Covariance Based Differential Geometry Segmentation Techniques for Surface Representation Using Vector Field Framework

Mana Eskandari[1] Denis Laurendeau[1]
[1]Computer Vision and System Laboratory, Dept. of ECE, Laval University
1065 Av. De la Médicine
Québec City, Canada, G1V 0A6
mana.eskandari.1@ulaval.ca
denis.lauredeau@gel.ulaval.ca

ABSTRACT

In this paper, the concepts of differential geometry traditionally applied to the segmentation of range maps is revisited in the context of implicit surface representation of unorganized point clouds. The paper shows that it is possible to combine covariance-based differential geometry and implicit surface representation methods to perform the segmentation of an unorganized point cloud (and not just a range map) into seven surface types. The acquisition of the point cloud data is achieved with handheld scanners used in metrology applications. The advantages of combining covariance-based differential geometry and implicit surface representation are that the segmentation does not require surface fitting nor does it require that all points be processed, thus reducing computational complexity. The segmentation approach is validated on synthetic data as well as point clouds borrowed from common datasets. Scans obtained from commercial metrologic handheld 3D sensors are also used for validation. The paper first presents the workflow commonly used for 3D scanning using handheld 3D scanners in the context of metrology. This is followed by a discussion on the different methods that are used for surface representation including the vector field, the implicit representation method exploited in this paper. Basic concepts of classical differential geometry for surface segmentation are presented. This is followed by the presentation of covariance-based differential geometry. The concepts of handheld 3D scanning, covariance-based differential geometry and implicit surface representation are then combined to achieve efficient segmentation of a point cloud into seven different surface types. Experimental results obtained on synthetic 3D data as well as real data demonstrate the segmentation approach.

Keywords
3D reconstruction, volumetric representation, geometry modeling.

1. INTRODUCTION

Applied metrology consists in the application of measurements in different fields such as quality control, inspection, product design and reverse engineering. If accuracy and precision are two important components of metrology, the time needed to achieve the measurements is a relevant issue, especially in the context of quality control in an industrial context for which a large number of parts have to be processed. The ease with which the scan can be obtained by users is also important when such users are domain specialists but not necessarily experts in 3D scanning.

Over the years, 3D sensors have become very popular because they are cheaper and easier to use than classical Coordinate Measuring Machines (CMM) while still achieving metrologic accuracy. In addition, 3D sensors can capture dense point clouds that convey the geometry of the object in real-time. Comparatively, capturing dense 3D point clouds with a CMM can be a very tedious and time consuming process. Capturing the geometry of parts is not only important in metrology but is also useful in many fields such as reverse engineering, design intent assessment and graphics rendering.

Among 3D sensors, handheld 3D sensors are of great interest because they allow the capture of 3D data on the specimens to be inspected in a very natural way which, in many aspects, resembles spray painting. As shown in Fig.1, scanning an object with a handheld 3D sensor consists in moving the sensor around the surface of the object of interest while the 3D coordinates of points at the surface are collected.

Most 3D handheld sensors are active sensors that project some sort of light pattern (laser point, laser stripes, laser crosses, white light patterns, Moiré fringes, etc., see [DANE2018] for an overview of different 3D sensing technologies) on the object to ease the image analysis process leading to the measurement of 3D coordinates. The 3D coordinates of points acquired from a pose (i.e. position and orientation) of the sensor are expressed in this local reference frame. The estimation of the rigid transformation (rotation and translation) between each pose of the sensor and a "global" reference frame is needed if the 3D points are to be expressed in a common reference frame. This global reference frame can be the initial pose of the sensor when the scanning process starts. The estimation of the rigid transformations is made easier if a real-time self-positioning strategy is used to compute the position and orientation of the sensor with respect to the object. One way of implementing self-positioning is to install markers (often retroreflective markers) at the surface of the object and to estimate the pose of the sensor with respect to these markers. The rigid transformation between different poses of the sensor is then readily available by exploiting registration algorithms such as the well-known Iterative Closest Point (ICP) approach [RUSI2001].

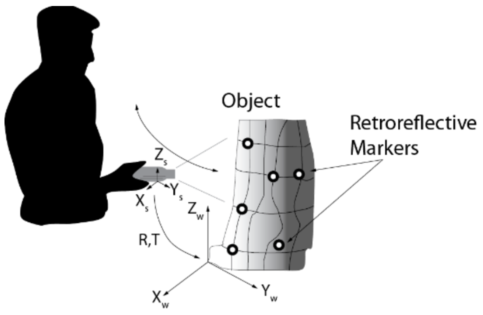


Figure 1. The 3D scanning process using a handheld scanner

Surface Representation

Once the cloud of unorganized 3D points covering the entire surface of the object has been captured, a model of the surface must be built if metrologic measurements are to be performed on the object. This model can also be used to analyze the geometry of the object. Two main representations can be exploited to build this model: *explicit* representations and *implicit* representations. The former representation makes the geometry supported by the point cloud explicit, for instance by building a triangular mesh [CHEN2012]. Such a mesh contains the connectivity between points in the cloud and is a compact representation of the geometry of the surface. This connectivity can take the form of a vertex-triangle list and a triangle-vertex list and can also include the information on the normal to the surface at each vertex. The latter representation rather encodes the geometric information contained in the point cloud implicitly into a volumetric structure composed of voxels. Two main types of voxel-based implicit representations have been proposed: the distance field [CURL2996] and the vector field [TUBI2002]. The volumetric structure must be processed a posteriori to produce a mesh representing the geometry explicitly. The Marching Cubes algorithm is often used for this task [LORE1987].

