Surface reconstruction of incomplete datasets: A novel Poisson surface approach based on CSRBF

Jules Morel*ab, Alexandra Bacb, Cédric Véga

a French Institute of Pondicherry, 11 Saint Louis Street, Pondicherry 605001, India
b Laboratoire des Sciences de l’Information et des Systèmes, 163 Avenue de Luminy, Marseille 13288, France
c Laboratoire d’Inventaire Forestier, Institut National de l’Informatique Géographique et Forestière, 14 rue Girardet, Nancy 54000, France

ARTICLE INFO

Article history:
Received 27 April 2018
Revised 11 May 2018
Accepted 14 May 2018
Available online 17 May 2018

Keywords:
Surface reconstruction
Approximation
Radial basis functions
Poisson surface
Terrestrial LiDAR scanning

ABSTRACT

This paper introduces a novel surface reconstruction method based on unorganized point clouds, which focuses on offering complete and closed mesh models of partially sampled object surfaces. To accomplish this task, our approach builds upon a known a priori model that coarsely describes the scanned object to guide the modeling of the shape based on heavily occluded point clouds. In the region of space visible to the scanner, we retrieve the surface by following the resolution of a Poisson problem: the surface is modeled as the zero level-set of an implicit function whose gradient is the closest to the vector field induced by the 3D sample normals. In the occluded region of space, we consider the a priori model as a sufficiently accurate descriptor of the shape. Both models, which are expressed in the same basis of compactly supported radial functions to ensure computation and memory efficiency, are then blended to obtain a closed model of the scanned object. Our method is finally tested on traditional testing datasets to assess its accuracy and on simulated terrestrial LiDAR scanning (TLS) point clouds of trees to assess its ability to handle complex shapes with occlusions.

© 2018 Elsevier Ltd. All rights reserved.

1. Introduction

The reconstruction of 3D surfaces from scattered data has received increasing attention with the emergence of new close-range 3D acquisition technologies, such as laser scanning devices (LiDAR), close-range photogrammetry and time-of-flight cameras. The dense 3D point clouds thus acquired accurately describe object surfaces (e.g., millimeter resolution for laser scanning). Such scanning processes have a wide range of applications: urban reconstruction and modeling, architecture, artifacts modeling, quality control for production, and medical imaging. However, despite their accuracy, the data acquired by these sensing technologies share common constraints, such as non-homogeneous sampling, occlusion and noise. In view of their characteristics and complexity, dedicated algorithms are required to segment, model and reconstruct objects of interest from raw point clouds. Terrestrial laser scanning (TLS) technology is broadly used in forest studies. TLS enables 3D forest structures to be acquired as point clouds in record time [1], with applications ranging from ecology (allometric relationships,1 and growth modeling carbon storage assessment) to forestry (forest monitoring, sustainable development) and industry (harvest planning, sawmill optimization). However, the features of such data are even more challenging with respect to reconstructing models and extracting information (e.g., more challenging than classic applicative data, such as urban environments or isolated objects, because the clouds are extremely dense, inhomogeneous, noisy and present large occlusions). These constraints arise both from the remote sensing technology and from the complexity of forest environments. The TLS point cloud sampling rate may vary from one scanner to another, and the spherical geometry of the sensor results in irregular sampling density. Moreover, the combination of TLS geometry and the vegetation itself (branches, leaves, low vegetation) results in large and numerous occluded areas that expand both in size and number far from the sensor. Noise contributes additional confusion at surface extremities and in foliage. Therefore, data obtained from a given tree have different characteristics from the base up to the crown. Forest measurements from TLS data also suffer from object-specific limitations. Stems bark can

1 Allometry consists of a set of general relations derived from a large compilation of forest measurements. It provides an estimate of the tree structure according to a few given parameters, such as the diameter at breast height (DBH, diameter of the stem 1.30 m above the ground) and the tree height.
be rough and therefore produce highly uneven surfaces. Moreover, the non-trivial topology of branches and the intricate occlusions produced by these branches are highly challenging for point cloud processing. Finally, let us note that LiDAR scanning of trees may be affected by wind and thus create multiple scan alignment issues. All these artifacts induce specific point cloud distortions and generate crooked objects. Therefore, in the context of LiDAR data, reconstruction is necessary to handle partially described objects. To overcome the data distribution characteristics inherent to TLS point clouds, especially for data acquired in forests, our idea was to rely on a priori knowledge about the forest elements expressed as geometrical models. This paper presents a novel surface reconstruction method that is specifically designed for the reconstruction of partially described objects based on 3D point clouds. We address this challenge by introducing a novel surface reconstruction method based on a Poisson scheme, building upon sturdy approximating basis functions. On the basis of this innovation, our algorithm lets us integrate a priori models of occluded areas, expressed using basis functions, to describe partially tubular objects. This approach provides good estimates of missing data and enables complete reconstruction of forest objects. This algorithm was tested and validated against “classic” datasets and on occluded almost-tubular shapes and tree sections. In Section 2, we present a short literature review of surface reconstruction and tree modeling from TLS samples.

2. State of the art

Pioneering works on surface reconstruction from raw point clouds date to the late nineties and are usually classified as explicit or implicit according to the underlying model (see [2–4] for complete surveys). There are two types of explicit surfaces, parametric and meshes, both of which have been investigated in this context. Parametric surfaces, such as B-splines [5,6] and NURBS [7], entail determining a 2D parameter space together with a set of associated control points. Such surfaces are controlled by these points but not as an approximation or interpolation. Therefore, complex surfaces are not easily representable, and parameterization is a complex issue for scattered data, especially in the presence of noise, inhomogeneity and occlusions. In [8], the authors define polynomial splines over locally refined parameter spaces in any dimension and thus successfully reconstruct sharp features and details from point clouds. However, although the approach performs well on terrain data, its parametric nature, as well as its complexity, limit its application to data from complex scenes. Triangular mesh reconstruction received substantial attention through Delaunay triangulation, alpha shape reconstruction and Voronoi diagrams [9–12]. However, scanning devices, such as LiDAR scanners, produce dense, noisy and potentially occluded point clouds that cannot be accurately modeled by meshes. The successive remeshing required to obtain an adequate model multiplies the cumbersome computations. Moreover, the inhomogeneity of sampling leads to unbalanced meshes, with larger polygons far from the sensor, tiny polygons close to it (where the point density can reach 1 million points/m²) and stretched polygons in occluded areas.

