Point Cloud Structural Parts Extraction based on Segmentation Energy Minimization

Bruno Cafaro[1], Iman Azimi[1], Valsamis Ntouskos[1], Fiora Pirri[1] and Manuel Ruiz[1]
[1]*Department of Computer Science, "Sapienza" University of Rome, Via L. Ariosto, 25*
{ *cafaro, ntouskos, pirri, ruiz* }*@dis.uniroma1.it,azimi.1449311@studenti.uniroma1.it*

Keywords: Point Clouds, Level Set Methods, Minimal Surface Energy, Segmentation, Meshing.

Abstract: In this work we consider 3D point sets, which in a typical setting represent unorganized point clouds. Segmentation of these point sets requires first to single out structural components of the unknown surface discretely approximated by the point cloud. Structural components, in turn, are surface patches approximating unknown parts of elementary geometric structures, such as planes, ellipsoids, spheres and so on. The approach used is based on level set methods computing the moving front of the surface and tracing the interfaces between different parts of it. Level set methods are widely recognized to be one of the most efficient methods to segment both 2D images and 3D medical images. Level set methods for 3D segmentation have recently received an increasing interest. We contribute by proposing a novel approach for raw point sets. Based on the motion and distance functions of the level set we introduce four energy minimization models, which are used for segmentation, by considering an equal number of distance functions specified by geometric features. Finally we evaluate the proposed algorithm on point sets simulating unorganized point clouds.

1 INTRODUCTION

This work is focused on 3D point cloud analysis. These are geometric data sets in which only the digitized 3D points are usually available. Such low-level data exhibits a huge gap with a high-level representation, which is necessary in many application areas as: surface compression, shape classification, hybrid rendering, meshing or simplification. We define simple scene elements, to extract their structural parts, relying only on the raw point cloud. The proposed framework is based on energy minimization using Level Set and the construction of specific distance functions using local geometric features, such as normals, surface variation and height of points. The novelty is due to the contextually guided mesh construction, which approximates the minimal surface energy of the considered structural components. More precisely, the proposed approach forms a mesh from a set of points only if these points share common local geometric features. Here we assume that structural parts of a simple scene element can be characterized by common geometric features values. Iteratively repeating the mesh construction for each structural part, segmentation is obtained, as can be seen in Figure 1.

The level set based approach proves to be quite general as it performs extremely well in surface energy minimization, despite it takes into account just few parameters. We tested the proposed method on point sets simulating unorganized point clouds of scene elements, highlighting the composition of structural parts, and adding typical points cloud noise level. Extension of this segmentation framework to more complex scene elements and very large point clouds might require some further work, though we expect that most of the approach can be elegantly generalized.

The paper is organized as follows: in Section 2 we review state of the art on 3D segmentation; in Section 3 we introduce the level set framework for surface reconstruction. In Section 4 we introduce the voxel space and the proposed 3D segmentation methods. Section 5 explains the main implementation issues and Section 6 illustrates the performed experiments. Finally, in the conclusion we address how the method could be used for object segmentation.

2 RELATED WORKS

Our approach is based on raw point cloud analysis, and survey (Nguyen and Le, 13) summarizes recent approaches for unorganized point cloud segmentation. The author propose to classify the segmentation

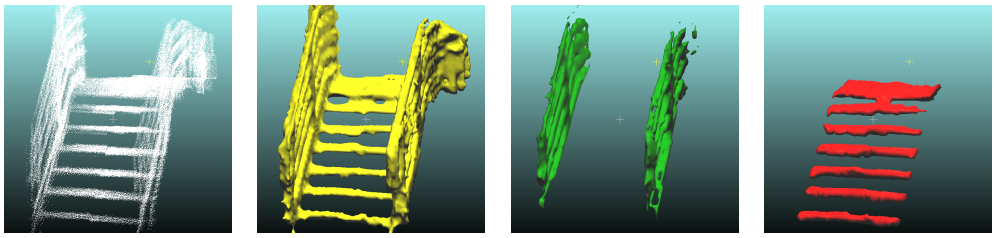


Figure 1: Results of model $\mathcal{M}_1$ segmentation and meshing on unorganized 3D laser scan data.

methods in five main methods: edge based, attribute based, graph based, region based, and model based. Among the edge based methods, (Ioannou et al., 12) propose to segment points adopting a difference of normal (DoN) operator. In (Rusu, 09) the author proposes an attribute based method exploiting euclidean clustering, and plane fitting, creating a queue for not labeled points, which are further assigned to one of the clusters according to features distance. Amid the graph based methods, (Ma et al., 10) propose to partition the point cloud data adopting a graph min-cut procedure, where the graph is defined according to points similarity; then spectral clustering is used to solve min-cut. In (Nurunnabi et al., 12) region growing exploits features such as color, normal vectors or curvature. In (Golovinskiy and Funkhouser, 09) the proposed graph-cut method computes an edge cost between points, and between points and source, and points and sink. Then a min-cut, max-flow approach is used to cut the created graph. In model based methods points are grouped using geometric primitives.

In (Kustra et al., 2014) the authors first estimate surface properties, then apply clustering in the Gauss sphere to group points in patches. Those are classified using a connectivity graph.