When handheld 3D sensors are used for *real-time* modelling (i.e. the model of the surface is built as the 3D points are measured), three tasks must be achieved: *i)* view registration, *ii)* view integration and *iii)* model visualization. View registration consists in the estimation of the rigid transformation between points of view from which the 3D data is collected. View integration aims at merging redundant 3D data common to two or more views. Finally, model visualization is the task of rendering the 3D model as it is being built so the user can observe the progression of the scan and plan the scanning strategy as points are being collected.

The advantage offered by an *explicit* representation is that a low-level model is readily available. However, it is not adapted to real-time modelling. The main reason for this is that the registration and integration steps rely on finding nearest neighbours and that the search for nearest neighbours to a point becomes too computationally expensive when the number of points increases. Updating a mesh as new 3D points are being collected is also impossible to achieve in real time. The advantage of using *implicit* representations is that some, such as the vector field, have demonstrated the ability to support the three modelling steps in real-time that cannot be achieved by other methods [KHAK2019]. As described next, a major advantage of the vector field representation is that it encodes the surface normal as well as information on the nearest neighbors in each voxel, thus enabling nearest neighbor search in linear time complexity. However, if an accurate model needs to be built, the voxel size must be small which may lead to huge memory requirements.

The rest of the paper is organized as follows. Related work is detailed in Section 2. The proposed method is explained in Section 3. Section 4 presents the experimental results which demonstrate the performance of the method. Section 5 concludes and proposes future work.

2. RELATED WORK

A Review of the Vector Field Implicit Representation for Real-time Modelling [TUBIC2002]

As shown in Fig.2, the vector field is composed of a regular grid made of cubic voxels with side length *L*.

Each voxel in the grid is addressed by the coordinates of its center v_{ijk} in a reference frame W_r. Let us assume that 3D points with coordinates p_m on the surface *S* are collected by the sensor in frame W_r. The covariance matrix $C_{i,j,k}$ of the points falling in voxel *ijk* is defined as in Eq. (1):

$$C_{i,j,k} = \frac{1}{N}\sum_{u=1}^{N}\left(p_u - \bar{p}\right)\left(p_u - \bar{p}\right)^t \qquad (1)$$

where $\bar{p}$ is the mean vector as defined in Eq. (2).

$$\bar{p} = \frac{1}{N}\sum_{u=1}^{N} p_u \qquad (2)$$

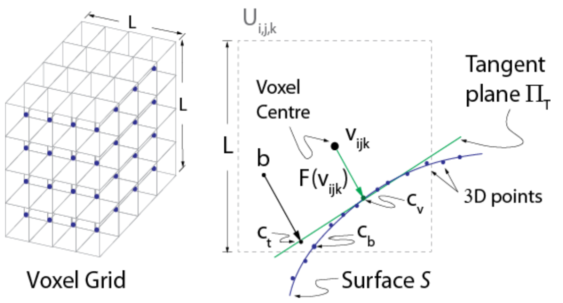


Figure 2. The Vector Field implicit representation (Adapted from [TUBI2002])

If the voxel size L is small enough, it is assumed that the object surface in the voxel can be approximated by the plane Π_T tangent to the surface. The normal vector to tangent plane Π_T is the eigenvector corresponding to the smallest eigenvalue λ_{min} of $C_{i,j,k}$. $F(V_{i,j,k})$, c_v and $C_{i,j,k}$ are stored in each voxel. In the voxel, point c_v on the tangent plane that is closest to $v_{i,j,k}$ is given by Eq. (3).

$$c_v = F(V_{i,j,k}) + V_{i,j,k} \qquad (3)$$

As new points are collected by the sensor, $C_{i,j,k}$, λ_{min}, $F(V_{i,j,k})$ and c_v can be updated in real-time. Now, let us assume that the closest point c_t to the surface approximated by Π_T has to be found for a point b falling in voxel i,j,k. The coordinates of c_t can be computed from the content of the vector field with Eq. (4) where $\langle,\rangle$ represents the scalar product and $\| \ \|$ is the norm of a vector.

$$c_t = b + F(V_{i,j,k}) + \frac{\langle F(V_{i,j,k}), V_{i,j,k} - b\rangle}{\|F(V_{i,j,k})\|} \qquad (4)$$

As shown in Fig.2, the closest point c_t estimated by Eq. (4) is a very good approximation of the true closest point on the surface c_b. In addition, for a single unit of data b, the computational complexity of finding its closest point on the surface is constant $O(1)$ and is of order $O(n)$ for n units of data (i.e. n points for which the closest points on the surface needs to be computed). This is more efficient than classical nearest neighbor finding approaches which show $O(n^2)$ or $O(n \log(n))$ computational complexity. The vector field representation thus allows the view registration and view integration steps to execute in real-time. As mentioned above, with the vector field representation, the price to pay for computational efficiency is the amount of memory that is needed to store the voxel grid at a resolution for which the planar approximation is valid.