Therefore, the unstructured, nonplanar nature of point clouds makes implicit surfaces a key modeling tool. Moreover, such models structurally smooth the noise by approximating the input points and are tolerant to inhomogeneity and limited occlusion. Implicit surface reconstruction is the process of finding a function that best fits the input data. However, the implicit representation of a surface needs to be post-processed to be visualized. Marching cube [13,14] is the best-known method to generate a triangulated surface from the implicit representation of the surface. Because the surface is extracted as a level set of an implicit function, the resulting mesh is guaranteed to be a watertight manifold.

Computing implicit functions from point clouds as an approximate of the signed distance function has been extensively studied [15]. However, such approaches prove to be unstable in the presence of nonuniform sampling. The moving least-squares method (introduced in [16], see [17] for a complete survey) addresses this problem but struggles in the presence of missing data, as noted in [2]: the large spatial support of basis functions required near holes spoils the reconstruction. Another class of methods, namely, global reconstruction methods, was proposed by Carr et al. [18]. These approaches are based on radial basis functions (RBFs) and take advantage of their approximating properties. RBFs are positive definite bases of functions and hence guarantee approximation feasibility (see [19,20] for more details). The benefits of modeling surfaces with RBFs are broadly recognized [21–24]. However, polyharmonic RBFs have global support; hence, reconstruction entails inverting dense ill-conditioned matrices. To mitigate this problem, further works focused on compactly supported radial basis functions (CSRBFs) (either used directly [25] or as blending functions between local reconstructions [26–28] or both [29]). The finite support of such functions enables faster filling of the interpolation matrix [30], which simultaneously becomes sparse. Matrix inversion can thus be accelerated by using a direct sparse matrix solver (see Morse [31] for a summary of the advantages of CSRBF over classic RBF). A different approach takes advantage of both global and local fitting schemes by approximating the field of estimated normals through Poisson reconstruction [32]. Owing to the representation as a Poisson problem, this method is robust to nonuniform sampling, noise, outliers and to a certain extent, missing data. These qualities make it a choice method for surface reconstruction from TLS point clouds. However, for computational reasons, implicit surfaces are computed and expressed in a basis of functions obtained by convolution of a box-filter with itself. Unfortunately, this basis is not positive definite (unlike radial basis functions) and thus it does not have sufficient approximation properties to express any a priori information about occluded areas. While these surface reconstruction methods have proven to produce sharp models from point clouds, none is able to fill the large gaps created by occlusion in LiDAR point clouds acquired in forests.

To solve this problem and extract the information needed in forestry, scientists rely on a common assumption: a woody tree structure is assumed to be a network of quasi cylinders. Thus, the tree structure, branching organization and branch size distribution are modeled through so-called quantitative structure models that summarize this information by describing the tree components in hierarchical order as a stack of elementary building blocks. This approach has been widely explored. Côté et al. [33] proposed an architectural model combined with a skeletal curve approach to retrieve the tree structure and allometric relationship to build the branching structure and further assess the amount of foliage. Dassot et al. [1] introduced a semi-automatic approach to model tree architecture using cylinders and to estimate tree parameters, such as tree volume. Raumonen et al. [34] introduced a method involving clustering and segmentation of the point cloud, followed by reconstruction of the tree architecture using cylinders. They combined this geometric and hierarchical information into the concept of quantitative structure modeling (QSM). A similar cylinder-based tree-reconstruction method was proposed by Hackenberg et al. [35] using a sphere-following approach to progressively reconstruct the tree structure from the ground to the apex. Switching from tree-level reconstruction to plot-level reconstruction is a challenging task. Intermingling crowns and occlusions due to branches and leaves in the signal path makes it difficult to accurately segment trees. Raumonen et al. [36] used a clustering approach to detect tree bases, followed by a distance-based expansion procedure to allocate the remaining clusters to the detected trees. Tao et al. [37] used clustering and shortest-path algorithms
to detect trees and segment associated crowns. Combining Hough transform and active contours, Ravaglia et al. [38,39] introduced a method to automatically detect and reconstruct largely occluded tubular shapes. This approach bears some similarities with skeleton extraction methods such as [40]. However, it mitigates the effects of noise and longitudinal occlusions which are both highly present in LiDAR point clouds. While these methods are effective for extracting structural parameters, they propose discontinuous models and rely on local cylindrical (and thus purely tubular) approximations, which has proven to be the best local approximation but can lead to substantial error. Indeed, the assumption that a tree is composed of cylinders can be far from reality, especially when we consider tropical trees. Chave et al. [41] found an error of 50% in the computation of the volume from field measurements at the tree level by such approximations in tropical forests. Therefore, even if QSM approaches are well suited to reconstruct the structure of trees from LiDAR data, they provide only a rough estimate of the structure. Hence, sharper 3D modeling is required to more accurately describe the shape and improve the geometric model to precisely assess tree properties.

In spite of this specific initial context, our approach is actually general: in order to reconstruct largely occluded, roughly tubular data, we assumed that a detailed reconstruction can be achieved only in reasonably sampled areas, but the reconstructed model should be resilient enough to integrate an a priori model of occluded (tubular) areas. Such “roughly tubular” shapes, locally well sampled and partly occluded are quite common in LiDAR data (such as archeological, urban or natural data for instance). The aim of our work is to build a model which proves both locally accurate (in properly sampled areas) and globally coherent (according to a priori information).

Using a priori models to guide the reconstruction is not new. Previous works are based on priors on the surface using Bayesian approaches (works on medical data), global matching of the shape using priors stored in a database (see [42] for instance) or local shape priors (see [43]). However such approaches are based on the assumption that the shape is globally or locally known a priori, and thus focus on matching data and priors. However in our context, we intend to reconstruct the shape as accurately as possible in visible areas (and the shape is unknown there) and priors are used only to reconstruct occluded areas in a coherent way.

3. Overview of the approach

Our work introduces two significant innovations. We propose a new 3D reconstruction method based on a Poisson scheme developed over CSRBF, whose approximating properties have already been mentioned and provide the “resilient” setting for further integration of a priori knowledge. In addition to the reconstruction efficiency demonstrated in our validation, this choice of basis function enables the expression and integration of a priori knowledge in occluded areas.