Other approaches develop on available mesh connectivity, extracting information also from the edges or the polygons of the mesh itself, for a review see (Shamir, 08) and references therein. Also mesh segmentation is grouped in five methods: region growing, hierarchical clustering, iterative clustering, spectral analysis and implicit methods. In iterative clustering the segmentation is formulated as a variational problem, then a solution is iteratively searched for the given number of clusters, see for example (Lloyd, 06). In (Karni and Gotsman, 00) spectral clustering exploits the same principles for meshes and point sets, but computing the Laplacian matrix for mesh vertices. In (Schreiner et al., 04) a mesh is divided into its parametrized surfaces of simplicial domains. Obviously many of the approaches for raw point cloud data share some common features with mesh based approaches, and can be easily implemented on both data structures, except for implicit methods.

The problem of most of these methods with respect to the one proposed in this paper is the lack of a clear mathematical framework that allows for a parallel computation, essential for large point clouds. This aspect is not true in Level Set based methods for medical image analysis (see survey (Wirjadi, 07) and references therein). Those methods can be applied when a volumetric representation of the data is available, and are based on implicit functions evolution, extending and refining the Chan-Vese algorithm for 2D images on 3D images.

Other works share common features with our approach, but are based on completely different mathematical framework. An example is (Schnabel et al., 07), in which a RANSAC based algorithm is used to detect clusters of 3D points which belongs to the same geometrical shape: planes, spheres, cylinders, cones and tori. This approach has been refined and extended to cope also with sharp edges in (Tran et al., 14). In (Turner and Zakhor, 14) the authors develop an approach based on the extrusion of the 2D map of a storey, then adding soil and ceiling to obtain a watertight mesh of that storey.

The roots of the proposed approach can be found in (Zhao et al., 98; Zhao et al., 01; Liang et al., 13), in which the authors solve the problem of surface reconstruction. These methods are suited for watertight reconstruction of an entire object. In (Liang et al., 13) the inner product field is used to compute a signed distance function whose zero lies close to the surface approximated by the input points. We extend this method by forming a mesh only for the input points which share common predetermined geometric feature values.

3 PRELIMINARIES AND THE LEVEL SET METHODS

Level set methods are numerical techniques widely adopted in computational physics, fluid-dynamics, computer vision, graphics, and many others fields, for tracking interface evolution and shapes in many different kind of mediums. The great advantage over

other methods is the possibility to perform numerical computations involving curves and surfaces on a fixed Cartesian grid, without having to parameterize these objects, but using their implicit definition instead.

Notation: In the following we shall use boldface letters, both lower and upper case, for vectors, matrices and vector fields such as $\mathbf{x}, \mathbf{w}, \mathbf{v}$, $\mathbf{X}$. We denote time with t, and use lower-case Greek letters for functions: ρ is used in particular for the distance function and ϕ for the level set function. We make arguments of a function explicit only when needed: we may write $\phi(\mathbf{w}, t)$, $\phi(t)$ or $\phi(\mathbf{w})$. We reserve upper case Greek or Latin letters for sets, in particular the point set is denoted S and the domain Ω. Points in Ω are denoted $\mathbf{w}$, and points in S by $\mathbf{x}$, $\mathbf{s}$ or $\mathbf{x}_i$ as we indicate with the subscript i the i-th elements of a vector, matrix or set. For the inner product we use both the $\cdot$ or nothing, and in this last case we make explicit the transpose with the top $\top$ as superscript. The partial derivative of a function φ with respect to parameter w is indicated by φ_w. Finally, we indicate by $\|\cdot\|$ the Euclidean norm, by $|\cdot|$ the absolute value.

By *structural component* of a simple scene element we intend that part of a point cloud approximating an unknown surface patch that could not be split into simpler elements. Within a scene element a structural component cannot be broken down into other scene elements of different kinds. In turn, by a *simple scene element* we intend any part of a complex scene.

Let Ω be a domain in the 3D Euclidean space, let $\mathbf{w} = (x, y, z)^\top \in \mathbb{R}^3$ and $\phi : \Omega \times \mathbb{R} \mapsto \mathbb{R}$, then the *boundary* among two regions, within Ω, can be specified according to an implicit representation: $\partial\Omega = \{\mathbf{w} \mid \phi(\mathbf{w}) = 0\}$. The surface separating these two regions is the zero level of the function $\phi(\mathbf{w})$, called the *level set function*. When adopting a level set approach, one usually starts with an initial guess for the level set function. Then the guessed function is evolved considering an algorithmic time, up to convergence to a minimum. This minimum is found defining another function governing the level set evolution, which is usually a signed or unsigned distance function. Typically the distance function from a point $\mathbf{w}$ to a set S is defined:

$$\rho(S, \mathbf{w}) = \min_{\mathbf{x}_i \in S}(\|\mathbf{w} - \mathbf{x}_i\|) \tag{1}$$

Which implies that when S is the boundary $\partial\Omega$, and $\mathbf{w} \in \partial\Omega$, then $\rho(\partial\Omega, \mathbf{w}) = 0$.

Adopting the Eulerian approach, which is the same as observing the variation of the level set function $\phi(\mathbf{w})$ in a given location in space, different models for the evolution of $\phi(\mathbf{w})$ can be defined. The level set equation is defined as:

$$\phi_t(t) + \mathbf{v} \cdot \nabla\phi(t) = 0. \tag{2}$$

Here $\mathbf{v}$ is the velocity which governs the level set evolution.

