SURFACE-BASED SEGMENTATION OF 3D POINT CLOUD DATA IN THE VECTOR FIELD REPRESENTATION

Van Tung Nguyen and Denis Laurendeau

Computer Vision and System Laboratory, Laval University, Quebec City, QC, Canada

Keywords: 3D Segmentation, Surface Modeling, Volumetric Representation.

Abstract: In this paper, we propose an approach for 3D segmentation of point cloud data based on the vector field representation. A volumetric surface representation named the vector field is first computed for the input point cloud. The data description in a voxel of the vector field is then classified into eight surface types by computing the sign of curvatures for the closest point of the voxel on the surface. A region-growing sheme based on bivariate fuctions fitting is finally applied to the closest points of these voxels for refining the segmented regions. This surface-based approach is entirely designed in the vector field surface representation and the advantages of using the vector field are discussed. Experiments demonstrate the peformance of the method.

1 INTRODUCTION

Segmentation is an essential step that can be found in many research areas in computer vision such as shape retrieval, compression, registration, and object recognition. Basically, 3D segmentation consists of segmenting the input data into a set of meaningful "geometric" parts or regions in the scene. Due to the importance of this step in 3D applications, many segmentation techniques have been developed that can roughly be classified in two main categories: edge-based and surface-based techniques (Hoover et al., 1996)(Besl and Jain, 1988). In general, the former use edge detection to create region boundaries (Hoover et al., 1996)(Zhang et al., 2008) while the latter use analytic surface description to divide the image into disjoint regions (Besl and Jain, 1988)(Besl and Jain, 1986)(Zheng et al., 2008)(Gelfand and Guibas, 2004).

In this paper, a surface-based segmentation technique for 3D point cloud data is presented. Our goal is to propose a segmentation technique that always minimizes computational load at every steps in the algorithm by using the vector field representation. The vector field representation, introduced in the next sections helps to reduce the complexity of neighbourhood operations often required by surface-based segmentation approaches. Our approach first computes the vector field from the input data. Then points belonging to regions with similiar geometric properties are classified into eight types of surface geometry based on local curvature properties computed from the vector field representation. A region growing scheme based on the fitting of low- order bivariate functions is then performed on the initial set of regions to refine the segmentation.

The paper is organized as follows. First the vector field framework and curvature estimation approach are presented in Section 2. Section 3 presents the proposed segmentation technique including the initial region finding process and the refinement process. Next, experiments show the results demonstrating the performance of the method. Section 5 discusses the results and proposes future work.

2 CURVATURE ESTIMATION IN THE VECTOR FIELD REPRESENTATION

We first introduce the terminology relevent to the vector field surface representation computed for point-set data and then extend the framework to introduce a computational method for curvature estimation on the vector field.

2.1 The Vector Field Representation

In 3D modeling, representation of 3D data is a very important issue since it has an impact on the computational complexity of the segmentation steps. Three main 3D representation methods- mesh, volumetric grids and polynomial representations- have been proposed in the literature(Zheng et al., 2008)(Tubic et al., 2002)(Zhang et al., 2008). Each type of representation has its advantages and disadvantages. In addition, the chosen representation of the input data affects the data segmentation and surface modelling steps significantly. In 2002, Tubic et al. (Tubic et al., 2002) proposed a new implicit surface representation called the "vector field". The vector field framework has demonstrated its capacity of unifying all steps of iterative 3D modeling (registration, integration, modelling and visualization) in one type of data representation with linear computational complexity (Tubic et al., 2004).

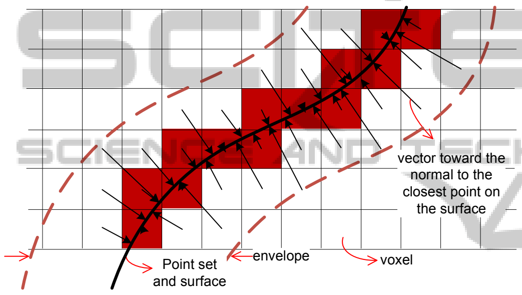

vector toward the normal to the closest point on the surface
Point set and surface
envelope
voxel

Figure 1: Vector field representation.