Our method hence focuses on accurately approximating data points while managing occlusions to produce a closed surface. To do so in our application context, the method identifies and separately processes visible and occluded areas. For the visible portions of the object, a novel Poisson surface algorithm finely reconstructs the surface. In occluded areas, the tree surface is represented using a a priori model characterized by a tubular profile that is adapted to approximate the shape of the tree. The smoothing characteristic of the CSRBF then guarantees the continuity of shape between models. The following two sections describe both pre-processing steps, upon which our method builds: the computation of the point normals, which are required by the Poisson surface solver, and the computation of the tubular shape (or QSM) to guide reconstruction in occluded areas.

3.1. Point normals computation

Let \( P = \{ \mathbf{p}_i \}_{i=1..N} \subset \mathbb{R}^3 \) be the input point cloud. As a preliminary processing step, Poisson surface reconstruction requires computation of a consistent normal field for each sample. We use the method introduced by Hoppe [15] to compute a normal \( \mathbf{n}_i \) at each point \( \mathbf{p}_i \). Actually, more accurate algorithms exist (local triangulation, quadric fitting...), however, LiDAR point clouds can be made of billions of points, making the computation time a critical issue. Following [15], we first study the neighborhood of each point \( \mathbf{p}_i \) to compute a normal \( \mathbf{n}_i \), then we direct consistently the normals by propagating their orientation on a minimum spanning tree.

3.2. Tubular guide estimation

In our next preliminary processing step, we compute a QSM of the tree (in the context of our application, we used the Hough transform approach introduced by Ravaglia et al. [38]). From this discontinuous stack of cylinders, the global outline of the tree is modeled by a continuous tubular shape, as introduced in [44]. Following this approach, in each slice along the Oz axis, the cylinder (centered in \( \mathbf{c}_i \)) is used as an initialization to fit a cylindrical quadratic function \( \mathbf{g}_i \), whose influence lies in the sphere of radius \( \sigma_i \). Each radius \( \sigma_i \) is thus computed so that the set of spheres \( S(\mathbf{c}_i, \sigma_i) \) covers the whole tree. The tubular model is then expressed as an implicit function obtained by blending the quadratic approximations \( \mathbf{g}_i \) together with a CSRBF, namely, Wenland’s CSRBF [30]. Therefore, the a priori model of the tree is expressed as:

\[
 f(\mathbf{q}) = \sum_i g_i(\mathbf{q}) \cdot \Psi_{\sigma i}(||\mathbf{q} - \mathbf{c}_i||),
\]

where \( i \) ranges over the indices of slices \( \{ \mathbf{z}_i \} \) along the Oz axis, \( g_i \) is a quadratic function and \( \Psi_{\sigma i} \) is the Wendland \( \phi_{3,1} \) CSRBF centered on \( \mathbf{c}_i \) and of support radius \( \sigma_i \).

4. CSRBF Poisson surface reconstruction guided by a model known a priori

As illustrated in Fig. 1, following the previous pre-processing steps, our method is based on five steps.

- We divide the space into two parts, namely, occluded and visible areas, by analyzing the angular distribution of points.
- We define a space of functions based on a CSRBF with high resolution near the points and near the surface of the tubular model and coarser resolution away from them.
- In visible areas, we set up and solve the Poisson equation.
- In occluded areas, taking advantage of the approximating properties of the CSRBF, we express the tubular shape in our space of functions and insert it into the solution.
- Finally, we extract an isosurface of the resulting implicit function.

The details of each steps are provided in the following subsections.

4.1. Mark off the occluded areas

As illustrated in Fig. 1, the QSM computation described in Section 3.2 provides a division of space into several slices \( \mathbf{z}_i \) along the stem axis. In each subset, an analysis of the angular distribution of points enables the detection of holes: among the points contained in \( \mathbf{z}_i \), if two points are separated by an angle larger than a given user-defined threshold, then the portion of space between them is considered to be occluded. In this way, we eventually obtain a boundary between the visible and occluded areas for the whole set of slices.
4.2. Problem discretization

To efficiently solve the Poisson scheme, our model needs to be refined close to the points and the tubular model surface. We perform this adaptive subdivision using an octree, which allows us to adjust the resolution locally while providing rapid access to neighborhoods and a low memory footprint. The octree subdivision process is illustrated in Fig. 2. We define the octree as the minimal octree so that every point falls into a leaf of a resolution fixed by the user. Then, we force octree division close to the tubular shape by analyzing the eight corners of each leaf falling in the occluded area; the sign of the implicit function is given in Section 3.2. Subdivision occurs if at least one corner of a leaf has a different sign. Finally, we refine this octree by allowing a maximum difference of levels equal to one between a leaf and its neighbors to guarantee smooth variation in resolution. We denote by \( O_{\text{visible}} \) and \( O_{\text{occluded}} \)
the leaves falling, respectively, in visible areas and occluded areas, as defined in Section 4.1. The resulting set of leaves \( O \) form a partition of space.

### 4.3. Definition of the function space

Let \( P \) denote the input point cloud (set of points of \( \mathbb{R}^3 \)). Given \([O_1, \ldots, O_N]\), a partition of \( P \) (in our case, the previously defined octree subdivision of space), for each subset \( O_i \) (or cell) we denote by \( o_i \) the center of \( O_i \). To build our implicit model, we define a space of functions as the span of translates of Wendland’s CSRFB: for every node \( o \in O \), we thus set \( \Phi_{o_i} \) to be the CSRFB centered on \( o_i \), with radius \( \sigma_o \), defined hereafter to ensure a sufficient coverage of space:

\[
\Phi_{o_i}(\|q - o_i\|) = \phi \left( \frac{\|q - o_i\|}{\sigma_o} \right).
\]

(2)

More precisely, we use \( \phi(r) = (1 - r)^4(1 + 4r) \), where \( + \) represents \( (x)_+ = x \) if \( x > 0 \) and \( (x)_+ = 0 \) otherwise. The radii \( \sigma_o \) are adjusted to recover the details of smaller cells while limiting the overlapping of supports for larger cells. With this in mind, we compute \( \sigma_o \) as:

\[
\sigma_o = \alpha \times \text{Size}_{\text{min}} + \sqrt{3}/2 \times \text{Size}_o.
\]

(3)

with \( \text{Size}_{\text{min}} \) the size of the smallest octree leaf, \( \text{Size}_o \) the size of leaf \( o \) and \( \alpha \) a parameter tuned by the user. \( \text{Span}(\Phi_{o_i}) \), the set of independent basis functions \( \Phi_{o_i} \), is used as a basis to express the implicit functions computed in the following sections.