3.1 Minimal Surface Energy Model

The level set function ϕ embeds the surface separating two regions, during its evolution. If we consider the samples collected from the surface of an object, such as $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}, \mathbf{x}_i \in \mathbb{R}^3$, we would like to obtain a continuous representation of the object, namely a mesh. To do so we need to represent the surface energy and minimize it to obtain the meshes. The surface energy for points in S is:

$$E(\Gamma) = \left[\int_\Gamma \rho_I^2(S, \mathbf{w}) ds\right]^{1/2} \tag{3}$$

Here Γ is an arbitrary surface in Ω, implicitly defined by the level set function, namely $\Gamma(t) = \{\mathbf{w} | \phi(\mathbf{w}, t) = 0\}$; ds specifies a surface element and $\rho(S, \mathbf{w})$ is the distance function defined in equation (1). This is a minimal surface energy model as defined in (Zhao et al., 98; Zhao et al., 01). During the evolution, equation (3) can be easily computed on a volume considering the Dirac delta function δ_ϕ of $\phi(\mathbf{w})$, or a smoothed version of it. This surface, embedded by the level set function, can be reconstructed considering meshing algorithms for implicit volumes, as marching cubes, see for example (Lorensen and Cline, 87).

3.2 Normal direction of motion

The motion model for the level set function evolution, defined by the normal direction of motion, can be applied when the velocity field is internally generated. In this case the surface is forced to evolve according to its normal direction. Expressing the velocity as $\mathbf{v} = (u, v, w)(\mathbf{n}, \mathbf{t}_1, \mathbf{t}_2)^\top$, where $\mathbf{n}$ is the outward normal to the level set function, and $\mathbf{t}_1$ and $\mathbf{t}_2$ are the tangents, and using this definition of $\mathbf{v}$ in equation (2), we obtain:

$$\phi_t(t) + \Big((u, v, w)(\mathbf{n}, \mathbf{t}_1, \mathbf{t}_2)^\top \nabla\phi(t)\Big) = 0 \tag{4}$$

Expanding the second term in the left-hand side of the above equation the following is obtained:

$$(u, v, w) \begin{pmatrix} n_1\partial\phi/\partial w + n_2\partial\phi/\partial y + n_3\partial\phi/\partial z \\ 0 \\ 0 \end{pmatrix} \tag{5}$$

Hence equation (4) reduces to:

$$\phi_t(t) + u\Big(\mathbf{n}^\top \nabla\phi(t)\Big) = 0. \tag{6}$$

And since $\mathbf{n} = \nabla\phi/\|\nabla\phi(t)\|$, for implicit defined surfaces, then eq. (6) can be simplified to:

$$\phi_t(t) + u\|\nabla\phi(t)\| = 0. \quad (7)$$

In our implementation $u = \rho(S, \mathbf{w})$ and, for example, $\phi(\mathbf{w}, 0) = \|\mathbf{w} - \mathbf{c}\| - r$, with $\mathbf{c} \in \Omega$, $r > 0$.

4 STRUCTURAL PARTS EXTRACTION

In this section we introduce four methods to extract the structural parts of a simple scene element. From now on we assume that each point $\mathbf{x}_i \in S$ is labeled with an approximated normal $\mathbf{n}_i$, computed by least squares, see (Hoppe et al., 92), and a surface variation value K_i, as in (Pauly et al., 03), which is a well known approximation to one of the principal curvatures. The computation proposed in (Pauly et al., 03) is based on (Hoppe et al., 92) computation of the normal, which actually returns the covariance $\Sigma_{\mathcal{N}(\mathbf{x})}$ of a neighborhood of $\mathbf{x} \in S$, and the eigen-decomposition of the covariance. The surface variation is given by $\lambda_1 / \sum_{i=1}^{3} \lambda_i$ with $\lambda_1 \leq \lambda_2 \leq \lambda_3$ the eigenvalues of $\Sigma_{\mathcal{N}(\mathbf{x})}$ eigen-decomposition. This quantity varies between 0 and $1/3$, and has the property of being high if the region is highly curved and it is small if the region is flat, however does not capture the principal curvatures and the induced classes. We use these neighborhood based methods because we keep the set S unorganized.

4.1 Voxel Space

The discrete voxel space $\Omega^\star$ is obtained from the domain Ω and the point cloud S as follows. Let M be the number of points in S, let $\mathcal{N}(\mathbf{x}_j) = \arg\min_{\mathbf{x}_i}(\|\mathbf{x}_j - \mathbf{x}_i\|)$, $j \neq i$, and $\zeta = (1/M)\sum_{j=1}^{M} \|\mathbf{x}_j - \mathcal{N}(\mathbf{x}_j)\|$, ζ is the mean distance of each point from its closest neighbor, then S is filtered using the normal density function defined from ζ and $\sigma = (1/M)\sum_{i=1}^{M}(\|\mathbf{x}_i - \mathcal{N}(\mathbf{x}_i)\| - \zeta)^2$ and ζ is redefined according to the filtered points.