In essence, the vector field is a volumetric representation where each voxel encodes the vector pointing from the voxel center to the closest point on the surface to be modeled. Suppose that point cloud data is available, the vector field is built by updating the covariance matrix C at each voxel located nearby data point $\mathbf{p}_i$ inside an envelope of size ε (Tubic et al., 2002):

$$C = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{p}_i - \tilde{\mathbf{p}})(\mathbf{p}_i - \tilde{\mathbf{p}})^T = \frac{1}{N}\sum_{i=1}^{N}\mathbf{p}_i\mathbf{p}_i^T - \mathbf{p}_i\mathbf{p}_i^T \quad (1)$$

where $\tilde{\mathbf{p}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{p}_i$ and N the number of points that have already participated in the voxel updates.

The eigenvector associated with the smallest eigenvalue of the covariance matrix represents the normal vector $\mathbf{N}$ to the tangent plane at the closest point, then the direction of the vector field $\mathbf{F}(v)$ at the voxel $\mathbf{v}$ is computed as:

$$\mathbf{F}(v) = \mathbf{N}\langle \mathbf{N}, \tilde{\mathbf{p}} - \mathbf{v}\rangle \quad (2)$$

where $\langle . \rangle$ is the scalar product.

Figure 1 shows a 2D view of a vector field representation, all the voxels within the envelope (bounded by the dashed line) are updated with the information of the point set on the surface (in black). The envelope is used to determine the number of local voxels surrounding the surface in which the vector field is updated for each incoming data point. The envelope size and voxel size is always a trade-off. A large envelope size or a small voxel size results in a dense vector field and increases the accuracy at the reconstruction step but increases computational complexity and memory requirements for storing the vector field. Practically, the envelope size is chosen twice the voxel size. As an expansion to this basic definition of the vector field, we define two types of voxels: *active voxels* and *surface voxels*. An *active voxel* is one that contains vector field information pointing toward the closest point on the surface (the voxels within the envelope in Figure 1). A *surface voxel* is an *active voxel* crossed by the surface (for the point cloud data, that is an *active voxel* contains the points on the surface). This condition can be easily checked by finding whether or not the closest point encoded by a *surface voxel* is located inside this voxel (the *surface voxels* are colored in red in the example on Figure 1). Given the extension, it will be shown next that only the curvatures at the closest points of the *surface voxels* are computed using the closest points of neighbouring *active voxels* and only the closest points of the *surface voxels* contribute to the estimation of the fitting functions in the segmentation process.

2.2 Curvature Estimation

Based on the vector field framework described above, we propose a scheme for estimating curvatures at the closest point of each *surface voxels* (Nguyen and Laurendeau, 2011). To do this, at a *surface voxel*, we detect the closest point on the surface by the encoded vector field value then, the other closest points of neighbouring *active voxels* are used to estimate the curvature at this point. This approach based on the vector field departs significantly from approaches that compute curvature directly from the point cloud data. As shown in Figure 2, to collect neighbouring points of a given point $\mathbf{p}$ for estimating the curvatures, all the points in the circle of radius r centered at $\mathbf{p}$ are searched. In Figure 2.a, all points in the green region are obtained by the search. This search is hard to achieve in a point cloud because nearest neighbours must be found. However, in the vector field representation, this search becomes an easy task by detecting the neighbouring voxels, then collecting the closest points of these *active voxel* that contribute to curvature estimation because nearest information is stored directly in the field. In Figure 2.c, all neighbour voxels of a given voxel containing $\mathbf{p}$ are detected and highlighted by light green colour, but only *active vox-*

els are considered for the curvature estimation (light green voxel in Figure 2.d). Changing the size of radius r to have more neighboring points in the search is approximated by adding more layers of neighbouring voxels in the curvature computation with the vector field.(Figure 2.e and Figure 2.f show example of two layers for detecting the neighbour voxels). Practically, the size of r or extention of more layers search in the vector field for collecting more neighboring points is chosen suitable with the geometry of the scene.

