Journal of Field Robotics

WILEY

RESEARCH ARTICLE OPEN ACCESS

Towards a Semiautonomous Young Spruce Forest Late Cleaning

Issouf Ouattara[1] | Jyri Hallikas[1] | Heikki Hyyti[2] | Arto Visala[1]

[1]Department of Electrical Engineering and Automation, Aalto University, Espoo, Uusimaa, Finland | [2]Department of Remote Sensing and Photogrammetry, Finnish Geospatial Research Institute (FGI), National Land Survey of Finland, Masala, Kirkkonummi, Finland

Correspondence: Issouf Ouattara (issouf.ouattara@aalto.fi)

Received: 7 June 2024 | **Revised:** 17 January 2025 | **Accepted:** 9 February 2025

Funding: We gratefully acknowledge the Strategic Research Council of Academy of Finland that funded this research through Grant/Award Numbers 293389 and 314312. In addition, the research is funded by the Technology Industries of Finland Centennial Foundation and Jane and Aatos Erkko Foundation in the Future Makers research program.

Keywords: forestry | human–machine interface | point cloud | semiautonomous | silviculture | unmanned aerial vehicle

ABSTRACT

Cleaning a seedling spruce stand is an important silvicultural task required to help the young spruce trees thrive. It is usually manual work with a clearing saw. Mechanized solutions have been proposed, but they have not worked out well, since the driver has challenges in seeing the seedling trees that should be left growing. Instead, we demonstrate that a semiautonomous forest machine could do the cleaning operation using a suitable map of the environment. We propose using a low-cost unmanned aerial vehicle (UAV) to map the area beforehand, and a real-time perception system onboard the forest machine performing the semiautonomous late cleaning of a seedling spruce stand with the help of a driver using a dedicated human–machine interface (HMI). In the proposed solution, overlapping color images are collected by a UAV to build a map that integrates both color and depth information. This map is used to detect the young spruce trees, among other species. We also propose a graph-based point-cloud matching method, which can precisely locate the forest machine on the map using forest machine-mounted sensors. An HMI is developed to improve the situational awareness of the operator in the forest machine cabin during the cleaning process. The proposed integrated solution has been demonstrated in a real seedling spruce forest.

1 | Introduction

Finland's forest industry, one of its oldest sectors, is crucial to its economy and accounts for about 20% of export revenue (Ministry of Agriculture and Forestry of Finland 2020). More than 75% of the country's land is covered by forests. According to Heinonen et al. (2017), the industry may need an additional 10–30 million m^3 of wood annually in the coming years. Given the importance of forestry to the Finnish economy and the increasing demands for production and profitability, new forest management approaches are required.

Efficient forest stand management, particularly tending operations, such as cleaning and thinning, is essential for promoting healthy forest growth and maximizing the value of harvested wood (Huuskonen et al. 2020). However, cleaning operations are often delayed (Korhonen and Ihalainen 2017) due to their labor-intensive nature and a shortage of skilled workers (Vestlund and Hellström 2006). This leads to increased costs that can double with delays of more than 2 years and to a decrease in timber quality (Uotila et al. 2014).

Manual methods with clearing saws are the most common practice (Hamberg et al. 2022; Uotila et al. 2020), but they are

laborious and inefficient, especially in late cleaning scenarios where dense foliage makes tree identification difficult. Mechanized solutions face difficulties in detecting the desired trees behind dense vegetation, often resulting in unintended damage to valuable spruces.

Robotics offers a promising avenue to overcome these challenges in forestry operations. L. F. Oliveira et al. (2021) discuss some general uses of robotics in forestry, such as environmental preservation, monitoring, wildfire firefighting, inventory operations, planting, pruning, and harvesting. In the review, the navigation of the robotics system is considered a challenge that needs to be solved. Vestlund and Hellström (2006) proposed general requirements for robotic systems performing cleaning operations in young forest stands, highlighting that the most challenging aspects are enabling robots to find, select, and handle trees throughout the forest environment and to move safely within it. In the context of the late cleaning of a seedling stand, the challenges facing a semiautonomous system can be summarized in three main points:

- the localization of the forest machine;
- the accurate detection of young spruces among leaf trees;
- the presentation of relevant information to the operator to safely move the machine and avoid damaging the young trees.

To address these challenges, we propose a semiautonomous robotized cleaning approach. By semiautonomous forest cleaning machine, we mean a forest machine with a cleaning tool at the tip of an instrumented forest crane, a human operator driving the machine and manipulating the crane, and several sensors to help the human operator perform his task safely and more efficiently; the pose of the cleaning tool can be measured as done in Hyyti (2023), which is based on the work in Hyyti et al. (2018) and Kalmari et al. (2013). The proposed solution comprises two steps. First, the forest area is mapped with a low-cost unmanned aerial vehicle (UAV) equipped with an RGB camera. This map is used offline to detect and locate the spruce trees, creating a map of the spruce trees. Second, a human–machine interface (HMI) utilizes this map to localize the machine platform in real-time by using the machine-mounted light detection and ranging (LiDAR) sensor and Global Navigation Satellite System (GNSS) measurements. A third step, which is not considered in this study, but is to be demonstrated in the future, is the automatic motion control of the crane in a cleaning scenario. The crane's motion control algorithms have been developed by Kalmari et al. (2014).

The contributions of this study are as follows:

- To the authors' best knowledge, this is the first study to consider combining data from a low-cost drone and forest–machine–mounted sensors in the context of seedling spruce stand late cleaning. Furthermore, this is the first study that uses an HMI with heterogeneous data in this context.
- Expanding on our previous work in spruce detection (Ouattara et al. 2020), we propose a simplified single-step spruce detection scheme, in which we augment the RGB image with the elevation model of the terrain, creating RGB-D images with four channels that serve as input for a deep segmentation model.
- Starting from a graph-based correspondence construction, we evaluate several alternative versions of point-cloud matching methods to determine the pose of the forest machine within the map created using the UAV data. The proposed methods do not need an (accurate) initial guess, making them more robust than the traditional point-cloud matching methods, such as the iterative closest point (ICP) method.
- We register point clouds from different sensor modalities and show the feasibility of semiautonomous forest cleaning operations through the use of an augmented reality (AR) HMI.

The rest of this paper is organized as follows: the related work is discussed in Section 2, the methods are proposed and explained in Section 3, the experimental setup is shown in Section 4, the results are demonstrated and discussed in Section 5, and conclusions are drawn in Section 6.

2 | Related Work

2.1 | Mapping and Individual Tree Detection

In a late cleaning situation, broad-leaved trees grow faster than the spruces, making it difficult for a machine operator to see the spruce trees occluded by competing vegetation. In this situation, the use of a camera and LiDAR on the machine to detect and locate the spruce trees, as was done by Vihlman et al. (2015), has a limitation related to the maximum height at which the sensor can be placed on the machine. As an alternative solution, a low-cost UAV can be used to map the forest area and detect the spruces before the cleaning phase. Since the data collected by the UAV are georeferenced, the detection result can be used to help the operator to have a better awareness of the location of the spruces and avoid damaging them in the cleaning process.

In the literature, tree species detection using UAV systems equipped with sensors such as RGB cameras, multispectral cameras, and LiDAR has been extensively studied (C. Zhang et al. 2015; Katoh et al. 2017; Nevalainen et al. 2017; Xu et al. 2020; Pearse et al. 2020; Raviraj et al. 2024). In this study, we focus on the use of UAVs equipped with RGB cameras, as they provide a more cost-effective alternative to multispectral and LiDAR sensors. In addition, RGB data has been shown to provide better accuracy in the detection of trees, particularly small ones, as demonstrated in the study by Thiel and Schmullius (2016). Several studies have investigated tree species detection using UAV-acquired RGB images and their derivatives, such as an orthomosaic, a digital surface model (DSM), and a digital terrain model (DTM). In the study by Huang et al. (2018), orthomosaic was used to detect individual trees using a method based on watershed segmentation (Meyer and Beucher 1990), without identifying tree species. The studies by Pearse et al. (2020) and Raviraj et al. (2024) used orthomosaics directly to detect individual tree crowns using the faster region-based convolutional neural network (R-CNN) detection model (Ren et al. 2017). A direct detection method does not provide a clear segmentation of the pixels belonging to seedling trees, which is

important in our case when presenting the detected trees in the user interface. Moreover, the elevation data, which are not used in these studies, could be beneficial for effective tree detection. In the study by Natesan et al. (2019), a DSM was used to segment the individual crowns of trees. The corresponding RGB crown is provided as input to a convolutional neural network (CNN) to infer the tree species. The approach is heavily dependent on the accuracy of the segmentation step. If a segmented tree image contains both a seedling tree and a leaf tree, the direct classification method can fail to classify the seedling tree. In our previous study (Ouattara et al. 2020), a canopy height model (CHM) was derived as the algebraic subtraction between the DSM and the DTM. The CHM is used in an individual tree identification (ITD) step to segment the tree crown. Given that the segmented crown image might contain more than one tree due to segmentation error, a deep learning detection method is used instead of a direct classifier. We will refer to this previous study as *ITD + detector* in the result section. In our work here, the ITD step is omitted. Instead, the CHM data are combined with the orthomosaic to form a four-channel image, which is directly used as input to a deep neural network model performing the segmentation. This approach simplifies the tree detection process without impacting negatively the accuracy.

2.2 | Localization of the Forest Machine

Modern forest machines are equipped with GNSS sensors for positioning. However, their accuracy is limited in forests due to occlusion, with positioning errors reaching more than 9 m (Kaartinen et al. 2015). LiDAR sensors, commonly used in robotics, offer an alternative solution as investigated in recent work in Simultaneous Localization and Mapping (SLAM) research for forest machinery (Faitli et al. 2023, 2024). The localization of vehicles based on point-cloud data from LiDAR sensors can be reduced to a point-cloud registration problem, which involves finding a spatial transformation to align two point clouds. The registration methods in the literature can be divided into local registration and global registration. Local registration methods rely on a good initial pose estimate to align the point clouds using heuristics based on the nearest neighbor search (Besl and McKay 1992; Holz et al. 2015; J. Zhang et al. 2022). Global registration algorithms commonly used for (re)localization and loop closing rely on feature descriptor or geometric primitives correspondences and do not require an initial guess (Li et al. 2020; Yang et al. 2021; Yuan et al. 2023; X. Zhang et al. 2023; Yin et al. 2023; Qiao et al. 2024). The studies by Li et al. (2020) and Yuan et al. (2023) use triangle feature matching. The studies by Yang et al. (2021), Yuan et al. (2023), X. Zhang et al. (2023), Yin et al. (2023), and Qiao et al. (2024) use the notion of maximal cliques to obtain a geometrically consistent set of correspondences. In all of these studies, the point clouds to be matched are generated from the same type of sensor, a LiDAR. In a point cloud generated by photogrammetry methods, the surfaces and peaks of objects tend to be averaged out, unlike in a point cloud from a LiDAR. This introduces two issues: (1) the same triangle can have different side lengths and surfaces; (2) the maximal clique used to establish correspondence between a point cloud generated by a LiDAR and a point cloud generated by a photogrammetry method can still contain outliers. In this study, we use additional measures such as tree height and intertree height consistency check, and we introduce an adaptive thresholding on distance check to establish correspondences.

2.3 | Human–Machine Interface

The goal of detecting the trees of interest is to present them to the human operator through an augmented reality (AR) HMI. Information of interest can be presented to the machine operator through three means. The first two approaches use a head-mounted display (HMD) or a head-up display (Santana-Fernández et al. 2010; Palonen et al. 2017; Pizzagalli et al. 2022). Palonen et al. (2017) demonstrated the use of AR in the cabin of a forest machine in a real-use scenario. The study proposes an HMD to augment the user's perception with additional data such as the three-dimensional (3D) point cloud of the environment and a simple wireframe model of the machine's crane and tool. Unfortunately, such a setup does not show the overall situational awareness of the machine in the work area, and the display data might occlude parts of the machine. This type of system tends to be more complex and increases the mental workload of the operator (Sitompul and Wallmyr 2019).

A third and simpler method is through the use of a flat screen (Fang et al. 2016; Fang and Cho 2017). The study by Fang et al. (2016) develops a proactive real-time safety assistance framework for mobile crane lifting operations. It implemented a user interface based on the Unity 3D game engine in which crane movement, a simplified reconstructed version of the working environment, and collision warning information are available to the machine operator. In Fang and Cho (2017), it was observed that the proposed AR framework could effectively improve the situational awareness of the machine operator. An advantage of a flat screen is that it is relatively easy to implement and gives the operator the choice to look at it only when necessary.

Our study uses the concept of a flat screen for its ease of implementation and rapid testing. The information presented to the user through the HMI includes the 3D point cloud of the forest environment, the location of young spruces, the location of the cleaning tool, and the location of the whole machine. This information is important for removing unwanted trees, avoiding damaging the young spruce trees, and keeping track of the forest machine situation. Although not done in this study, the measured location of the spruces could also be used to plan the overall cleaning process.