High-Level Surface Segmentation Using Differential Geometry

An explicit representation such as a triangular mesh is a low-level model of a surface that provides the connectivity between points and that is good for visualization in computer graphics and for performing some metrologic measurements. However, it does not convey high-level information on the geometry of the surface in the neighborhood of a point. Differential geometry is a popular approach for describing a surface. In differential geometry, the coefficients of the first and second fundamental forms of a surface patch $\sigma(u,v)$ on a surface S in a differential neighborhood of a point P completely describe its intrinsic and extrinsic properties and, ultimately, its shape [DOCA1976]. As shown in Fig.3, given a parameterization (u,v), a differential surface patch $\sigma(u,v)$ at a point P has a surface normal N_s that is orthogonal to the tangent plane Π_t at P. A tangent vector v in Π_t can be expressed as a linear combination of σ_u and σ_v. Defining the linear maps $du(v) = \lambda$ and $dv(v) = \mu$, we obtain Eq. (5).

$$v = \lambda\sigma_u + \mu\sigma_v \qquad (5)$$

Applying the inner product $\langle \ \rangle$ to vector v and using Eq. (5) yields Eq. (6).

$$\langle v,v\rangle = \lambda^2\langle\sigma_u,\sigma_u\rangle + 2\lambda\mu\langle\sigma_u,\sigma_v\rangle + \mu^2\langle\sigma_v,\sigma_v\rangle \qquad (6)$$

Writing $E = \|\sigma_u\|^2$, $F = \sigma_u \cdot \sigma_v$, *and* $G = \|\sigma_v\|^2$ and using the maps $du(v)$ and $dv(v)$ above, the expression for $\langle v,v\rangle$ writes as Eq. (7).

$$\langle v,v\rangle = Edu^2 + 2F du\, dv + Gdv^2 \qquad (7)$$

Eq. (7) is referred to as the *First Fundamental Form I* of the surface. In an infinitesimal neighborhood of P, *I* describes the measurement of a length on the surface. Although E, F, G, σ_u and σ_v depend on the parameterization of the surface, the first fundamental form I depends only on S and P. It is an intrinsic property of the surface since it is independent of how the surface is embedded in 3D space. This can be better understood by visualizing the distance between two points on a flat sheet of paper. When the sheet is bent (without being folded), the distance between the points remains the same. It is thus an invariant property of *S* and is not affected by rotations and translations.

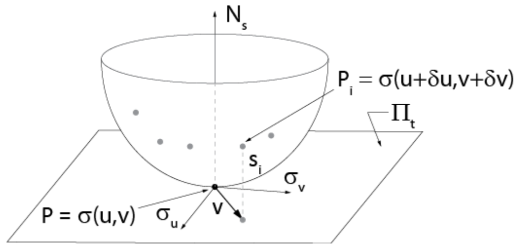


Figure 3. A small patch near a point P on surface σ(u,v) and the tangent plane Π_t

The *Second Fundamental Form II* of S describes the extrinsic properties of the surface around a point and is linked to the *curvature* of this surface, i.e. the way the surface pulls away from the tangent plane *Π_t* at *P*. Considering again Fig.3, it can be seen that the surface at point P_i pulls away from the tangent plane at P by a distance given in Eq. (8).

$$\left\langle \left(\sigma\left(u+\Delta u, v+\Delta v\right)-\sigma\left(u,v\right)\right), N_s \right\rangle \quad (8)$$

Approximating $\left(\sigma\left(u+\Delta u, v+\Delta v\right)-\sigma\left(u,v\right)\right)$ by its Taylor expansion and neglecting the high order terms, one obtains Eq. (9).

$$\sigma_u \Delta u + \sigma_v \Delta v + \frac{1}{2}\left(\sigma_{uu}\left(\Delta u\right)^2 + 2\sigma_{uv}\Delta u \Delta v + \sigma_{vv}\left(\Delta v\right)^2\right) \quad (9)$$

Since σ_u and σ_v are tangent to the surface and are thus perpendicular to N_s, Eq. (9) becomes Eq. (10)

$$\frac{1}{2}\left(L\left(\Delta u\right)^2 + 2M\Delta u \Delta v + N\left(\Delta v\right)^2\right) \quad (10)$$

with $L=\sigma_{uu},\ M=\sigma_{uv},\ N=\sigma_{vv}$. For small Δu and Δv, Eq. (10) can be written as Eq. (11) (if ½ is dropped)

$$\Pi = Ldu^2 + M\,du\,dv + Ndv^2 \quad (11)$$

Eq. (11) is called the *Second Fundamental Form II* of the surface at P.

The Shape Operator, also called the Weingarten Map, S in Eq. (12) can be defined at a point using the coefficients of the first and second fundamental forms [DOCA1976].

$$S = \left(EG-F^2\right)^{-1}\begin{bmatrix} LG-MF & MG-NF \\ ME-LF & NE-MF \end{bmatrix} \quad (12)$$

The eigenvectors of S determine the directions in which the surface bends at each point and the eigenvalues κ_1 and κ_2 are the principal curvatures (i.e. the maximum and minimum normal curvatures at the point). It is possible to compute two very important invariant surface properties of a surface at a point: the Mean curvature *H* and the Gaussian curvature *K. H* and *K* are defined in Eq. (13) and (14) respectively.

$$H = \frac{\kappa_1+\kappa_2}{2} \quad (13)$$

$$K = \kappa_1\ \kappa_2 \quad (14)$$

Although Eq. (13) and (14) are useful, it is more convenient to use the coefficients of *I* and *II* to compute H and K. The Gaussian curvature *K* is given by Eq. (15) while the Mean curvature *H* is given by Eq. (16) [PRES2010].

$$K = \frac{LN-M^2}{EG-F^2} \quad (15)$$

$$H = \frac{LG-2MF+NE}{2(EG-F^2)} \quad (16)$$

Looking at Eq. (15) and (16), *K* and *H* can be obtained from differentials. Using the signs of K and H, it is also possible to characterize the type of surface to which a 3D point on a surface belongs to [BESL1988]. As shown in Table 1, seven types of surface can be described by the combination of the signs of K and H.