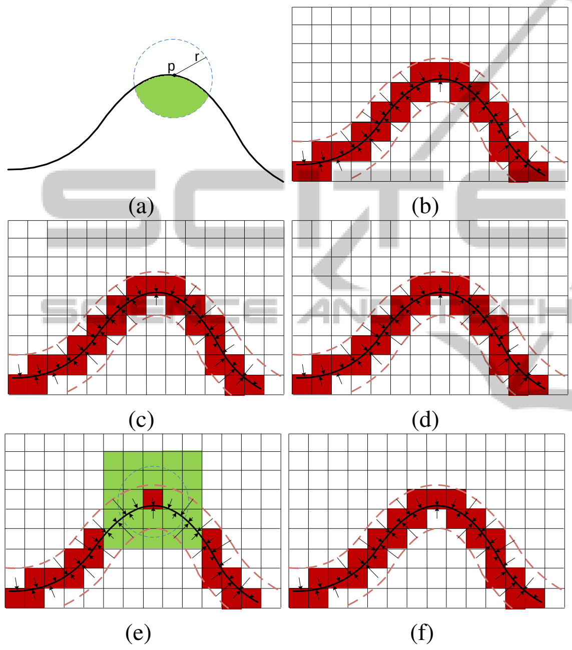


Figure 2: Illustration of neighbouring voxels detection for cuvature computation using the vector field. (a) shows the search with point cloud directly, (b) is the encoded vector field, (c) shows all the neighbouring voxels detected in green for the center voxel in red and (d) shows the neighbouring active voxels taken into account for curvature estimation, (e),(f) shows an example of the two voxel layers search.

After detecting the neighbours to be considered for curvature estimation, the method described by Chen and Schmitt (1992) is used for estimating the curvatures at a point using normal vectors and their neighbouring points (Chen and Schmitt, 1992)(Dong, 2005). In our case, the normal vectors at the closest points are implicitly encoded in the vector field (as shown in Figure 1) since the field contains the direction of the closest point at the surface and Chen and Schmitt's method is exploited as follows.

Let us assume that a point $\mathbf{p}$ has m neighbours. The main idea in this method is to choose a suitable coordinate system $(\hat{\mathbf{e}}_1,\hat{\mathbf{e}}_2)$ for a group of unit length tangent vectors $\mathbf{t}$ on the local tangent plane at point $\mathbf{p}$. The normal vectors at that point and its neigbours are used to estimate the normal curvatures $k_n(\mathbf{t}_i)$ along the tangent direction $\mathbf{t}_i$ on the local tangent plane at $\mathbf{p}$ as follows (Chen and Schmitt, 1992)(Dong, 2005):

$$k_n(\mathbf{t}_i) = -\frac{\langle \mathbf{p}_i - \mathbf{p}, \mathbf{N}_i - \mathbf{N}\rangle}{\langle \mathbf{p}_i - \mathbf{p}, \mathbf{p}_i - \mathbf{p}\rangle} \tag{3}$$

$$\mathbf{t}_i = \frac{(\mathbf{p}_i - \mathbf{p}) - \langle \mathbf{p}_i - \mathbf{p}, \mathbf{N}\rangle \mathbf{N}}{\|(\mathbf{p}_i - \mathbf{p}) - \langle \mathbf{p}_i - \mathbf{p}, \mathbf{N}\rangle \mathbf{N}\|} \quad (i\text{=}1,2..m) \tag{4}$$

