Point Set Segmentation

1 Introduction

Point sets are 3D positions, which can be considered as sampling of a continuous surface. Point sets are emerged as a new standard for the representation of largely detailed models. One of the reasons for this is that point sets are simple and flexible and the operations on point sets are simplified due to the absence of any connectivity or topological information. Another reason is acquiring the point set has become more easy because the range scanning devices are becoming fast and economical.

Segmentation of point sets is a process of classifying the point sets in to subsets according to some common features or characteristics. Segmentation is a necessary and intermediate process in tasks such as object recognition, classification, matching, autonomous navigation and reverse engineering. In general, segmentation is classified into two types: part type segmentation and surface-type (patch-type) segmentation. The goal of part type segmentation is to segment the given input into meaningful, mostly volumetric, parts. The objective of surface-type segmentation is to partition the given input into patches under some criteria.

Our objective is to find out a method for point set segmentation, analyze the time and space complexity of the method, implement the method, compare the method with other existing methods and to see where all the method can be applied. The absence of any connectivity or topological information of the input point set makes the point set segmentation challenging than mesh segmentation and image segmentation. It is more interesting to see how a given point set is segmented without converting it into any intermediate model. As a first step towards our objective, we have done a literature survey on the existing methods for point set segmentation and related techniques. We picked up the common characteristics of the existing methods.

The report is organized as follows: Section2 discusses about the methodologies used in the reference papers, section3 is about the observations we made about the existing methods, section4 talks about the method proposed, section5 presents the implementations we have tried out and we conclude with the last section with pointers to the future directions.

2 Existing methods for point set segmentation and related techniques

In this section we are trying to categorize the existing methods into three groups according to their purpose: 1) methods for segmentation 2) methods for feature extraction and 3) methods for normal and curvature estimation.

2.1 Methods for segmentation

Given a point cloud, the methods for segmentation are proposed in around thirty five papers we have read. The main methods used for segmentation can be classified in to Graph based, Visibility based, Surface based and Hybrid methods.

2.1.1 Graph based methods for segmentation

The multi phase segmentation process [YNBH10] consists mainly of two phases viz., super node extraction and hierarchical segmentation. Super node extraction is of constructing and analyzing a discrete function defined over the input point set. The discrete function is constructed based on the concept of centrality (average geodesic distance from the point to all points over the surface model). Centrality can be approximated as the shortest path between the points in the k-nearest graph G. An approximate centrality value is calculated as the average shortest path distance from the point p to uniformly distributed sample points U over the graph G. Tip of an elongated feature is represented by a local maximum of the function, even though not all local maximum represent the features in the underlying surface. The feature boundaries are identified by saddle points, where multiple super nodes merge at the largest centrality value. To remove the insignificant supernodes, a local refinement process is performed, by which not only the quality of the segmentation is improved but also the segmentation time is reduced. As part of the local refinement process, small supernodes are merged up and skinny supernodes are merged down. If a supernode contains disconnected regions after the local refinement, then it is either merged with a supernode or becomes a supernode. In the second phase of segmentation of the multi phase segmentation process, a weighted graph is constructed based on the supernodes identified. The supernodes serve as vertices of the graph. With this weighted graph, some features could not be segmented out and because of that a new weighted graph that captures the connectivity within each supernode is constructed. The vertex set of the new graph consists of two types of vertices corresponding to each supernode - a representative vertex and a member vertex. One type of edge is created between a representative and member vertices and another type of edge between the member vertices. The weight of an edge between a representative and member vertices measures the importance of the feature identified by the corresponding super node. The weight of an edge between two member vertices measures the similarity between the corresponding super nodes. Normalized graph cut is performed such that the recursive bisection of each subset results in a hierarchical segmentation of the vertex set of the graph.