A single voxel is a cube with size $(dx, dy, dz)^\top = (2\zeta, 2\zeta, 2\zeta)^\top$. The diagonal of a voxel is $dc = (\sqrt{12})\zeta$ and the vertices of the voxel grid are specified by (i, j, k). The (i, j, k) indices of voxel $\omega^\star_{i,j,k}$ are defined as follows: the leftmost bottom vertex of the grid has indices $i_0 < x_q, j_0 < y_q, k_0 < z_q$, for all $(x_q, y_q, z_q)^\top = \mathbf{x_q} \in S$, and $(i, j, k) = (i_0, j_0, k_0) + (i\,dx, j\,dy, k\,dz)$.

Each generated cuboid has axes parallel to the reference frame of the point cloud, and it is padded with an amount $\delta > 3$ of the cuboid edge size, inducing a total number of voxels added for padding equal to $\delta^3 + 3\delta^2 r + 3\delta r^2$ (here r is the obtained resolution). The (i, j, k)-th voxel centroid $\mathbf{m}_{i,j,k}$ is defined as $\omega^\star_{i,j,k} + (1/2)(dx, dy, dz)$, and the closest point to $\mathbf{m}_{i,j,k}$ is the point $\mathbf{x}_i \in S$ specified by $\mathcal{N}(\mathbf{m}_{i,j,k})$.

Table 1: Special cases for the c_i constants of equation (8).

$F(\mathbf{w})$	c_1	c_2	c_3
$K \neq 0$	$c_1 > 1$	$c_2 = 0$	$c_3 = 0$
$K = 0$	$c_1 = 0$	$c_2 > 1$	$c_3 > 1$
$K = 0 \wedge n^\star \neq [0, 0, \|1\|]$	$c_1 = 0$	$c_2 = 0$	$c_3 > 1$

Note that the voxel space is a grid and that operations on it are related to the distance function ρ, to its extensions, as described afterwards, and to the grid centroids $\mathbf{m}_{i,j,k}$ labels, namely the geometric features values. It follows that a centroid $\mathbf{m}_{i,j,k}$ is meaningful in terms of the distance to points $\mathbf{x}_i \in S$ and in terms of the features labeling the closest point $\mathbf{x}_i$ to $\mathbf{m}_{i,j,k}$.

4.2 Feature extended distance functions

The labeled voxel space is used to define a distance function accomodating geometric features. This distance function and its extensions are used to obtain four structural parts segmentation models, as explained in the following. For each point in S the features K, h and $\mathbf{n}$, as described in the previous paragraph, are collected into a descriptor $F(\mathbf{x}_i) = \{K, h, \mathbf{n}\}$, with K the mean curvature and h the height. Hence each voxel is labeled with features values, according to $F(\mathbf{m}_{i,j,k}) := F(\mathcal{N}(\mathbf{m}_{i,j,k}))$.

The transformed distance function, taking into account the features descriptor, is defined as follows:

$$H = c_1|K - K^\star| + c_2|h - h^\star| + c_3\left(\cos^{-1}\left(\frac{\mathbf{n}\cdot\mathbf{n}^\star}{|\mathbf{n}||\mathbf{n}^\star|}\right)\right) \quad (8)$$

$$\rho_F(S, \mathbf{w}) = \begin{cases} \rho_I(S, \mathbf{w}) + H & \text{if } F \neq F^\star \\ H & \text{otherwise} \end{cases} \quad (9)$$

Note that here the c_i are scalars, whose role is to keep the distance function within a specified range, and provide relative importance to the feature difference terms in equation (8). Moreover the surface variation K is best used alone than together with the normal vectors, in some contexts. In such cases the normals term in equation (8) must be set to zero. These special cases are reported in Table 1. The starred variables indicate a reference feature value, which determines the points in S having a minimum for equation (8). These values are obtained as described in Section 5.

Now, let $\ell = 0.05 \max_\Omega(\rho_F)$ be an offset, let $\mathbf{B}$ be a binary tensor:

$$\mathbf{B} = \begin{cases} 1 \text{ if } \rho_F(S, \mathbf{w}) > \ell \\ 0 \text{ if } \rho_F(S, \mathbf{w}) \leq \ell \end{cases} \quad (10)$$

and let $DT(B)$ be the Euclidean distance transform applied to the binary tensor, we introduce a new function $\gamma_\Omega : [0,1] \mapsto \mathbb{R}$ defined as follows: $\gamma_\Omega(\mathbf{B}) = DT(\mathbf{B}) + DT(\mathbf{I}-\mathbf{B}) + 0.5\mathbf{B}$. Let b be a scalar value corresponding to a level of the level set function ϕ, the four models can be defined modulating the initial level set function ϕ_0 and the velocity $\mathbf{v}$, according to the function γ_Ω, and the extended distance function:

$$\begin{aligned}
\mathcal{M}_1: &\quad \rho_F(S,\mathbf{w}) = \rho_I(S,\mathbf{w}) + H, \mathbf{v} = \gamma_\Omega(\mathbf{B}), \\
&\quad \phi(\mathbf{w},0) = \|\mathbf{w}-\mathbf{c}\| - r \\
\mathcal{M}_2: &\quad \rho_F(S,\mathbf{w}) = \rho_I(S,\mathbf{w}) + H, \mathbf{v} = \rho_F \\
&\quad \phi(\mathbf{w},0) = \gamma_\Omega(\mathbf{B}) - b \\
\mathcal{M}_3: &\quad \rho_F(S,\mathbf{w}) = H, \mathbf{v} = \gamma_\Omega(\mathbf{B}), \\
&\quad \phi(\mathbf{w},0) = \|\mathbf{w}-\mathbf{c}\| - r \\
\mathcal{M}_4: &\quad \rho_F(S,\mathbf{w}) = H, \mathbf{v} = \rho_F \\
&\quad \phi(\mathbf{w},0) = \gamma_\Omega(\mathbf{B}) - b
\end{aligned} \tag{11}$$

