Multi-tree woody structure reconstruction from mobile terrestrial laser scanner point clouds based on a dual neighbourhood connectivity graph algorithm

Valeriano Méndez[a,d,e] , Joan R. Rosell-Polo[b], Miquel Pascual[c], Alexandre Escolà[b]

[a] Department of Applied Mathematics. Polytechnic University of Madrid. Ciudad Universitaria, s/n, 28040 Madrid, Spain.

[b] Research Group on AgroICT & Precision Agriculture. Department of Agricultural and Forest Engineering. University of Lleida - Agrotecnio Center: Av. Rovira Roure, 191, 25198 Lleida, Spain.

[c] Department of Horticulture, Fruit growing, Botany and Gardening. University of Lleida. Rovira Roure, 191, 25198 Lleida, Spain.

[d] Corresponding author. Tel.: +34 917 308 355. E-mail: valeriano.mendez@upm.es

[e] Proofs correspondence. Valeriano Méndez. Department of Applied Mathematics. E.T.S. Ingenieros Agrónomos. Polytechnic University of Madrid. Ciudad Universitaria, s/n, 28040 Madrid, Spain.

ABSTRACT

A process is presented for the vector reconstruction of fruit plantations based on the model developed by Verroust and Lazarus. To solve occlusion problems, the use of a dual graph of local and extended connectivity is proposed. The process allows vegetation variables such as the length and volume of the ligneous structure to be measured, enabling studies such as intensity of pruning operations. The process has been tested against simulated models and real trees with different training systems: open-vase system (peach trees) and central leader hedgerow system (pear trees). The cost of the algorithm will be given by the cost of the implementation of Dijkstra's algorithm, which in its standard version is of potential ($O(n^2)$). Algorithm accuracy was checked against point clouds of virtual trees. The reconstruction was also applied before and after a pruning operation of real trees to enable a study of the evolution of the vegetation indices. Results showed the algorithm to be suitable for multi-tree reconstruction of both central leader and open-vase training systems.

KEYWORDS

Multi-tree reconstruction; LiDAR; mobile terrestrial laser scanner; point cloud, tree training, ligneous structure.

Variable	***Description***
b	Number of sets in a branch.
c	Centroid of a group of points, coordinates $\begin{pmatrix} x_c & y_c & z_c \end{pmatrix}$

d_e	Maximum extended distance over which neighbour points are selected.
d_l	Maximum local distance over which neighbour points are selected.
E	Cloud of enveloping points to each branch of a virtual tree. The point cloud is obtained from an enveloping mesh on the cylindrical surfaces so that there are no occlusions in the cloud.
G_s	Geodesic graph of tree s
$GD_{s,i}$	Geodesic distance from point P_i to root r_s
HMT	Hidden Markov Tree
k	Maximum number of k-level sets in which the cloud points are grouped.
KPI	Key performance indicator
LS	SimLidar-obtained cloud with simulation of lateral scan of a virtual tree
m	Total number of trees
M	Connectivity matrix
md_s	Maximum geodesic distance to root r_s
MTLS	Mobile terrestrial laser scanner
n	Total number of points in cloud
nb	Total number of branches in reconstructed model
NB_e	Extended neighbourhood graph obtained with d_e
NB_l	Local neighbourhood graph obtained with d_l
$\vec{n}_i^j(t)$	Surface area which encloses the 3D reconstructed branch object
PC	Point cloud
P_i	Individual point of cloud
p	Total number of points in a given set
q	Number of sections in which the total md_s geodesic distance is divided
θ_k	Polar angle which defines a spherical sector to select the closest point at a distance smaller than d_l or d_e
φ_p	Azimuth angle which defines a spherical sector to select the closest point at a distance shorter than d_l or d_e
r_s	Point of the base of the tree s which is taken as root of the geodesic graph.
$\vec{r}(t)$	Piecewise polynomial curve which defines the axis of a branch

r	Radius
rd	Minimum radius
ru	Maximum radius
s_i	Set of points
t	Linear parameter in $\vec{r}(t)$ used for least squares fit of the radii distribution
TLS	Terrestrial laser scanner

1 INTRODUCTION

The geometric reconstruction of a tree is fundamental for a detailed analysis of its structure. Using massive data support with information about geometry, measurements can be made of direct (leaf area, canopy volume or wood volume) and indirect tree vegetation parameters (LAI, leaf density, canopy permeability or radiation interception), which provide information about the productive characteristics of trees related to their shape and structure. The direct use of rasterised information or image analysis, from photographs for example, can allow obtaining some of these parameters (Phattaralerphong and Sinoquet, 2007). The vector reconstruction of the geometry of the tree provides support for these objectives and lays the foundation for the implementation of virtual construction models, such as the use of the statistical framework of the hidden Markov tree (HMT) model introduced by Crouse et al. (1998) and used to undertake realistic constructions of apple trees by Durand et al. (2005) and Costes et al. (2008).

In parallel, with the use of massive data provided by photogrammetry or airborne laser scanning (ALS) for tree detection and general parameter estimation, geometry at individual tree level has been studied using two main approaches. The first comprises the use of digital photographs (Shlyakhter et al., 2001; Mizoue and Masutani, 2003; Phattaralerphong and Sinoquet, 2005 and 2007; Tan et al., 2008). Image information is processed to determine the existence of vegetation and, based on sensor parameters (horizontal distance from camera to tree and tree height), a projection is made onto a voxel space through which the crown volume and leaf area are estimated (Phattaralerphong and Sinoquet, 2007). The use of a smaller voxel size to increase precision dramatically increases running time.

The second approach involves the use of a terrestrial LiDAR system or terrestrial laser scanners (TLS), which allows dense point clouds to be obtained from which a detailed description of the geometry can be extracted. Detection of the woody geometry from the TLS was considered by Simonse et al. (2003) using Hough transforms, while Gorte and Winterhalder (2004) and Gorte and Pfeifer (2004) generated the topology of the skeleton from a voxel space. The use of a TIN (triangulated irregular network) to obtain vector information about the ligneous structure of a tree is limited as a result of the presence of a large number of small branches (Fig. 1). Pfeifer et al. (2004) and Méndez et al. (2014) obtained a model of the scaffold branches and stems from a cylinder fit. Other mixed methods, which combine scanner data with high resolution image-obtained texture information, have been proposed by Reulke and Haala (2005). ICP (Iterative Closest Point) algorithms have also been employed, used to minimise the difference between two point clouds. The algorithm iteratively revises the rotations and translations required to minimise the distance between the points of a cloud with respect to another cloud taken as reference. The ICP algorithms have been used to register point clouds, i.e. fit the orientations obtained in different scans (Besl and McKay, 1992; Henning and Radtke, 2006). Pfeifer et al. (2004) used cylinders in a kind of a following-the-line approach to do the reconstruction. Hackenberg et al. (2015) used a similar approach but changing the cylinders to spheres. In Raumonen et al. (2013), "the model is constructed by a local approach in which the point cloud is covered with small sets corresponding to connected surface patches in the tree surface".

Figure 1 should be placed here

Assigning the point of a cloud obtained with the TLS to the different components of the plant is easy in the case of the trunk and scaffold branches. However, when it comes to the higher order branches, particularly the shoots, assigning a particular cloud point to a particular object can be a tricky business. Neighbourhood graphs, geodesic graphs and different cluster balancing algorithms are used to obtain the skeleton of the tree together with the radius of each branch. Searches for close points to construct the neighbourhood graphs are kd-tree based. Verroust and Lazarus (2000) generated the skeleton of the tree from the neighbourhood graph, geodesic graph and k-levels set. Verroust and Lazarus (2000) implements a Dijkstra's algorithm (1959) from a point-proximity neighbour graph to get the geodesic graph and using the geodesic distance to the root of the tree the points are separated in k-levels set, finally the sets fit the cylinders of branches. Yan et al. (2009), from a kd-tree structure, applied Lloyd's iteration (1982) to undertake segmentation of the cloud in clusters which are reconstructed in cylinders. Delagrange and Rochon (2011) used the model of Verroust and Lazarus (2000) to obtain the skeleton framework and, selecting centroids in the skeleton, applied a clustering process to group together the points pertaining to each branch. Runions et al. (2007) and Preuksakarn et al. (2010) used a space colonisation algorithm (SCA), which is initiated with a seed point and advances by adding points according to a normalised surrounding point's minimum distance, as a clustering function.