A tree is an undirected graph in which any two vertices are connected by exactly one simple path. In other words, any connected graph without cycles is a tree. An octree is a tree data structure in which each internal node has exactly eight children. Octrees are most often used to partition a three dimensional space by recursively subdividing it into eight octants. Jan Fransens et al.[FR06] proposes an approach where in, planar parts are extracted using Principal Component Analysis(PCA)on an octree, in a recursive manner, from the set of samples given. At any stage of recursion, the data contained in the octant under consideration is tested for planarity, comparing the eigen values covariance matrix. If the data is found to be planar, no further division of the octant is performed, else the octant is split and the algorithm recurses on the children. After PCA is performed, the aggregation of planar patches into planar clusters is performed. Merging of planar patches is considered as a clustering problem which is solved by Minimum Spanning Tree (MST) analysis. A MST of a weighted graph G is the minimum-weight connected, acyclic subgraph G' containing all nodes of G. The choice of an appropriate weight function which accounts for the global plane parameters and the average distance between the samples of any two patches, is crucial for a successful clustering. To gather both of the above parameters, the average linkage distance in combination with the Mahalanobis metric is used. Mahalanobis distance can be considered as the inverse probability the sample s belongs to the P population. A fully connected graph G is constructed with the planar patches P as vertices and with the edge weights as the average Manhalanobis linkage weights. The MST of G is determined by Kruskall's algorithm. The planar clusters are got by iteratively cutting the MST G' along the higher valued edges, until all resulting subgraphs consist of co-planar patches. Then, every subgraph defines a planar cluster.The planar segmentation is finalized by extending each cluster with points not contained in any planar cluster, if they conform to the statistics of the cluster.

An interactive unstructured point cloud segmentation is discussed in [SZ09]. To get interactive segmentation, the unstructured point-cloud segmentation with graph cut algorithm is used. The graph cut algorithm is designed to minimize energy function in weighted graph where the energy function defines segmentation. The method used for segmentation is similar to the region growing. The graph construction process is as follows: The weighted graph G contains all 3D points (vertices) V from the input point cloud and two imaginary vertices called terminals. The terminals represent assignment of points from V to two sets representing Object (S) or Background (T) points. Terminals corresponding to these sets are called source s and sink t. Every vertex from V is initially connected with N nearest neighbors from V where each edge E connecting two vertices is weighted with a calculated capacity. The edges between vertices V and terminals are not set in initialization phase. The omission of edges to terminals has two advantages for graph cut algorithm. First advantage is decreasing the edges amount in final graph which results in higher speed in max flow computing. The second advantage for point-cloud segmentation is in segmentation behavior. Segmentation is presented as a minimal graph cut problem. In segmentation the V set is separated into two disjunctive sets labeled S and T while the energy function $E(L)$ describing equilibrium between segmentation and point cloud behavior is minimized. The segmentation is computed using max flow algorithm.

A min-cut based segmentation algorithm to compute a foreground background segmentation of point clouds is presented in[GF09]. The inputs to the segmentation algorithm are a $2d$ location and a suspected background radius. The ground plane is estimated with iterative plane fitting, and remove points close to the ground (within .2m) A 4-nearest graph is created for representing the point cloud. The edges of the graph has weights that decrease with distance. A soft background penalty, which can be set to reflect any background prior, is created using the given background radius. The point wise background penalty is added to the total cost of the cut for every point chosen to be in the foreground. The background penalty is a function of the horizontal distance to the object location (relative to the background radius). The penalty makes the background points near to the background radius. The algorithm proposed can be run in two regimes: Automatic regime and Interactive regime. In automatic regime, a hard constraint on the foreground region is made. The min-cut algorithm for several iterations is run to automatically determine the best background radius for the segmentation. In interactive regime, the user iteratively adds removes points as hard background or foreground constraints and the segmentation is re-calculated as the min-cut under these constraints.

