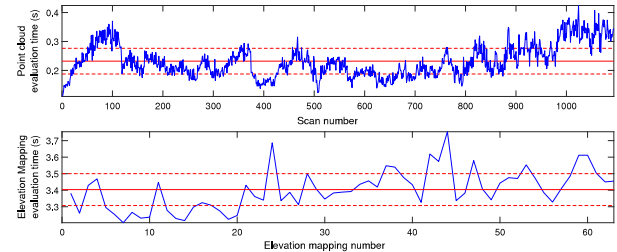


Fig. 14. Evolution of computation time, taken over the 622 s to walk along the Wytham Woods transect, for the two major components. **Top:** The total time for tree segmentation and fitting. **Bottom:** Time for the elevation map filtering process (computed every 8 m of travel distance). The mean and standard deviation are shown by the red solid and dashed lines respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

time for each component. Fig. 14 demonstrates the time required for the two major components of the system – (individual) scan processing in *Tree Tracking* and (accumulated) elevation map filtering. A breakdown of individual modules of the *Tree Tracking* pipeline is given in Table 3.

The *Tree Tracking* system takes about 0.23 s (about 4 Hz) to process an individual scan, which is almost two times faster than the output of our 2 Hz LiDAR odometry.

The elevation map filtering process takes 3.4 s on average. The elevation map is processed every ~8 m and walking speeds of ~1 m/s are expected. Therefore, on average, this process takes less than half the time available. Future work aims to optimize this block to run closer to sensor frame rate.

7.6. Mapping with loop closures

In this experiment, we demonstrate that our mapping system is capable of scaling to larger areas of forest. We demonstrate our approach on two additional large-scale datasets recorded at Wytham woods, Oxford and a commercial forest one hour north of Helsinki, Finland respectively.

Fig. 15 illustrates the mapping results from the Wytham woods dataset. This dataset covers an area of about 0.5 ha traversed in a lawn-mower pattern with a spacing of about 10 m. Several loop-closures have been detected towards the end of each row allowing the SLAM system to correct for drift error. Fig. 15 shows the point cloud reconstruction as well as the extracted trees. We also see the terrain profile of the forest in the cross sectional view.

In Fig. 16, we highlight the effect of the loop closures by plotting the estimated trajectory using LiDAR odometry only (in black), and the SLAM trajectory which uses loop closures (in green). This dataset was recorded the commercial forest in Finland with the length of the total trajectory being 1.3 Km. The trajectory begins and ends at the same position (indicated by

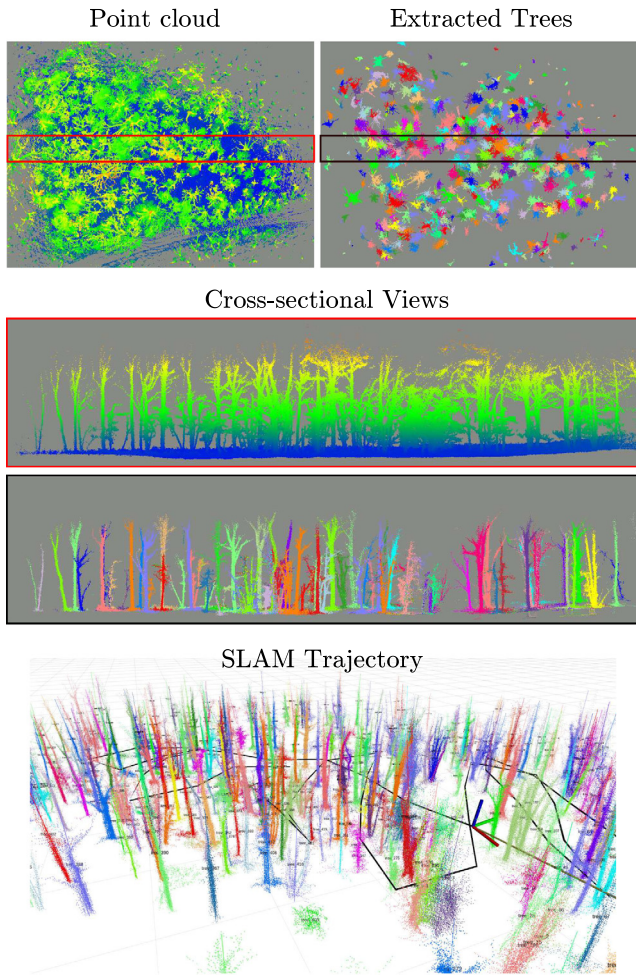


Fig. 15. Mapping results on Wytham Woods dataset surveyed in a lawn-mower pattern. **Top:** Accumulated point cloud reconstruction and the extracted trees. **Middle:** Cross-sectional views of the clouds shown above. **Bottom:** Zoomed-in view of the extracted trees, showing the path of the camera in black.