Note that the distance function ρ and its gradient are used as the velocity of the level set function evolution in normal direction. Therefore, the extended distance function can enforce the level set function to converge to the desired location avoiding unwanted points. The four models are resumed as follows:

Model $\mathcal{M}_1$ Here the distance function ρ_F has a global minimum located at the voxels closer to the point $\mathbf{w}$ belonging to the structural part to be singled out. In fact, ρ_F is always positive, as it is each term in equation (8), though it is not smooth, except near the target points. This is the reason why we introduced the offset ℓ. Furthermore, $\mathbf{B}$ is a binary tensor taking on values 1 or 0 and the function γ_Ω is used to define the velocity $\mathbf{v}$ in (11). Level set function is initialized using a general shape.

Model $\mathcal{M}_2$ In this model the extended distance function is the same as in the previous one, which is here used as velocity to evolve the level set function. Conversely the distance transform function γ_Ω is used for the level set function initial guess, namely for ϕ_0. Note that here $\phi_0 = \gamma_\Omega(\mathbf{B}) - b$, where b is needed to extrapolate the function. The evolution of this initialized level set function is governed by ρ_F, adopting the *normal direction of motion* model. Smoothness in the region close to target points ensures convergence only if b is suitably trimmed.

Model $\mathcal{M}_3$ The same parametrization applied to model $\mathcal{M}_1$ applies to this model, the major difference being in the definition of ρ_F, here lacking the first term ρ. Also in this case ρ_F is not smooth (except close to the target points), but has a global minimum close to the points corresponding to the structural part to be singled out. Like in $\mathcal{M}_1$, applying the function γ_Ω a smooth function is obtained, whose minimum is still located where the minimum of ρ_F is located. As in model $\mathcal{M}_1$ this smooth function is used as velocity to evolve the level set function, considering a general shape for the initial guess.

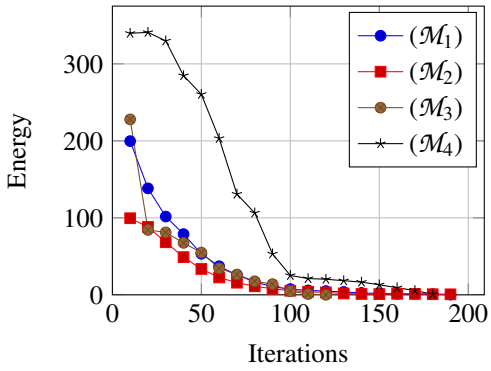


Figure 2: Energy VS iterations in the four methods, for the first scene element of Figure 3.

Model $\mathcal{M}_4$ For this method we can just note that it is the same as model $\mathcal{M}_2$ apart from the definition of ρ_F here lacking the first term ρ as in model $\mathcal{M}_3$. The term ρ has been dropped as ρ has been used to deviate the reference normal assigned to the domain Ω.

Finally, energy minimization with the above models amounts to use the energy function of eq. (3), and minimizing with respect to the above defined feature based distance functions, namely:

$$\arg\min_{\mathbf{w}} E(\Gamma, \rho_F(S,\mathbf{w})) \tag{12}$$

5 IMPLEMENTATION

As previously highlighted the voxel space $\Omega^\star$ is meaningful only for feature labeling, and for the computation of the distance function $\rho(S,\mathbf{w})$. Let V be the crust of voxels close to the original input points. For each $\omega_{i,j,k} \in V$ we define a window $\mathcal{W}(\omega_{i,j,k})$ of fixed size and the smoothness of K_i is checked within this window. More precisely, smoothness for each voxel in $\mathcal{W}(\omega_{i,j,k})$ is defined to be the squared difference between the norm of the curvature gradient and the average of the curvature gradient norm in $\mathcal{W}(\omega_{i,j,k})$. If this value is less than a threshold τ then the local window is accepted as smooth. The geometric based distance function $\rho_{F^\star}(S,\mathbf{w})$ is, then, computed for the specified $F^\star$. Minimization is finally obtained and a mesh is extracted for the selected points. At the end all voxels close to the final mesh are excluded for further processing. The procedure is iterated, choosing another voxel in V randomly, up to when V is emptied. This procedure is summarized in Algorithm 1.

As noted in Section 4, the initialization of the level set function can use any shape, except for the ap-

Input : Unorganized point cloud S, geometric features $F(\mathbf{x}_i) = \{K, h, \mathbf{n}\}$, threshold τ
Output: Minimal surface energy $E(\Gamma, \rho_{F^\star}(S, \mathbf{w}))$ for each model $\mathcal{M}_i$, $i \in \{1,2,3,4\}$.