A reordering based approach for the multi resolution decomposition of a point cloud is presented in [SN06]. In this approach points are first paired up optimally such that the sum of distance between the pairs is minimum. The minimum weighted perfect matching is used for the pairing up of points. A perfect matching in a graph G is a subset of edges such that each node in G is met by exactly one edge in the subset. Given a real weight c_e for each edge e of G, the minimum-weight perfect-matching problem is to find a perfect matching M of minimum weight $\Sigma(c_e : e \in M)$. After applying the perfect matching, the points are partitioned into Odd and Even halves. A lower resolution representation of the model is given by the odd half. Reorder the points such that the odd points are in position (1,...,$N/2$) and even points are in ($N/2$+1, ..., N).When a point in the odd set is reordered, its matching point in the even set is also reordered. This process is repeated recursively with successive Odd sets to get a lossless multi resolution decomposition of the point set.The even point can be replaced by a vector from the matching odd point. If the pairing is done well, these vectors have similar valued and compress well. The approximation point set is same as the odd point set. The detail point set is obtained by taking difference between the matched even point set and the odd point set for every even point in the even point set. After k levels of decomposition, the approximation is A_k and the detail is D_k. When approximation A_{i-1} is split into A_i and D_i, the coordinates of A_i are stripped off the least significant bit. Thus, the coordinates at level i are represented using p bits. Hence, k level decomposition of points consists of A_k, D_k, E_k, D_{k-1}, E_{k-1}, D_{k-2}, E_{k-2},..., D_1, E_1.

Ruwen Schnabel et al.[SWWK08] present a flexible framework for the rapid detection of semantically motivated shapes in large point clouds. The algorithm proposed consists of two main steps: Firstly a shape based representation of the data is derived by detecting primitive shapes in the unstructured point data and constructing a topology graph that captures the neighborhood relations between the different shapes. Then, in the second step, this topology graph is searched for characteristic subgraphs corresponding to semantic entities that have been defined by the user. At the first step, the fundamental building blocks are established that lay the ground for all further operations. At first primitive shapes are detected that are then subsumed in a topology graph. The detection algorithm is based on the Random Sampling Consensus (RANSAC) paradigm which generates a large amount of primitive shape hypotheses by drawing random sets of sample points that each uniquely determine the parameters of a primitive. The sampling technique in RANSAC can make use of the observation that points that are close in space are more likely to belong to the same shape than points distributed over the whole point-cloud. Therefore drawing sample points that are contained in a sphere of small diameter increases the probability of estimating correct shape hypotheses, which leads to an overall reduced number of necessary shape candidates.The topology graph describes the neighborhood relations between the primitive shapes detected in the point-cloud data. For each detected shape a vertex is inserted into the graph. Two shapes are considered connected if their supports are neighboring in space. The vertices associated to connected shapes are joined by an edge in the graph. Thus, to find the graph edges, the spatial proximity between the support of all detected shapes has to be determined. A 3D grid is used for this task . In order to avoid discretization dependencies due to the location of the cells, eight shifted and overlapping grids are used. All points are sorted into the grid and for all grid cells that contain points belonging to two different shapes, an edge is added to the graph connecting the graph vertices corresponding to these two shapes. The width of the grid cells defines how far apart shapes are allowed to be in order to still get connected in the topology graph. Once the graph is complete the first step of the method is finished and a shape based representation of the point-cloud data has been derived. For the second step of the algorithm ie. shape matching, a query graph is defined for an entity as a graph that captures its characteristic shape configuration. Basically a query graph is a topology graph with the difference that it does not stem from a point-cloud, but from knowledge about the shape of an entity which is introduced by the user. The matching of a semantic entity to the data then corresponds to a matching of the query graph to a subgraph of the topology graph. The query graphs are small in size, and a simple brute-force implementation of subgraph matching performs well. To make the detection more reliable, the user can add constraints to the query graph.

An algorithm is proposed in [BL08b] for skeletonisation and segmentation of point clouds viz. Collapsing And Merging Procedures IN Octree graphs(CAMPINO) algorithm. In CAMPINO, first of all, an octree is generated from the point cloud, then a graph is extracted from the octree. As a third step, cycles of the graph are removed and then vertices with many incident edges are splitted. During octree generation, subdivision of the tree is controlled by the number of points incident on the side of the octree cells.Octree graph contains two types of vertices: M-vertex and T-vetrex. Some of the neighboring T-vertices are merged based on the constraints on subdivision level. Cycles in the graph are removed, considering the number of neighboring M-vertices and T- vertices.The cycle removal process may result in M- vertices with more than three incident edges, known as knots. The knots are split and the skeletons and consequently segments of point clouds are got.