where $\|.\|$ is the Euclidean norm of a vector and N_i and N are normal vectors at $\mathbf{p}_i$ and $\mathbf{p}$. Suppose that $k_n(\mathbf{t}_{id})$ is the maximum of the normal curvatures corresponding to the tangent direction t_{id} . Then, the special coordinate system $(\hat{\mathbf{e}}_1,\hat{\mathbf{e}}_2)$ on the tangent plane at $\mathbf{p}$ can be chosen as follows:

$$\hat{\mathbf{e}}_1 = \mathbf{t}_{id}, \qquad \hat{\mathbf{e}}_2 = \frac{\hat{\mathbf{e}}_1 \times \mathbf{N}}{\|\hat{\mathbf{e}}_1 \times \mathbf{N}\|} \tag{5}$$

From this, a set of m equations is obtained according to Chen and Schmitt's method:

$$k_n(\mathbf{t}_i) = a\cos^2(\theta_i) + b\cos(\theta_i)\sin(\theta_i) + c\sin^2(\theta_i) \tag{6}$$

where θ_i is the angle between t_i and $\hat{\mathbf{e}}_1$ $(i\text{=}1,2..m)$. A least-squares method is used for solving the set of equations for the coefficients a, b and c. This leads to the Gaussian curvature(K), mean curvature (H) and two principal curvatures $k_{1,2}$ as follows:

$$K = ac - b^2/4, \quad H = (a+c)/2, \quad \text{and}$$
$$k_{1,2} = H \pm \sqrt{H^2 - K} \tag{7}$$

3 SEGMENTATION IN THE VECTOR FIELD REPRESENTATION

As presented by Besl in 1988, using the sign of the Gaussian and mean curvatures at points on a smooth surface, the surface can be decomposed into a union of simple surface patches that are approximated by bivariate polynomials with order lower than four (Besl and Jain, 1986). In this section, we borrow Besl's approach for segmenting surfaces based on the sign of the curvatures and then use a region growing fitting process to refine the initial segmentation. With some assumptions, Besl's approach is implemented in the vector field representation.

3.1 Initial Segmentation with the Sign of Curvatures

In section 2.2 we presented a method for estimating the curvatures at the closest points of the *surface voxels* in the vector field representation. Based on the fact that surface characteristics are almost identical on a small region surrounding a point, we use the closest points as *"prominent points"* that characterize surrounding regions occupied by the *surface voxels*. This means that, based on the surface shape determined by the curvature signs at a *"prominent point"* on the surface, the region occupied by the corresponding *surface voxel* belongs to the same surface type. Figure 3 shows the eight fundamental surface shapes possible based on the Gaussian and mean curvatures at a *prominent point* (Besl and Jain, 1986).

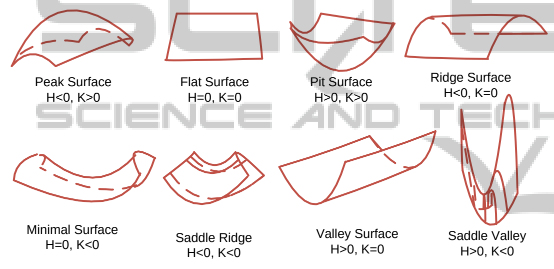


Figure 3: Eight fundamental surface types (Besl and Jain, 1986).

Therefore, in this initial segmentation step, the surface is decomposed into small disjoint regions belonging to one of eight possible surface types by labeling surface types of *"prominent point"*. Figure 4 shows an example of how this initial segmentation is obtained. A surface (Figure 4.a) is encoded by the vector field as shown in Figure 4.b. The *surface voxels* with different colours express different shape types (illustrated in Figure 3) of the *"prominent point"* of these voxel respectively. Then the regions occupied by these *surface voxels* have the same shape type (Figure 4.c). Figure 4.d shows the surface classified by regions with different shape type expressed by different colours. Section 4 will shows some typical results obtained by the first segmentation step.