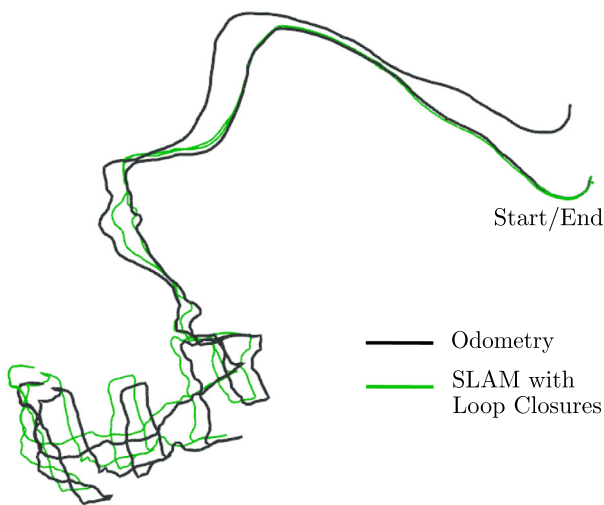


Fig. 16. Effect of loop closures on the estimated trajectory of the device. Odometry estimate (in black) and SLAM pose estimate with loop closures (in green) for a 1.3 Km long trajectory. The odometry estimate drift can be clearly seen at the start/end position. The drift accumulated for this trajectory is around 25 m. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Start/End) with several loops made inside the forest area. The odometry estimate drifted over time, and accumulated an error of 25 m on returning to the start position. In contrast, we see that SLAM trajectory returns to the same location on finish indicating that there is no drift error. This has been possible thanks to the corrections made due to the loop closures.

7.7. Mapping over multiple sessions

In this experiment, we demonstrate the multi-session mapping capability of our system. We record the data in a commercial pine tree forest in Finland over three sessions. In each session, the human operator starts and ends at an operator vehicle and walks down an access road to a plot of trees. The first session inspected a relatively sparse forest with clear separation between trees. The second was somewhat denser and overlapped with the first. Finally the third area was very dense with un-thinned pine sapplings. Fig. 17 illustrates the mapping results of our approach where the area covered in the three sessions is shown. The pose-graphs from individual sessions are visualized in Fig. 18 with a zoomed-in view of a portion of the forest show in Fig. 19. The total travel distance of the combined missions is over 4 kms with each session being roughly equal in length. Note that the three sessions have overlapping areas which is recognized by the loop closure module. An example of such a loop closure is shown in Fig. 20.

This dataset consists of several challenging elements, including long corridor like passages, dense forest patches with limited LiDAR range due to occlusions, transitions in between areas with different densities. Further, as the dataset is captured with a human operator walking on rough surface, the sensor module is often subject to shaking and fast motions. Despite these challenges, our mapping system was successfully able to map each of sessions as well as to merge them together in a post-processing step.

The performance of the *ScanContext* loop proposal mechanism was impressive. Despite being originally developed for automatic localization, it was still successful in our forestry application. We noticed that loops could be found in the forests with managed tree separation. Performance degraded in unmanaged forests where trees were typically smaller samplings and with little clear separation. A further study would be necessary to properly evaluate place recognition algorithms in this type of environment.

7.8. Mapping with different sparsity levels

In this section, we show that our mapping and segmentation pipeline can deal with different sparsity levels of the LiDAR without additional tuning. To test this capability, we recorded datasets using Ouster LiDAR OS0-128 and Ouster LiDAR OS1-64, which are 128 beam and 64 beam LiDARs respectively. As seen in Fig. 21, OS0-128 provides a much denser point cloud in comparison to OS1-64. The two sensors also have different vertical field of views (FOVs) and maximum ranges. The OS0-128 LiDAR has a FOV of 90° with a claimed maximum range of 50 m whereas OS1-64 has a FOV of 45° with a claimed maximum range of 120 m. We found that in practice the maximum range of the OS0-128 was about 25 m.