The usual approach that was proposed for finding the type of surface at a given point consisted in fitting a quadratic surface model in the *N x N* neighborhood of each point in a smoothed range map and then in computing the partial derivatives needed to extract K and H [BESL1988]. A range map is a 3D image defined as in Eq. (17) for which the surface parameterization is such that there is a depth value z corresponding to a coordinate pair (x,y) in a plane. The connectivity between 3D points in a range map is thus known compared to a point cloud for which the connectivity between points is unknown.

$$z(x,y) = f(x,y) \quad (17)$$

Although this fitting approach can achieve good results, computing the derivatives on the raw depth map (i.e. without fitting) is unpractical because of sensor noise. For large depth maps or, in a more general case for large point clouds, the fitting step is very time consuming. In addition, a different fit is implemented at each point even when points lie in the same neighborhood and may belong to the same surface type. However, as pointed out in [BESL1988], correcting this may require a priori assumptions on the surface type. Making such assumptions is very restrictive and does not allow generalization of the approach. When each point has been labelled with a given surface type, it is possible to group connected points sharing the same label and to fit a high-order polynomial (or spline model) to the region in order to obtain a high-level model.

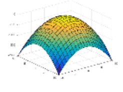
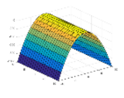
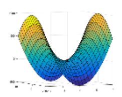
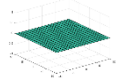
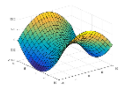
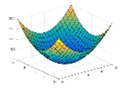
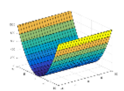
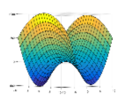
		K		
		+	0	-
H	-	Peak 	Ridge 	Saddle Ridge 
	0	none	Flat 	Minimal surface 
	+	Pit 	Valley 	Saddle Valley 

Table 1. Types of surface as a function of the signs of the Gaussian and Mean Curvatures

Covariance-Based Differential Geometry

A different approach to exploit differential geometry for finding the surface type at each point of a range map consists in using covariance-based differential geometry [BECK1994]. Returning to Fig.3, one can compute a local covariance matrix C_I at point P of a range map as Eq. (18).

$$C_I = \frac{1}{N}\sum_{i=1}^{N}\left(\underline{P}-\underline{P}_m\right)\left(\underline{P}-\underline{P}_m\right)^t \qquad (18)$$

P_m is defined as in Eq. (19) where $\underline{P}_i$ is a point in the neighborhood of P on the range map. It is assumed that N points are selected in the neighborhood of P.

$$\underline{P}_m = \frac{1}{N}\sum_{i=1}^{N}\underline{P}_i \qquad (19)$$

As described in [BECK1994], the eigenvectors of C_I are three orthogonal vectors, two of which, t_1 and t_2, lie on the tangent plane to the surface at P (plane Π_t in Fig.3) and the third one, corresponding to the smallest eigenvalue of C_I, is the normal Ns to the tangent plane (and the surface) as suggested in [LIAN1990]. In [BECK1994], the two-dimensional covariance matrix in Eq. (20) is defined. In Eq. (20), W_i is a two-dimensional vector defined as in Eq. (21) with s_i being defined as in Eq. (22).

$$C_{II} = \frac{1}{N}\sum_{i=1}^{N}\left(W_i - W_m\right)\left(W_i - W_m\right)^t \qquad (20)$$

$$W_i = s_i\begin{bmatrix}\left(P_i - P\right)^t\ t_1 \\ \left(P_i - P\right)^t\ t_2\end{bmatrix} \qquad (21)$$

$$s_i = \left(P_i - P\right)^t\ N_s \qquad (22)$$

As shown in Fig.4, vector W_i is thus the difference between a point P_i in the neighborhood of P projected on the vectors in the tangent plane weighted by the distance s_i between P_i and the tangent plane Π_t at P.

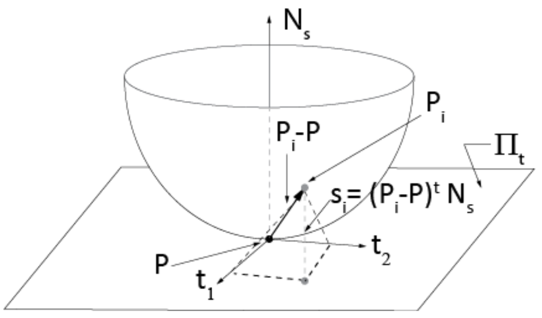


Figure 4. Geometry for the vectors in Equations (16), (17) and (18)

Beckman *et al.* define the quadratic form in Eq. (23) as a "covariance-based Weingarten map" for a vector v in the tangent plane.

$$II_C = \underline{v}^t\ C_{II}\ \underline{v} \qquad (23)$$

Beckman *et al.* claim that the eigenvectors of C_{II} are the principal directions on the surface, i.e. the directions of minimum and maximum normal curvature. They also define a covariance-based approach analogous to the Gauss map at a point P of the range map as in Eq. (24) with vector v_i defined as in Eq. (25).