A four step process to extract fruit shape from unorganized point cloud is proposed in [HjY11]. As the first step, octree partitioning is performed on the given point set. Starting with a root node, the point cloud is partitioned to eight, nonempty, disjointed compact octant partitions. The partitioning process stops when there is no node to be partitioned in the point cloud, or normals of points is consistent or number of points in a cell is less than pre-specified threshold. As the second step- each octree splat is assigned an MLS surface for fitting the fruit local shape. In the third step, by the PCA algorithm, the geometric features like normals, variance and curvature of the splat are estimated. As the last step, extraction of fruit shape from splats by a recursive procedure is performed. The extraction algorithm starts with an octree seed and it is pushed in to a stack. In the recursive process, pop the splat P_k from the stack, get the 8-connected neighbor splatsP_i, which are not yet processed. Compute the geometric similarity of the splat as the dot product of the normals and surface variance difference between P_k and P_i.If the dot product is greater than 0.9 and surface variance difference is less than 0.025, push P_i to stack. The process is iterated until the stack is empty.

[GKF09] proposes a system which takes point cloud of a city as input and creates as output a segmentation and labeling, where every point in the city is associated with a segment and every segment has a semantic label. One of the steps among the four step system is segmentation. After potential object locations are found, the segmentation is performed with the following purposes:1) identify the object shape and 2) identify the segmentations and assign points to objects. Three approaches to segmentation are proposed: Method I: Use all above-ground points within a present horizontal radius, and under a pre-set height. Even though, this method has high recall, it has a low precision. Method II: Start with the closest point to the predicted object location at a pre-defined height and define the foreground object as all connected points, if they lie in some distance threshold. Even though the method works well, it may not be so if there is noise, sparse sampling and proximity to other objects. Method III: The method is based on the proximity of points between back ground and forward points. To measure the proximity, the nearest neighbor graph is used and the degree to which foreground is connected to the back ground is quantified by the cut. The foreground points are extracted from the given object location by a min cut. After the segmentation process by the graph cut method, the features are extracted describing the shape of the objects as well as their context.

An octree based 3 D grid method for segmentation is proposed in [WKWL02]. First step of the process is point normal estimation by surface fitting or Delauny triangualtion. After calculating the normals, it is stored in point data structures. After normal estimation, the points are not merged or ordered. This requires additional operation such as octree method, which is used for segmenting data. In the octree method, each cubical octant in the pyramid structure has eight daughters created halving the point cell along the x, y and z. directions. In approximating a geometric object, the octants completely inside or outside the object are not subdivided further while those octants which contain a portion of the object's boundary continue to be subdivided. A bounding box that can encase the point cloud is created and the initial grids are generated based on the box. The shortest axis among the bounding box's three axis is selected. The size of the cell is determined by dividing the shortest axis with a user defined number. Among the initial cells, unnecessary cells, which do not contain any data are removed. The standard deviation of points in a cell indicates the level of changes in the shape of the part with in each cell. When the standard deviation is larger than the user defined tolerance, the cell is divided into cells using octree decomposition. Segmentation using 3D-grids, separates each region representing a different surface. The points in highly curved areas are extracted first by selecting smaller size cells. Removal of these cells automatically separates the point cloud into several regions by leaving gaps between different regions.