Build the voxel space $\Omega^\star \subset \Omega$;
$\forall\, \mathbf{w} \in \Omega^\star$ compute $\rho(S, \mathbf{w})$;
$F(\omega^\star_{i,j,k}) = F(\mathcal{N}(\omega^\star_{i,j,k}))$;
$V = \{\omega_{i,j,k} \mid \rho(S, \omega^\star_{i,j,k}) < 0.01\}$;
while $V \neq \varnothing$ **do**
  Select a voxel $\omega_{i,j,k} \in V$;
  if $\forall\, \omega_{i,j,k} \in \mathcal{W}(\omega_{i,j,k})$, $\left(\|\nabla K(\omega_{i,j,k})\| - \frac{1}{|\mathcal{W}(\omega_{i,j,k})|}\int_{\mathcal{W}(\omega_{i,j,k})} \|\nabla K(\omega_{i,j,k})\| d\omega\right)^2 \leq \tau$ **then**
    $F^\star = F(\omega^\star_{i,j,k})$;
    $\forall\, \mathbf{w} \in \Omega^\star$ compute $\rho_{F^\star}(S, \mathbf{w})$;
    Initialize level set: $\phi(\mathbf{w}, 0)$;
    Compute $E(\Gamma, \rho_{F^\star}(S, \mathbf{w}))$;
    $V^o = \{\omega_{i,j,k} \mid argmin_{\mathbf{w}} E(\Gamma, \rho_{F^\star}(S, \mathbf{w}))\}$;
    $V = V \setminus V^o$;
  else
    Label $\omega_{i,j,k} \in V$ as *edge*;
    $V = V \setminus \mathcal{W}(\omega_{i,j,k})$;
  end
end

Algorithm 1: Structural parts extraction.

proaches $\mathcal{M}_2$ and $\mathcal{M}_4$. This is the main reason why the algorithm requires many iterations to converge (see Figure 2), as typically the starting surface lies very far away from the input points. For the level set evolution we adopt a WENO5 approach, and the Godunov scheme for the normal direction of motion. For stability, during evolution, the correct space derivative (right or left), is chosen according to the sign of the velocity of $\mathbf{w}$. On the other hand the time step is defined according to the CLF conditions (see (Osher and Fedkiw, 03)). We control the time step by taking only a fraction of the maximum allowed time step. This slows down the convergence too, though it guarantees better final results. Level set evolution is also implemented to run on a GPU. This makes the total evolution time reasonable, but not yet useful for real-time applications. Moreover, especially for the models $\mathcal{M}_2$ and $\mathcal{M}_3$ the level set update must be evaluated only in the narrow band close to the interface, and a periodic reinitialization, according to (Sussman and Fatemi, 99), must be performed to ensure a good reconstruction.

The proposed implementation exploits GPU to compute the distance functions and to evolve the level set toward its minimum. The bottleneck in computation time is given by the distance functions evaluation, which is strongly affected by the size of S. The level set evolution is also time expensive, but this is due to the initialization and the chosen time step value, as noted before.

6 Experiments

The proposed approach has been tested with manually built point sets of simple scene elements of an approximate size of $2 \times 2 \times 2$ meters. This to evaluate the structural component separability in terms of planes, which are separated by their normal direction, and have a zero curvature, as in the first row of Figure 3. Moreover, different horizontal planes in the scene are separated according to the h value, as in the second row of Figure 3. The surface variation of (Pauly et al., 03) is useful to evaluate different shapes, but it is not enough to distinguish between a cylinder and a sphere, but a sphere and a cone, as can be seen in the third and fourth rows of Figure 3.

In Figure 3 the meshes extracted after level set evolution are shown. To evaluate accuracy of the reconstruction we look at the average distance among the vertex of the reconstructed mesh and the input point cloud S. This distance is reported in Table 2. One can see that the approximation of the minimal surface energy is quite accurate. This is because the average distance, in the most meaningful cases, is less than a centimeter, given that the objects in Figure 3 have size of meters. A parameter which influences this final distance is the offset ℓ, but only for the reconstruction models $\mathcal{M}_1$ and $\mathcal{M}_3$, as the transformed distance function $\gamma_\Omega(\mathbf{B})$ is used in these models to drive the level set evolution. As expected the resulting mesh is more noisy for the model $\mathcal{M}_3$, but especially for model $\mathcal{M}_4$ as in this case the level set evolution is governed only by H, which is smooth only very close to the target points. This aspect is overcome in model $\mathcal{M}_2$, due to the addition of $\rho(S, \mathbf{w})$.

Results for Gaussian added noise, with increasing variance, are shown in Figure 4. To compute precision and recall we consider that an input point belongs to

Table 2: Accuracy results for the four proposed models. Both mean and max distances are reported in meters.