$$C_P = \frac{1}{N}\sum_{i=1}^{N}\left(v_i - v_m\right)\left(v_i - v_m\right)^t \qquad (24)$$

$$v_i = \begin{bmatrix} n_i^t\ t_1 \\ n_i^t\ t_2\end{bmatrix} \qquad (25)$$

The 2 x 2 matrix C_P in Eq. (24) is the covariance matrix of the projections of the normal vector n_i at points P_i in the neighborhood of P, v_m being the average vector of the projections. The eigenvectors of C_P are the principal directions. The eigenvalues of C_P provide information on the way the surface normal in the neighborhood of P projects onto the tangent plane. For instance, if the surface in the neighborhood of P is a plane, the normal vectors all map into a single point,

point P itself. When one eigenvalue is large and the other is small, the projection of the surface normal vectors map on a straight line and the underlying surface is a developable parabolic surface. Finally, when both eigenvalues are large, the surface is locally curved near P. Beckman et al. have applied the above covariance-based approach to segment points in a range map. In comparison with pure differential geometry approaches such as the one presented in [BESL1988], Beckman's covariance-based approach can only identify three different types of surface: planar, parabolic and curved. A planar surface is identical to the "flat" surface type (with K = H = 0) in Table 1. A parabolic surface covers the cases of ridge (K=0, H<0) and valley (K=0, H>0) in Table 1 while a curved surface covers the other in Table 1. Consequently, the segmentation obtained by covariance-based differential geometry is less rich than the one obtained with classical differential geometry for the reason that, as demonstrated in [DIGN2014], even though the eigenvectors of C_P correspond to the principal directions, the eigenvalues of C_P are not equal to the principal curvatures κ1 and κ2 but are rather functions of their squared value as expressed in Eq. (26) and Eq. (27) (where r is the radius of the ball centered at P).

$$\lambda_{1-C_P} = \frac{\kappa_1^2 r^4 \pi}{4} + o\left(r^4\right) \tag{26}$$

$$\lambda_{2-C_P} = \frac{\kappa_2^2 r^4 \pi}{4} + o\left(r^4\right) \tag{27}$$

Because of this, it is not possible to find the sign of H and, consequently, to differentiate between ridge or valley or peak / pit in Table 1.

The following sections explain how these limitations can be circumvented and how the 3D data can be segmented into the seven surface types in table 1(here, minimal surface is considered as a saddle).

3. PROPOSED APPROACH

As mentioned previously, the sign ambiguity of the eigenvalues obtained by C_{II} prevents us from distinguishing some surface types. We propose a new technique that combines covariance-based differential geometry and the vector field implicit surface representation to segment the 3D data. Instead of working with each point and the neighborhood around that point, we rather work with the voxels in the volumetric grid containing the vector field and its 26 possible neighbors in the grid.

As mentioned above, handheld scanners for metrologic applications use retroreflective markers or natural features to estimate the pose of the sensor with respect to a reference frame chosen as the "world" reference frame. Secondly, using the vector field implicit surface representation, view registration, view integration and the estimation of the normal to the surface in each voxel of the field can be performed in real time as the 3D data is collected by the handheld sensor. The vector field is also built in the world reference frame. Since the pose of the sensor in the world reference frame is estimated in real-time, it is also possible to know on which side of the surface the sensor is when 3D data is collected and integrated in a voxel of the vector field. Knowing on which side of the surface the sensor is located when the data is collected and the surface normal in the voxel allows the orientation of this normal to be defined with respect to the direction of the optical axis of the sensor. This also allows the differentiation between peak/pit or ridge/valley and eliminates the limitations of covariance-based differential geometry (expressed by Eq. (26) and Eq. (27)) for surface segmentation. Based on the above, the strategy that is proposed for surface segmentation is to apply covariance-based geometry on the vector field implicit surface representation instead of on individual points, thus reducing the computational load considerably since hundreds of points if not thousands fall in a single voxel.

In the context of the differential geometry, when the orientation of the normal vector is given, then it is possible to observe that if

- The distance between the tangent plane and all of the points of a certain region around the particular point are positive, then the point is a peak surface.
- The distance between the tangent plane and all of the points of a certain region around the particular point are negative, then the point is a pit surface.
- The distance between the tangent plane and all of the points of a certain region around the particular point is both positive and negative, then the point is a saddle surface.

- The distance between the tangent plane and all of the points of a certain region around the particular point is both positive and zero, then the point is a ridge surface.
- The distance between the tangent plane and all of the points of a certain region around the particular point is both negative and zero, then the point is a valley surface
- The distance between the tangent plane and all of the points of a certain region around the particular point is all zero, then the point is a plane.

We extend this concept into the vector field framework. Therefore, instead of using points we rather use a voxel and the neighbours around the voxel. With the additional knowledge of the direction of the surface normal (available in each voxel of the field), it is possible to exploit Eq. (22) on the neighborhood of a voxel to identify to which type of surface the points in this voxel belong to.