2.1.2 Visibility approach for segmentation

Star-shaped decomposition partitions a model into a set of star-shaped components. A model is star- shaped iff there exists at least one point which can see all the points in the model. A method to partition a point set in to approximately star shaped components is proposed [Lie07], where in a point x is selected at random from the point set P that is not visible from any existing guards. Add x as guard and x's ε-visible region as an ε-A^* to the decomposition. The process is iterated until all points in P are ε- visible by at least a guard. Given a point set P and a guard x, x is used to partition P. To do so, the points of P is converted to a spherical coordinate system centered at x. Then, the spherical coordinate system is partitioned into $2(\Pi/\varepsilon)^2$ equally sized buckets. Each bucket represents an ε-view of x. Finally, each point of P is assigned to a bucket according to its coordinate. In the implementation, an enclosing box is used instead of a sphere, and P is projected to the boundary of the box. For each bucket, the closest back point p of x is found. Then all points in the bucket that are closer than p to x are visible by x. A box tree is used to pre-process the input point set and to improve the performance of the algorithm. A box tree is similar to an octree except that, in the box tree, each node is always a smallest (axis-aligned) bounding box of the enclosed points.The construction of the box tree starts with the minimum bounding box of the entire point set. If there are more than k points in a box, these points are partitioned evenly at their center into eight point sets. Similar to the radial partitioning of the entire point data, a radial partitioning is performed on the box tree constructed. The idea is to traverse the box tree top-down and find a set of boxes that can fit into the buckets. Starting with the bounding box (the root), if all of its vertices belong to the same bucket is checked. If so, the box is assigned to the bucket. Otherwise, the box is opened and the same test for each of the eight child boxes are performed and repeated until no unfit boxes left. A point p is ε-visible from a point x if p is neither a back point nor an occluded point of x. Then, a set of ε-visible points is called the ε-visible region of x. To compute the ε-visible region, radial partitioning and ε- visible point identification are performed. Algorithm to build a box tree from the point set P is of O(kn log n) time complexity, where n and k are the number of points in P and guards respectively.

An operator (named as Hidden Point Removal(HPR) operator) that computes visibility directly from a point cloud, without going for reconstruction is proposed in [KTB07].The operator consists of two steps: Insertion and Convex hull construction. Insertion is done by spherical flipping. A D- dimensional sphere with radius R, centered C is considered. Spherical flipping reflects every point p_i internal to the sphere along the ray from C to p_i. The convex hull of the point cloud and the center of the sphere is calculated. The HPR operator extracts the points that reside on the convex hull and thus determines the visible points.

2.1.3 Surface based methods for segmentation

A typical work flow from point set acquisition to geometrical surface model consists four steps: Calibration, Registration, Minimization of the influence of measurement of noise and Surface modeling. The last two steps are discussed in [DN07]. It is assumed that the resulting segments meet the requirements as:1) resulting segments are partition of original point clouds, 2) the points of the resulting segments should be connected to each other, 3) the points of one resulting segment must lie in the same instance of the planar patch and 4) the points belonging to two adjacent segments lie on two separate planes.The connectivity requirement is checked by using the Euclidean Metric ie. the resulting segment is a connected component if a distance of any point to its nearest neighbor in the segment is below a given threshold. It is also assumed that the normal vectors of the local regression planes of the same segment are almost identical. The method to find out the local regression planes is based on the Fast Minimum Covariance Determinant (FMCD) approach. An appropriate distance measure is needed for hierarchical clustering by the local regression planes. Mean squared distance between the planes is taken.

A method for automatic detection and classification of artifacts located at the ground level is proposed by Hernndez et al.[JH09]. The detection is two fold: i) Filtering of the structures such as cars, pedestrians, lampposts to facilitate the modeling process from buildings/facades and ii) Re-introduction of some elements (lampposts, sign boards, bus stop, etc), improving visual realism in modeled scene. There are two assumptions as follows on the 3D data: i)Facade building information is principally located on a vertical plane (z coordinate) and in front of the acquisition system and ii)Ground data is perpendicular to facade data. 3D data is exploited using range and accumulation images and the images are generated by projecting 3D points onto a plane using a virtual camera. The range image is a representation of 3D information where the pixel intensity is a function of the measured distance between 3D points and the camera plane. If several points are projected on the same image coordinates, Z-buffer algorithm is used to determine which distance is stored. The accumulation image counts the number of points which are projected on the same pixel(u, v) of the camera image. City block separation is done to handle dense point clouds. Hough transform is used to detect the facade direction. To detect the building height's profile variation, the profile is initially filtered and then a watershed segmentation is performed. The divisions are traced in perpendicular direction to the facade direction to split them into city blocks. Once the 3D data is split, the detection is applied to each urban block separately. First the ground detection and then artifact detection is done. The method used for ground detection is based on a raw segmentation of range image, by using λ- flat zones labeling algorithm. As the height variations on the ground level are small, a range image segmentation with a height $\lambda = 2$m will cluster ground pixels into the same region. The method of artifact detection is based on hole filling algorithm.