	Mean	Max		Mean	Max
$\mathcal{M}_1$	0.007	0.035	$\mathcal{M}_3$	0.014	0.011
$\mathcal{M}_2$	0.004	0.09	$\mathcal{M}_4$	0.021	0.21

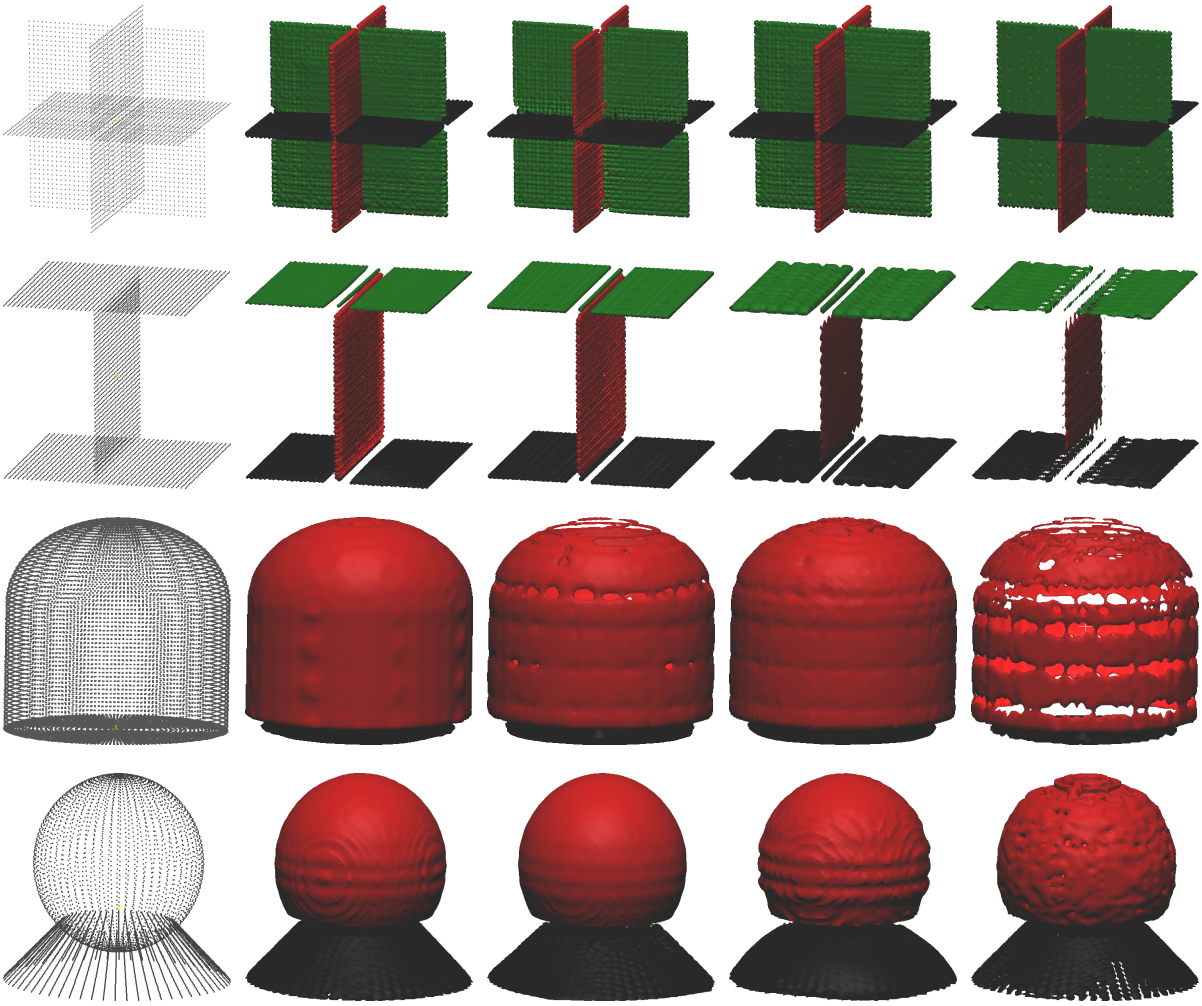


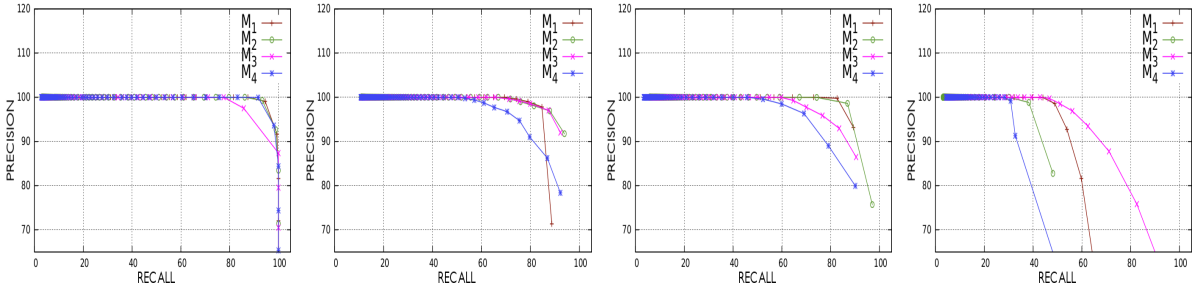
Figure 3: From left to right: results according to the four models in ascending order. Objects' bounding box are approximately $2 \times 2 \times 2$ meters. Number of points is approximately $10k$, for each object.

a segment if its distance from its closer vertex, in the extracted mesh, is less than a threshold. Charts of Figure 4 illustrate the results obtained increasing this threshold. 100% of precision is obtained when the threshold distance is quite high. On the other hand if the noise level is low, decreasing this threshold does not affect the good performance of precision.