Experimental results suggest that a narrow FOV and sparse point clouds are sufficient for our application of estimating the DBH and motion from odometry. Our algorithms have a voxel filtering step at the beginning which ensures that the subsequent steps are robust to sensor changes.

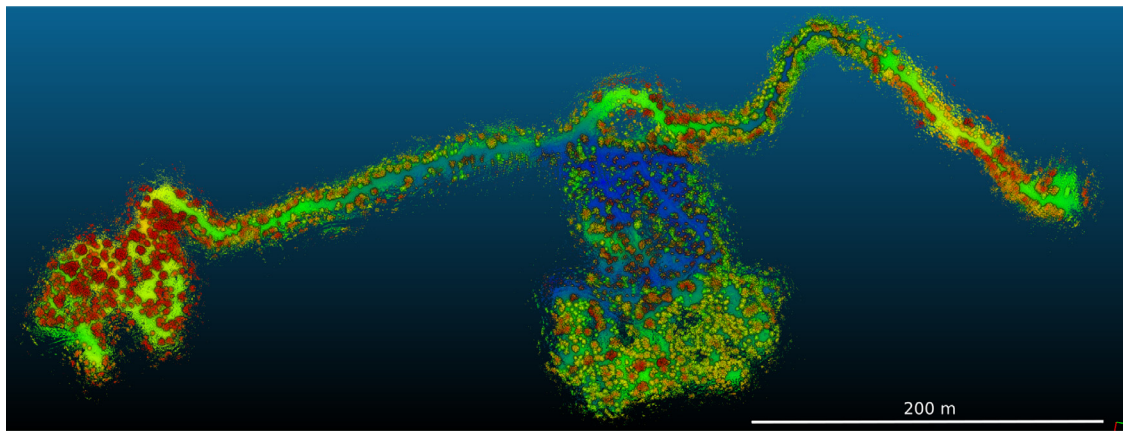


Fig. 17. Mapping results on a large-scale dataset from a commercial pine tree forest in Finland. Figure shows the point cloud reconstruction of the forest obtained by merging all the local maps after joint pose-graph optimization.

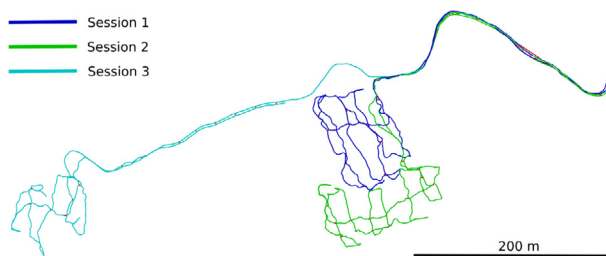


Fig. 18. Pose-graphs from individual sessions visualized in different colours. Loop closures both with-in the same session, and in-between different sessions is shown in red. The total travel distance of all the sessions is 4 km. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

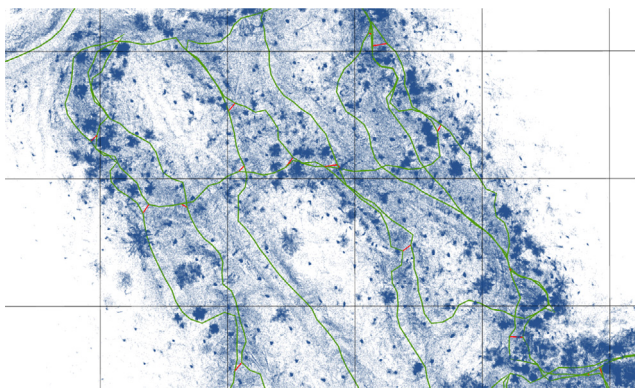


Fig. 19. Zoomed-in view of a portion of the forest shown in Fig. 17 showing dense clusters corresponding to individual trees. The trajectory traversed (green) and the detected loop closures (red) are also shown. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

8. Conclusion and future work

This paper presented a mobile LiDAR scanning system for mapping large forest areas, automatic segmentation of trees and the online estimation of their DBH. The architecture of a tree tracking functional block was introduced to facilitate the on-line operation of this system. We tested the system's performance using data from a natural forest, Wytham Woods, as well as a commercial forest one hour north of Helsinki, Finland to