Implementation

Since the covariance matrix of the points falling inside a voxel is computed in real time, computation of the normal vector is also achieved in real time and is the eigenvector of the covariance matrix corresponding to the smallest eigenvalue. By implementing the vector field framework, the normal vector, the closest point to the surface (Eq. (3)), the sensor position and the voxel center are all stored in a voxel of the 3D grid. Therefore, all of the values needed to compute the orthogonal distance from the tangent plane of a particular voxel to the neighboring voxels are provided. For each voxel in the 3D volumetric grid, there are 26 possible neighbors. We have to consider the neighboring voxels which contain much more than 3 points inside in order to have a reliable covariance matrix. This is not a problem since modern scanners can capture 250,000 points per second. We define the orthogonal distance form as

$$Distance_j = (c_v(j) - c_v(0))^T.n \qquad (28)$$

where j = 1 to 26 is the number of the neighbouring voxels around the voxel v_0 . $c_v(j)$ is the point on the plane approximating the surface in the jth neighboring voxel (point c_v in Figure 2), $c_v(0)$ is the point on the plane approximating the surface in the voxel of interest, n is the normal vector obtained from the covariance matrix in the voxel of interest of the vector field framework. $Distance_j$, is a $j \times 1$ vector stored in each voxel of the vector field framework. An illustration in 2D to simplify visualization is given in Fig. 5.

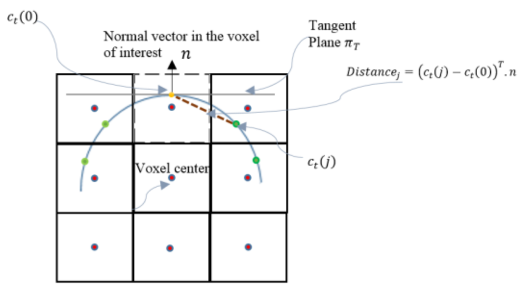

$c_t(0)$
Normal vector in the voxel of interest n
Tangent Plane π_T
$Distance_j = (c_t(j) - c_t(0))^T.n$
$c_t(j)$
Voxel center

Figure 5. An illustration of $Distance_j$ stored in a voxel of interest.

The information of this matrix allows us to recognize the surface type in a given voxel as follow:

- If all of the values of the $Distance_j$ are positive, then the surface in the particular voxel is a peak surface.
- If all the values of the $Distance_j$ are negative, then the surface in the particular voxel is a pit surface.
- If all of the values of the $Distance_j$ are both positive and negative, then the surface in the particular voxel is a saddle surface.
- If all of the values of the $Distance_j$ are both positive and negative, and the eigenvalues obtained by Eq.(26), Eq. (27) are equal then the surface in the particular voxel is a minimal surface. Which, in this paper are considered as belonging to the same category as the saddle surfaces.
- If all of the values of the $Distance_j$ is both positive and zero, then the surface in the particular voxel is a ridge surface.
- If all of the values of the $Distance_j$ is both negative and zero, then the surface in the particular voxel is a valley surface.
- If all of the values of the $Distance_j$ are zero, then the surface in the particular voxel is planar.

  No filtering or fitting operation are performed on the data.

4. EXPERIMENTAL RESULTS

In this section, the experimental results are presented to demonstrate the performance of our approach. The method is applied to 3D synthetic data: Plane, Sphere (peak, pit), Cylinder (Valley, Ridge) and Saddle surface, as well as 3D data which was obtained from a real scanner (Stanford repository) and a HandyScan 3D Laser Scanner by Creaform. The following experiments validate our method. The color map for different surface types is shown in Table 2.

Surface Type	Colour list
Pit surface	Yellow
Ridge surface	Green
Valley surface	Black
Saddle and Minimal surface	Cyan
Peak surface	Blue
Plane surface	Red

Table2. Color map corresponding to different surface types

Fig.6 to Fig.14 show the result of the voxels segmented into different surface types. The color in the voxels is coherent with the surface types in each voxel. The results on synthetic spherical surfaces (peak in blue and pit in yellow) and the results on synthetic cylindrical surfaces, which are valley (black) and ridge (green) surface are shown in Fig.6 and Fig.7 respectively.

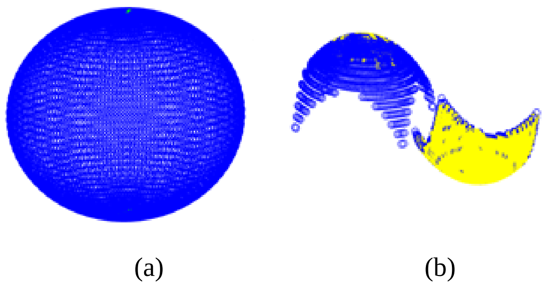


Figure 6. The result of the proposed covariance differential geometry segmentation on a synthetic sphere. (a) Shows peak in blue. (b) Shows peak in blue and pit in yellow color

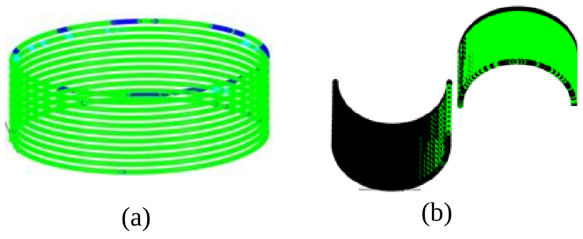


Figure 7. The result of the proposed covariance differential geometry segmentation on a synthetic cylinder. (a) Shows the ridge surface in green (b) Shows the ridge surface in green and the valley surface in black color.

Fig.8 shows the result of our approach on synthetic saddle and minimal surface, in this paper we classify minimal surfaces in the same category as the saddle surfaces because they are a special case of saddle for which the $|k_1| = |k_2|$. So both are shown in the same color.