3.2 Refined Segmentation with Region Growing

The region growing scheme is responsible for merging the coarse segmented regions obtained in the above initial step to create larger surface patches. In fact, it is the process of spreading out initial patches from a seed region. As presented by Besl(Besl and

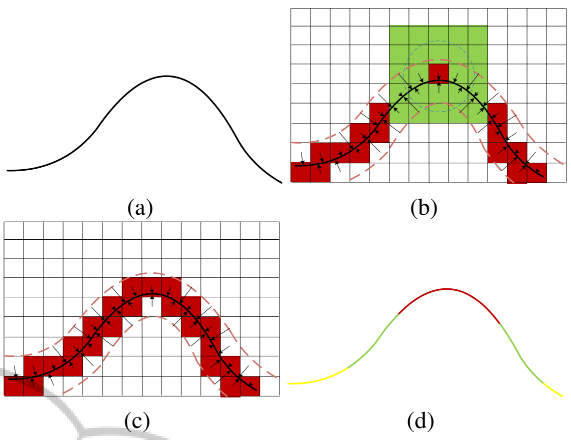


Figure 4: Illustration of ideal results obtained from the initial segmentation. (a) is an example of input surface; (b) is the vector field with the surface types stored in the surface voxels in different colours; (c) regions crossed by the voxels have the same surface types; (d) regions in different colour are obtained by the initial segmentation.

Jain, 1988), the curvature-sign primitives classified by the initial step presented above can be approximated well by a bivariate function with order lower than four as illustrated next.

Suppose that a smooth surface is a twice-differentiable function *z=f(x,y)*. The form of the fitting function can be written as follows:

$$\begin{aligned}\hat{f}(m,\mathbf{a};x,y) &= \sum_{i+j\leq m} a_{ij}x^i y^j \\ &= a_{00}+a_{10}x+a_{01}y+a_{11}xy+a_{20}x^2 \\ &+a_{02}y^2+a_{21}x^2y+a_{12}xy^2+a_{30}y^3 \\ &+a_{03}y^3+a_{31}x^3y+a_{22}x^2y^2 \\ &+a_{13}xy^3+a_{40}x^4+a_{04}y^4 \end{aligned} \quad (8)$$

Where, m$\leq$4 is the order of the function and the function can be expressed for a planar, biquadratic, bicubic and biquartic surface with the length of the coefficient vector **a** corresponding to 3, 6, 10 and 15. The surface fitting function minimizes the least-squares error metric

$$\varepsilon = \frac{1}{N}\sum_{i=1}^{N}(\hat{f}(m,\mathbf{a};x,y)-f(x,y))^2 \quad (9)$$

Where N is the number of points contributing to the estimation of the fitting function. The region-growing scheme based on this fitting of functions in the vector field representation is as follows.

First, the surface shape types of the prominent points set (the closest points of *surface voxels*) obtained at the initial step are stored at the corresponding *surface voxels* and coordinates of the prominent points are extracted and stored in a list of set of prominent points for the next processing step as shown in Figure 5 (different colours of voxels are used for different shape types). This configuration helps to avoid checking all voxels to find a *surface voxel* as a starting point. Instead of checking all voxels to detect a *surface voxels*, starting from a point in this list allows to detect which *surface voxel* the point falls into.

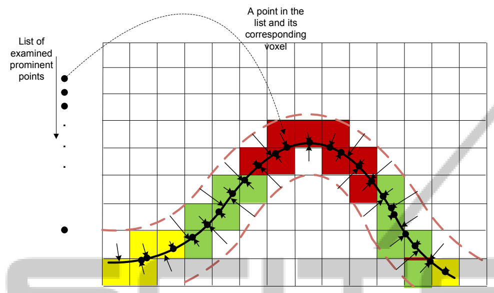

List of examined prominent points
A point in the list and its corresponding voxel

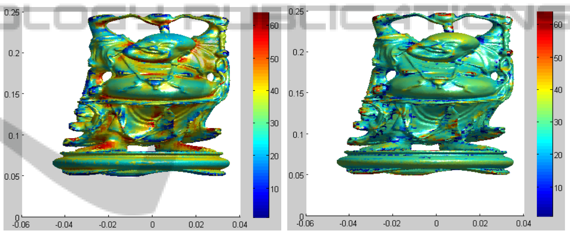
Figure 5: Explanation of the configuration for growing fitting in the vector field.