The method employed by Verroust and Lazarus is relatively stable and not as dependent on configuration parameter values and point cloud shape compared to the method of Pfeifer et al. (2004) which requires fitted parameters as described in the study by Méndez et al. (2014). Even so, the quality of the point cloud, as a result of precision related and laser scanner operational problems, as well as tree part occlusions, has an important impact on the quality of the final result. Essentially, the point clouds obtained are affected by various error sources associated with measurements carried out using LiDAR systems: ranging and angular LiDAR accuracy, tree part occlusions, the mixed-pixels phenomenon (partial impacts of the laser beam on different parts of the objects), LiDAR alignment and aiming errors, positioning and georeferencing system and inertial system errors, vibrations of the LiDAR-vehicle combination (Sanz et al., 2011a; Lichti and Skaloud, 2010), etc. The method will therefore not always converge to the real solution. Côté et al. 2009 implements a woody material reconstruction based in Verroust and Lazarus (2000) where the foliage are added using L-System productions.

This present work offers a variation on Dijkstra's algorithm (1959) which reduces occlusion problems in a point cloud obtained by mobile terrestrial laser scanning (MTLS) consisting of using a dual (local and extended) neighbourhood connectivity graph. The process allows vegetative variables such as the length and volume of the ligneous structure in fruit orchards to be measured from 3D point clouds generated by MTLS. The first step is to determine the skeleton of the tree to subsequently adjust cylinders to it. In our algorithm, the model surface is obtained at the end of the process once the 3D skeleton is determined. Other previous methods such as those developed by Pfeifer et al. (2004), Hackenberg et al. (2015) and Raumonen et al. (2013) use different approaches. The results can be directly applied in the objective and quantifiable evaluation of the intensity of pruning operations (Sun et al. 2006). Indirectly, the results of the algorithm could be used in the generation of decision support systems for pruning operation and even in the automation of such operations.

The algorithm has been tested against simulated models and against real trees with different training systems. In a first case, the reconstruction is presented of an isolated tree with open-vase training (peach tree, *Prunus persica* (L.) Batsch). A second case involves the reconstruction of a single tree in a row of central-leader trained pear trees (*Pyrus communis* L.), while a third case deals with the multi-tree reconstruction of various individuals in the tree row. The algorithm also returns the vegetative measurements distributed according to branch order following the terminology proposed by De Reffye et al. (1988).

In this way, the aim of the present study is to implement the Verroust and Lazarus method, introducing the novel use of a dual matrix of connectivity in Dijkstra's algorithm (1959), and test its suitability in the reconstruction of ligneous structures of commercially grown orchards. The use of the dual matrix of connectivity allows working with compact point subsets at local level as well as the interconnection of separated subsets due to occlusions of objects situated on a plane closer to the sensor, for example. An analysis is also undertaken of the feasibility of obtaining vegetation indices of interest for the agronomic analysis of the orchard trees. Three vegetation indices are implemented: number of terminal apices, branch length and wood volume. Testing is undertaken of whether the estimation of the obtained indices is realistic or not. These vegetation indices are used as key performance indicators (KPI) for the validation of the reconstructed models.

2 MATERIALS AND METHODS

The first step of the present work comprises testing of the algorithm for a complex but simulated (Méndez et al., 2013) formation. A direct point cloud was obtained of a cylindrical structure of a tree with abundant branching. As a difference to point clouds obtained with MTLS, the simulated cloud presented no noise and no occlusions. Nonetheless, the problem of indeterminacy was evident in fine and close neighbouring branches. The total number of terminal apices and the total length and volume of the branches obtained in the reconstruction (the KPI) were compared to the corresponding values for the simulated model, being used as goodness-of-fit measures of the reconstructions.

In the following step, cloud points were obtained from real MTLS operations, considering one side and both sides of the tree row. These scans were performed before and after a tree pruning process. Direct test of the goodness-of-fit of the estimations was performed by comparing the difference in branch volume, before and after pruning, against the mass of pruned wood. The reconstruction method used was the one proposed by Verroust and Lazarus (2000), comprising the construction of a series of graphs: Neighbourhood - NB, Geodesic - G, Level Sets - L and Skeleton - K.

2.1 Neighbourhood Graph – NB

The neighbourhood graph of a point cloud $PC=\{P_i, con\, i=1\ldots n\}$ relates each point with all the points with which it is connected. The employment of a tetrahedralisation using the conditions of Delaunay allows the optimum graph connecting each point with the minimum number of possible neighbours to be constructed. However, the high cost of processing a tetrahedralisation has resulted in the use of alternatives which lead to analogous results but at lower computational cost. Generally, tetrahedralisation is replaced with a neighbourhood graph in which all the points $P_j \in PC$ given $\|P_i P_j\| < d_j$ will be neighbours of a point P_i. Delagrange et al. (2014) proposed the suitability of this approach to enhance the density of the graph that is acquired. A graph of higher density implies a higher cost in obtaining the geodesic graph, with minimum cost when tetrahedralisation is used since the point connectivity based in tetrahedron edges is optimal. The edges obtained by tetrahedralisation are minimum in number, although the cost of the process is high. In the present study, the approach of Verroust and Lazarus has been followed, selecting the points P_i at a minimum distance d_j within a $k*p$ sector of the sphere $\{P_i, d_j\}$ in intervals of the polar angle ($0<\theta_k \leq 2\pi$) and azimuth angle ($\frac{-\pi}{2} < \varphi_p \leq \frac{\pi}{2}$). The neighbourhood graph thus obtained will be seen as $(0, d_l)$ = {(P_i, P_j)} $\forall P_i$ fulfilling the condition $0 \leq \|P_i P_j\| < d_l$. The classical implementation of a neighbourhood graph of a cloud of n points is a matrix M, such that from point i of the cloud comes a connection to j if $M_{ij}=1$ and there will be no connection when $M_{ij}=0$. The connection will be bidirectional when $M_{ij}=M_{ji} \forall i, j$, as happens in our case. Given that the “1” values in the matrix are limited, a n-dimensional vector structure can be used to store the effective connections as an alternative to the matrix n^2.

As reported by Delagrange et al. (2014), the choice of d_j has an impact on the quality of the reconstruction. Obtaining an accurate reconstruction depends initially on the quality of the point cloud. Problems such as branch occlusion or the accuracy of the scanner itself affect the quality of the reconstruction. But even when starting with an ideal homogenous cloud, without occlusions or accuracy problems, such a ramified tree model will lead to a different result in the reconstruction depending on the chosen value of d_l. Choosing a low value facilitates a detailed reconstruction of the small branches of the tree at the cost of leaving isolated point subsets without reconstruction when gaps are found as a result of scanner inaccuracy or occlusions. To avoid this problem, the use is proposed of two neighbourhood graphs $NB_j(0, d_l)$ and $NB_e(d_l, d_e)$, with the condition that $d_l < d_e$. NB_j will be used to obtain the geodesic graph applying Dijkstra's algorithm (see Table 1), while NB_e will be used to connect subsets of points isolated with respect to NB_j.

2.2 Geodesic Graph – G

The geodesic graph of a point cloud defines the minimum distance from each point to a vertex chosen as origin or root, using a hierarchical path from a vertex to its predecessor. So, for each vertex, the minimum distance to the root is defined as well as the predecessor vertex which is used to arrive at that root. The root vertex will have no predecessor and will have a distance zero. An isolated vertex will similarly have no predecessor and its distance to the root will be ∞.

In order to obtain NB_j, a multi-tree option of Dijkstra's algorithm has been implemented, which consists of calculating the geodesic graph of each point to the base (root, r_s, with $s=1,\ldots,m$) for each of the m trees. The choice of each r_s is made by taking the vertex of lowest z coordinate value in the region where each tree is found. In practice, the axes on which the roots go are defined as input parameters so that they are located in the appropriate place. Introducing these parameters is facilitated by a graphical assistant on the original point cloud. Therefore, the reconstruction of m trees requires m geodesic graphs to be obtained, which will be written as G_s with $s=1,\ldots,m$.