### 4.4. Setup of the Poisson problem

Poisson surface reconstruction methods are motivated by the computation of the characteristic function \( \chi \) of a solid \( M \) with surface boundary \( \partial M \). In [32], the authors highlighted an integral relation between the gradient \( \nabla \chi \) (where \( \chi \) has been smoothed to become at least \( C^2 \)) and the field of oriented point normals. Hence, \( \chi \) can be computed as an implicit function \( \chi \) whose gradient approximates a vector field \( \nabla \chi \) induced by the normals of the sample points. Given \( p \in \partial M \):

\[
\nabla \chi(p) \cdot \nabla = \int_{\partial M} \mathcal{F}(q - p) \cdot \mathbf{n} \, dp.
\]

(4)

where \( \mathcal{F} \) is a smoothing filter. Because \( \nabla \) is rarely integrable, this variational problem is transformed into a standard Poisson problem by applying the divergence operator:

\[
\Delta \chi = \nabla \cdot \nabla \chi.
\]

(5)

The definition of the discrete approximation of \( \nabla \) is a critical step. The main point is to discretize and approximate Eq. (5). First, following [32], Eq. (5) is discretized as a sum over data points by assuming that any point \( p \) of normal \( n \) describes an elementary patch of surface \( A_p \), which leads to the following formula:

\[
\nabla \chi(p) = \sum_{p \in P} A_p \cdot \mathcal{F}(q - p) \cdot \mathbf{n}.
\]

(6)

Then, for computational efficiency, in order to avoid this integral over data points, the influence of each sample point and its normal are distributed in the surrounding octree cells. Given a point \( p \in P \), a coefficient \( \alpha_{o,p} \) is introduced to account for the share of \( p \) attributed to cell \( o \). We define these coefficients using the basis functions themselves. Let \( N(p) \) be the neighborhood of a sample point \( p \), that is, \( N(p) = \{ o \in O \text{\ visible}, \| p - o_i \| < \sigma_o \} \). We define \( \alpha_{o,p} \) as:

\[
\alpha_{o,p} = \frac{\Phi_{o_i}(\|p - o_i\|)}{\sum_{o \in N(p)} \Phi_{o_i}(\|p - o_i\|)}.
\]

(7)

Now, before defining the vector field \( \mathcal{V} \), let us consider the data inhomogeneity. As mentioned in [32], local variations in point density should be considered to adjust the contribution of data points. We follow a kernel density estimation approach and estimate the local density as:

\[
W(q) = \sum_{p \in P} \alpha_{o,p} \Phi_{o_i}(\|q - o_i\|).
\]

(8)

with \( \Phi^{(r)}(r) = 1 - r \) being a Wendland’s CSRFB of first order.

We can now approximate the vector field defined by Eq. (6) as:

\[
\mathcal{V}(q) = \sum_{p \in P} \frac{1}{W(p)} \sum_{o \in O \text{\ visible}} \alpha_{o,p} \Phi_{o_i}(\|q - o_i\|) \cdot \mathbf{n}.
\]

(9)

### 4.5. Resolution of the Poisson equation

We express \( \chi \) in the space \( \text{Span}(\Phi_{o_i}) \) as the linear combination:

\[
\chi(q) = \sum_{o \in O \text{\ visible}} x_o \cdot \Phi_{o_i}(\|q - o_i\|).
\]

(10)

By projecting the Poisson equation onto this space of functions, we obtain:

\[
\sum_{o \in O \text{\ visible}} \Delta \chi(q) = \sum_{o \in O \text{\ visible}} x_o \cdot \Delta \Phi_{o_i}(\|q - o_i\|).
\]

(11)

Denote by \( \mathcal{L} \) the matrix of elements \( \Delta \Phi_{o_i}(\Phi_{o_i'}) \). \( \mathcal{V} \) the vector of \( \langle \nabla \cdot \mathcal{V}, \Phi_{o_i} \rangle \) elements and \( \chi \) the unknown vector composed of the linear coefficient of \( \chi \). Because our basis function is not orthonormal, we solve the Poisson problem by minimizing \( \| \mathcal{L} \cdot x - \chi \|^2 \) in the least-squares sense. In our case, the matrix \( \mathcal{L} \) is self-adjoint and positive definite owing to the properties of functions \( \Phi_{o_i} \) and \( \Delta \Phi_{o_i} \). Moreover, it is sparse because of the compact support of the basis functions; thus, solving this problem directly is not expensive. Therefore, we use the LDL^T Cholesky decomposition for sparse matrices implemented in the Eigen library. In the next section, we provide further details about the efficient resolution of this Poisson problem with CSRFB. In fact, we make explicit the system of equations and deduce the symmetry properties that enable us to optimize its resolution.

### 4.6. Optimization

The Poisson Eq. (5) can be expressed in its matrix form as:

\[
\begin{bmatrix}
\vdots \\
\langle \Delta \Phi_{o_i}, \Phi_{o_i'} \rangle \\
\vdots \\
\end{bmatrix} \begin{bmatrix}
\vdots \\
x_o \\
\vdots \\
\end{bmatrix} = \begin{bmatrix}
\vdots \\
\langle \nabla \cdot \mathcal{V}, \Phi_{o_i} \rangle \\
\vdots \\
\end{bmatrix}.
\]

(12)

The dot product is that of \( L^2(\Omega) \): given \( f: \mathbb{R}^3 \to \mathbb{R} \) and \( g: \mathbb{R}^3 \to \mathbb{R} \), two compactly supported functions, and \( \Omega \), the union of their compact supports, the dot product can be expressed as follows:

\[
\langle f, g \rangle = \int_{\Omega} f(q) \cdot g(q) \, dq = \int_{\Omega} f(q) \cdot g(q) \, dq.
\]

(13)

Keeping with the definitions introduced earlier in this document, we denote by \( \mathcal{L} = \langle \Delta \Phi_{o_i}, \Phi_{o_i'} \rangle \) the Laplacian matrix and by \( \mathcal{V} \) the vector \( \langle \nabla \cdot \mathcal{V}, \Phi_{o_i} \rangle \).