3 | Methods

In this section, we first show the map-building process using the images collected by a UAV. Second, we show the preprocessing of the point cloud derived from the UAV data. We then show how the point cloud collected by the LiDAR on the forest machine is constructed and fitted to the UAV-based point cloud to reveal the position of the forest machine. Finally,

we show the structure and capabilities of the developed HMI. The overall pipeline of the process is shown in Figure 1.

3.1 | Mapping and Young Spruces Detection

We first present the mapping process using the UAV platform and the relevant data obtained from the collected images. We then present the proposed single-step spruce detection and segmentation method. These processes are summarized in Figure 2 and are explained in detail next.

Mapping: The mapping of the forest environment is performed before the forest machine is deployed in the forest stand. The mapping process includes collecting overlapping images (70% overlap) using a UAV equipped with an RGB camera. The collected images were used to produce an RGB orthomosaic, a DSM, a DTM, and a point-cloud representation of the environment

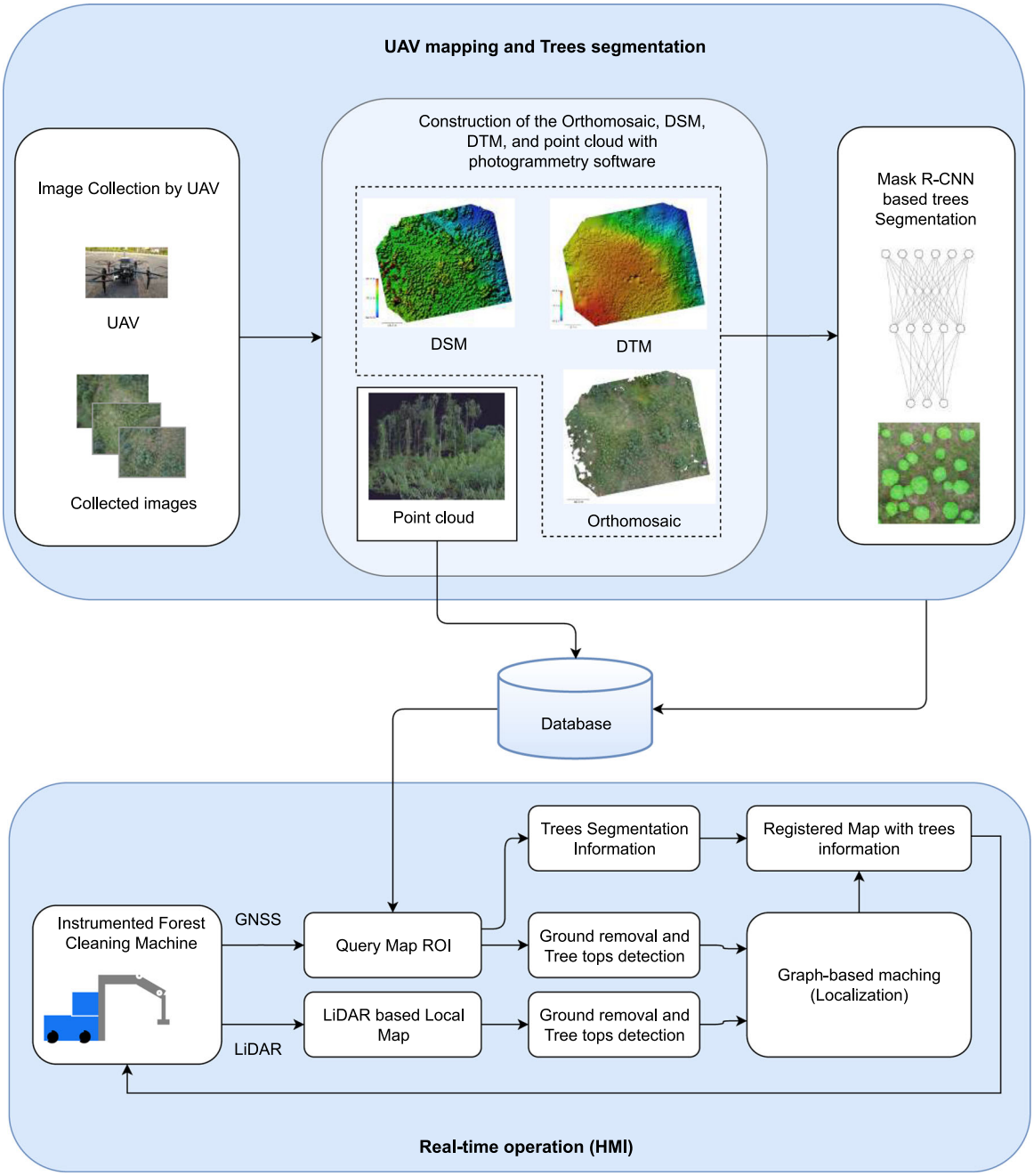

FIGURE 1 | An overview of the proposed method, which includes collecting overlapping images of the environment with a UAV system, the construction of the point cloud, the orthomosaic, the DTM, and the DSM maps with a photogrammetry software, and the segmentation of the trees based on the orthomosaic and a canopy height model (CHM). The CHM is an algebraic subtraction between the DSM and the DTM. The results are saved in the database and later used during forest cleaning operation to localize the forest machine, its tool, and the young trees to be saved during the cleaning process. DSM, digital surface model; DTM, digital terrain model; GNSS, Global Navigation Satellite System; HMI, human–machine interface; LiDAR, light detection and ranging; R-CNN, region-based convolutional neural network; ROI, region of interest; UAV, unmanned aerial vehicle. [Color figure can be viewed at wileyonlinelibrary.com]

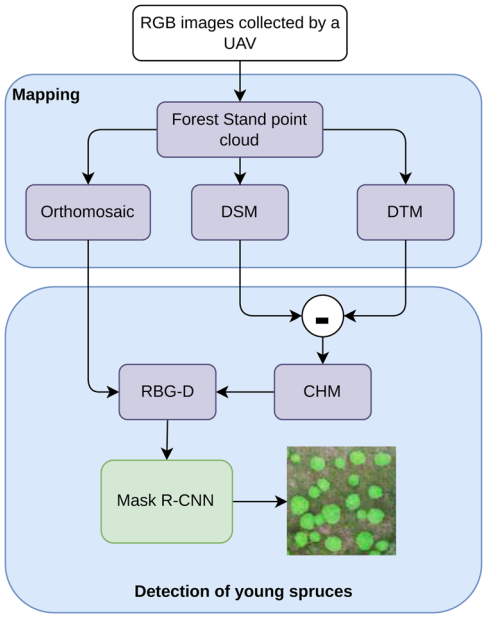

FIGURE 2 | An overview of the proposed mapping and seedling spruce segmentation process. The CHM is added as a fourth layer to the orthomosaic to form an RBG-D image. This resulting image is partitioned into patches of size 520 × 520 pixels, which are used as inputs to the Mask R-CNN network. CHM, canopy height model; DSM, digital surface model; DTM, digital terrain model; R-CNN, region-based convolutional neural network; UAV, unmanned aerial vehicle. [Color figure can be viewed at wileyonlinelibrary.com]

using Agisoft Metashape Professional v1.5.5 software (Agisoft and St. Petersburg 2019) which is used in several research involving photogrammetry-based mapping (Giannetti et al. 2018; Iglhaut et al. 2019; Ouattara et al. 2020; R. A. Oliveira et al. 2020). The study by Jiménez-Jiménez et al. (2021) provides an overview of other software used in photogrammetry mapping. A typical photogrammetry mapping pipeline includes feature detection and matching followed by a structure-from-motion (SFM) algorithm to produce a sparse point cloud. These steps are followed by a multiview stereo approach to produce a dense point cloud (Iglhaut et al. 2019). The DSM is generated in Agisoft Metashape by directly using the dense point cloud. It represents the elevation of the Earth's surface, including all natural and man-made features, such as vegetation and buildings. The dense point cloud is then segmented into two sets: a ground point cloud and a non-ground point cloud. The ground point cloud is used to generate the DTM, which represents the elevation of the bare ground. The DTM and the DSM have a resolution of 5.4 cm/pixel, while the RGB orthomosaic has a resolution of 2.5 cm/pixel, in this study. All the obtained data are expressed in the WGS84/UTM zone 35° N (EPSG:32635) reference projection frame. Apart from the photogrammetry process, which is done using Agisoft Metashape software, all the remaining steps in this study are implemented using either Python or C++ code.

Detection of young spruces: After obtaining the RGB orthomosaic, the DTM, and the DSM, the spruce detection is implemented. A CHM is created by subtracting the DTM from the DSM. This model represents the height of objects above ground. Assuming that the CHM is valuable in distinguishing between the trees and the rest of the environment (grass and other undergrowth), we combined it with the RGB orthomosaic to form a four-channel image of the study area. This can be considered as an RGB-D model of the environment, where "D" stands for *depth*.

The RGB orthomosaic and elevation maps are very large. In general, the RGB orthophoto has a higher resolution (this is desired for tree detection) than the elevation model, which does not need to have a centimeter-level resolution in practice. These large maps are partitioned into smaller subimages. First, the RGB orthomosaic is partitioned into image patches, each with dimensions of 520×520 pixels. Using the geospatial transformation information embedded in the orthomosaic and the elevation data, we utilize the Rasterio Python API to determine the patches in the CHM that correspond to the patches in the RGB orthomosaic (Gillies et al. 2013). We concatenate each RGB patch with its corresponding CHM patch to form RGB-D patches, which are used to train a deep neural network to segment individual spruce trees. We also train the same model using only the RGB patches to confirm the assumption that the elevation information is important for tree detection. The training labels were obtained by manually annotating the RGB patches using Label Studio, an open-source labeling tool (Tkachenko et al. 2020).

The Mask R-CNN object instance segmentation framework is used to detect the seedling spruce trees (He et al. 2017). Mask R-CNN is built on Faster R-CNN, and it has three outputs: the object class, its bounding box, and the object mask. The Mask R-CNN framework is flexible, allowing for different choices of feature extraction backbone. In this study, the Inception V3 architecture is used as the backbone (Szegedy et al. 2016). We removed the final two inception modules along with the convolution layers that follow them to keep the number of output feature maps to 1280 instead of 2048. Due to the limited amount of training data available, a pretrained model with ImageNet data set (Deng et al. 2009; Paszke et al. 2019) is used and retrained using our data. The original pretrained model accepts images with three channels only. To adapt the model to our RGB-D images, the first convolution layer has been modified to accept four channels. The weights of the newly introduced channel were initialized by replicating the weights from the first channel of the pretrained model. An overview of the Mask R-CNN model's structure is shown in Figure 3. The Region Proposal Network and the Region of Interest Align follow the Inception V3 backbone. For more details about the Mask R-CNN, see the work by He et al. (2017) since the layers at the end of our network are similar to their work. The output of the model is a mask around each spruce tree.

The segmented mask for each tree is used to mark the corresponding tree in the point cloud. This is easily done because the point cloud and the orthomosaic are generated in the same reference system (WGS84/UTM zone 35° N). In the HMI, an approximate radius of the segmented mask is computed and used to draw a cone around the detected tree in 3D. Figure 4 shows an example where the detected spruce information is used in the point cloud.

3.2 | Processing of Point Clouds

The HMI developed in this study needs to handle several data sources: the inertial measurement unit (IMU) and GNSS data, the *forest-stand point cloud* obtained from the drone data, and

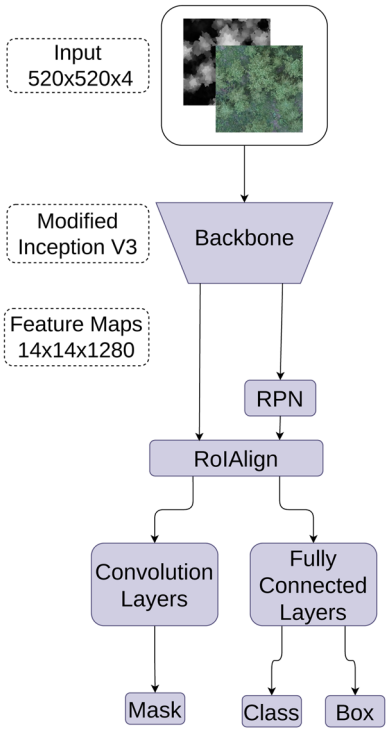

FIGURE 3 | An overview of the Mask R-CNN model. The input layer is modified to accept a four-channel image formed by stacking the RGB and the CHM patches. The backbone is an Inception V3 model modified by removing the final two inception modules and the convolution layers that follow them. See the study by He et al. (2017) for more details about the Mask R-CNN network. CHM, canopy height model; R-CNN, region-based convolutional neural network; RoIAlign, Region of Interest Align; RPN, Region Proposal Network. [Color figure can be viewed at wileyonlinelibrary.com]

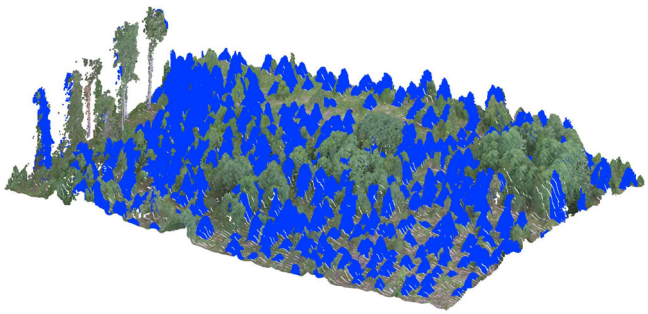

FIGURE 4 | Sample detected spruce trees in the point-cloud map. The spruces are shown in blue. [Color figure can be viewed at wileyonlinelibrary.com]

the *real-time point cloud* from the LiDAR onboard the forest machine. The different point clouds must be preprocessed before being visualized in the HMI. The whole process is shown in Figure 5, and its parts are explained next.