7 Conclusions

The paper introduces a novel approach to the extraction of the structural parts of simple scene elements, contextually obtaining a mesh for these parts. The methods return a set of meshes labeled with the feature values used to extract it, and return no meshes for points belonging to edges amid different structural parts. The main contribution is the mesh reconstruction of singled out point clouds, despite the point cloud is a finite set of unorganized points. Results are accurate in general, despite the fact that the framework is quite simple requiring only few parameters for features computation. In addition, voxel size (time step) is determined by the average density of the provided point cloud. The reported results show that the first and the second methods are the most accurate.

This work is a preliminary step for 3D object segmentation. 3D object segmentation requires structural parts decomposition of the unknown surface approximated by the point cloud, together with a precise parts labeling. We cannot yet evaluate completely whether the geometric features are sufficient to single out all the structural parts of a scene. Still, the method can be easily extended to fulfil further features requirements,



(a) $\sigma^2 = 0$ (b) $\sigma^2 = 0.1$ (c) $\sigma^2 = 0.5$ (d) $\sigma^2 = 2$

Figure 4: From Left to right: precision and recall charts for the simulated objects, for the four methods, with increasing noise.

adding further features in the computation of equation (8) and (9). Also more precise geometric features, such as the principal curvatures could provide exact categories for structural parts.

ACKNOWLEDGEMENTS

This research is currently supported by EUFP7-ICT-Project TRADR 60963

REFERENCES

Golovinskiy, A. and Funkhouser, T. ('09). Min-cut based segmentation of point clouds. In *IEEE Workshop on Search in 3D and Video (S3DV) at ICCV*.

Hoppe, H., DeRose, T., Duchamp, T., Mcdonald, J., and Stuetzle, W. ('92). Surface reconstruction from unorganized points. In *SIGGRAPH*, pages 71–78.

Ioannou, Y., Taati, B., Harrap, R., and Greenspan, M. A. ('12). Difference of normals as a multi-scale operator in unorganized point clouds. In *3DIMPVT*, pages 501–508.

Karni, Z. and Gotsman, C. ('00). Spectral compression of mesh geometry. In *27th Conf. on Comp. Graphics and Int. Techniques*, SIGGRAPH, pages 279–286.

Kustra, J., Jalba, A., and Telea, A. (2014). Robust segmentation of multiple intersecting manifolds from unoriented noisy point clouds. pages 73–87.

Liang, J., Park, F., and Zhao, H. ('13). Robust and efficient implicit surface reconstruction for point clouds based on convexified image segmentation. *J. Sci. Comput.*, 54(2-3):577–602.

Lloyd, S. ('06). Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137.

Lorensen, W. E. and Cline, H. E. ('87). Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169.

Ma, T., Wu, Z., Feng, L., Luo, P., and Long, X. ('10). Point cloud segmentation through spectral clustering. In *ICISE*, pages 1–4.

Nguyen, A. and Le, B. (13). 3d point cloud segmentation: A survey. In *6th Conf. in Robotics, Automation and Mechatronics (RAM)*, pages 225–230.

Nurunnabi, A., Belton, D., and West, G. (12). Robust segmentation in laser scanning 3d point cloud data. In *DICTA*, pages 1–8.

Osher, S. and Fedkiw, R. (03). *Level Set Methods and Dynamic Implicit Surfaces*. Springer.

Pauly, M., Keiser, R., and Gross, M. ('03). Multi-scale feature extraction on point-sampled surfaces. *Computer Graphics Forum*, 22(3):281–289.

Rusu, R. B. ('09). *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Computer Science dpt. TUM.

Schnabel, R., Wahl, R., and Klein, R. ('07). Efficient RANSAC for point-cloud shape detection. *Comput. Graph. Forum*, 26(2):214–226.

Schreiner, J., Asirvatham, A., Praun, E., and Hoppe, H. ('04). Inter-surface mapping. SIGGRAPH, pages 870–877, NY, USA. ACM.

Shamir, A. (08). A survey on mesh segmentation techniques. *Comput. Graph. Forum*, 27(6):1539–1556.

Sussman, M. and Fatemi, E. ('99). An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. *SIAM J. Sci. Comput.*, 20(4):1165–1191.

Tran, T., Cao, V., Nguyen, V., Ali, S., and Laurendeau, D. ('14). Automatic method for sharp feature extraction from 3d data of man-made objects. In *GRAPP 2014*, pages 112–119.

Turner, E. and Zakhor, A. ('14). Floor plan generation and room labeling of indoor environments from laser range data. In *GRAPP 2014*, pages 22–33.

Wirjadi, O. (07). Survey of 3d image segmentation methods. Technical Report 123, Fraunhofer (ITWM).

Zhao, H. K., Stanley, O., Barry, M., and Myungjoo, K. ('98). Implicit, nonparametric shape reconstruction from unorganized points using a variational level set method. *Computer Vision and Image Understanding*, 80:295–319.

Zhao, H. K., Stanley, O., and Ronald, F. ('01). Fast surface reconstruction using the level set method. In *IEEE Workshop on Variational and Level Set Methods in Computer Vision*, pages 194–201.