A study of the symmetries of this system, presented in detail in section Appendix A, allows to reduce the number of computations required to solve the system. Table 1, in particular, illustrates the drastic reduction in the number of computations for reconstruction of the Stanford bunny. For each explored radii value, among the hundreds of thousands of matrix entries that need to be estimated, only a few dozen are actually computed thanks to the symmetry; the remaining are retrieved by a simple search in the hash table.
Table 1: Evolution of the number of integral computations $\mathcal{E}_{a,r}$ and $\mathcal{D}_{a,r}$ to reconstruct the Stanford bunny model according to the radii $\sigma_r$ of the basis functions (parameter $a$ in Eq. (3) for an octree resolution of 1 mm).

<table>
<thead>
<tr>
<th>$a$</th>
<th>Non-zero entries in $\mathcal{E}$</th>
<th>Entries actually computed for $(\nabla \cdot \Phi, \Phi)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5</td>
<td>4773,131</td>
<td>14</td>
</tr>
<tr>
<td>1.6</td>
<td>4781,877</td>
<td>16</td>
</tr>
<tr>
<td>1.7</td>
<td>4785,120</td>
<td>18</td>
</tr>
<tr>
<td>1.8</td>
<td>4785,766</td>
<td>19</td>
</tr>
<tr>
<td>1.9</td>
<td>4786,511</td>
<td>21</td>
</tr>
<tr>
<td>2.0</td>
<td>4786,627</td>
<td>22</td>
</tr>
</tbody>
</table>

4.7. Injection of the tubular shape

To include the tubular shape defined in Section 3.2, the implicit function $f$ describing the tubular shape must be approximated in the space $\text{Span}(\Phi_{\sigma_r})$. We compute the function by interpolating $f$ at collocations $\{o_o, o \in \mathbb{C}\}$. To accomplish this task, we consider the system:

$$
\begin{bmatrix}
A_{o,o'} & \\
\vdots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \vdots \\
\end{bmatrix}
\begin{bmatrix}
y_o \\
\vdots \\
\vdots \\
\vdots \\
\end{bmatrix}
=
\begin{bmatrix}
f(o_o') \\
\vdots \\
\vdots \\
\vdots \\
\end{bmatrix}
$$

(14)

where $A_{o,o'} = (\Phi_{o_o}, \Phi_{o_o'})$.

According to the properties of Wendland’s CSRBF $\Phi_{\sigma_r}$, this matrix is self-adjoint and positive definite. Thus, the system is inverted by using the LDL’ Cholesky decomposition implemented in the Eigen library.

Finally, we obtain the linear decomposition of $f$ in $\text{Span}(\Phi_{\sigma_r})$ as:

$$
f(q) = \sum_{o \in \mathbb{C}_{\text{archmed}}} y_o \cdot \Phi_{\sigma_r}(|q - o|).$$

(15)

4.8. Smoothing between visible and occluded areas

In most cases, the transition between $\chi$, the implicit function defined in the visible areas, and $f$, the implicit function defined in the occluded areas, is handled by the smoothing characteristic of Wendland’s CSRBF. Nevertheless, for complex tree shapes, the transition might be too rough. To overcome this issue, we define an overlap portion of space, whose size $\Theta_{\text{max}}$ is defined by the user, in which we linearly mix the implicit functions. In this sub-part of $\mathbb{C}_{\text{visible}}$, called $\mathbb{C}_{\text{overlap}}$, we have Poisson reconstruction $\sum y_o \cdot \Phi_{\sigma_r}$ and a priori model $\sum y_o \cdot \Phi_{\sigma_r}$. Thus, we consider the implicit function $k$ defined as:

$$
k(q) = \sum_{o \in \mathbb{C}_{\text{overlap}}} [\alpha_o \cdot x_o + (1 - \alpha_o) \cdot y_o] \cdot \Phi_{\sigma_r}(|q - o|).$$

(16)

where $\alpha_o = 1 - \frac{\Theta_{\text{o}}}{\Theta_{\text{max}}}$ is the coefficient of the linear interpolation between $x_o$ and $y_o$ according to the angular position $\Theta_o$ of the center of $o$ in $\mathbb{C}_{\text{overlap}}$.

Finally, our resulting implicit function $h$ is expressed as: $h = \chi + k + f$, as illustrated in Fig. 3.

4.9. Surface extraction

Following from our modeling choices (namely, Wendland’s RBF and quadrics), the implicit function $f$ is $C^2$. Hence, it is possible to use a polygonization algorithm to represent the zero-level set of implicit function $h$. In the space defined by the union of spheres of radius $\sigma_r$, $h$ is discretized into a scalar field, whose resolution is set by the user. From this scalar field, the polygonizer computes a set of triangles approximating the implicit surface $h$ by using a marching-cube-like algorithm [45].

5. Complexity

In the visible areas, the scheme transforms the integral on the points into an integral on octree cells by distributing each point influence in the neighboring cells through the $\alpha_{o,p}$ coefficients. The complexity thus mainly arises from construction of the sparse $N \times N$ matrix, where $N$ is the octree size and the resolution of the
Fig. 5. Mean Hausdorff distance between the computed mesh and the reference for the three objects considered.

Fig. 6. Insight of the results obtained on Quasimoto.

Fig. 7. Insight of the results obtained on the gargoyle.

Fig. 8. Insight of the results obtained on the anchor.

associated linear system. Owing to the symmetries presented in Appendix A, the matrix construction involves at most one hundred integrals.

In the occluded areas, the complexity originates from the expression of the a priori model in the CSRBF basis. The main operation is also the inversion of a symmetric and positive definite sparse matrix.