3.2.1 | The Forest-Stand Point-Cloud Preprocessing

The forest-stand point cloud[1] contains more than 14 million points with RGB channels. The point cloud is downsampled using the voxel grid method (Han et al. 2017) to enable real-time processing by the HMI. The ground points are also segmented using the progressive morphological ground removal algorithm (Keqi et al. 2003). This preprocessing is done offline using the Point-Cloud Library (PCL) (Rusu and Cousins 2011).

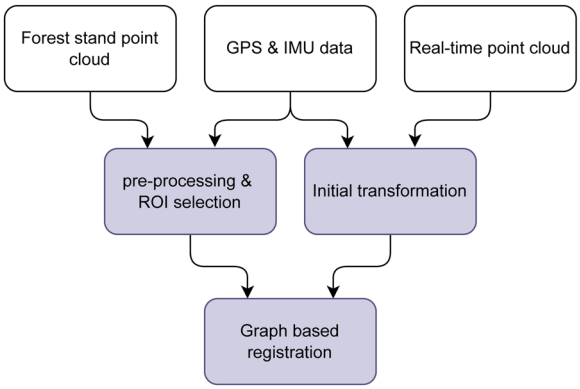

FIGURE 5 | An overview of the point-clouds processing. All these processes, except the preprocessing of the forest stand-point cloud, are performed in real-time in the HMI. GPS, global positioning system; HMI, human–machine interface; IMU, inertial measurement unit; ROI, region of interest. [Color figure can be viewed at wileyonlinelibrary.com]

3.2.2 | The Real-Time Point-Cloud Processing

The LiDAR device onboard the forest machine is a SICK LMS221 2D laser scanner that takes real-time measurements on a vertical plane parallel to the crane plane. The same sensor is also used for the crane pose estimation by Hyyti et al. (2018). This crane pose is also needed in the HMI. The laser scanner measures 75 scan lines consisting of 180 individual range measurements on a plane at 1° intervals each second. A separate angular encoder is used to measure the rotation of the crane pillar on which the laser scanner is mounted. Consequently, the 2D scanner can be used to accumulate 3D points of the surrounding environment by rotating the crane. The sequences of measurements are accumulated into the same point cloud during one full rotation of the crane from left to right, thus building a local map of the environment. The point cloud is thinned using an octree with a fixed voxel size $d \times d \times d$, ensuring that only one point is retained per voxel, where d is the voxel's side length. The construction of the local map using the LiDAR onboard the forest machine is done in real-time. The point cloud is also segmented by removing the ground points using the well-known RANSAC algorithm (Fischler and Bolles 1981) and noisy points are removed using a radius outliers filter (Rusu and Cousins 2011). For each point in the point cloud, the radius outliers filter searches for neighboring points within a specified radius. If the number of these neighboring points is below a given threshold, the point is considered an outlier and is removed from the point cloud.

3.2.3 | Graph-Based Registration of the Forest-Stand Point Cloud and the Real-Time Point Cloud

The forest-stand point cloud is expressed in the WGS84/UTM zone 35N coordinate system. The position of the forest machine, measured in real-time by an onboard GNSS receiver, is converted from latitude and longitude to the UTM coordinate.

Given this measured location of the forest machine and the knowledge of the heading of the forest machine from the IMU, the real-time point cloud is transformed into the same reference frame as the forest-stand point cloud. If $p^{(i)}_{\text{lidar}}$ is a point measured by the LiDAR onboard the forest machine, and if t_{utm} and R_{heading} are, respectively, the position and orientation of the forest machine, then the coordinate of the measured point in UTM coordinate is $p^{(i)}_{\text{utm}} = R_{\text{heading}}\, p^{(i)}_{\text{lidar}} + t_{\text{utm}}$.

The real-time point cloud only covers a local neighborhood around the forest machine. To quickly register that point cloud with the forest-stand point cloud, a region of interest (ROI) is defined in the forest-stand point cloud. This ROI is centered around the measured coordinates t_{utm} of the forest machine and extends to a radius of 20 m. After the initial transformation, a local maximum filter is used to find the local maximum points (in terms of z-coordinate values) in the real-time point cloud. The local maximum points are also searched from the forest-stand point cloud. These local maximum points correspond to the tops of the trees. We will refer to the local maximum from the real-time point cloud and those from the forest-stand point cloud as LM_{rt} and LM_{stand}, respectively. If the forest machine's position t_{utm} and heading R_{heading} were accurate, LM_{rt} and LM_{stand} would align. However, this is not the case due to inaccuracies in GNSS measurements.

Algorithm 1. Graph-based point-cloud registration.

1: **Input:** real-time point-cloud P_{rt}, forest-stand point-cloud P_{stand}, search radius r

2: **Output:** transformation, T

3: Remove ground points from P_{rt}.

4: Detect local maximum points from P_{rt}; Save as LM_{rt}.

5: Detect local maximum points from P_{stand}; Save as LM_{stand}.

6: For each point $p_i \in LM_{\text{rt}}$, find all potential correspondences $g^k_i \in LM_{\text{stand}}$ such that $\|p_i - g^k_i\| \le r$. The set of all matches is C.

7: *Trees height consistency* check: remove matches from C, for which $|h_{p_i} - h_{g_i}| > \epsilon_h$.

8: Build the adjacency matrix M such that $M(i,j)$ is defined as in Equation (3).

9: *Intertrees height consistency* check: Set $M(i,j) = 0$ if $c_i = (p_i, g_i)$ and $c_j = (p_j, g_j)$ are such that $|\Delta h_p - \Delta h_g| > \epsilon_{ij}$ where $\Delta h_p = h_{p_i} - h_{p_j}$ and $\Delta h_g = h_{g_i} - h_{g_j}$.

10: Build a graph G based on M. Compute the maximal cliques in G. Each clique contains a set of correspondences.

11: Use one of the six methods mentioned in Section 3.2.3 to compute the transformation T by the mean of singular value decomposition and RANSAC if necessary.

12: Return T

A graph-based method, inspired by the work of Lusk et al. (2021), is used to overcome the need for an (accurate) initial pose. The method is based on the principle that Euclidean distances of points within the same point cloud are invariant under rigid transformation. The registration algorithm is summarized in Algorithm 1 and explained next.

Following line 6 of Algorithm 1, for each local maximum point $p_i \in LM_{\text{rt}}$, the neighbors within a search radius r are found within LM_{stand}. These neighbors are the potential matches for the point p_i. The radius r can be chosen large enough to ensure that enough inlier matches are included in the set of all matches. These matches are used to build an adjacency matrix M from which a graph $G(V, E)$ can be built. V, the set of vertices, represents the correspondences, and E is the edges between geometrically consistent correspondences. The number of elements of E is also defined as the size of the graph. If $p_i \in LM_{\text{rt}}$ and $g_i \in LM_{\text{stand}}$ are potential correspondences, then the tuple $c_i = \{p_i, g_i\}$ is a vertex in the graph $G(V, E)$, that is, $c_i \in V$.

Let point $p_j \in LM_{\text{rt}}$ have a potential correspondence $g_j \in LM_{\text{stand}}$, so $c_j = \{p_j, g_j\} \in V$. Because a rigid transformation does not change distances, we say that the two edges c_i and c_j are geometrically consistent if the distances $\|p_i - p_j\|$ and $\|g_i - g_j\|$ are (almost) equal. In this case, an edge $e_{ij} \in E$ is created between the vertices c_i and c_j in the graph $G(V, E)$. The adjacency matrix M is a square matrix used to represent the connections between the vertices in the graph $G(V, E)$. The matrix has dimensions $n \times n$, where n is the number of vertices in the graph (also the number of matches). For example, the entry $M(i,j)$ is set to zero if there is no edge between the vertices c_i and c_j, and conversely.

In the literature (see, e.g., Lusk et al. 2021), given two tuples of correspondences $c_i = \{p_i, g_i\}$ and $c_j = \{p_j, g_j\}$, it is common to build an edge between c_i and c_j if the distance difference $s(c_i, c_j)$ in Equation (1) is less than a fixed threshold ϵ. In that case, all points for which $\|p_i - p_j\| \le \epsilon$ and $\|g_i - g_j\| \le \epsilon$ will always produce an edge in the graph. This tends to include more outliers in the graph. Instead, we introduce an adaptive threshold formulation: an edge is added between c_i and c_j if $s(c_i, c_j)$ given in Equation (1) is less than a threshold β, which expression is given in Equation (2). The introduction of the terms $\alpha \cdot min(\|p_i - p_j\|, \|g_i - g_j\|)$ indicates how much error we are willing to accept for a given distance. It avoids automatically adding an edge between c_i and c_j when $\|p_i - p_j\| \le \epsilon$ and $\|g_i - g_j\| \le \epsilon$. In our experiments, this adaptive threshold reduced the size of the graph by more than a thousand edges compared with the case where ϵ alone is used. Since most of the reduced edges were outliers, this reduction in the number of edges was found to be significant when processing a graph.

$$s(c_i, c_j) = abs(\|p_i - p_j\| - \|g_i - g_j\|), \tag{1}$$

$$\beta = min(\epsilon, \alpha \cdot min(\|p_i - p_j\|, \|g_i - g_j\|)), \tag{2}$$

$$M(i,j) = \begin{cases} \exp\left(-\frac{s(c_i,c_j)^2}{2}\right) & \text{if } s(c_i, c_j) \le \beta, \\ 0 & \text{Otherwise.} \end{cases} \tag{3}$$

To find the inlier matches among all the matches, the maximal cliques of the graph $G(V, E)$ can be computed (Lusk

et al. 2021). The maximal cliques are the largest complete subgraphs in $G(V, E)$. They represent the largest sets of correspondences with consistent geometric constraints between any pair of correspondences.

The construction of the graph and the search for maximal cliques can be accelerated by pruning obvious outliers. Recall that the local maximum points represent the tops of the seedling trees. If g_i is a potential match for the point p_i, we compute the height of the points p_i and g_i with respect to the average ground points in the relevant point cloud. These heights are, respectively, referred to as h_{p_i} and h_{g_i}. In line 7 of Algorithm 1, if the difference between h_{p_i} and h_{g_i} is greater than a given threshold ϵ_h, then the corresponding matching is discarded (trees with significantly different height values cannot be good correspondences). In our experiments, this condition reduced the number of potential matches (i.e., vertices in the graph) by more than half. The adjacency matrix M is built in line 8 of Algorithm 1 using the remaining matches.

Because the two point clouds are obtained from different sensor modalities, there can be a systematic height difference. For example, the point cloud from the photogrammetric solution (a priori mapped forest-stand point cloud) tends to have an underestimated height value for objects above ground. This must be taken into account in setting the threshold ϵ_h. Because this systematic error is not known in advance, choosing a high value for ϵ_h is necessary. To compensate for this high value of ϵ_h, an *intertree* heights consistency is introduced in line 9 of Algorithm 1 to make the adjacency matrix much sparser. Given two tuples of correspondences $c_i = \{p_i, g_i\}$ and $c_j = \{p_j, g_j\}$, if the tree corresponding to p_i is higher than the tree corresponding to p_j (i.e., $h_{p_i} > h_{p_j}$), then the same should be true for the tree corresponding to g_i and g_j (i.e., $h_{g_i} > h_{g_j}$), and vice versa. In practice, we compare $\Delta h_p = h_{p_i} - h_{p_j}$ and $\Delta h_g = h_{g_i} - h_{g_j}$. Note that Δh_p and Δh_g are computed using points from the same point cloud; therefore, they are not affected by a systematic height error. Given a user-defined threshold ϵ_{ij}, if $|\Delta h_p - \Delta h_g| > \epsilon_{ij}$, then the entry $M(i, j)$ of the adjacency matrix is set to zero. We found this to significantly reduce the number of edges of the graph and to reduce the search time for maximal cliques. This condition could also use other attributes of the trees, such as the width.