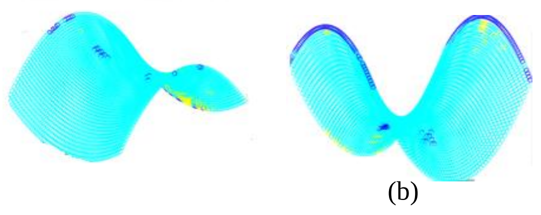


Figure 8. The result of the proposed covariance differential geometry segmentation on a synthetic saddle surface. (a) Shows the minimal surface (in this paper considered as saddle) in cyan (b) Shows the saddle surface in cyan color

Fig.9 is an example of segmentation of a planar surface.

Figure 9. Result of the covariance differential geometry segmentation on a synthetic planar surface

Fig.10 shows the segmented regions in a bunny's head. One can observe on the model data the regions around the muzzle and cheek are pit and the regions around the ears are mostly cylindrical (valley and ridge). So as expected, Figure 9 (b) shows the region around muzzle and cheek in blue and voxels around the ear in green and black which prove the efficiency of our approach.

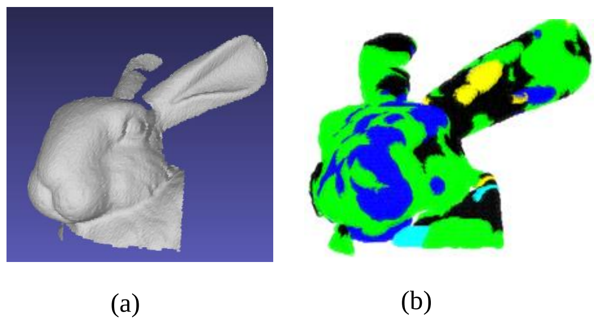


Figure 10. The result of the proposed covariance differential geometry segmentation on a 3D point provided by the Stanford repository. (a) Object mesh data (b) The segmented regions on 3D points of bunny's head

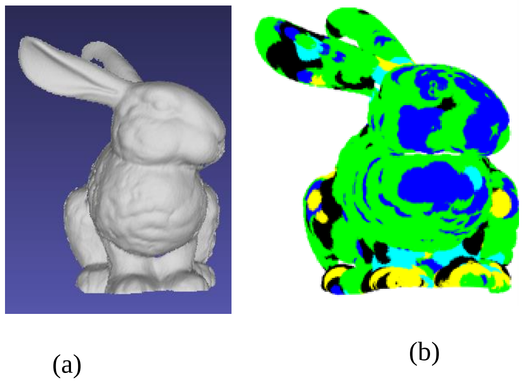


Figure 11. The result of the covariance differential geometry segmentation on a 3D point provided by Stanford repository. (a) Object mesh data. (b) The result of the segmentation on the 3D points

By referring to Fig.11 (a), it can be observed that the upper side of the bunny's belly is peak surface and under the belly is more ridge and the regions around the claws of the bunny are mostly pit and valley. Clearly the regions betweens the belly and claws are saddle, and Fig.11 (b) provide a qualitative validation of our approach on the 3D points of the bunny.

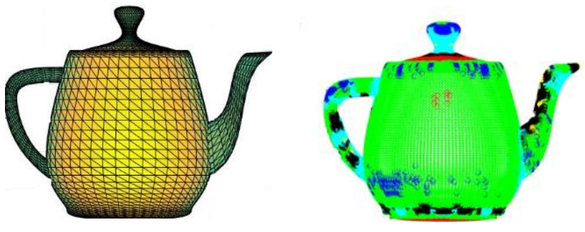

(a) (b)

Figure 12. The result of the covariance differential geometry segmentation on a teapot. (a) Object mesh data. (b) The result of the segmentation on 3D points

It is apparent that for the teapot the main body is basically cylindrical. And it is also clear that the knob in the teapot is peak and the spout is a saddle surface. Fig. 12 and Fig.13 show the main body in green color which is a ridge surface and the knob in blue colour which demonstrate that the region is peak as expected. The spout in cyan color is also coherent with saddle surfaces.

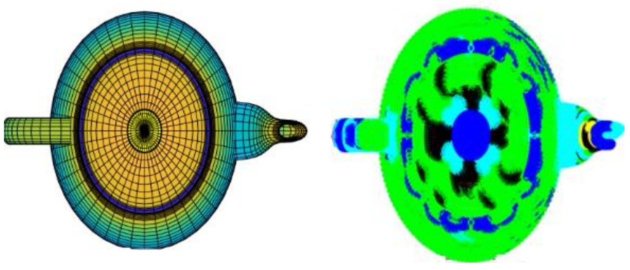

(a) (b)

Figure 13. The result of the covariance differential geometry segmentation on a teapot. (a) Object mesh data. (b) The result of the segmentation on 3D points

The object shown in Fig.14 was scanned by a HandyScan scanner by Creaform. As labeled in Fig.14 (a) the red arrow shows the region that are curved and is a mixture of peak and ridge surfaces. The region shown by purple arrow is ridge. The blue arrow corresponds to a saddle surface. The regions on the object shown by the yellow arrow are peak. We obtain corresponding segmentation in Fig.14 (b) as expected.