6. Results and validation

The quality of our algorithm is assessed based on three experiments: in Section 6.1, we compare the surface generated by our approach to the Poisson surface reconstruction found in the literature, namely, the un-screened version of the Poisson reconstruction developed in [32]. Then, in Section 6.2, we compare how the two methods address occlusions. Finally, in Section 6.3, we analyze the improvement to tree modeling.
6.1. Comparison without occlusion

To test the robustness of our version of Poisson surface reconstruction, we compare the surface we generate to the one created by Kazhdan, which acts as a reference in this field of research. We solved the Poisson problem on a non-occluded well-known standard model (the Stanford bunny) and directly produced a surface model without injecting any shape in the solution, i.e., we ran only the part of the algorithm described in Section 4.4, 4.5 and 4.9. Fig. 4 presents the reconstructed surfaces of the bunny.

Then, we estimated the distances to the reference for the un-screened Poisson surface reconstruction and for our method. This quality assessment was performed by comparing the reference mesh model with the outputs of the algorithms using the Hausdorff distance implementation available in Meshlab. As an additional validation step, we evaluate the reconstruction force of our method on the data set provided by Berger et al. [4]. Among the simulated point clouds available in this benchmark, we considered the ones describing the anchor, the gargoyles, and Quasimodo. For each of these objects, surfaces were reconstructed on 16 point clouds, for an octree resolution of 0.5 cm. Fig. 5 presents the Hausdorff distances computed between the surfaces produced by our algorithm and the reference, which is also provided in the benchmark. Figs. 6–8 gives an insight of the reconstructed surfaces quality.

6.2. Occlusion management

The development of our method ensues from the need to manage holes in surface reconstruction from 3D point samples collected in forests. Indeed, the state-of-the-art approaches in the literature struggle to address holes. In particular, when considering trees described by single TLS scans, only the bark facing the scanner is described by the point samples: the entire back side remains occluded. To compare our approach to traditional PSR in this situation, we generated point clouds distributed on a cylinder with a 1 meter radius and a decreasing angular distribution to simulate a widening occlusion. Visual assessment of the surface reconstruction on the occluded cylinders (Fig. 9) demonstrated how our approach can handle data gaps over 50% of the surface owing to the completion of the shape by the a priori model. In this case, the point samples are distributed only on half of the cylinder, similar to the conditions of a single scan in a forest. On the opposite side, traditional PSR shows shape inconsistencies starting at 15° and fails to close the shape when the occluded areas exceed 20°.

6.3. Modeling of trees

The following step of our validation focuses on testing the efficiency of our reconstruction approach with respect to the
7. Discussion

First, in order to demonstrate the generality of our Poisson reconstruction scheme, we compared our method, used without any a priori information, to the un-screened Poisson surface reconstruction that is well-known in the computer graphics field of surface reconstruction. Despite the longer computation time required to compute the matrix terms due to the Monte Carlo integration, our method proved to be slightly more efficient to rebuild the surface mesh of the Stanford bunny, with less noise, higher accuracy and no holes. Indeed, the distance between the input point cloud and the reconstructed surface presented in Table 2 has the same magnitude for both methods (slightly lower for our approach). Our method produces a slightly more detailed surface owing to the smaller radius of the kernels. The un-screened Poisson reconstruction surface produces a smoother and less-detailed surface for the same octree refinement. As a second validation step without priors, we estimated the difference in between our reconstructions on the simulated scans proposed in the benchmark [4] and the supplied references. At a 5 mm resolution, our method showed good results on the rounded objects that are the gragoyle and Quasimoto, with errors close to 2 mm. Errors increased to almost 3 mm on the more angular object, the anchor, presenting sometimes artifacts on the sharp edges. Those unsteadies, where the normal direction tends to change abruptly, will be tackle by a better discretization induced by the multiscale approach discussed later on.

The next step of the validation was to evaluate the effect of occlusion for simple tubular shapes on both methods. The results of the experiment presented in Fig. 9 clearly show the limit of un-screened Poisson surface reconstruction: for an angular occlusion greater than 20°, which corresponds to a hole of 35 cm on a cylinder of points with a 1 m radius, the method diverges (the surface represented in Fig. 9 is the extraction of the isosurface in the bounding box of the cylinder, thus divergence appears as a hole in the reconstructed surface). On the other hand, our approach solves this occlusion issue up to 180° by guiding the reconstruction using an a priori shape.

In the last part of the validation, we “measured” the added value of our method for modeling trees and computing their volume. The model that our method produces was compared to the tubular shape based on quadrics and CSRBF described in Section 3.2 (which was itself 50% more accurate than standard QSM approaches). Overall, the proposed Poisson CSRBF based approach provided an improved approximation of the object shape and volume. The method is particularly well suited to reconstruct irregular shapes and is thus more flexible to handle the diversity of shapes found in natural environments.

The method is particularly useful for reconstructing crown branches whose shapes can be distorted due to gravity, as well as tree buttresses, which remain difficult to model and could...
contain a significant portion of the tree volume and biomass. Fig. 13 presents an example of surface reconstruction in an unfavorable case of a point cloud distributed on the buttress at the base of a tree. While the tubular shape struggles to fit the points, our method correctly approximates the shape of the buttress while preserving the tubular shape in the occlusion. Figs. 10–11 highlights the improvement in tree modeling from the tubular shape to our Poisson surface reconstruction based on CSRBF. For the whole set of trees used in the validation, the tubular shape model remains greater than 1.5 cm on average. Our method reduces this distance to an average of 6 mm. The remaining distance between the model and the points is due to the smoothing of the implicit function described in Section 4.8, where we set aside input points and the surface to handle complex tree shapes. Moreover, because of the point-density differences, our results are slightly less accurate on thinner trees, for instance, tree 4. These issues highlight several ways to improve the method. First, we will improve the occlusion detection to more accurately merge the surface from the resolution of the Poisson equation to the shape known a priori. Then, we plan to improve the combination of the models produced in the visible and occluded areas by defining a feedback loop to deform the a priori model by checking the local curvature. Finally, we intend to implement a multiscale approach to Poisson surface reconstruction to better handle variations in points density. By following the framework defined in [46] where the matrices are partially recomputed at each refinement iteration, we also intend to build an adaptive solver for our Poisson equation by taking advantage of the refinability of the function space. This process will also lead to better memory management and faster computation.