The graph $G(V, E)$ is built in line 10 of Algorithm 1, and the maximal cliques are computed using *igraph*, a C language-based library (Csardi and Nepusz 2006). The *igraph* library uses the algorithm developed by Eppstein et al. (2010) to find the maximal cliques. The algorithm is an improved version of the well-known recursive maximal clique finding *Bron–Kerbosch algorithm* (Bron and Kerbosch 1973; Eppstein et al. 2010). It measures the graph's sparsity in linear time using the degeneracy of the graph, which is the smallest value d such that every nonempty subgraph contains a vertex of degree[2] at most d. Eppstein et al. (2010) set the order of vertices at the outer level of recursion of the *Bron–Kerbosch algorithm* (Bron and Kerbosch 1973). By doing so, all maximal cliques can be listed in time $O(dn3^{d/3})$, where d is the degeneracy and n is the number of vertices of the graph, respectively. Therefore, the sparser the graph, the shorter the algorithm's running time.

If MC is a maximal clique of the graph $G(V, E)$, for any two vertices c_i and c_j in MC, the distance difference $s(c_i, c_j) \leq \beta$. If $\beta = 0$, then the two subsets of points $LM^{MC}_{\text{rt}} = \left\{p_1, p_2, ..., p_{n_{MC}}\right\}$ and $LM^{MC}_{\text{stand}} = \left\{g_1, g_2, ..., g_{n_{MC}}\right\}$ associated with the n_{MC} vertices of MC have the same 3D configuration up to a rigid transformation (Dokmanic et al. 2015; Grinberg and Olver 2019).

In practice, however, for point clouds obtained from different sensor modalities, as is the case in the current study, $s(c_i, c_j)$ can be very small for inlier matches but rarely equal to zero. This is due to the noise in the points' positions. The consequence is that several maximal cliques are found, and the maximal cliques might share some common vertices. A question arises: Which maximal clique should be used to solve the point-cloud's registration? To answer this, we evaluated six methods for computing the relative pose using singular value decomposition from matches once the maximal cliques are identified in the graph:

1. *largest clique:* use the matches from the first largest maximal clique;
2. *intersection of cliques:* use the matches from the core clique, which is the set intersection of a given number of maximal cliques;
3. *union of cliques:* use the matches from the union of a given number of maximal cliques;
4. *hypothesis fusing:* separately compute the pose from each maximal clique and fuse the obtained poses;
5. *min RMSE clique:* separately compute the poses from each maximal clique and select the pose that results in the smallest point-to-point matching root mean square error (RMSE);
6. *max score:* get from the first largest clique, a given number of edges (c_i, c_j), which have the largest values for the scores M_{ij} and compute the pose using the corresponding matches.

The other methods are quite self-explanatory, but the *hypothesis fusing* method, which involves the fusion of poses, is worth detailing. This method assumes that the poses computed from the matches resulting from individual maximal cliques are independent measurements of the actual pose. Consequently, computing the actual pose becomes a pose-averaging problem. To optimally fuse these poses, we employ the method outlined by Barfoot (2017, pp. 280–285). Let $T_1, T_2, ..., T_K$ be the individual poses, and K be the number of maximal cliques. The optimal fusion of these poses can be obtained by optimizing the cost function in Equation (4),

$$J(T) = \sum_{i=1}^{K} e_i(T)^T \Sigma_i e_i(T), \tag{4}$$

in which

$$e_i(T) = \ln\left(T T_i^{-1}\right)^{\vee}, \tag{5}$$

where $e_i(T)$ is the error vector obtained from the logarithmic map of the error pose matrix $TT_i^{-1} \in SE(3)$ (see Barfoot 2017

for more details on the logarithmic map and Lie algebra); Σ_i is the covariance matrix associated with the measurement T_i. In this study, this matrix is replaced with a scalar weight w_i that is proportional to the point-to-point matching RMSE. Let $T = exp(\xi^\wedge)T_{\text{op}}$, with T_{op} being an initial value for the optimal pose, which can be set to T_1 or the identity transformation, for example. The error value $e_i(T)$ can be linearized, and the linear approximation is given in Equation (6) in which $\mathcal{J}$ is a Jacobian term and its expression is given by Barfoot (2017, pp. 235–236).

$$e_i(T) \approx e_i(T_{\text{op}}) - \mathcal{J}(-e_i(T_{\text{op}}))^{-1}\xi, \tag{6}$$

Using this linearized expression of $e_i(T)$ in the expression of $J(T)$, taking the derivative with respect to ξ and equating it to zero, we obtain a linear equation that can be solved for ξ. Using the computed value for ξ, the optimal pose (which is also the linearization point) can be updated by $T_{\text{op}} \leftarrow exp(\xi^\wedge)T_{\text{op}}$. The process is repeated, and we solve for a new value ξ and update the operating point T_{op} until the cost function does not change or ξ becomes very small.

3.3 | Human–Machine Interface

The HMI is organized into two separate modules: a visualization and interaction module, and a processing module, as shown in Figure 6. In the figure, the main components of the HMI and the information flow are visualized. The processing module performs the real-time processing operations that have been discussed previously (see Sections 3.2.2 and 3.2.3). These operations include handling the GNSS, the IMU, and the LiDAR real-time measurements, selecting the ROI in the forest-stand point cloud, and aligning the real-time and forest-stand point clouds.

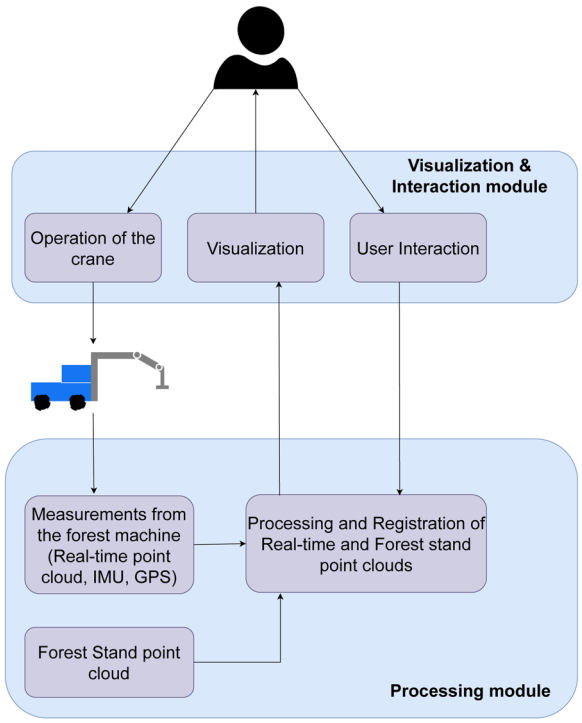

FIGURE 6 | The structure of the HMI with two main modules: a *visualization and interaction module* that allows the operator to see and interact with the views; a *processing module* that acts as a backend where the point-clouds processing and registration are done. GPS, global positioning system; HMI, human–machine interface; IMU, inertial measurement unit. [Color figure can be viewed at wileyonlinelibrary.com]

The visualization and interaction module allows the user to visualize the point cloud and the overall working environment, including the overall view of the forest machine, the location of the forest machine crane tool, and the location of the young spruce trees. It also allows the user to interact with the views and intervene during the program's start to correct the forest machine's pose if needed. Given that GNSS location information can be seriously degraded in the forest environment, this possibility of giving control to the user to intervene in pose correction is important to ensure a safe operation. Note that this kind of manual correction is only necessary at the beginning of the process and rarely needed because the graph-based point-cloud matching algorithm, introduced in Section 3.2.3, is able to locate the forest machine reliably.

The user interaction module provides two default views of the scene. They are referred to as *view 1* and *view 2*. The *view 1* represents the visualization of the overall scene. The user can perform rotation and zooming operations in this view. It allows the user to observe the environment more freely and overcome problems caused by occlusion. The focus point of this view is fixed to the base frame of the forest machine. The *view 2* is a tool-focused view that has a focus point on the tip of the crane of the forest machine. The viewpoint moves along with the movement of the crane. The *view 2* could be seen as a virtual camera attached to the crane, with an automatically moving focus point tracking the tool's tip. The forest machine and its crane presented by the HMI are modeled using simple 3D objects in the PCL library (Rusu and Cousins 2011). The 3D model considers the real dimensions of the forest machine and the crane.

4 | Experimental Setup

The overall system has been tested in a young forest area that had not yet been cleaned. It is located in Lohja, a municipality in the southern region of Finland and has an approximate area of 8 acres. Although it had a limited number of leaf trees, it still functioned effectively as a test area to evaluate the detection of spruces. It included young conifer and deciduous trees, mainly spruces and birches, respectively. The heights of the seedling trees range from 30 cm to approximately 3 m. Figure 7 shows a bird's eye view of the test area.

A UAV shown in Figure 8 was first used to collect images in the test area. It was equipped with a Zenmuse Z3 RGB camera, which is capable of taking high-quality color images with a resolution of 4000×3000 pixels. These images were used to generate the orthomosaic, the DSM, the DTM, and the point cloud, as presented in Section 3.1.

The data collection for mapping was done in two phases, approximately 2 months apart. First, a smaller data set was used to train and validate the tree detection method, and a larger

data set was used to evaluate the accuracy of the detection method against manually marked reference tree positions. The tree detection models were trained on a laptop equipped with an i7 processor, 32 GB of RAM, and an 8 GB NVIDIA GPU. The data set for training and validating the models contains about 200 training images of size 520 × 520 pixels and 50 validation images of the same size. Each of these images typically contains more than one spruce tree. The validation images were used only to tune the model's hyperparameters. To use the map in the later cleaning operation, a plan to conduct the cleaning operation for the stand needs to be done. In this study, a trivial plan was given to clean the surroundings of all detected trees and the cleaning machine was initially located in an accessible location. Naturally, any more complex plan could be used here instead. The plan can be integrated into the developed HMI.

Later, the forest machine, shown in Figure 9, was transported to the test area to study how well the proposed solution works. The forest machine, built from a Valtra T123 agricultural tractor with an installed Kesla 305T hydraulic forestry crane (Kalmari et al. 2014), was used to test the semiautonomous cleaning. The crane is connected to the tractor through an ISO 11783-compliant hydraulic unit, enabling the human operator to control the crane via joysticks (Kalmari et al. 2013). In addition, a swinging saw-like object is attached to the end joint of the crane to simulate a saw that could be used to clean a young forest stand. The system also includes two sensors that are attached to the crane: a SICK LMS221 2D laser scanner and IMU sensors for the estimation of the pose of the crane and the swaying cutting tool (Hyyti 2023; Hyyti et al. 2018; Kalmari et al. 2013).

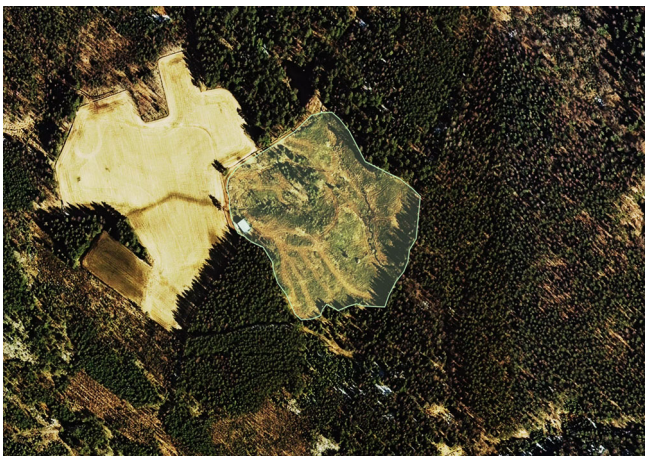

FIGURE 7 | A bird-eye view of the test area in Lohja, Finland. [Color figure can be viewed at wileyonlinelibrary.com]

4.1 | Evaluation of the Registration Methods

Providing the exact reference position for the forest machine is a challenge. For example, as already noted earlier, the GNSS positioning accuracy under the forest canopy is limited. An option would be to manually align the forest-stand point cloud and the real-time point cloud and use the obtained transformation as the ground truth to evaluate the registration algorithm. However, this manual alignment is prone to errors. Another challenge in the manual alignment is the noise in the position of the detected tree top from the two different point-cloud sources.

Instead of relying on a reference pose obtained from manual alignment, we used an alternative solution. A subset of the forest-stand point cloud in a given ROI was copied to simulate the measurement from the forest machine. Note that only the detected local maximum points representing the tops of the trees are used for the registration. The selected subset of local maximum points is first transformed using a reference transformation matrix T_{ref}. Second, to mimic the local maxima acquired from the real-time measurements, local random position perturbations are added to each point within this subset. The perturbation is selected to be Gaussian with zero mean and a given covariance. The objective of adding these perturbations is to test the robustness of the different pose computation methods against the slight variation in the position of the detected top of trees obtained from different sensor modalities (photogrammetry-based map versus LiDAR onboard a ground machine), as well as wind effects that cause minor movements of seedling trees.