Dijkstra's algorithm (Table 1) begins with marking the chosen root r_s as treated, assigning to it a zero value as the geodesic distance (line 2), and, finally, relaxing it. The relaxation consists of setting the distance to the root of all untreated points adjacent to r_s. Given P_i, if P_j is adjacent and untreated, the sum of the geodesic distance from P_i plus the Euclidean distance between both points ($GD_{s,i}+\|P_i P_j\|$) is stored as the geodesic distance to r_s ($GD_{s,j}$), provided the resulting value is lower than the previous value of $GD_{s,j}$. The algorithm iterates searching for the closest untreated point (line 6), which is subsequently marked as treated and then relaxed. The loop ends when there are no untreated points left.

Table 1 should be placed here

The search for a close point to a given point (in the MinimumVertex function) requires the point to be connected by graph NB_j with a treated point. The existence of a totally treated graph is a necessary condition to enable the calculation of the geodesic distance of all points of the cloud to the root. A d_l value high enough is all that is needed for this to happen.

However, in models with many concave elements, as occurs in ramified structures such as trees, choosing a high value of d_j causes the reconstruction to merge neighbouring branches, joining them through the concavities that separate them.

To minimise this problem, a variation is considered of Dijkstra's algorithm with the use of a dual neighbourhood graph. On the one hand, a local neighbourhood graph, NB_l, is employed with a similar use to that considered in the standard algorithm. NB_l allows to construct the geodesic graph within isolated sets of neighbourhood graph formed by interconnected points (Fig. 2). On the other hand, an extended neighbourhood graph, NB_e, constructed with a longer distance, d_e, is used to connect the different islands, thereby forming a complete graph.

Figure 2 should be placed here

The algorithm is fitted so that, when a connected minimum point is not found in graph NB_l, a search is made for a minimum vertex through graph NB_e(using the MinimumExtVertex function). This function searches for an untreated and connected point in NB_e to a treated point with the shortest distance to the root. Once this minimum extended point is found (Fig. 1), its geodesic distance is stored (SetDistanceMinExt function), reported as treated and relaxed. A view of the resulting new code for the algorithm is shown in Table 2.

Table 2 should be placed here

2.3 Level Sets – L

The geodesic graph allows a classification of the vertices to be made in levels. Knowing the geodesic distance (md_s) of the most distant point from r_s, all the points of the geodesic graph are distributed according to classes of the geodesic distance. If the maximum number of classes is k, a point P_i will belong to the class l if its geodesic distance is $\frac{l}{k}\cdot md_s \leq GD_{s,i} < \frac{l-1}{k}\cdot md_s$.

Using the neighbourhood graph along with the geodesic graph and the above distribution of classes, it is possible to define sets or groups of homogenous points with a hierarchic structure which will serve to define the branches of the tree. Each set will be formed by P_i points which are in the same class l and which are interconnected according to NB_j.

Each set is formed by all those points that are neighbours and of equal class. The centroid of each set can be obtained as $c=\left(x_c \quad y_c \quad z_c\right)$, where $x_c=\frac{\sum x_i}{p}$, $y_c=\frac{\sum y_i}{p}$ and $z_c=\frac{\sum z_i}{p}$, where p is the total number of points in a given set. Lloyd's iteration can be applied to the group of points thus formed. Lloyd's iteration aims at distributing the points of the initial cloud among the sets such that the distance from the points to the centroids is a minimum. Lloyd's iteration is restricted to points belonging to the same tree.

In a class l there will generally be one or more sets. More than one set will be formed when there are vertices which are locally interconnected but disconnected from the rest. Sets of different class may be hierarchically interconnected, a hierarchy which is inherited from the existing hierarchy between the vertices in the geodesic graph. There will be a set to which the root vertex belongs which is predecessor to the rest and which is the only one which has no predecessor. The rest of the sets will all have a predecessor. The set which contains the preceding vertex with minimum geodesic distance will be taken as the predecessor set to a given set (Table 3).

The choice of the value of k in the formation of the sets, along with the choice of d_j in the formation of the neighbourhood graph, has a considerable impact on the quality of the reconstruction. High values of k allow more accurate reconstruction of the smaller-sized terminal branches and more detailed ramifications. If the geodesic distance of a set ($\frac{md_s}{k}$) is smaller than the trunk radius, a unique section of trunk become rebuilt in different sets. Then, false branches appear around the kernel of the trunk. We substitute a fixed k value by a series of values $\left\{k_1, k_2, \ldots, k_q\right\}$. The total md_s geodesic distance is divided in q sections, and section i is divided in k_i sets. For example {1, 2, 4} divides the tree into 3 sections, creating 1, 2 and 4 sets in each one.

Table 3 should be placed here

2.4 Definition of branches and branch axes

Each branch is defined as the surface of revolution over a smooth curve (B-spline) which fits over the sequence of centroids of the sets which make up the branch. Given a branch B composed of the sets $\left\{s_1, s_2, \ldots, s_b\right\}$, the respective centroids $\left\{c_1, \cdots, c_i, \ldots, c_b\right\}$

are used to build a polynomial piecewise curve $\{\vec{r}_1(t),\cdots,\vec{r}_i(t),\ldots,\vec{r}_b(t)\}$. The parameter t is chosen such that $\vec{r}_i(0)=c_i$ and $\vec{r}_i(1)=c_{i+1}$ and so the piecewise curve must meet the following conditions:

- $\vec{r}_i(1)=\vec{r}_{i+1}(0)=c_{i+1}$
- $\vec{r}_i^{\,'}(1)=\vec{r}_{i+1}^{\,'}(0)$
- $\vec{r}_i^{\,''}(1)=\vec{r}_{i+1}^{\,''}(0)$
- $\vec{r}_1^{\,''}(1)=\vec{r}_b^{\,''}(0)=0$

The obtained curve $\vec{r}_i(t)$, is the axis of the branch. The branch will be reconstructed as a surface of revolution with the vector $\vec{r}_i(t)+\vec{n}_i^j(t)$ where $\vec{n}_i^j(t)$ is a normal vector $with\ j=1,\ldots,p$ in steps of $\frac{2\pi}{P}$.

It is then determined whether the concatenation of sets which go from the root set to a leaf set (set which is not a predecessor of any other set) is a branch continuation or ramification. In this aspect, the De Reffye's criteria (1988) have been followed to order the ramifications. Order 1 is given to the main trunk where the root is found. If the concatenation of sets does not deviate more than a given angle threshold, then it is considered that there is no change of order, otherwise the order is increased by one unit to the next order.

In a real model, branch radius should decrease with respect to that of the predecessor branch. In the reconstruction, as a result of inaccuracies in the point cloud or because points of different and very small branches are very close to each other, branches may be generated with a radius greater than that of the predecessor branch. In these cases, a debugging algorithm consisting of a deconstruction process is applied to the set s_i by extracting the outsider point (the point furthest from the axis of the initial branch). Subsequently, a new set s_i^* is created containing it. Following this, all the points of s_i which are closer to the centroid of s_i^* than to its own centroid are transferred to s_i^*. This process is iterated until there is no branch left with a radius greater than its predecessor. If the resulting new set is an isolated point, then it is discarded.

2.5 Determination of mean radius

Knowing the direction $d_i=c_{i-1}c_i$ of each set s_i of an axis $\{s_1,s_2,\ldots,s_n\}$, where c_i is the centroid, the mean radius of the points of the class to the axis is determined. For all the points $\{\forall P_j \in s_i\, with\ j=1\ldots q\}$ of the set, the distribution of the q radial distances of each point in the shape $\{(t_j, r_j)\}$ is determined. Using a least squares fit, a straight line can be obtained which defines in each class a minimum radius (rd_i) and a maximum radius (ru_i), where the subscript i refers to the fact that there will be a value in each of the classes which forms the axis which we have fitted with $\vec{r}(t)$. This means that $\vec{r}(t)$, rd_i and ru_i form a trunk-cylinder curve which defines the axis. For the sake of simplicity, and given that the result is neutral in axis volume calculations, a mean radius value of ($\frac{rd_i+ru_i}{2}$) is taken.