8. Conclusion

Our original Poisson surface reconstruction based on CSRBF proved to be efficient to reconstruct surfaces from dense point clouds. Moreover, using this function basis, we can model any smooth prior. Hence, for tubular shapes, the method addresses occlusion by integrating an a priori model of the shape to reconstruct closed surface models. For scanned trees, we thus improve both volume (ie. biomass) and shape estimation but the method actually applies to any quasi tubular shape (such data actually arise frequently in archeological, urban or natural data). In future work, we intend to enhance the method to handle even higher point-density shifts by following a multiscale approach, which will also accelerate the resolution of the Poisson equation. We also plan to improve the control of transitions between the visible and occluded areas by adding feedback from the Poisson reconstruction to the a priori model. Moreover, as any smooth prior can be expressed using proper collocations for CSRBF, we are currently working on the efficient integration of general priors for occluded areas.

Acknowledgments

The authors would like to express their thanks to the Stanford Scanning Repository for their generosity in distributing their
3D models. The authors would also like to thank the French Institute of Pondicherry for supporting the fellowship of Jules Mored, Alexandre Piboule at ONF (Office National des Forêts, France) to assist in the development of the Computree plugin and Nicolas Barbier at UMR AMAP (Botanique et Modélisation de l’architecture des plantes et des végétations, France) for providing test data. Those unpublished data from Cameroon were collected in collaboration with Alpican company within the IRD project PPR FTH-AC Change- ment globaux, biodiversité et santé en zone forestière d’Afrique Centrale. Cédric Véga was supported by the Laboratory of Excellence for Advanced Research on the Biology of Tree and Forest Ecosystems (ARBRE) and the European Unions Horizon 2020 research and innovation programme under grant agreement No 633464 (DIA-BOLO).

Appendix A. Optimization

A1. Evaluation of the integral terms

Let us first compute the coefficients of the matrix \( L \), that is, \( \langle \Delta \Phi_{\sigma_o}, \Phi_{\sigma_o} \rangle \). Our basis functions have been chosen to be \( c^2 \); therefore, by integrating the compactly supported functions by parts, we obtain \( l_{ij} f^i \cdot g = -l_{ij} f^j \cdot \partial_x g \). Thus, we can express the terms of the matrix as:

\[
\langle \Delta \Phi_{\sigma_o}, \Phi_{\sigma_o} \rangle = -\frac{\partial \Phi_{\sigma_o}}{\partial x} \cdot \frac{\partial \Phi_{\sigma_o}}{\partial y} - \frac{\partial \Phi_{\sigma_o}}{\partial z} \cdot \frac{\partial \Phi_{\sigma_o}}{\partial x}
\]

As noted in Section 4.3, we use Wendland’s CSRB to define our basis, that is, for \( q(x,y,z) \in \mathbb{R}^3 \), \( \Phi_{\sigma_o}(q) = \left(1 - \frac{\|q - \sigma_o\|}{\sigma_o}\right)^4 (1 + 4 \frac{\|q - \sigma_o\|}{\sigma_o}) \). The derivative along the first coordinate thus gives:

\[
\frac{\partial \Phi_{\sigma_o}}{\partial x} = \frac{20}{\sigma_o^2} \left(1 - \frac{\|q - \sigma_o\|}{\sigma_o}\right)^3 \frac{\|q - \sigma_o\|}{\sigma_o}^2 \frac{\partial q}{\partial x}
\]

where \( (v)_w \) denotes the coordinate of a vector \( v \) along the \( w \) axis (with \( x, y, z \)). Let us set \( q_x = q - \sigma_o \) and \( q_y = q - \sigma_o \). The combination of Eqs (A.1) and (A.2) gives:

\[
\langle \Delta \Phi_{\sigma_o}, \Phi_{\sigma_o} \rangle = -\frac{200}{\sigma_o^2 \sigma_o' \sigma_o'} \int_\Omega \left(1 - \frac{\|q - \sigma_o\|}{\sigma_o}\right)^3 \frac{\|q - \sigma_o\|}{\sigma_o}^2 \langle q_x, q_y \rangle dq.
\]

We then simplify Eq. (A.3) as:

\[
\langle \Delta \Phi_{\sigma_o}, \Phi_{\sigma_o} \rangle = -\frac{400}{\sigma_o^2 \sigma_o' \sigma_o'} \mathcal{L}_{\sigma_o', \sigma_o'}
\]

where

\[
\mathcal{L}_{\sigma_o', \sigma_o'} = \int_\Omega \left(1 - \frac{\|q - \sigma_o\|}{\sigma_o}\right)^3 \frac{\|q - \sigma_o\|}{\sigma_o}^2 \langle q_x, q_y \rangle dq.
\]

Following the same process, we express the projection of the divergence of \( v \) on our base of functions by:

\[
\langle \nabla \cdot v(q), \Phi_{\sigma_o} \rangle = \sum_{p \in \mathbb{P}} \sum_{\sigma_o \in \Omega} \alpha_{p,\sigma_o} \int_\Omega \nabla \cdot [\Phi_{\sigma_o}(\|q - \sigma_o\|) \cdot n] \Phi_{\sigma_o}(\|q - \sigma_o\|) dq.
\]

where

\[
\nabla \cdot [\Phi_{\sigma_o}(\|q - \sigma_o\|) \cdot n] = (\nabla \Phi_{\sigma_o}(\|q - \sigma_o\|) \cdot n) = -\left(\frac{20}{\sigma_o^2}\right) \left(1 - \frac{\|q - \sigma_o\|}{\sigma_o}\right)^3 \langle q, n \rangle.
\]

Finally, we rewrite Eq. (A.6) as:

\[
\langle \nabla \cdot v(q), \Phi_{\sigma_o} \rangle = \sum_{p \in \mathbb{P}} \sum_{\sigma_o \in \Omega} \alpha_{p,\sigma_o} \left(\frac{20}{\sigma_o^2}\right) \sum_{i \in \{x,y,z\}} \mathcal{D}_{\sigma_o,ij} \langle n_i, n \rangle \sigma_o dq.
\]

A2. Study of symmetries

To accelerate and simplify the computation of the terms of Eq. (12), we study the symmetries of \( \mathcal{L}_{\sigma_o', \sigma_o'} \) and \( \mathcal{D}_{\sigma_o,\sigma_o', \sigma_o'} \) and show that only a few of these terms need to be computed.