Since the primary goal of computing the maximal cliques in Algorithm 1 is to obtain accurate matches, it is essential to perform an ablation study to analyze how the user-defined parameters ϵ, α, and ϵ_{ij} affect the precision of detecting the matches. In our case, we systematically vary these parameters and compute the precision of the matches using Equation (7), where the

FIGURE 8 | DJI Matrice 100 drone equipped with a Zenmuse Z3 RGB camera. [Color figure can be viewed at wileyonlinelibrary.com]

FIGURE 9 | The instrumented forest machine in the test area. [Color figure can be viewed at wileyonlinelibrary.com]

matches used to calculate the true positives (TPs) and false positives (FPs) are obtained by merging the matches from the first five largest maximal cliques. The precision metric is chosen because it allows us to compare the rate of TP matches relative to all the matches, including both TP and FP. We do not consider false negatives (FN) because we can accurately compute the pose with sufficient TP 3D point matches. In addition, we examine how these variables affect the size of the graph $G(V, E)$. Specifically, we show how the size of $G(V, E)$ varies with ϵ. We show how the graph size relative reduction varies as a function of α and ϵ_{ij}. The graph size relative reduction can be defined as the proportion of the edges removed from the graph as a consequence of introducing the parameters α and ϵ_{ij}.

The graph size relative reduction (rr_α) for the parameter α is calculated using Equation (8), where SG_ϵ is the size of the graph when $\beta = \epsilon$ in Equation (2) and SG_α is the size of the graph when $\beta = min(\epsilon, \alpha \cdot min(\|p_i - p_j\|, \|g_i - g_j\|))$. This indicates the relative change in the size of the graph when a fixed threshold is used to construct the adjacent matrix M in line 8 of Algorithm 1 versus the use of the newly introduced adaptive threshold in Equation (2).

The relative reduction of the graph size for the parameter ϵ_{ij} is computed using Equation (9), where $SG_{\epsilon_{ij}=1m}$ is the size of $G(V, E)$ when ϵ_{ij} assumes a high value (here 1 m) and $SG_{\epsilon_{ij}\in(0,1]}$ is the size of the $G(V, E)$ when $\epsilon_{ij} \in (0, 1]$.

No ablation study was done on the parameter ϵ_h in line 7 of Algorithm 1 because it is only used to prune gross outliers, and its value is highly dependent on the bias of the tree height in the photogrammetry-based map. Its value can be chosen to be arbitrarily large. In this study, a value of $\epsilon_h = 1.0$ m is chosen.

$$Precision = \frac{TP}{TP + FP}, \tag{7}$$

$$rr_\alpha = \frac{SG_\epsilon - SG_\alpha}{SG_\epsilon}, \tag{8}$$

$$rr_{\epsilon_{ij}} = \frac{SG_{\epsilon_{ij}=1m} - SG_{\epsilon_{ij}\in(0,1]}}{SG_{\epsilon_{ij}=1m}}. \tag{9}$$

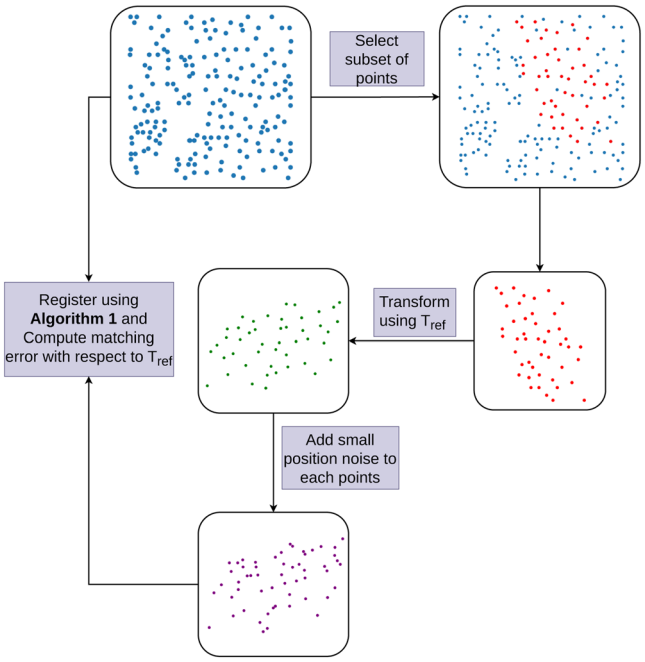

FIGURE 10 | An overview of the registration evaluation setup. T_{ref} is kept constant. The steps of adding local position noise to each point and the registration and error computation are repeated several times so that we have several registration error values, which are used to produce the angle and position error plots in Section 5.2.2. [Color figure can be viewed at wileyonlinelibrary.com]

After choosing the optimal values for the parameters ϵ, α, and ϵ_{ij}, two experiments are carried out to evaluate the six proposed pose estimation methods. In the first experiment, the position perturbation covariance is kept constant, and the number of perturbed points varies. In the second experiment, 100% of the points are perturbed, and the induced noise level varies from a variance of 0–0.4 m^2 (i.e., the maximum standard deviation is 0.63 m and $3\sigma = 1.89$ m). The overall process of the evaluation setup of the registration methods is shown in Figure 10.

5 | Results and Discussions

In this section, we present the result of the detection of the seedling spruces followed by the point-cloud registration and the HMI.

5.1 | Young Spruce Detection

Table 1 shows the detection accuracies of the *ITD + detector* method, the method using only the RGB orthomosaic (Mask R-CNN with only RGB image), and the proposed approach using the RBG-D data. The detection accuracy of the *ITD + detector* method reaches, on average, a detection rate of 95% on two separate data sets collected in summer and autumn (Ouattara et al. 2020). The proposed method that combines the RGB orthomosaic with the CHM has the highest accuracy, reaching 98%. This corroborates our assumption that elevation data are important for deep learning tree detection. The proposed method can also detect seedling trees located within a concentration of leaf trees but visible from above.

TABLE 1 | Accuracy of the different tree detection methods.

Method	Accuracy (%)
Previous method (*ITD + detector*)	95.1
Mask R-CNN with only RGB image	94.47
Proposed (Mask R-CNN with RGB-D)	98.36

Abbreviations: ITD, individual tree identification; R-CNN, region-based convolutional neural network.

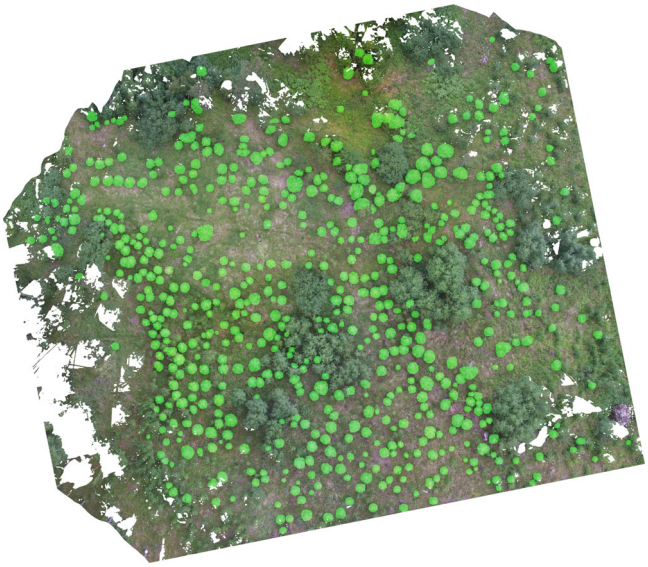

FIGURE 11 | The detection results obtained using the proposed Mask R-CNN pipeline on the smaller data set. The highlighted green overlay represents the detected spruces. R-CNN, region-based convolutional neural network. [Color figure can be viewed at wileyonlinelibrary.com]

Figures 11–13 show the results of the proposed approach. Figure 13, in addition to showing the detected trees highlighted by the light green color, also shows the manual annotations (as red dots) for the accuracy computation. Note that the deep learning method is trained using only part of the map in Figure 11.

5.2 | Point-Cloud Matching and HMI

5.2.1 | Ablation Study

Figure 14 shows the precision of the detected matches and the size of the graph $G(V, E)$ as a function of the parameter ϵ in Algorithm 1. A precision greater than 86% is achieved for $\epsilon > 0.4$. It can also be observed that the graph size increases linearly with ϵ. To achieve a high precision while keeping a manageable graph size, $\epsilon = 0.65$ is selected for subsequent experiments.

Figure 15 shows the precision of the detected matches and the relative reduction in the graph size as a function of the parameter α used in Equation (2) and line 8 of Algorithm 1. Figure 15 also shows the precision of the detected matches when $\beta = \epsilon = 0.65$ is used as the distance threshold instead of the full expression in Equation (2). It can be seen that by choosing $\alpha \in [0.04, 0.1]$, we can achieve a reduction between 44% and 8. 8% in the size of the graph without negatively impacting the precision of the detected matches. In this study, we select $\alpha = 0.05$, which reduces the number of edges by 52,000 (33%) while maintaining a precision of more than 90%.

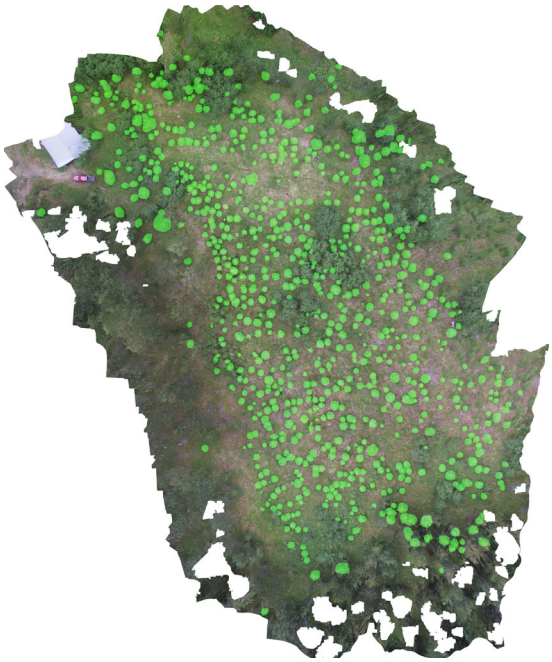

FIGURE 12 | The detection results obtained using the proposed Mask R-CNN pipeline on the large data set. The highlighted green overlay represents the detected spruces. R-CNN, region-based convolutional neural network. [Color figure can be viewed at wileyonlinelibrary.com]

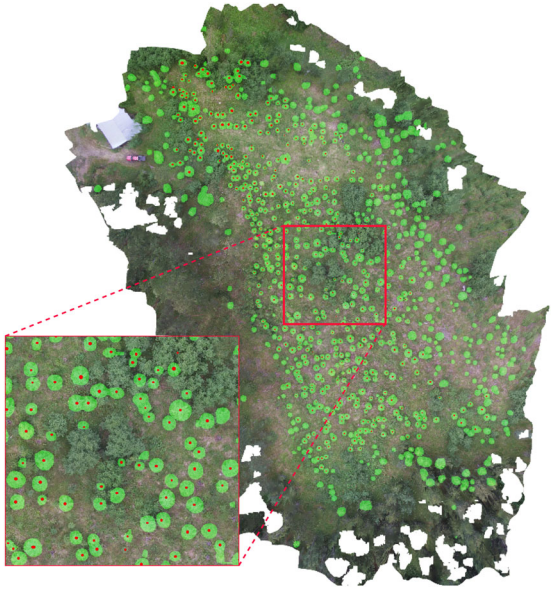

FIGURE 13 | The detection results obtained using the proposed Mask R-CNN pipeline and the manual marking for the accuracy computation. The highlighted green overlay represents the detected spruces; the red dots are the manual markings. R-CNN, region-based convolutional neural network. [Color figure can be viewed at wileyonlinelibrary.com]

Figure 16 shows the precision of the detected matches and the relative reduction in the graph size as a function of the

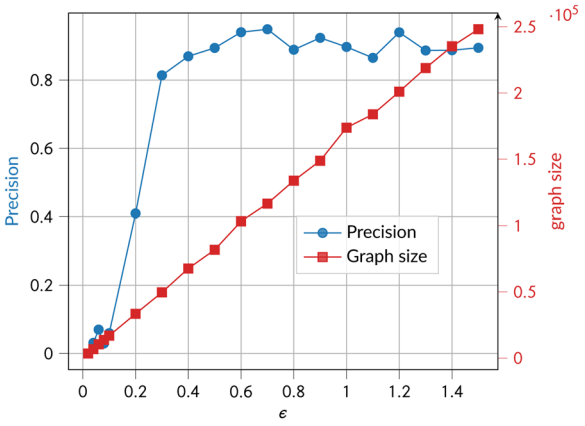

FIGURE 14 | The graph size and the precision of the detected matches as functions of the parameter ϵ used in Equation (2). [Color figure can be viewed at wileyonlinelibrary.com]

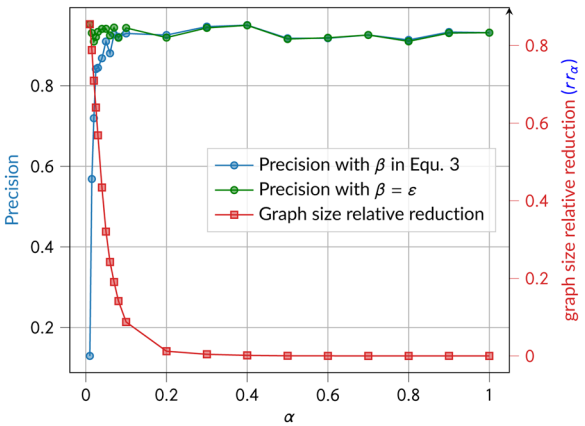

FIGURE 15 | The precision of detected matches and the relative reduction of the graph size as a function of the parameter α used in Equation (2) are shown. The relative reduction of the graph size represents the fraction of the edges removed from the graph as a consequence of using the extra term $\alpha \cdot min(\|p_i - p_j\|, \|g_i - g_j\|)$. Its computation is shown in Equation (8). [Color figure can be viewed at wileyonlinelibrary.com]

parameter ϵ_{ij}. It can be seen that by choosing $\epsilon_{ij} \in [0.3\,\text{m}, 0.6\,\text{m}]$, a high precision (more than 90%) is maintained while achieving a reduction in graph size between 61% and 30%. We selected $\epsilon_{ij} = 0.35$ for the subsequent experiments.