2.6 Processing cost of the algorithm

A potential process limit was estimated in order to establish the algorithm's processing cost. The upper limit of growth (O) of each function is estimated in Table 4 according to the total number of cloud points (n), total number of trees (m) and total number of branches (nb). The cost of generating the kd-tree is $O(n*\log(n))$ (Cormen et al., 2009). The creation of the neighbourhood matrix requires access to the kd-tree structure with a cost $O(\log(n))$ (Cormen et al., 2009) for each point, and so the total cost will also be $O(n*\log(n))$. The cost of Dijkstra's algorithm in its standard construction is $O(n^2)$ (Leyzorek et al., 1957) in its standard implementation. When a minimum point is not found in the local matrix (MinimumVertex function), a complementary search needs to be made in the extended matrix (MinimumExtVertex) and the geodesic distances assigned of the new minimum point (SetDistanceMinExt). In its standard version, the MinimumExtVertex function will have a cost complexity $O(n)$, while SetDistanceMinExt will have a lower complexity that we can estimate in $O(\log(n))$. It can be concluded that the complexity of the proposed variation maintains the initial complexity of Dijkstra's algorithm, $O(n^2)$. When implementing the multi-tree algorithm, the maximum computational cost is $O(m*n^2)$, where m is the number of trees. As m is very small compared with the number of points, a final cost can be taken of $O(n^2)$. The process of generation of sets from the geodesic graph has a cost $O(n)$, while the clustering algorithm employed (Méndez et al., 2014) has a cost $O(n*\log(nb))$. Finally, the process of obtaining the branches from the Level Sets graph is $O(nb*\log(nb))$. In conclusion, the proposed algorithm has a processing cost $O(n^2)$.

Table 4 should be placed here

2.7 Test against simulated tree and cloud

The algorithm was tested against a virtual model. An apple tree model was obtained using a HMT model (Méndez et al., 2013). The different internode structures evolve according to a probability matrix (Durand et al. 2005, p. 818), which allows realistic models of trees to be obtained. A point cloud is extracted from the virtual apple tree, which permits an absence of noise and occlusions. The point cloud is obtained selecting a mesh or envelope over the cylindrical surface. In this point cloud, it is possible to verify that the choice of the dual matrix of connectivity does not affect the result of the reconstruction due to the non-existence of occlusions.

Simulated MTLS operations were also obtained (Méndez et al., 2012 & 2013) with a guarantee of the non-existence of noise, but not of occlusions. In these simulations, the extended connectivity matrix allows isolated point subsets to be reconstructed.

A leafless virtual apple tree is chosen of sufficient complexity to include small size branches (shoots). The number of terminal apices is known, as well as total branch length and volume, when the virtual apple tree is generated. These indices are compared with those generated in the reconstruction and are used as KPI to validate the model since they may be relatively easy to measure in field conditions. The simulated apple tree can generate small overlapping branches, which implies indeterminacy when obtaining the reconstruction. Two small and very close branches cannot be accurately differentiated, which will mean the same number of terminal apices as in the original model will not always be reconstructed. The total length of the estimated branches is derived from the reconstructed apices and this, together with the diameter, enables an evaluation of the volume.

2.8 Testing against real models

The present study was based on measurements obtained with an MTLS of various pear and peach fruit trees which were central-leader and open-vase trained, respectively, as is common practice in commercial fruit orchards (see Fig. 3). The measurements were only of the ligneous structure (that is, without leaves or fruits) as they were taken during winter (at plots run by the School of Agrifood and Forestry Science and Engineering of the University of Lleida).

Figure 3 should be placed here

A time-of-flight 2D LiDAR (Light Detection and Ranging) sensor, model UTM30-LX-EW (HOKUYO, Osaka, Japan) was used to scan the pear and peach orchards. The LiDAR has a range of 30 m, a scanning window of 270º with an angular resolution of 0.25º, providing 1081 first-return signal measurements per scan at a scanning frequency of 40 Hz, which results in more than 43,000 points s^{-1}. It also has multi-return capabilities, providing up to 3 distance measurements associated with partial impacts of the same emitted laser pulse on different objects (Escolà et al., 2014 & 2015). An RTK GPS 1200+ receiver (Leica geosystems AG, Heerbrugg, Switzerland) was used to geolocate the LiDAR sensor to subsequently geolocate the measurement points. Additionally, the LiDAR sensor was mounted on a gimbal to dynamically stabilise it in a horizontal position. The MTLS scanned each side of specific row sections, including various trees with no leaves, before and after pruning. After data processing, several 3D georeferenced point clouds were obtained from the sampled row sections: a point cloud obtained when scanning from the right hand side of the row, a second point cloud obtained when scanning from the left hand side, and a third point cloud fusing both previous point clouds. A reconstruction of the ligneous structure was undertaken, separating the ligneous formation into individual trees.

The selection of an extended matrix allows the extension of the reconstruction to point subsets of the cloud which would otherwise remain isolated.

3 RESULTS AND DISCUSSION

3.1 Test of algorithm performance

A test was performed using cylinder-based models of virtual trees with different branching degrees (4, 11, 24 and 92 terminal apices). Taking points directly from the enveloping surfaces of the cylinders (E) generates a point cloud without occlusions, as shown in Fig. 4 (top). Besides, another point cloud is generated by simulating the performance of a one-sided MTLS lateral scan (LS) where occlusion problems appear.

Figure 4 should be placed here

The number of generated terminal apices, the length and total volume of the branches are used as KPI for reconstruction validation purposes. Reconstruction from the cylinder enveloping points enables testing the correct implementation of the model of Verroust and Lazarus (2000). The point clouds obtained with simulated LS allow the effect of the use of a dual connectivity to be verified in the case of occlusions. The results obtained are shown in Table 5. The enveloping point clouds allow us to conclude that, without occlusion problems, the reconstruction provides good results in terms of identification of the number of free apices and the total branch length.

Table 5 should be placed here

However, branch volume, which is affected by diameter estimation, shows high levels of discrepancy. The reconstructed volumes are systematically lower than reference virtual tree volumes. The clustering applied with the Lloyd's iteration compacts the point sets, tending to give volume underestimation. Additionally, when there is a large number of shoots, they are superposed and the probability to assign points to wrong branches is high. Furthermore, the error in the volume estimation may be of importance since wrong assignments can greatly affect shoot diameters, which are originally small. Moreover, diameter mis-estimations are quadratic in the volume calculation. That is why a debugging process was implemented to ensure that all branches have smaller diameters than their predecessors. When the debugging process is applied, the reconstructed model tends towards reality. Despite this, the reconstructed volumes can be used as relative values (i.e. qualitatively) in pruning operations or fertilisation activities, or to estimate potential yield, allowing relative comparison studies in a first stage. On the other hand, future efforts will be devoted to refine the developed method in order to improve the accuracy of the computed volume of reconstructed trees.

The reconstruction process requires handling of a series of parameters which, *a priori*, are unknown. The possible contrast with the virtual model enables their determination. The basic tuning parameters are the number of sets and the intervals of connectivity.

The way to calibrate the number of sets is to start from a value and reduce it if the internodes are seen to clump together or are easily resolved. In the trunk area it is better to use a low number of sets, while in smaller branches, a high number of sets allow more detailed reconstructions. The local connectivity interval is adjusted considering the point cloud density (related to the scanning settings), whereas the extended connectivity interval is found increasing the local parameter value to avoid occlusion until a value that permits the reconstruction of all the point cloud. Firstly, a reconstruction is performed using the same interval for local and extended connectivities. The initial selected value is small and it is increased in subsequent reconstructions until a significant part of the tree is built. Secondly, the local interval value is frozen and subsequent reconstructions are undertaken increasing the extended interval until the tree is totally reconstructed.

It has been observed in the tests that, once the exact number of apices to reconstruct has been attained, further adjustment may cause variations in the resultant number of apices. Transferring this experience to the reconstruction of MTLS-acquired point clouds of real trees, suggests the use of simulated MTLS to virtually calibrate the parameters of the reconstruction.

In the point clouds without occlusions, obtained from the cylinder enveloping meshes, it can be verified that the choice of the dual matrix of connectivity does not affect the result of the reconstruction due to the non-existence of occlusions (Fig. 4 bottom). Only in clouds derived from simulated MTLS with complex and multi-branched virtual models the use of dual connectivity is required. Dual connectivity improves reconstruction without the need to implement a process of occlusion-filling.