Symmetries for the computation of \( \mathcal{L}_{\sigma_o, \sigma_o'} \)

Let us define the function \( f \) as \( f(x) = (1 - x)^2 \). Eq. (A.4) then becomes:

\[
\mathcal{L}_{\sigma_o, \sigma_o'} = \int_\Omega f \left(\frac{\|q - \sigma_o\|}{\sigma_o}\right) f \left(\frac{\sigma_o - \sigma_o'}{\sigma_o'}\right) dq.
\]

In this section, we assume \( \sigma_o \) and \( \sigma_o' \) are given and fixed. Let us first show that \( \mathcal{L}_{\sigma_o, \sigma_o'} \) depends only on vector \( \sigma_o - \sigma_o' \). We set \( p = q - \sigma_o \) (and let \( r \) be the corresponding translation); by change of variables in Eq. (A.10), we obtain:

\[
\mathcal{L}_{\sigma_o, \sigma_o'} = \int_\Omega f \left(\frac{\|r + \sigma_o - \sigma_o'\|}{\sigma_o'}\right) f \left(\frac{\|r - \sigma_o - \sigma_o'\|}{\sigma_o}\right) \left|\det(\mathcal{J}(r))\right| dq.
\]

where \( \det(\mathcal{J}(r)) \) is the determinant of the Jacobian of the translation \( r \), whose value is 1. As a consequence, \( \mathcal{L}_{\sigma_o, \sigma_o'} \) depends only on the vector \( \sigma_o - \sigma_o' \) and radii \( \sigma_o \) and \( \sigma_o' \).

Let us now proceed a step further and show that \( \mathcal{L}_{\sigma_o, \sigma_o'} \) depends only on \( \|\sigma_o - \sigma_o'\| \). Let us consider the rotation \( r : q \rightarrow \mathcal{R} \cdot q \), whose matrix in the canonical basis is \( \mathcal{R} \), and send the vector \( \|\sigma_o - \sigma_o'\| \cdot e_1 \) to \( \sigma_o - \sigma_o' \) along the x axis, with \( e_1 = (1, 0, 0) \). As \( f \) is a radial basis function, for any vector \( u, f(u) = f(\|\mathcal{R} u\|) \). Moreover, the scalar product of two vectors is clearly invariant under simultaneous rotation. Hence, Eq. (A.11) becomes:

\[
\mathcal{L}_{\sigma_o, \sigma_o'} = \int_\Omega f \left(\frac{\|\mathcal{R} p\|}{\sigma_o'}\right) f \left(\frac{\|\mathcal{R} p + \sigma_o - \sigma_o'\|}{\sigma_o'}\right) \left|\det(\mathcal{J}(r))\right| dq.
\]

Now, under the change of variables \( u = \mathcal{R} p \), we obtain:

\[
\mathcal{L}_{\sigma_o, \sigma_o'} = \int_\Omega f \left(\frac{\|u\|}{\sigma_o'}\right) f \left(\frac{\|u + \sigma_o - \sigma_o'\|}{\sigma_o'}\right) \left|\det(\mathcal{J}(r))\right| dq.
\]

As the rotation \( \mathcal{R} \) is isometric, it maintains the norms and dot products. Therefore, by using \( |\det(\mathcal{J}(r))| = 1 \), Eq. (A.13) becomes:

\[
\mathcal{L}_{\sigma_o, \sigma_o'} = \int_\Omega f \left(\frac{\|u\|}{\sigma_o'}\right) \left(\frac{\|u + \sigma_o - \sigma_o'\|}{\sigma_o'}\right) \left|\det(\mathcal{J}(r))\right| dq.
\]

Therefore, Eq. (A.14) proves that \( \mathcal{L}_{\sigma_o, \sigma_o'} \) depends only on \( \sigma_o, \sigma_o' \) and \( \|\sigma_o - \sigma_o'\| \).

Symmetries for the computation of \( \mathcal{D}_{\sigma_o, \sigma_o', \sigma_o'} \), \( t \in \{x,y,z\} \)

The same principle can be used to study the symmetries of Eq. (A.9):

\[
\forall t \in \{x,y,z\}, \mathcal{D}_{\sigma_o, \sigma_o', \sigma_o'} = \int_\Omega f \left(\frac{\|q\|}{\sigma_o}\right) \Phi_{\sigma_o}(\|q + \sigma_o - \sigma_o'\|) \langle q, n \rangle dq.
\]

Although the results are not shown here, it can be demonstrated that \( \mathcal{D}_{\sigma_o, \sigma_o', \sigma_o'} \) depends only on \( \sigma_o, \sigma_o' \), \( \|\sigma_o - \sigma_o'\| \) and the t-coordinate of \( \sigma_o - \sigma_o' \).
A3. Resolution

The integrals $L_{\Omega_d}$ and $D_{\Omega_d}$ are computed with the MISER algorithm of Press and Farrar [47] implemented in GSL. This Monte Carlo technique reduces the overall integration error by concentrating the integration points in the regions of highest variance.

Moreover, the use of the invariance properties we just proved drastically reduces the number of integrals to compute. Indeed, we showed that the integrals in $L_{\Omega_d}$ depend on only $\sigma_\alpha, \sigma_\beta$ and $|\mathbf{o}_\alpha - \mathbf{o}_\beta|$ and that the integrals in $D_{\Omega_d}$ depend on only $\sigma_\alpha, \sigma_\beta, |\mathbf{o}_\alpha - \mathbf{o}_\beta|$ and $(\mathbf{o}_\alpha - \mathbf{o}_\beta)$. In practice, we stored the values of the integrals in hash tables whose hashes were computed from the variables $\sigma_\alpha, \sigma_\beta, |\mathbf{o}_\alpha - \mathbf{o}_\beta|$ and the extra $(\mathbf{o}_\alpha - \mathbf{o}_\beta)_i$ for the integrals in $D_{\Omega_d}$. Those hash keys can be converted into integer values to avoid floating point precision problems. For each couple $\alpha, \beta$, we thus compute the integrals $L_{\Omega_d}$ and $D_{\Omega_d}$ only if the value is absent in the hash table.

References