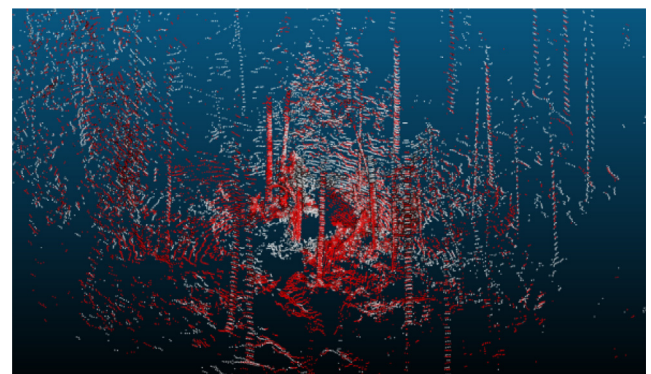


Fig. 20. Example of a loop closure detected between Session 1 and Session 2. Top: The corresponding point clouds overlaid on each other after ICP registration. Point cloud from Session 1 is shown in red whereas white points are from Session 2. Bottom: Corresponding camera images of the loop closure candidates are shown. Note that the loop closures are detected only from the LiDAR data, and the images are shown for illustrative purposes only.

prove that acceptable results can be achieved online which compares favourably to the performance achieved by a commercial software package conducted in post-processing.

In this work, we focused on extracting tree trunks and computing the DBH parameter. In the future, we would like to extend the breadth of metrics used to characterize trees. In particular, we would like to quantify information on the branching of the tree, as well as properties related to the tree crown. We plan to integrate information from the cameras as well as extract semantics to estimate a richer set of quantitative and qualitative metrics for the forest inventory.

One of the limitations identified in the experimental analysis was the downgraded performance on small trees. We like to address this issue by considering an adaptive segmentation algorithm such as in [14] to improve performance on the smaller trees.

Finally, by fully using the screen we intend to provide feedback to the operator to better direct their actions during operation.

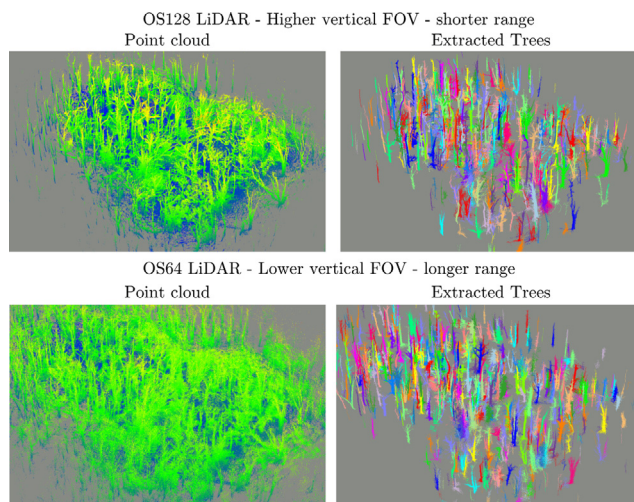


Fig. 21. Reconstruction and tree extraction results using LiDAR scanners with different sparsity levels. Top: 128 beam LiDAR resulting in a denser point cloud reconstruction. Bottom: 64 beam LiDAR resulting in a sparser point cloud reconstruction. The proposed tree extraction approach is robust to different sparsity levels.

CRediT authorship contribution statement

Alexander Proudman: Conceptualization, Methodology, Software, Writing. **Milad Ramezani:** Conceptualization, Methodology, Software, Writing. **Sundara Tejaswi Digumarti:** Conceptualization, Methodology, Software, Writing. **Nived Chebrolu:** Conceptualization, Methodology, Software, Writing. **Maurice Fallon:** Conceptualization, Methodology, Software, Writing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Maurice Fallon reports financial support was provided by ORCA Robotics ub. Maurice Fallon reports financial support was provided by Royal Society University Research Fellowship. Sundara Tejaswi Digumarti reports a relationship with The University of Sydney that includes: employment. Milad Ramezani reports a relationship with Commonwealth Scientific and Industrial Research Organization that includes: employment. Nived Chebrolu reports a relationship with University of Bonn that includes: employment.

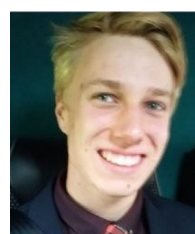
Acknowledgements

This research is supported by the UKRI/EPSC ORCA Robotics Hub, UK (EP/R026173/1). M. Fallon is supported by a Royal Society University Research Fellowship, UK. The authors would like to Taneli Vuornos (Silvere) for the collection of the data used in Section 7.7.