5.2.2 | Registration Methods

The results of the first experiment of the registration are shown in Figures 17 and 18 for rotation and translation errors, respectively. The percentage of perturbed points is changed, and the translation and rotation errors of each pose estimation method are computed and plotted. The results of the second experiment are shown in Figures 19 and 20 for rotation and translation errors, respectively. The value of the perturbation covariance is changed, and the translation and rotation errors of each pose estimation method are computed and plotted. All the points are perturbed in this second experiment.

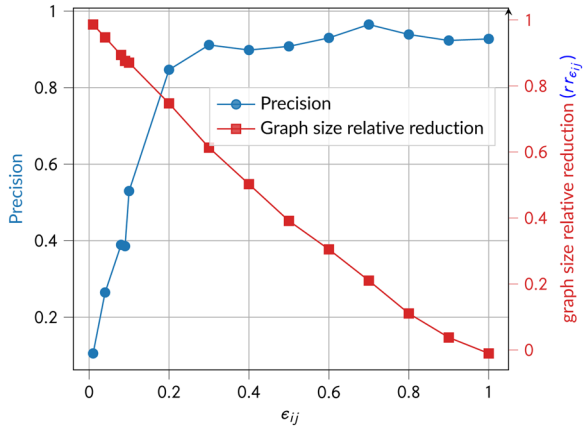

FIGURE 16 | The precision of detected matches and the relative reduction of the graph size as a function of the parameter ϵ_{ij} used in line 9 of Algorithm 1. The relative change in graph size is calculated relative to a reference graph size when $\epsilon_{ij} = 1.0$ using Equation (9). [Color figure can be viewed at wileyonlinelibrary.com]

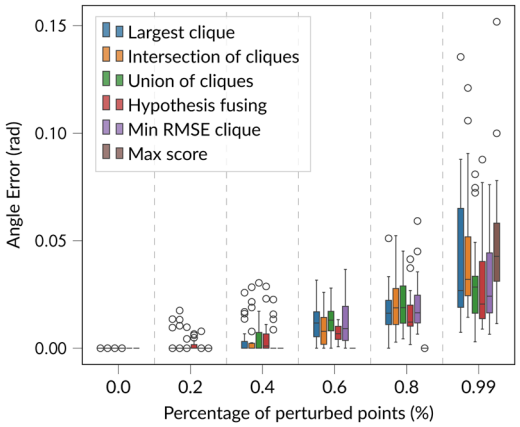

FIGURE 17 | Rotation errors as a function of the percentage of perturbed points. The Boxes indicate the interquartile ranges, while whiskers denote the 25% and 75% quartiles, and the center line indicates the median. The perturbation standard deviation for the position of the points is $\sigma = 0.63$ m. RMSE, root mean square error. [Color figure can be viewed at wileyonlinelibrary.com]

As can be observed in Figures 17 and 18 when there are enough points that are not perturbed, the *max score* method based on the matches with the highest score can recover the ground truth pose. This is expected as those matches correspond to edges in the graph for which $s(c_i, c_j) = 0$; they are true matches. When all the points are perturbed, the *hypothesis fusing* method and the *min rmse clique* method provide better results. They both maintain a low median error and a small interquartile across varying perturbation values, as shown in Figures 19 and 20. The median translation errors for both experiments are lower than 0.5 m, and the median rotation errors are less than 3°.

Since being the most robust against added noise, the *hypothesis fusing* method is used to compute the registration of the actual real-time point cloud with the forest-stand point cloud. We compare our method and state-of-the-art registration methods generalized-ICP (GICP) (Segal et al. 2009), Teaser++ (Yang et al. 2021), and G3Reg (Qiao et al. 2024). Here, we only show

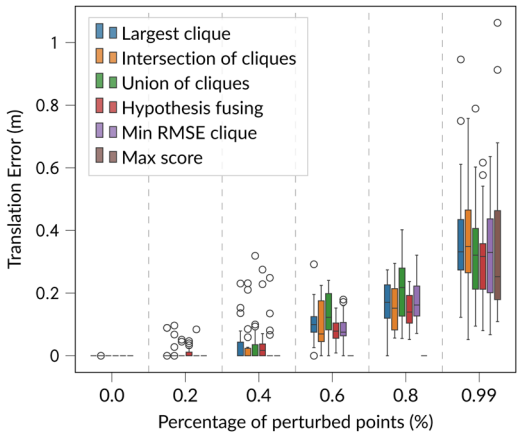

FIGURE 18 | Translation errors as a function of the percentage of perturbed points. The Boxes indicate the interquartile ranges, while whiskers denote the upper and lower quartiles, and the center line indicates the median. The perturbation standard deviation for the position of the points is $\sigma = 0.63$ m. RMSE, root mean square error. [Color figure can be viewed at wileyonlinelibrary.com]

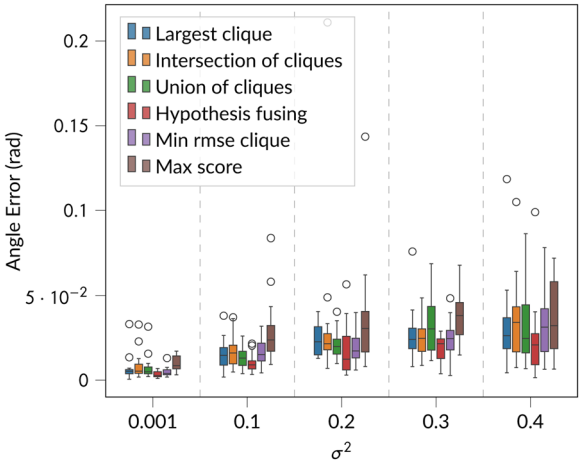

FIGURE 19 | Rotation errors as a function of the variance of the added perturbations. The Boxes indicate the interquartile ranges while whiskers denote the upper and lower quartiles, and the center line indicates the median. [Color figure can be viewed at wileyonlinelibrary.com]

qualitative registration results in Figure 21. It can be observed that only the proposed method can correctly register the two point clouds.

5.2.3 | The User Interface

The registered point clouds in the HMI are shown in Figures 22 and 23. The *view 1* is shown in Figure 22. It provides the operator with an overall situational awareness when moving the forest machine to a new location. It allows the operator to move the crane without the risk of hitting large trees around the machine.

The *view 2*, or working view, is shown in Figure 23. In this view, the green cones represent the spruce trees. These are the same spruce trees detected by our proposed deep learning methods. This perspective displays the machine's crane tool view. As can

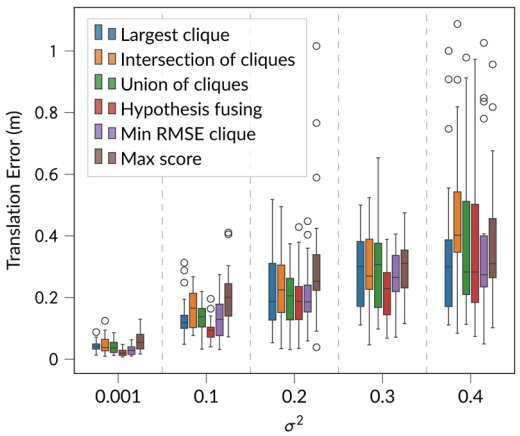

FIGURE 20 | Translation errors as a function of the variance of the added perturbations. The Boxes indicate the interquartile ranges while whiskers denote the upper and lower quartiles, and the center line indicates the median. RMSE, root mean square error. [Color figure can be viewed at wileyonlinelibrary.com]

be seen, the operator can use this view to remove the unwanted trees while keeping the target spruce trees.

These two views remove the challenges caused by blind spots in the operation of the forest machine and the crane.

5.3 | Limitations and Future Work

The UAV was flown above the canopy to collect the images used to build the a priori map. Therefore, the GNSS positioning of the UAV is not affected by the forest canopy. However, any GNSS device has a limited accuracy. The absolute location and height information of the detected spruce trees is directly tied to the accuracy of the GNSS device onboard the drone and the SFM method used to produce the point cloud. Therefore, it is important to further investigate the accuracy of the height and position information of the detected trees in future work. In fact, it was observed during the test that the height of the spruces in the forest-stand point cloud has a systematic error of about 40 cm compared with the height reported by the LiDAR mounted on the forest machine. However, since the real-time measurements from the forest machine are fitted into the forest-stand point cloud, small absolute positioning errors of the UAV data or the generated map do not have an impact on the cleaning operation itself.

Although random noise was added to the coordinates of the detected tree tops to simulate slight wind effects that cause minor movements of seedling trees, more extensive testing may be necessary to account for a broader range of wind conditions that could impact the proposed localization method.

Given that the forest environment is changing during the cleaning process, it is important to continuously update the forest-stand point cloud to ensure correct matching with the real-time point cloud. However, no update was performed in our tests as we were not allowed to perform the actual cleaning operation in the test forest. We just simulated the cleaning operation using a dummy weight, as shown in Figure 9. Instead,

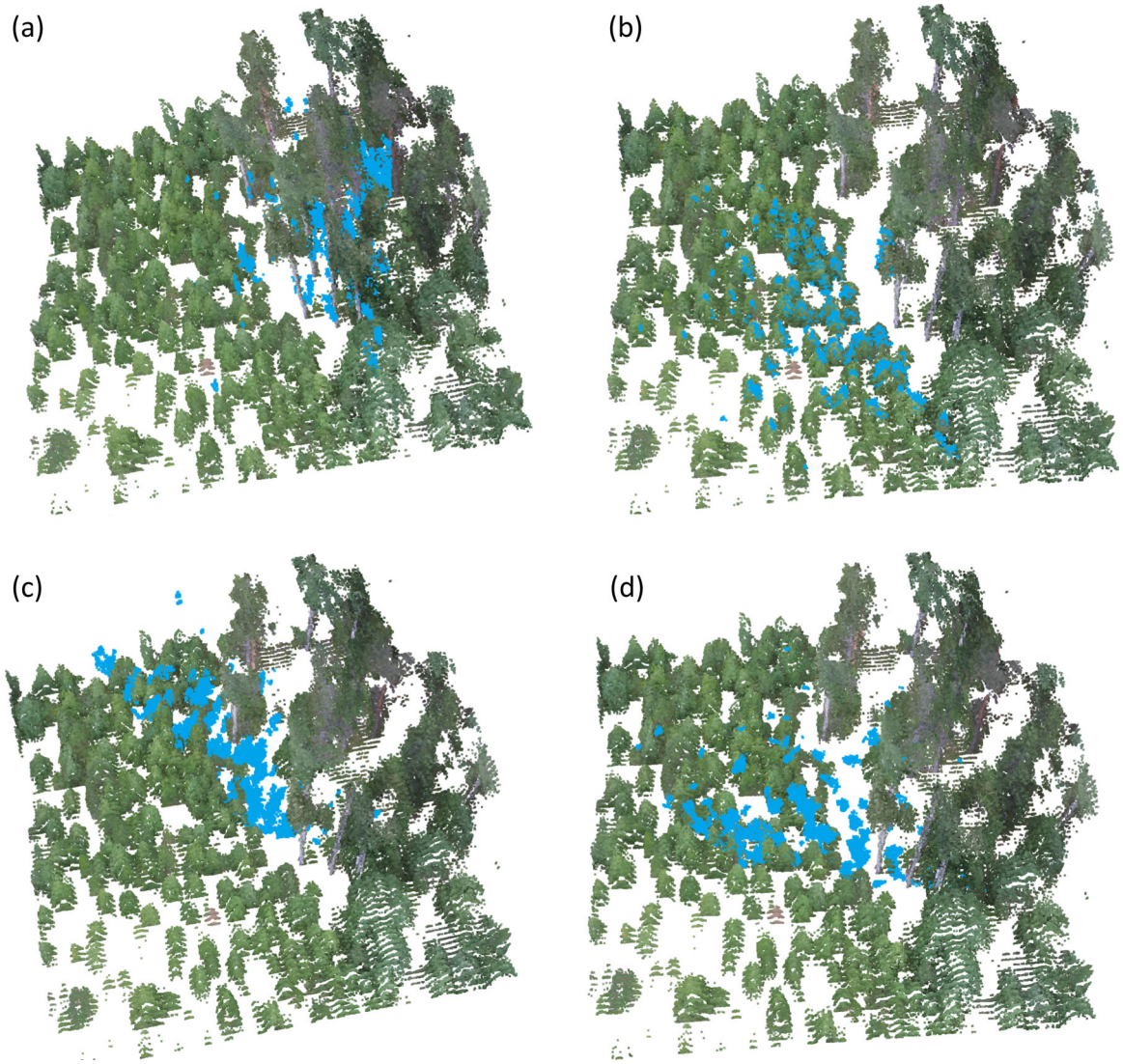

FIGURE 21 | Results of the registration with our method compared with state-of-the-art methods GICP (Segal et al. 2009), Teaser++ (Yang et al. 2021), and G3Reg (Qiao et al. 2024). (a) Registration with GICP method (Wrong registration result). (b) Registration with our method (Hypothesis fusing method is used). (c) Registration with G3Reg method (Wrong registration result). (d) Registration with Teaser++ method (Wrong registration result). GICP, generalized iterative closest point. [Color figure can be viewed at wileyonlinelibrary.com]

updating the map or the forest-stand point cloud has been left as a future work. We hypothesize that the real-time point cloud can be used to update the forest-stand point cloud using a SLAM approach (see, e.g., Faitli et al. 2023, 2024; Hyyti 2023) in which the area under cleaning is either removed from the forest-stand point cloud or marked as unreliable before merging new real-time measurements into the forest-stand point cloud.