In addition, real MTLS measurements were made of different types of tree training systems: open-vase trained peach trees and central-leader trained pear trees. A structure of polylines is extracted from the reconstruction which makes up the skeleton framework shown together with the point cloud in the CloudCompare v2.6.2 software (Girardeau-Montaut, 2006); at their side the cylinder-based reconstructions are also shown (Fig. 5). Finally, the result is shown of the multi-tree reconstruction of a row of five central-leader trained pear trees (Fig. 6). MTLS measurements were made before and after tree pruning. The MTLS operations were also made along one side of the row and along the other, left and right, with separate reconstruction of the plants based on each lateral point cloud. A bilateral reconstruction was also made based on the fusion of the two point clouds. The result is shown in Table 6.

Estimation of the mass of pruned wood of each tree was used as numerical test of the reconstructions. The existence of high overlapping between terminal branches, as has been seen to occur in the simulated models, as well as the typical errors of LiDAR sensor-based systems, cause uncertainty in branch radius estimations. Pruned branch length is also calculated as the difference in total branch length before and after pruning. Given that branch length estimation is more stable, pruned branch length together with a proposed average radius allow the pruned branch mass to be estimated from an estimated density of 0.6 kg dm^{-3} in peach (*Prunus persica*; Meier 2007) and 0.69 kg dm^{-3} in pear (*Pyrus communis*; Meier 2007). Experimentally obtained pruned branch mass values were 1.463 kg and 0.716 kg, for peach and pear trees, respectively.

Figure 5 should be placed here

Figure 6 should be placed here

Table 6 should be placed here

4 CONCLUSIONS

The algorithm that is presented allows reconstruction of multi-tree structures with abundant small-sized branching and occlusions in the point cloud. Accuracy of the algorithm was verified against simulated clouds, and was tested according to the three KPI: the number of terminal apices and total branch length and volume. The fundamental parameters in the reconstruction process are the two connectivity matrix intervals and the distribution of sets.

The use of the dual matrix of connectivity has been shown to favour reconstruction in the case of occlusions in the point cloud. When the distribution of terminal branches shows no clumping together, the obtained KPI indicate a reconstruction of good quality, with reliable measurements of length, volume and total number of apices in the ligneous structure. If terminal branches overlap, the clustering process erroneously assigns points from one apex to the adjacent one. This affects the determination of total branch volume.

The complexity (cost) of the algorithm is of the potential order ($O(n^2)$).

Potential lines for future research that have been identified include optimisation of the algorithm of grouping into highly populated branch formations. Such optimisation would allow the reliable computation of both total branch length and volume. A first approach has been made to the determination of branch order. Further investigations into this aspect will be undertaken in future studies given the great interest in KPI distribution by branch order. We consider these KPI to be a useful tool for following the evolution of a tree over its lifetime with respect to, for example, pruning operations or fertilisation activities, or to estimate potential yield.

Table Captions

- **Table 1**. Implementation of Dijkstra's algorithm for calculation of multi-tree geodesic graph.
- **Table 2**. Implementation of Dijkstra's algorithm for calculation of the multi-tree geodesic graph. Version including management of a matrix of dual connectivity.
- **Table 3**. Implementation of the algorithm to obtain the Predecessor Group.
- **Table 4**. List of functions used in the algorithm with estimated cost, where O is the upper limit of growth of the algorithm time cost with the increase of n (number of points in cloud), nb (number of branches) and m (number of trees).
- **Table 5**. Reconstructions of virtual trees. The point cloud type used is generated from an enveloping mesh on the cylinders without occlusions (E) or from a simulated one-sided MTLS lateral scan with occlusions (LS). Connectivity shows

whether dual connectivity was used (two values shown). Debugging shows whether it was necessary as a result of the detection of branch diameters larger than those of their predecessors.

- **Table 6**. Reconstruction of peach tree (*Prunus persica*) with connectivity matrix 50/300 and k-level sets 16;32 and pear tree (*Pyrus communis*) with connectivity matrix 50/150 and k-level sets 1;4;8;16. Pruned branch volume is estimated from measured branch length and an average radius of 7 mm. For pruned wood mass estimations, a density of 0.6 kg dm^{-3} and 0.69 kg dm^{-3} is considered in peach and pear trees, respectively.

Figure Captions

- **Figure 1**. Construction of the geodesic root from the root vertex to a leaf vertex. The solid arrows represent the oriented geodesic path. In red and thick line the choice of vertices of extended scope.
- **Figure 2**. Local (blue circle) and extended (dashed red circle) neighbourhood graph.
- **Figure 3**. Pictures of the pear (a) and peach (b) trees before (1) and after (2) pruning.
- **Figure 4**. Simulated point cloud (top) and cylinder reconstruction result (bottom).
- **Figure 5**. Views of the point clouds with skeletons generated in the form of polylines and of their respective cylinder-based reconstructions.
- **Figure 6**. Simultaneous reconstruction of a hedgerow of five central-leader trained pear trees.

Acknowledgements

This research was partially funded by the Spanish Ministry of Economy and Competitiveness (projects SAFESPRAY: AGL2010-22304-C04-03 and AGVANCE: AGL2013-48297-C2-2-R).

References

Besl, P. J., & McKay, N. D. (1992). Method for registration of 3-D shapes. In Robotics-DL tentative (pp. 586-606). International Society for Optics and Photonics.

Côté, J. F., Widlowski, J. L., Fournier, R. A., & Verstraete, M. M. (2009). The structural and radiative consistency of three-dimensional tree reconstructions from terrestrial lidar. *Remote Sensing of Environment*, *113*(5), 1067-1081.

Costes, E., Smith, C., Renton, M., Guédon, Y., Prusinkiewicz, & P., Godin, C. (2008). MAppleT: simulation of apple tree development using mixed stochastic and biomechanical models. Functional Plant Biology, 35: 936–950.

Crouse, M.S., Nowak, R.D., Baraniuk, & R.G. (1998). Wavelet-based statistical signal processing using hidden Markov models. Signal Processing, IEEE Transactions on, vol. 46, no. 4, pp. 886-902.

Delagrange, S., & Rochon, P. (2011). Reconstruction and analysis of a deciduous sapling using digital photographs or terrestrial-LiDAR technology. Annals of Botany, 108: 991–1000.

Delagrange, S., Jauvin, Ch., Rochon, P. (2014). PypeTree: A Tool for Reconstructing Tree Perennial Tissues from Point Clouds. Sensors 2014, 14, pp. 4271-4289; doi:10.3390/s140304271

De Reffye, P., Edelin, C., Jaeger, M., & Puech, C. (1988). Plant models faithful to botanical structure and development. Computer Graphics, 22, 151–158.

Delagrange, S., Messier, C., Lechowicz, M. J., & Dizengremel, P. (2004). Physiological, morphological and allocational plasticity in understory deciduous trees: importance of plant size and light availability. Tree physiology, 24(7), 775-784.

Dijkstra, E. W. (1959). A note on two problems in connection with graphs. Numer. Math., vol. 1, no. 1, pp. 269–271.

Durand, J.B., Guédon, Y., Caraglio, Y., & Costes, E. (2005). Analysis of the plant architecture via tree-structured statistical models: the hidden Markov tree models. New Phytologist, 166: 813–825.

Escolà A., Sanz R., Martínez-Casasnovas J.A., Masip J., Sebé F., Arnó J., Gregorio E., Rufat J., Arbonés A., Ribes-Dasi M., Pascual M., Villar J.M., del Moral I., & Rosell-Polo J.R. (2014). Obtaining and mapping relevant characteristics of olive trees canopies using a georeferenced multi-echo mobile terrestrial laser scanner (MTLS). International Conference of Agricultural Engineering AgEng 2014. Zürich (CH).

Escolà, A., Martínez-Casasnovas, J.A., Rufat, J., Arbonés, A., Sanz, R., Sebé, F., Arnó, J.. Masip, J., Pascual, M., Gregorio, E., Ribes-Dasi, M., Villar, J.M., & Rosell-Polo, J.R. (2015). A mobile terrestrial laser scanner for tree crops: point cloud generation, information extraction and validation in an intensive olive orchard. 10th European Conference on Precision Agriculture - ECPA 2015. Tel-Aviv (Israel).