Two key elements of a region growing process are finding a starting seed point and defining a stopping criterion. For finding a good seed region, we start with a prominent point in the list to detect to which *surface voxel* it belongs to. Based on the detected *surface voxel*, we consider its neighbouring *surface voxels* to check whether or not these voxels are labeled with the same surface type. If it is the case, then we have a "trustable" seed region to start the region growing process with the closest points corresponding to the *surface voxels*. The expansion process to find neighboring closest points for the surface fitting procedure becomes easy by detecting neighboring *surface voxels* (the same as shown in the Figure 2). When a *surface voxel* is considered and its prominent point has successfully contributed to the surface fitting estimation, it is assigned the label 'visited' to avoid multiple scan for the corresponding closest point. For a seed region, the stopping criterion is when the error metric in the equation (9) is greater than a threshold while the highest order of the bivariate function has been reached (m=4). The idea of variable-order polynomial fitting proposed by Besl is used: each time a seed region is chosen, we first try to fit with a plane to the region (length of the coefficent **a** is 3). If it is not suitable (error metric is larger than the threshold), we move to higher order surfaces (biquadratic, bicubic, and biquartic surface, where **a** is 6, 10 and 15 respectively). The overall decision to stop the fitting process of the region growing on the surface is to check if all selected prominent points from the list fall into a 'visited' *surface voxel*. This means that all regions on the surface have been visited.

4 EXPERIMENTAL RESULTS

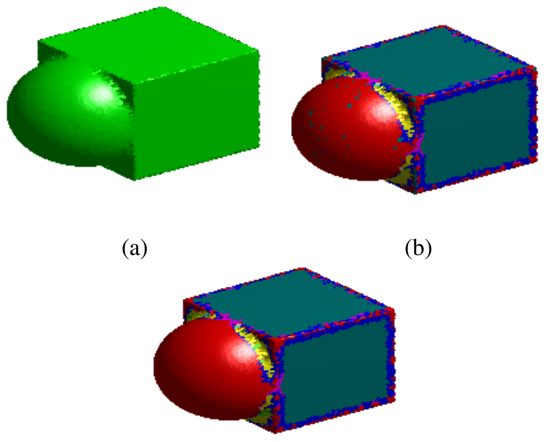
For validation, the proposed method is applied to some popular testing objects. The envelope size of the vector field is always chosen as twice of voxel size, and the size of voxel is chosen depending on resolution of object. To get the sign of Gaussian and Mean curvatures, curvature is set to zero when it is smaller than 0.4% of the maximum absolute value of curvature. The threshold for the error in the fitting process is set to 0.5 of the voxel size.

For curvature estimation using the vector field, Figure 6 shows the colormap of Gaussian and Mean curvatures estimated for the budda model. Colour change corresponds to a change in minimum to maximum estimated value of the curvatures. The resolution of the budda model is good, so the estimated curvature values are reliable given by the smooth change in colours in the figure. With this model, the voxel size is set 0.001.



(a) Gaussian curvature. (b) Mean curvature.

Figure 6: Colormap of estimated curvatures for the budda model.



(a) (b)

(c)

Figure 7: Surface segmentation on a simple object. (a) view of the surface of the surface object, (b) initial segmentation, (c) refined segmentation.