References

- [1] P. Raunonen, M. Kaasalainen, M. Åkerblom, S. Kaasalainen, H. Kaartinen, M. Vastaranta, M. Holopainen, M. Disney, P. Lewis, Fast automatic precision tree models from terrestrial laser scanner data, *Remote Sens.* 5 (2) (2013) 491–520.
- [2] K. Calders, G. Newnham, A. Burt, S. Murphy, P. Raunonen, M. Herold, D. Culvenor, V. Avitabile, M. Disney, J. Armston, et al., Nondestructive estimates of above-ground biomass using terrestrial laser scanning, *Methods Ecol. Evol.* 6 (2) (2015) 198–208.
- [3] J. Gonzalez de Tanago, A. Lau, H. Bartholomeus, M. Herold, V. Avitabile, P. Raunonen, C. Martius, R.C. Goodman, M. Disney, S. Manuri, et al., Estimation of above-ground biomass of large tropical trees with terrestrial LiDAR, *Methods Ecol. Evol.* 9 (2) (2018) 223–234.

- [4] P. Raunonen, M. Åkerblom, M. Kaasalainen, E. Casella, K. Calders, S. Murphy, Massive-scale tree modelling from tls data, in: *ISPRS Annals of the Phot., Remote Sensing and Spatial Inf. Sciences*, 2, 2015.
- [5] J. Trochta, M. Krůček, T. Vrška, K. Král, 3D forest: An application for descriptions of three-dimensional forest structures using terrestrial LiDAR, *PLoS One* 12 (5) (2017) e0176871, <http://dx.doi.org/10.1371/journal.pone.0176871>.
- [6] L. Zhong, L. Cheng, H. Xu, Y. Wu, Y. Chen, M. Li, Segmentation of individual trees from TLS and MLS data, *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* 10 (2) (2016) 774–787.
- [7] A. Burt, M. Disney, K. Calders, Extracting individual trees from lidar point clouds using treeseg, *Methods Ecol. Evol.* 10 (3) (2019) 438–445.
- [8] H.K. Heo, D.K. Lee, J.H. Park, J.H. Thorne, Estimating the heights and diameters at breast height of trees in an urban park and along a street using mobile LiDAR, *Landsc. Ecol. Eng.* 15 (3) (2019) 253–263.
- [9] V. Pratt, Direct least-squares fitting of algebraic surfaces, *SIGGRAPH* 21 (4) (1987) 145–152.
- [10] S. Zhou, F. Kang, W. Li, J. Kan, Y. Zheng, G. He, Extracting diameter at breast height with a handheld mobile LiDAR system in an outdoor environment, *Sensors* 19 (14) (2019) 3212.
- [11] A.J. Trevor, S. Gedikli, R.B. Rusu, H.I. Christensen, Efficient organized point cloud segmentation with connected components, in: *Semantic Perception Mapping and Exploration, SPME*, 2013.
- [12] F. Westling, D.J. Underwood, D.M. Bryson, Graph-based methods for analyzing orchard tree structure using noisy point cloud data, 2020, arXiv preprint [arXiv:2009.13727](https://arxiv.org/abs/2009.13727).
- [13] A.K. Aijazi, P. Checchin, L. Malaterre, L. Trassoudaine, Automatic detection and parameter estimation of trees for forest inventory applications using 3D terrestrial LiDAR, *Remote Sens.* 9 (9) (2017) URL: <https://www.mdpi.com/2072-4292/9/9/946>.
- [14] S.T. Digumarti, J. Nieto, C. Cadena, R. Siegwart, P. Beardsley, Automatic segmentation of tree structure from point cloud data, *IEEE Robot. Autom. Lett.* 3 (4) (2018) 3043–3050, <http://dx.doi.org/10.1109/LRA.2018.2849499>.
- [15] S. Tejaswi Digumarti, L.M. Schmid, G.M. Rizzi, J. Nieto, R. Siegwart, P. Beardsley, C. Cadena, An approach for semantic segmentation of tree-like vegetation, in: *IEEE Intl. Conf. on Robotics and Automation, ICRA*, 2019, pp. 1801–1807, <http://dx.doi.org/10.1109/ICRA.2019.8793576>.
- [16] L. Windrim, M. Bryson, Detection, segmentation, and model fitting of individual tree stems from airborne laser scanning of forests using deep learning, *Remote Sens.* 12 (9) (2020) 1469.
- [17] C.R. Qi, H. Su, K. Mo, L.J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, in: *Proc. of the IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2017, pp. 652–660.
- [18] S. Krisanski, M.S. Taskhiri, S. Gonzalez Aracil, D. Herries, P. Turner, Sensor agnostic semantic segmentation of structurally diverse and complex forest point clouds using deep learning, *Remote Sens.* 13 (8) (2021) 1413.
- [19] D. Wisth, M. Camurri, S. Das, M. Fallon, Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry, *IEEE Robot. Autom. Lett.* 6 (2) (2021) 1004–1011.
- [20] F. Pomerleau, Applied Registration for Robotics: Methodology and Tools for ICP-Like Algorithms (Ph.D. thesis), ETH Zurich, 2013.
- [21] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, R. Siegwart, Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes, in: *2016 IEEE International Conference on Robotics and Automation, ICRA*, 2016, pp. 4304–4311, <http://dx.doi.org/10.1109/ICRA.2016.7487628>.
- [22] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, F. Dellaert, iSAM2: Incremental smoothing and mapping using the Bayes tree, *Intl. J. Robot. Res.* 31 (2) (2012) 216–235.
- [23] G. Kim, A. Kim, Scan context: Egocentric spatial descriptor for place recognition within 3D point cloud map, in: *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS*, 2018, pp. 4802–4809, <http://dx.doi.org/10.1109/IROS.2018.8593953>.
- [24] P. Fankhauser, M. Hutter, A universal grid map library: Implementation and use case for rough terrain navigation, in: *Robot Operating System, ROS*, Springer, 2016, pp. 99–120.
- [25] P. Fankhauser, M. Bloesch, M. Hutter, Probabilistic terrain mapping for mobile robots with uncertain localization, *IEEE Robot. Autom. Lett.* 3 (4) (2018) 3019–3026.
- [26] R. Schnabel, R. Wahl, R. Klein, Efficient RANSAC for point-cloud shape detection, *Comput. Graph. Forum* 26 (2) (2007) 214–226, <http://dx.doi.org/10.1111/j.1467-8659.2007.01016.x>.