Girardeau-Montaut, D. (2006). Development of CloudCompare, an open source project for comparison and analysis of huge 3D point cloud data and CAD models, available in www.cloudcompare.org.

Gorte, B. (2006). Skeletonization of laser-scanned trees in the 3D raster domain. In Lecture Notes in Geoinformation and Cartography: Innovations in 3D Geo Information Systems. Berlin, Germany: Springer-Verlag, pp. 371–380.

Gorte, B., & Pfeifer, N. (2004). Structuring laser-scanned trees using 3D mathematical morphology. International Archives of Photogrammetry and Remote Sensing, 35(B5), 929-933.

Gorte, B., & Winterhalder, D. (2004). Reconstruction of laser-scanned trees using filter operations in the 3D raster domain. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 36(Part 8), W2.

Hackenberg, J., Spiecker, H., Calders, K., Disney, M., & Raumonen, P. (2015). SimpleTree—An Efficient Open Source Tool to Build Tree Models from TLS Clouds. *Forests*, *6*(11), 4245-4294.

Henning, J.G., & Radtke, P.J. (2006). Detailed stem measurements of standing trees from ground-based scanning lidar. For. Sci. 2006, 52, 67–80.

Leyzorek, M., Gray, R. S., Johnson, A. A., Ladew, W. C., Meaker Jr, S. R., Petry, R. M., & Seitz, R. N. (1957). Investigation of Model Techniques–First Annual Report–6 June 1956–1 July 1957–A Study of Model Techniques for Communication Systems. Case Institute of Technology, Cleveland, Ohio.

Lichti, D., & Skaloud, J. (2010). Registration and Calibration. In Airborne and Terrestrial Laser Scanning. G. Vosselman and H-G Maas, Eds. Dunbeath, Scotland, UK: Whittles Publishing, pp. 83-133.

Lloyd, S. P. (1982). Least squares quantization in PCM. Information Theory, IEEE Transactions on, 28(2), 129-137.

Meier, E. (2007). The Wood Database, available in hhttp://www.wood-database.com/about/.

Méndez, V., Catalán, H., Rosell, J.R., Arnó, J., Sanz, R., & Tarquis, A. (2012). SIMLiDAR – Simulation of LiDAR performance in artificially simulated orchards. Biosystems Engineering, 111(1): 72-82.

Méndez, V., Catalán, H., Rosell, J.R., Arnó, J., & Sanz, R. (2013). LiDAR simulation in modelled orchards to optimise the use of terrestrial laser scanners and derived vegetative measures. Biosystems Engineering, 115, 7-19.

Méndez, V., Rosell-Polo, J.R., Sanz, R., Escolà, A., & Catalán, H. (2014). Deciduous tree reconstruction algorithm based on cylinder fitting from mobile terrestrial laser scanned point clouds. Biosystems Engineering, vol. 124, pp. 78–88. doi:10.1016/j.biosystemseng.2014.06.001.

Mizoue N., & Masutani, T. (2003). Image analysis measure of crown condition, foliage biomass and stem growth relationships of Chamaecyparis obtusa. Forest Ecology and Management 172: , 79–88.

Phattaralerphong J., & Sinoquet, H. (2005). A method for 3D reconstruction of tree crown volume from photographs: assessment with 3D-digitized plants. Tree Physiology 25: , 1229–1242.

Pfeifer N., Gorte B., & Winterhalder, D. (2004). Automatic reconstruction of single trees from terrestrial laser scanner data, ISPRS—Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., vol. 35, pt. B5, pp. 114–119.

Phattaralerphong J., & Sinoquet, H. (2007). Tree analyser: software to compute tree structure parameters from photographs. User manual. PIAF-INRA. http://www2.clermont.inra.fr/piaf/eng/download/download.php.Verroust, A. & Lazarus, F. (2000). Extracting skeletal curves from 3D scattered data. Visual Comput., vol. 16, no. 1, pp. 15–25.

Preuksakarn, C., Boudon, F., Ferraro, P., Durand, J.-B., Nikinmaa, E., & Godin, C. (2010). Reconstructing plant architecture from 3D laser scanner data. In Proceeding of the 6th International Workshop on Functional-Structural Plant Models, FSPM10, University of California, Davis, CA, USA, 14–16.

Raumonen, P., Kaasalainen, M., Åkerblom, M., Kaasalainen, S., Kaartinen, H., Vastaranta, M., Holopainen, M., Disney, M. & Lewis, P. (2013). Fast automatic precision tree models from terrestrial laser scanner data. Remote Sensing, 5(2), 491-520.

Reulke, R., & Haala, N. (2005). Tree species recognition with fuzzy texture parameters. In Combinatorial Image Analysis (pp. 607-620). Springer Berlin Heidelberg.

Runions, A., Lane, B., & Prusinkiewicz, P. (2007). Modeling Trees with a Space Colonization Algorithm. *NPH*, *7*, 63-70.

Sanz, R., Llorens, J., Rosell, J.R., Gregorio, E., & Palacín, J. (2011). Characterisation of the LMS200 laser beam under the influence of blockage surfaces. Influence on 3D scanning of tree orchards. Sensors 11(3), 2751-2772.

Simonse, M., Aschoff, T., Spiecker, H., & Thies, M. (2003). Automatic determination of forest inventory parameters using terrestrial laser scanning. Proceedings of ScandLaser Workshop, 3-4 September 2003, Umea, Sweden, 251-257.

Shlyakhter, I., Rozenoer, M., Dorsey, J., & Teller, S. (2001). Reconstructing 3D tree models from instrumented photographs. *IEEE Computer Graphics and Applications*, 21, 53–61.

Sun, R., Li, W., Tian, Y., & Hua, L. (2006). Automatic identification for standing tree limb pruning. *Frontiers of Forestry in China*, *1*(2), 150-153.

Tan P., Fang T., Xiao J., Zhao P., & Quan L. (2008). Single image tree modeling. ACM Transactions on Graphics 27: Article 108. doi:10.1145/1409060.1409061.

Verroust, A., & Lazarus, F. (2000). Extracting skeletal curves from 3D scattered data. Vis. Comput. 2000, 16, 15–25.

Yan, D. M., Wintz, J., Mourrain, B., Wang, W., Boudon, F., & Godin, C. (2009). Efficient and robust tree model reconstruction from laser scanned data points. In Proceedings of the 11th IEEE International conference on Computer-Aided Design and Computer Graphics (pp. 572-576).

Figure 1. Construction of the geodesic root from the root vertex to a leaf vertex. The solid arrows represent the oriented geodesic path. In red & thick line the choice of vertices of extended scope.

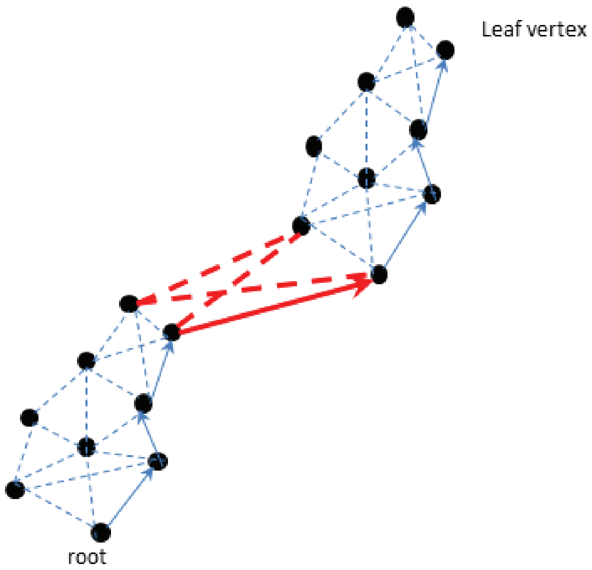

Figure 2. Local (blue circle) and extended (dash & dot red circle) neighbourhood graph.