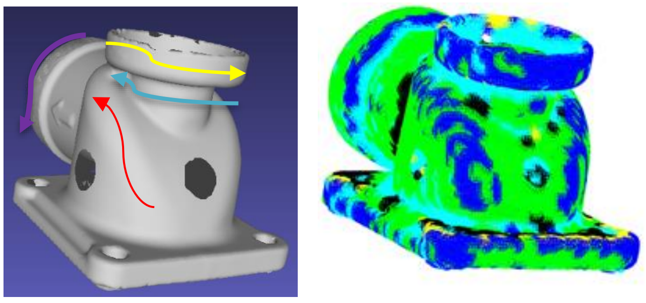

(a) (b)

Figure 14. The result of the covariance differential geometry segmentation on a 3D point collected by HandyScan (Creaform) scanner. (a) Object mesh data. (b) The result of the segmentation on the 3D points

Performance Evaluation

To evaluate the performance of the proposed method, a comparison between quadratic fitting at each point of the point cloud and the proposed method using the vector field has been done. In this section, the details of estimating the performance timing in MATLAB (R2016a) installed on the system with CPU (Intel(R)Core(TM) i7-5820K CPU @ 3.30 GHz 3.30 GHz) and memory (RAM 48.0 GB) for the "teapot" object is presented. The total number of 3D points for the "teapot" object is 41472 points. To evaluate the performance time, we sample 20 points in the neighborhood of a point of the teapot. After sampling, a quadratic equation was fitted on the sampled points. The execution time was "37.3792" seconds for the 20-point neighborhood. For a total number of points of 40000 points, the estimated time to perform the quadratic fitting would be around 74000 seconds. On the other hand, the time required to run the proposed method on the vector field representation using covariance-based differential geometry is 1027.6 seconds, which is much shorter than the performance time for quadratic fitting on the points. This performance is achieved for a vector field grid composed of 35991 voxels with 23661 non-empty voxels. Only the non-empty voxels are processed.

5. CONCLUSION and FUTURE WORK

In this paper, we revisit the concepts of differential geometry used for the segmentation of range maps into different surface types in the more recent context of implicit surface representation and demonstrate that differential geometry can be used efficiently for surface segmentation without the need of surface fitting or the estimation of derivatives.

The paper also extends the concepts used for range maps to unorganized point clouds. There is a significant advantage of combining covariance-based differential geometry approaches and the vector field framework since the segmentation does not require surface fitting. The key point in our approach is that instead of working with points, we are working with voxels and their neighbors, which reduces the computational complexity. The future work of our approach is choosing a higher-level reconstruction method for the surface in the segmented regions, then investigating the continuity between the segmented voxels and their neighbors which do not require high order surface representation.

6. ACKNOWLEDGMENTS

The authors acknowledge the support of the NSERC/Creaform Industrial Research Chair on 3-D Scanning for conducting the work presented in this paper.

7. REFERENCES

[BECK1994] J. Beckman, T. Caelli, "Computation of Surface Geometry and Segmentation Using Covariance Techniques," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 16, no. 11, Nov. 1994, pp. 1114-1116

[BESL1988] P. Besl, R.C Jain, "Segmentation Through Variable-Order Surface Fitting," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 10, no. 2, March 1988, pp. 167-192

[CHEN2012] S.-W. Cheng, T.K. Dey, J.R. Shewchuk, "Delauney Mesh Generation," Chapman & Hall / CRC computer and information science series, 2012, 386 p.

[CURL2996] B. Curless, M. Levoy, "A volumetric method for building complex models from range images," SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, August 1996 Pages 303–312

[DANE2018] M. Daneshmand, A. Helmi, E. Avots, F. Noroozi, F. Alisinanoglu, H. Sait Arslan, J. Gorbova, R. E. Haamer, C. Ozcinar, G. Anbarjafari, "3D Scanning: A Comprehensive Survey," Jan. 2018, arXiv:1801.08863v1

[DIGN2014] J. Digne, J.M. Morel, "Numerical Analysis of Differential Operators on Raw Point Clouds," Numerische Mathematik, (2014), 127:255-289, DOI 10.1007/s00211-013-0584-y

[KHAK 2019] H. Khaksari-Haddad, D. Laurendeau," Alignment of Point Clouds for Comparison of Infrastructures in Engineering on Quality Control", Journal of WSCG,Vol 27, no1, October 2019.

[DOCA1976] M.P. do Carmo, "Differential Geometry of Curves and Surfaces," Prentice-Hall, © 1976, 503 p.

[LIAN1990] P. Liang and J. S. Todhunter, "Representation and recognition of surface shapes in range images," Computer Vision, Graphics and Image Processing, vol. 52, no. 10, pp. 78-109, 1990

[LORE1987] W.E. Lorensen, H.E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," SIGGRAPH '87 Conference Proceedings, 21(4):163–169, 1987

[PRES2010] A. Presley, "Elementary Differential Geometry," Springer, 2010, 468 p.

[RUSI2001] S. Rusinkiewicz, M. Levoy, "Efficient variants of the ICP algorithm," Proceedings Third International Conference on 3-D Digital Imaging and Modeling (3-DIM), 28 May-1 June 2001, DOI: 10.1109/IM.2001.924423, pp. 145-152

[TUBI2002] D. Tubić, P. Hébert, D. Laurendeau, « A volumetric approach for interactive 3d modeling," In Proceedings of First International Symposium on 3D data Processing Visualization and Transmission (3DPVT), volume 1, June 2002, pp. 150–158.