One limitation of the proposed method is that it has been evaluated within a limited forest environment due to practical constraints. To further assess the robustness and generalizability of our approach, future work will involve testing the method in larger and more diverse forest scenarios.

In addition, tests to show how much productivity can be increased with the proposed system would be needed in future work. These could show if the working speed could be increased, if errors made by the driver could be reduced, or if the quality of the cleaning operation thus improved. Moreover, the system could be automatized even further. We leave the fully automatic cleaning process in which the crane would be controlled, for example, with the model predictive control by Kalmari et al. (2014) to clean the young forest stand using the cleaning plan generated over the mapped stand to be later demonstrated in the future work.

6 | Conclusions

In this work, we presented a way for a semiautonomous forest machine to perform a young spruce stand's late-cleaning operation with the help of a prior tree map and its onboard sensors for positioning itself on the map. A low-cost UAV equipped with a RGB camera was used to collect the images needed to generate the prior tree map.

For building the tree map, we developed a tree detection scheme to identify the trees of interest that must be left growing

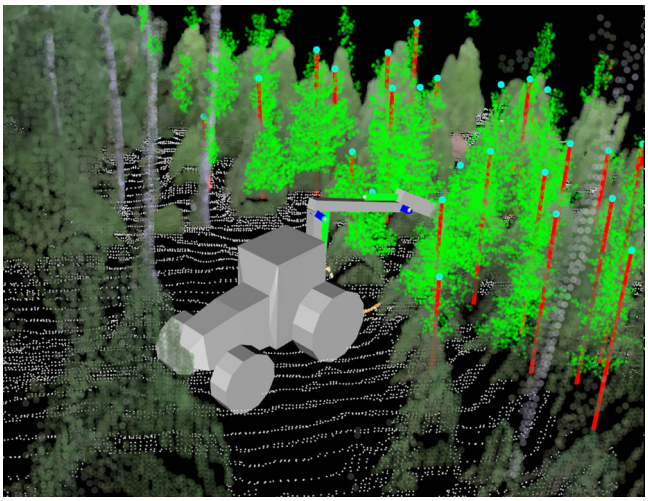

FIGURE 22 | The matched point clouds and tree visualization. The blue dots indicate tree tops, and red vertical lines connect each tree's base to its top, improving the visualization of detected trees. This also represents the *view 1* in which the machine's location with respect to its surroundings can be seen. The operator can manually zoom in and out. It can also be rotated if needed. [Color figure can be viewed at wileyonlinelibrary.com]

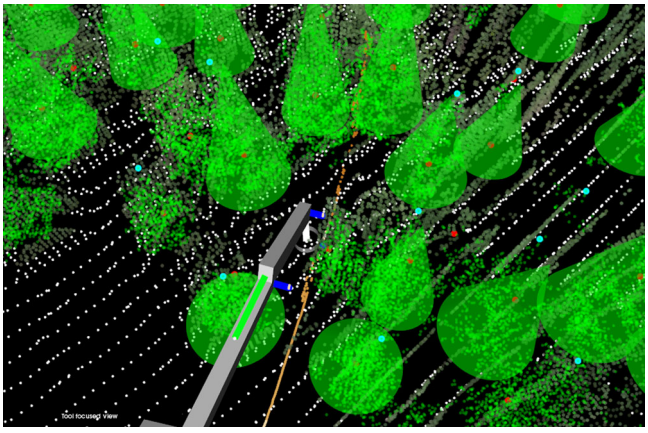

FIGURE 23 | The tool-focused view or working view (*view 2*) of the HMI. The green cones represent the detected spruces from the UAV data. The white points represent the ground. The other green points represent leaf trees. HMI, human–machine interface; UAV, unmanned aerial vehicle. [Color figure can be viewed at wileyonlinelibrary.com]

during the cleaning operation. The proposed method combines an aerial orthomosaic with a CHM as an RGB-D image, which is used as input for a modified Mask R-CNN deep learning model. We utilized a pretrained model, which we retrained to predict the masks for the spruce trees. The proposed tree detection method achieved a decent spruce detection accuracy of 98%, which is significantly better than any of the methods compared in our tests.

To use the detected tree map in a cleaning operation performed with the semiautonomous forest machine, first, a work plan for the cleaning operation was made and integrated with the map containing the detected trees. The detection result was then merged into the forest-stand point cloud obtained from the images collected by the UAV. This point cloud was used to locate the semiautonomous forest machine in real-time.

We proposed the use of an HMI in the forest machine cabin to aid the operator. It incorporates the forest-stand point cloud with the tree detection result and the real-time point cloud from a LiDAR onboard the forest machine to provide better situational awareness to the machine operator. We used two different views in the HMI: The first view (shown in Figure 22) to show the overview of the machine and its surroundings, and the second view (shown in Figure 23) to help the cleaning process since the driver sitting in the forest machine cabin cannot always see the target spruces among other vegetation.

We evaluated six different versions of a graph-based point-matching method used to register the real-time point cloud and the forest-stand point cloud. We found that the *hypothesis fusing* version of the method was the most reliable, and it best withstood the induced noise in our tests. It made only around 0.4 m errors in the tests when all measurements were perturbed with a Gaussian noise, which has a standard deviation of 0.63 m. When only less than half of the points were perturbed with the same noise, the method showed only minor errors, less than 0.1 m.

In future work, a method for updating the forest-stand point cloud using the real-time measurements from the forest machine is needed, since the cleaning process is expected to modify the environment significantly, thus making the positioning increasingly difficult using the map or data collected before the operation. We believe that this could be done by modifying the forest-stand point cloud with new real-time measurements during the cleaning process. Furthermore, tests to show how much productivity can be increased with the proposed system would be needed to show whether the working speed can be increased, whether errors made by the driver can be reduced, and whether the quality of the cleaning operation can thus be improved.

Acknowledgments

The authors contributed in the following ways: Mr. Issouf Ouattara designed and developed the mapping and spruces detection method, the point-cloud matching methods, and prepared the manuscript. He also improved the human–machine interface. Mr. Jyri Hallikas designed and developed the human–machine interface. Issouf Ouattara, Jyri Hallikas, Heikki Hyyti, and prof. Arto Visala performed the experimental tests. Dr. Heikki Hyyti guided the study and supported writing the manuscript. Prof. Arto Visala acquired funding and supervised the project. We gratefully acknowledge the Strategic Research Council of Academy of Finland that funded this research through Grant/Award Numbers 293389 and 314312. In addition, the research is funded by the Technology Industries of Finland Centennial Foundation and Jane and Aatos Erkko Foundation in the Future Makers research program.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Endnotes

[1]This point cloud is exported from Agisoft Metashape software.

[2]The degree of a vertex is the number of edges connected to it.

References

Agisoft, L. L. C., and R. St. Petersburg. 2019. *Agisoft Metashape User Manual, Professional Edition, Version 1.5 [Computer Software Manual]*. Agisoft LLC. https://www.agisoft.com/pdf/metashape-pro_1_5_en.pdf.

Barfoot, T. D. 2017. *State Estimation for Robotics*. Cambridge University Press. https://doi.org/10.1017/9781316671528.

Besl, P., and N. D. McKay. 1992. "A Method for Registration of 3-D Shapes." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14, no. 2: 239–256. https://doi.org/10.1109/34.121791.

Bron, C., and J. Kerbosch. 1973. "Algorithm 457: Finding All Cliques of an Undirected Graph." *Communications of the ACM* 16, no. 9: 575–577. https://doi.org/10.1145/362342.362367.

Csardi, G., and T. Nepusz. 2006. "The Igraph Software Package for Complex Network Research." *InterJournal, Complex Systems* 1695: 1–9. https://igraph.org.

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. "ImageNet: A Large-Scale Hierarchical Image Database." In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255. IEEE.

Dokmanic, I., R. Parhizkar, J. Ranieri, and M. Vetterli. 2015. "Euclidean Distance Matrices: Essential Theory, Algorithms, and Applications." *IEEE Signal Processing Magazine* 32, no. 6: 12–30. https://doi.org/10.1109/MSP.2015.2398954.

Eppstein, D., M. Löffler, and D. Strash. 2010. "Listing All Maximal Cliques in Sparse Graphs in Near-Optimal Time." In *Algorithms and Computation*, edited by O. Cheong, K.-Y. Chwa and K. Park, 403–414. Springer.

Faitli, T., T. Hakala, H. Kaartinen, J. Hyyppä, and A. Kukko. 2023. "Real-Time Lidar-Inertial Positioning and Mapping for Forestry Automation." *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 48: 145–150.

Faitli, T., E. Hyyppä, H. Hyyti, et al. 2024. "Integration of a Mobile Laser Scanning System With a Forest Harvester for Accurate Localization and Tree Stem Measurements." *Remote Sensing* 16, no. 17: 3292. https://doi.org/10.3390/rs16173292.

Fang, Y., and Y. K. Cho. 2017. "Effectiveness Analysis From a Cognitive Perspective for a Real-Time Safety Assistance System for Mobile Crane Lifting Operations." *Journal of Construction Engineering and Management* 143, no. 4: 05016025. https://doi.org/10.1061/(ASCE)CO.1943-7862.0001258.

Fang, Y., Y. K. Cho, and J. Chen. 2016. "A Framework for Real-Time Pro-Active Safety Assistance for Mobile Crane Lifting Operations." *Automation in Construction* 72: 367–379. https://doi.org/10.1016/j.autcon.2016.08.025.

Fischler, M. A., and R. C. Bolles. 1981. "Random Sample Consensus: A Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography." *Communications of the ACM* 24, no. 6: 381–395. https://doi.org/10.1145/358669.358692.

Giannetti, F., G. Chirici, T. Gobakken, E. Naesset, D. Travaglini, and S. Puliti. 2018. "A New Approach With DTM-Independent Metrics for Forest Growing Stock Prediction Using UAV Photogrammetric Data." *Remote Sensing of Environment* 213: 195–205. https://doi.org/10.1016/j.rse.2018.05.016.

Gillies, S., B. Ward, and A. Petersen. 2013. *Rasterio: Geospatial Raster I/O for Python Programmers*. Github Repository. https://github.com/rasterio/rasterio.

Grinberg, D., and P. J. Olver. 2019. "The N Body Matrix and Its Determinant." *SIAM Journal on Applied Algebra and Geometry* 3, no. 1: 67–86. https://doi.org/10.1137/18M1175410.

Hamberg, L., M. Strandström, and T. Saksa. 2022. "Cutlink Cleaning Head With a Spreading Feature for Biological Sprout Control." *Croatian Journal of Forest Engineering: Journal for Theory and Application of Forestry Engineering* 43, no. 1: 152–164. https://doi.org/10.5552/crojfe.

Han, X.-F., J. S. Jin, M.-J. Wang, W. Jiang, L. Gao, and L. Xiao. 2017. "A Review of Algorithms for Filtering the 3D Point Cloud." *Signal Processing: Image Communication* 57: 103–112.

He, K., G. Gkioxari, P. Dollar, and R. Girshick. 2017. "Mask R-CNN." In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE.