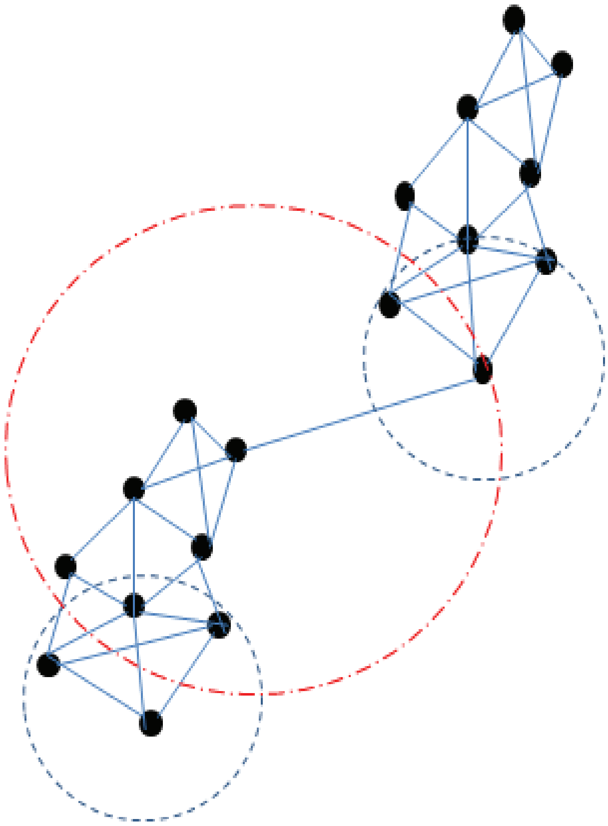

Figure 3. Pictures of the pear (a) and peach (b) trees before (1) and after (2) pruning.

Figure 4. Simulated point cloud (top) and cylinder reconstruction result (bottom).

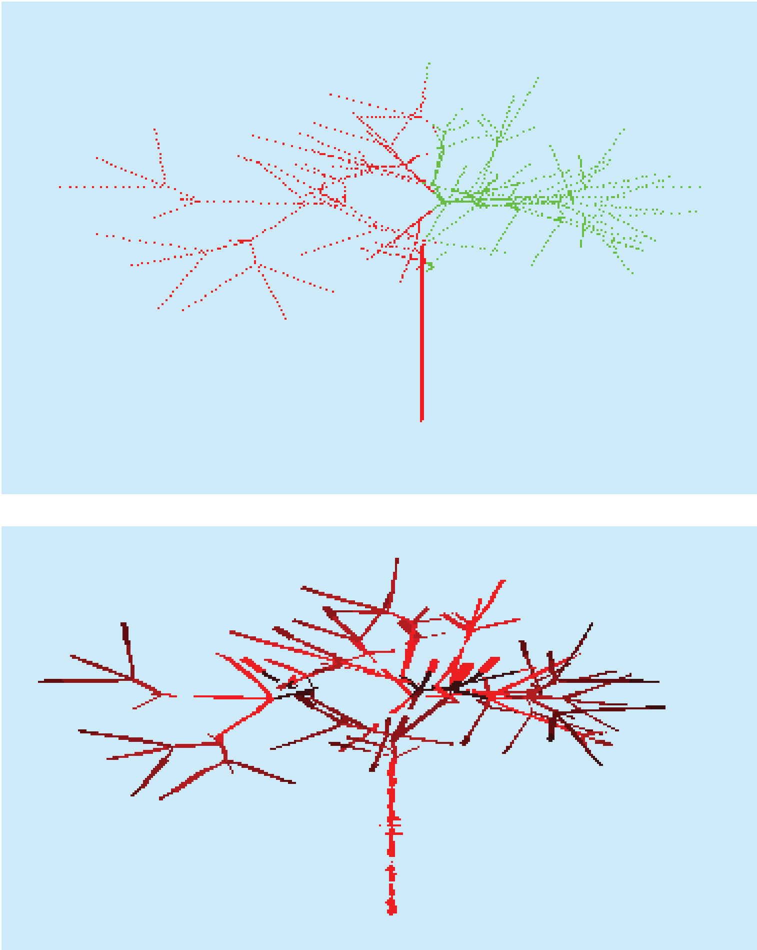

Figure 5. Views of the point clouds with skeletons generated in the form of polylines and of their respective cylinder-based reconstructions.

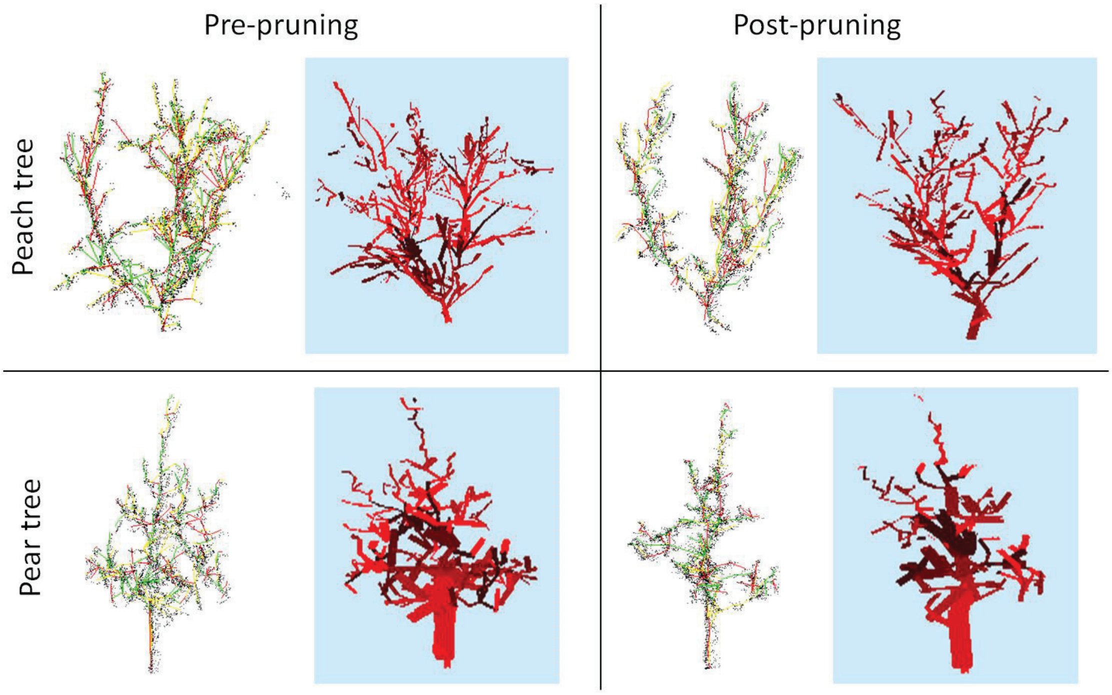

Figure 6. Simultaneous reconstruction of a hedgerow of five central-leader trained pear trees.

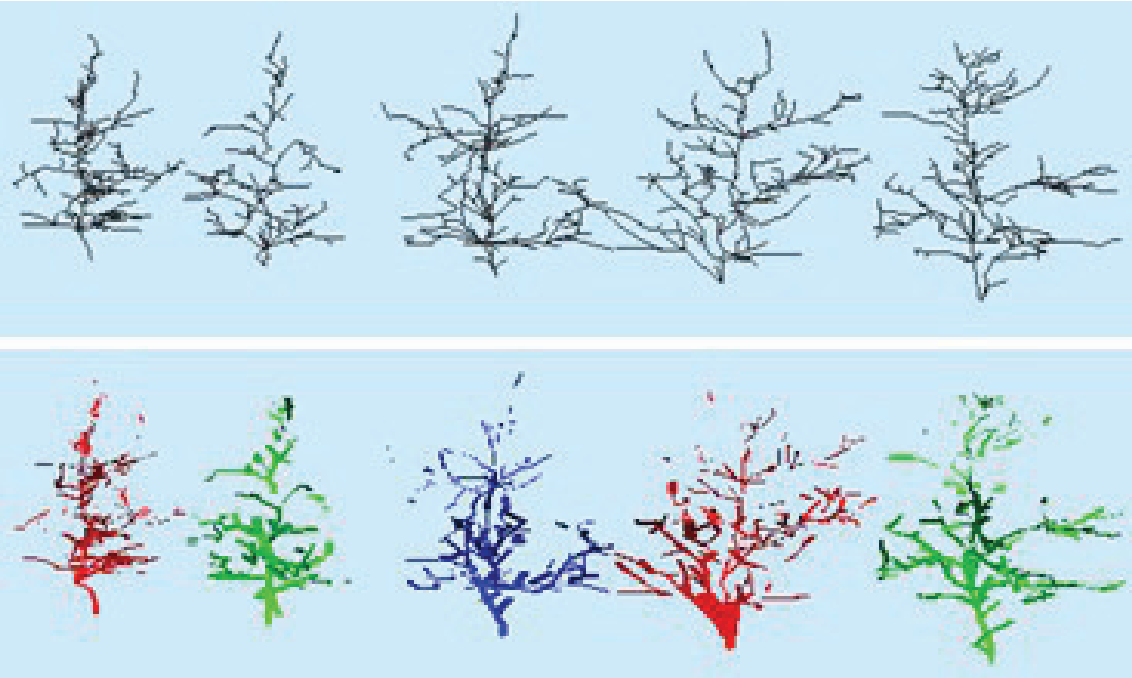

Table 1. Implementation of Dijkstra's algorithm for calculation of multi-tree geodesic graph.