Alexander Proudman completed his M.Eng from the University of Oxford. He is also associated with Wadham College, Oxford. He worked with the Dynamic Robot Systems Group at the Oxford Robotics Institute for his masters thesis. His research interests are in developing robust SLAM systems and real-time semantic interpretation from sensor data.



Milad is a Postdoctoral Fellow in the Embodied AI research team of the Robotics and Autonomous Systems Group at CSIRO Data61. He studied computer vision and photogrammetry at K.N. Toosi University of Technology, Iran. He carried out his Ph.D. research in localization via GPS/IMU integration and visual-inertial odometry at the University of Melbourne. He joined the DRS Group in Oxford Robotics Institute as a postdoc under Assoc. Prof. Maurice Fallon (2018–2021). His research focused on robust and accurate localization and mapping systems deployable on various robotics

platforms.



Tejaswi is a post-doc at the DRS Group at the ORI. He received his B.Tech in Electrical Engineering from IIT Jodhpur, India and M.Sc. and Ph.D. in Robotics from ETH Zurich, Switzerland. From 2019–2021 he was a post-doc at ACFR, Australia, where he worked on semantic segmentation of large scale point clouds of forests and learning-based approaches to interpret novel camera systems. His research focuses on using semantics and geometry to improve navigation, scene understanding and 3D reconstruction.



Nived Chebrolu is a post-doctoral research associate at the Oxford Robotics Institute, University of Oxford. His research interests are in developing robust localization and mapping techniques for field robotics applications. He obtained his Ph.D. from the University of Bonn in 2021, where he developed long-term registration techniques for SLAM systems which were deployed on agricultural fields within the Flourish H2020 EU project.



Maurice Fallon received his B.Eng (electrical engineering) from University College Dublin, Ireland and his Ph.D. (acoustic source tracking) from University of Cambridge, UK. From 2008–2012 he was a postdoc and research scientist in MIT Marine Robotics Group working on SLAM. Later, he was the perception lead of MIT's team in the DARPA Robotics Challenge. From 2017 he has been a Royal Society University Research Fellow and Associate Professor at University of Oxford, UK. He leads the Dynamic Robot Systems Group in Oxford Robotics Institute. His research focuses on probabilistic methods for localization, mapping, dynamic motion planning, and multi-sensor fusion.