Heinonen, T., T. Pukkala, L. Mehtätalo, A. Asikainen, J. Kangas, and H. Peltola. 2017. "Scenario Analyses for the Effects of Harvesting Intensity on Development of Forest Resources, Timber Supply, Carbon Balance and Biodiversity of Finnish Forestry." *Forest Policy and Economics* 80: 80–98. https://doi.org/10.1016/j.forpol.2017.03.011.

Holz, D., A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke. 2015. "Registration With the Point Cloud Library: A Modular Framework for Aligning in 3-D." *IEEE Robotics Automation Magazine* 22, no. 4: 110–124. https://doi.org/10.1109/MRA.2015.2432331.

Huang, H., X. Li, and C. Chen. 2018. "Individual Tree Crown Detection and Delineation From Very-High-Resolution UAV Images Based on Bias Field and Marker-Controlled Watershed Segmentation Algorithms." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 11, no. 7: 2253–2262. https://doi.org/10.1109/JSTARS.2018.2830410.

Huuskonen, S., S. Haikarainen, T. Sauvula-Seppälä, et al. 2020. "Benefits of Juvenile Stand Management in Finland–Impacts on Wood Production Based on Scenario Analysis." *Forestry: An International Journal of Forest Research* 93, no. 3: 458–470. https://doi.org/10.1093/forestry/cpz075.

Hyyti, H. 2023. "Perception Systems for Autonomous Forest Machinery." Doctoral diss., Aalto University.

Hyyti, H., V. V. Lehtola, and A. Visala. 2018. "Forestry Crane Posture Estimation With a Two-Dimensional Laser Scanner." *Journal of Field Robotics* 35, no. 7: 1025–1049. https://doi.org/10.1002/rob.2018.35.issue-7.

Iglhaut, J., C. Cabo, S. Puliti, L. Piermattei, J. O'Connor, and J. Rosette. 2019. "Structure From Motion Photogrammetry in Forestry: A Review." *Current Forestry Reports* 5, no. 3: 155–168. https://doi.org/10.1007/s40725-019-00094-3.

Jiménez-Jiménez, S. I., W. Ojeda-Bustamante, M. d. J. Marcial-Pablo, and J. Enciso. 2021. "Digital Terrain Models Generated With Low-Cost UAV Photogrammetry: Methodology and Accuracy." *ISPRS International Journal of Geo-Information* 10, no. 5: 2280–2292. https://doi.org/10.3390/ijgi10050285.

Kaartinen, H., J. Hyyppä, M. Vastaranta, et al. 2015. "Accuracy of Kinematic Positioning Using Global Satellite Navigation Systems Under Forest Canopies." *Forests* 6, no. 9: 3218–3236.

Kalmari, J., J. Backman, and A. Visala. 2014. "Nonlinear Model Predictive Control of Hydraulic Forestry Crane With Automatic Sway Damping." *Computers and Electronics in Agriculture* 109: 36–45. https://doi.org/10.1016/j.compag.2014.09.006.

Kalmari, J., H. Hyyti, and A. Visala. 2013. "Sway Estimation Using Inertial Measurement Units for Cranes With a Rotating Tool." *IFAC Proceedings Volumes* 46, no. 10: 274–279.

Kalmari, J., T. Pihlajamäki, H. Hyyti, M. Luomaranta, and A. Visala. 2013. "Iso 11783 Compliant Forest Crane as a Platform for Automatic Control." *IFAC Proceedings Volumes* 46, no. 18: 164–169.

Katoh, M., S. Deng, Y. Takenaka, et al. 2017. "Development of Smart Precision Forest in Conifer Plantation in Japan Using Laser Scanning Data." *ISPRS—International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII–3/W3: 95–100. https://doi.org/10.5194/isprs-archives-XLII-3-W3-95-2017.

Keqi, Z., S.-C. Chen, D. Whitman, M.-L. Shyu, J. Yan, and C. Zhang. 2003. "A Progressive Morphological Filter for Removing Nonground Measurements From Airborne Lidar Data." *IEEE Transactions on Geoscience and Remote Sensing* 41, no. 4: 872–882.

Korhonen, K. T., and A. Ihalainen. 2017. *2017 valtakunnan Metsien Inventointi: Puuston Kasvu Noussut Edelleen-Pohjois-Suomessa Metsät Järeytyvät (National Forest Inventory: The Growth of Forests Continues Increasing-in Northern Finland Forests Continue To Grow)*. https://www.luke.fi/uutiset/valtakunnan-metsien-inventoinnin-tulosjulkistus-2017/.

Li, Q., P. Nevalainen, J. PeñaQueralta, J. Heikkonen, and T. Westerlund. 2020. "Localization in Unstructured Environments: Towards Autonomous Robots in Forests With Delaunay Triangulation." *Remote Sensing* 12, no. 11: 1870. https://doi.org/10.3390/rs12111870.

Lusk, P. C., K. Fathian, and J. P. How. 2021. "CLIPPER: A Graph-Theoretic Framework for Robust Data Association." In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 13828–13834. IEEE.

Meyer, F., and S. Beucher. 1990. "Morphological Segmentation." *Journal of Visual Communication and Image Representation* 1, no. 1: 21–46. https://doi.org/10.1016/1047-3203(90)90014-M.

Ministry of Agriculture and Forestry of Finland. 2020. "Forests and the Economy." Accessed 2020-12-03. https://mmm.fi/en/forests/forestry/sustainable-forest-management/forests-and-the-economy.

Natesan, S., C. Armenakis, and U. Vepakomma. 2019. "Resnet-Based Tree Species Classification Using UAV Images." *ISPRS—International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 4213: 475–481. https://doi.org/10.5194/isprs-archives-XLII-2-W13-475-2019.

Nevalainen, O., E. Honkavaara, S. Tuominen, et al. 2017. "Individual Tree Detection and Classification With UAV-Based Photogrammetric Point Clouds and Hyperspectral Imaging." *Remote Sensing* 9, no. 3: 185. https://doi.org/10.3390/rs9030185.

Oliveira, L. F., A. P. Moreira, and M. F. Silva. 2021. "Advances in Forest Robotics: A State-of-the-Art Survey." *Robotics* 10, no. 2: 53.

Oliveira, R. A., R. Näsi, O. Niemeläinen, et al. 2020. "Machine Learning Estimators for the Quantity and Quality of Grass Swards Used for Silage Production Using Drone-Based Imaging Spectrometry and Photogrammetry." *Remote Sensing of Environment* 246: 111830. https://doi.org/10.1016/j.rse.2020.111830.

Ouattara, I., H. Hyyti, and A. Visala. 2020. "Drone Based Mapping and Identification of Young Spruce Stand for Semiautonomous Cleaning." *IFAC-PapersOnLine* 53, no. 2: 15777–15783.

Palonen, T., H. Hyyti, and A. Visala. 2017. "Augmented Reality in Forest Machine Cabin." *IFAC-PapersOnLine* 50, no. 1: 5410–5417.

Paszke, A., S. Gross, and F. Massa, et al. 2019. "Pytorch: An Imperative Style, High-Performance Deep Learning Library." In *Advances in Neural Information Processing Systems*, 8024–8035. Association for Computing Machinery (acm.org). https://proceedings.neurips.cc/.

Pearse, G. D., A. Y. S. Tan, M. S. Watt, M. O. Franz, and J. P. Dash. 2020. "Detecting and Mapping Tree Seedlings in UAV Imagery Using Convolutional Neural Networks and Field-Verified Data." *ISPRS Journal of Photogrammetry and Remote Sensing* 168: 156–169. https://doi.org/10.1016/j.isprsjprs.2020.08.005.

Pizzagalli, S. L., Y. Bondarenko, and B. C. Baykara, et al. 2022. *Forestry Crane Immersive User Interface for Control and Teleoperation (Volume 2B: Advanced Manufacturing)*. ASME (The American Society of Mechanical Engineers). https://doi.org/10.1115/IMECE2022-94975.

Qiao, Z., Z. Yu, B. Jiang, H. Yin, and S. Shen. 2024. "G3reg: Pyramid Graph-Based Global Registration Using Gaussian Ellipsoid Model." *IEEE Transactions on Automation Science and Engineering* 22: 1–17.

Raviraj, A., M. Johenneken, A. Drak, A. Asteroth, and S. Houben. 2024. "Individual Plant Detection on Post-Harvest Forest Floor Using Aerial Imagery." In *2024 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 1–6. IEEE. https://doi.org/10.23919/SoftCOM62040.2024.10721958.

Ren, S., K. He, R. Girshick, and J. Sun. 2017. "Faster R-CNN: Towards Real-Time Object Detection With Region Proposal Networks." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, no. 6: 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031.

Rusu, R. B., and S. Cousins. 2011. "3D Is Here: Point Cloud Library (PCL)." In *2011 IEEE International Conference on Robotics and Automation*, 1–4. IEEE. https://doi.org/10.1109/ICRA.2011.5980567.

Santana-Fernández, J., J. Gómez-Gil, and L. Del-Pozo-San-Cirilo. 2010. "Design and Implementation of a GPS Guidance System for Agricultural Tractors Using Augmented Reality Technology." *Sensors* 10, no. 11: 10435–10447. https://doi.org/10.3390/s101110435.

Segal, A. V., D. Hähnel, and S. Thrun. 2009. "Generalized-ICP." In *Robotics: Science and Systems*. https://api.semanticscholar.org/CorpusID:231748613.

Sitompul, T. A., and M. Wallmyr. 2019. "Using Augmented Reality to Improve Productivity and Safety for Heavy Machinery Operators: State of the Art." In *17th International Conference on Virtual–Reality Continuum and Its Applications in Industry*. Association for Computing Machinery (acm.org). https://doi.org/10.1145/3359997.3365689.

Szegedy, C., V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. 2016. "Rethinking the Inception Architecture for Computer Vision." In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2818–2826. IEEE. https://doi.org/10.1109/CVPR.2016.308.

Thiel, C., and C. Schmullius. 2016. "Comparison of UAV Photograph-Based and Airborne Lidar-Based Point Clouds Over Forest From a Forestry Application Perspective." *International Journal of Remote Sensing* 38: 1–16. https://doi.org/10.1080/01431161.2016.1225181.

Tkachenko, M., M. Malyuk, A. Holmanyuk, and N. Liubimov. 2020–2022. *Label Studio: Data Labeling Software*. Open Source Software. https://github.com/heartexlabs/label-studio.

Uotila, K., J. Miina, T. Saksa, R. Store, K. Kärkkäinen, and M. Härkönen. 2020. "Low Cost Prediction of Time Consumption for Pre-Commercial Thinning in Finland." *Silva Fennica* 54, no. 1: 10196. https://doi.org/10.14214/sf.10196.

Uotila, K., T. Saksa, J. Rantala, and N. Kiljunen. 2014. "Labour Consumption Models Applied to Motor-Manual Pre-Commercial Thinning in Finland." *Silva Fennica* 48: 982. https://doi.org/10.14214/sf.982.

Vestlund, K., and T. Hellström. 2006. "Requirements and System Design for a Robot Performing Selective Cleaning in Young Forest Stands." *Journal of Terramechanics* 43, no. 4: 505–525. https://doi.org/10.1016/j.jterra.2005.07.001.

Vihlman, M., H. Hyyti, J. Kalmari, and A. Visala. 2015. "Detection and Species Classification of Young Trees Using Machine Perception for a Semi-Autonomous Forest Machine." In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 1543–1548. IEEE. https://doi.org/10.1109/ICRA.2015.7139394.

Xu, Z., X. Shen, L. Cao, et al. 2020. "Tree Species Classification Using UAS-Based Digital Aerial Photogrammetry Point Clouds and Multispectral Imageries in Subtropical Natural Forests." *International Journal of Applied Earth Observation and Geoinformation* 92: 102173. https://doi.org/10.1016/j.jag.2020.102173.

Yang, H., J. Shi, and L. Carlone. 2021. "Teaser: Fast and Certifiable Point Cloud Registration." *IEEE Transactions on Robotics* 37, no. 2: 314–333. https://doi.org/10.1109/TRO.8860.

Yin, P., S. Yuan, H. Cao, X. Ji, S. Zhang, and L. Xie. 2023. "Segregator: Global Point Cloud Registration With Semantic and Geometric Cues." In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2848–2854. IEEE.

Yuan, C., J. Lin, Z. Zou, X. Hong, and F. Zhang. 2023. "STD: Stable Triangle Descriptor for 3D Place Recognition." In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 1897–1903. IEEE.

Zhang, X., J. Yang, S. Zhang, and Y. Zhang. 2023. "3D Registration With Maximal Cliques." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 17745–17754. IEEE.

Zhang, J., Y. Yao, and B. Deng. 2022. "Fast and Robust Iterative Closest Point." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, no. 7: 3450–3466.

Zhang, C., Y. Zhou, and F. Qiu. 2015. "Individual Tree Segmentation From Lidar Point Clouds for Urban Forest Inventory." *Remote Sensing* 7, no. 6: 7892–7913. https://doi.org/10.3390/rs70607892.