Figure 7 shows the segmentation of a simple object. Figure 7.b shows the result of the initial segmentation. Surface regions of different surface types are shown in different colours. The initial step includes some fragmented regions caused by wrong estimated curvature sign. Figure 7.c shows the improvement of the result after applying the refinement segmentation process. Because the vector field implicitly contains reconstruction information, the algorithm for the refinement process is designed to refine the shape type of the surface to get segmented regions separated by shape boundaries. The voxel size is set 0.18 for this object.

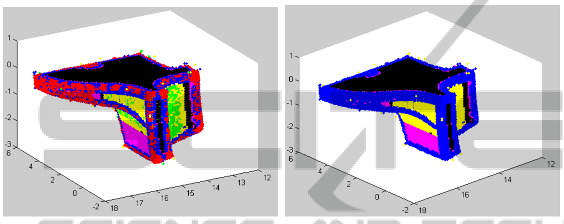

(a) (b)

Figure 8: Surface segmentation on an object. (a) initial segmentation, (b) refined segmentation.

Figure 8.a shows the prominent points (closest points on the surface of surface voxels) reconstructed for another object after the initial segmentation step. Different colours express different surface types of the points. Figure 8.b shows the improved result composed of reliable regions after the refinement process. Voxel size is set 0.065 for this object. The algorithm is currently being improved to reduce the thickness of boundaries on these objects.

5 CONCLUSIONS

A new mechanism for 3D segmentation was introduced for point-set data in the vector field surface representation. An initial segmentation process is first proposed by segmenting the surface into disjoint regions labeled by eight fundamental surface types. Then the segmented regions are improved by a region growing process based on bivariate function fitting. Designing the segmentation mechanism in the vector field allows to keep computation simple and avoid complex nearest neighbour search for curvature estimation and spreading out in the region growing process. Several directions are possible for future work. For instance an adaptive scaling in computing local features such as curvatures could help to avoid seed region to be too fragmented. Robustness to noise will also be investigated.

REFERENCES

Besl, P. J. and Jain, R. C. (1986). Invariant surface characteristics for 3D object recognition in range images. *Computer Vision, Graphics, and Image Processing*, 33:3380. ACM ID: 7056.

Besl, P. J. and Jain, R. C. (1988). Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(2):167–192.

Chen, X. and Schmitt, F. (1992). Intrinsic surface properties from surface triangulation. In *Proceedings of the Second European Conference on Computer Vision*, ECCV '92, pages 739–743, London, UK. Springer-Verlag.

Dong, C. Wang, G. (2005). Curvatures estimation on triangular mesh. *Journal of Zhejiang University (SCIENCE)*, Journal of Zhejiang University (SCIENCE):128–136.

Gelfand, N. and Guibas, L. J. (2004). Shape segmentation using local slippage analysis. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '04, page 214223, New York, NY, USA. ACM.

Hoover, A., Jean-baptiste, G., Jiang, X., Flynn, P. J., Bunke, H., Goldgof, D., Bowyer, K., Eggert, D., Fitzgibbon, A., and Fisher, R. (1996). An experimental comparison of range image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Nguyen, V. and Laurendeau, D. (2011). A global registration method based on the vector field representation. In *2011 Canadian Conference on Computer and Robot Vision (CRV)*, pages 132–139. IEEE.

Tubic, D., Hebert, P., Deschenes, J., and Laurendeau, D. (2004). A unified representation for interactive 3D modeling. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, page 175182.

Tubic, D., Hebert, P., and Laurendeau, D. (2002). A volumetric approach for the registration and integration of range images: towards interactive modeling systems. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 3, pages 283–286 vol.3.

Zhang, X., Li, G., Xiong, Y., and He, F. (2008). 3D mesh segmentation using mean-shifted curvature. In *Proceedings of the 5th international conference on Advances in geometric modeling and processing*, GMP'08, page 465474, Berlin, Heidelberg. Springer-Verlag.

Zheng, B., Takamatsu, J., and Ikeuchi, K. (2008). 3D model segmentation and representation with implicit polynomials. *IEICE - Trans. Inf. Syst.*, E91-D(4):11491158.