Function	GeodesicGraphTree
Input	iTree
Output	Geodesic Graph

```
Imin = GetIndexRoot(iTree) //Return the root vertex of a Tree
GeoDist[iTree][Imin] ← 0  //The distance of the root to itself must be zero
GeoDone[iTree][Imin] ← true //The vertex of the root is done
RelaxVertex(iTree, Imin) //Set the distance from closed connected vertex to
the root
Iter From I = 1 To length(List_CloudPoints)
        iCur ← MinimumVertex(iTree) //
        If iCur = -1 Then Break(Iter) End(If) //End Iteration
        GeoDone[iTree][iCur] ← true
        RelaxVertex(iTree, iCur) //Calculating distance to root using
        distance to new vertex
End(I)
Return
```

Function	MinimumVertex
Input	iTree
Output	IMin

```
DistMin ← -1
IMin ← -1
Iter From I = 1 To length(List_CloudPoints)
        If GeoDone[iTree][I] = false and GeoPredec[iTree][I] > -1 Then
                1f DistMin = -1 Then
                        IMin ← I
                        DistMin ←GeoDist[iTree][I]
                Else GeoDist[iTree][I] < DistMin Then
                        IMin ←I
                        DistMin ←GeoDist[iTree][I]
                End(If)
        End(If)
End(I)
Return IMin
```

Function	RelaxVertex
Input	iTree, iCur
Output	Void

```
VertCurr ← List_CloudPoints[iCur]
Iter From I = 1 To length(List_CloudPoints)
        If NeighBMatrix[iCur][I] = 1 Then
                VertAdyacen ← List_CloudPoints[I]
                Dist ← Distance(VertCurr , VerAdyacen)
                If Dist + GeoDist[ITree][iCur] < GeoDist[ITree][I] Then
```

```
            GeoDist[ITree][I] ← GeoDist[ITree][iCur] + Dist
            GeoPredec[ITree][I] ← iCur
        End(If)
    End(If)
End(I)
Return
```

Table 2. Implementation of Dijkstra's algorithm for calculation of the multi-tree geodesic graph. Version including management of a matrix of dual connectivity.

Function GeodesicGraphTree

Input iTree

Output Geodesic Graph

```
Imin = GetIndexRoot(iTree) //Return the root vertex of a Tree
GeoDist[iTree][Imin] ← 0  //The distance of the root to itself must be zero
GeoDone[iTree][Imin] ← true //The vertex of the root is done
RelaxVertex(iTree, Imin) //Set the distance from closed connected vertex to the root
Iter From I = 1 To length(List_CloudPoints)
        iCur ← MinimumVertex(iTree) // Find next local vertex to process
        If iCur = -1 Then //There is no local next vertex
                iCur ← MinimumExtVertex(iTree) // Find next extended vertex
                If iCur = -1 Then Break(Iter) End(If) //End Iteration
                SetGeodesicDistance(iTree, iCur) //Set Geod. Distance and parent of iCur
                GeoDone[iTree][iCur] ← true
                RelaxVertex(iTree, iCur) //Distance to root using distance to new vertex
        Else //There is a local connect vertex to process
                GeoDone[iTree][iCur] ← true
                RelaxVertex(iTree, iCur) //Distance to root using distance to new vertex
        End(If)
End(I)
Return
```

Table 3. Implementation of the algorithm to obtain the Predecessor Group.

Function	MeanGroupPredecessor
Input	GroupId
Output	PredecId
	PredecId ← NULL //Empty pointer for predecessor group
	wGeoDist ← ∞
	Iter From I = 1 To GroupId.IndexPoints //Go through every vertex of GroupId
	PredVer ← GeoPredec[GroupId.iTree][I] //Get predecessor in Geodesic Graph
	otherGrId ← GeoGroup[PredVer] //Get the Group of a Vertex
	If otherGrId ≠ GroupId and GeoDist[PredVer]<GeoDist **Then** //A vertex predecessor of a different group and shorter geodesic distance
	PredecId ← otherGrId
	wGeoDist ← GeoDist[PredVer]
	End(If)
	End(I)
	Return

Table 4. List of functions used in the algorithm with estimated cost, where O is the upper limit of growth of the algorithm time cost with the increase of n (number of points in cloud), nb (number of branches) and m (number of trees).

Function	Cost
KTree()	$O(n*\log(n))$
NeighbourMatrix()	$O(n*\log(n))$
GeodesicGraph()	$O(m*n^2)$
LevelSets()	$O(n)$
Clustering()	$O(n*\log(nb))$
FinalCylinders()	$O(nb)+O(nb)+O(nb)*O(\log(nb))=$ $O(n*\log(nb))$
MainFunction()	$O(n^2)$

Table 5. Reconstructions of virtual trees. The point cloud type used is generated from an enveloping mesh on the cylinders without occlusions (E) or from a simulated one-sided MTLS lateral scan with occlusions (LS). Connectivity shows whether dual connectivity was used (two values shown). Debugging shows whether it was necessary as a result of the detection of branch diameters larger than those of their predecessors.

Virtual tree			Scan	Reconstruction				
Length (m)	Volume (dm^3)	Apices	Cloud type	Connectivity (mm)	Length (m)	Volume (dm^3)	Apices	Debugging
1.56	0.60	4	E	20	1.53	0.45	4	No
			LS	15	1.54	0.29	4	
2.40	0.75	11	E	20	2.32	0.62	11	No
			LS	15	2.35	0.36	10	
			LS	15/30	2.38	0.36	11	
7.02	1.27	24	E	20	6.73	0.98	24	No
			LS	15	1.34	0.22	5	
			LS	15/40	5.41	0.86	22	
24.81	1.89	92	E	20	23.60	0.92	93	Yes
			LS	20	3.52	0.37	13	
			LS	20/60	17.21	1	75	
			LS	20/70	20.88	1.02	105	

Table 6. Reconstruction of peach tree (*Prunus persica*) with connectivity matrix 50/300 and k-level sets 16;32 and pear tree (*Pyrus communis*) with connectivity matrix 50/150 and k-level sets 1;4;8;16. Pruned branch volume is estimated from measured branch length and an average radius of 7 mm. For pruned wood mass estimations, a density of 0.6 kg dm^{-3} and 0.69 kg dm^{-3} is considered in peach and pear trees, respectively.

Scanned tree type	**Actual mass of pruned wood (kg)**	**MTLS derived point cloud**	**Pre-pruning**		**Post-pruning**		**Pruned**	
			Number of apices	**Branch length (m)**	**Number of apices**	**Branch length (m)**	**Wood volume (dm^3)**	**Wood mass (kg)**
Peach (open-vase)	1.463	Left scan	365	112.30	228	69.59	1.64	0.98
		Right scan	387	113.14	211	66.81	1.78	1.06
		Scanning both sides	502	144.97	284	80.32	2.48	1.49
Pear (central-leader)	0.716	Left scan	156	49.27	93	30.10	0.74	0.51
		Right scan	157	44.98	95	29.31	0.60	0.42
		Scanning both sides	176	54.76	93	32.71	0.85	0.59