A method to search for a CAD model based on a point cloud generated by a partial scan is presented in [IG07]. The method consists of mainly three steps: i)Segmentation of point cloud and CAD meshes into surface patches using an identical algorithm ii) Identification of the matching patches in point cloud and CAD meshes and iii) Aligning the point cloud with the CAD meshes and evaluating the error associated with the alignment. The first step of the approach (Segmentation) is based on the comparison of curvatures. The point cloud and CAD meshes is segmented to local patches and use as matching units. Any extra patches from the CAD mesh is ignored. Insignificant patches, which may be noise, from the point cloud is also discarded. The surface patches of point cloud and mesh are computed according to their surface curvature values. Normal vectors distribution of local neighborhoods is used to compute the total curvature. Normal vectors of the point cloud are determined by normals of the best fitted planes of small neighborhood of points. By the norm of the covariance matrices of its normal vectors, the total curvature of a small neighborhood is estimated. Neighboring points and triangles that share similar curvature are grouped into patches. In the second step of the method (Matching) sorting of the patches in point cloud and CAD mesh are done based on the surface area. By performing binary search for a point cloud on list of CAD meshes, the matching CAD mesh is found out. When the point cloud and CAD mesh do not share a patch, the CAD mesh is eliminated. As the third step of the method(Alignment), principal components of matching pairs are computed by analyzing the eigen values of the covariance matrices. Covariance matrices are found out by aggregating the distances between the points to its center of mass. When the point cloud matches with the CAD mesh, the distance is small and the matching procedure returns true and terminates.

An approach for plane detection by integrating RANdom SAmple Consensus (RANSAC) and Minimum description length (MDL) principle is proposed by Michael Ying Yang et al. in [Yan10]. MDL principle is used to deal with several competing hypothesis. The principle of RANSAC algorithm is to search the best plane among a 3D point cloud. Based on the observation that RANSAC may find wrong planes if the data has a complex geometry, the following 3-step scheme for plane extraction is proposed: 1)The point cloud is partitioned into small rectangular blocks to make sure that there will be a maximum of three planes in one block. 2) RANSAC is applied to extract planes in each block. 3) The MDL principle is employed to decide how many planes are in each block. Eventually, there are zero to three planes in each block.

Mingrui DAI et al.[DZZJ09] propose a technique of segmenting 3D points of a real tree to distinguish its crown and all branches. Given a single-scanned point cloud of a tree, the principal direction and principal curvature of each point are computed firstly. Then an energy function that presents the possibility that a point is on branches is defined. The energy of each point is estimated and the segmentation of leaves and branches is determined by the energy. Then, points from different branches are divided into different groups based on principal directions. Branches of a tree are approximately cylinders; therefore points from branches have the feature of cylinders in shape. The following two aspects should be specially considered: (1)local geometric features at point p is nearly parallel to the direction of cylindrical axis. (2) Points on branches distribute nearly uniformly in its axial direction. These properties are applied to distinguish points sampled from branches and those from the leaves. The neighbor points of each point considered help to determine the local shape of a surface. This neighborhood is selected by K-nearest neighbor (KNN) algorithm. To compute the principal curvature, local normal for each point must be estimated first, by plane fitting. Then a local coordinate system with point as the origin and normal of the point as the z axis is established. To obtain the principal curvature, a quadric surface is fitted to approximate the local point cloud surface. The points in branches have a regular distribution of principal directions, while those from leaves have scattered distribution, which means that the neighbor points from branches usually have similar principal directions and neighbor points from leaves do not have this feature. After obtaining each points $C(p)$ value (Neighbor Point Distribution along Cylinder Axis of a point p), a threshold is selected, points with a C(p) value that is bigger than the threshold is considered as seed points, all the seed points and their neighbor points in their cylinder are labeled as points from branches.

[LWZL03] takes a point cloud data as input and does segmentation in threes stages:- data preprocessing, data modeling and extraction of the Rapid Prototyping(RP) layer contours. Data subdivision and data reduction are done in data preprocessing. In the data subdivision stage, a reference plane is specified by the user and the 3D point cloud is divided in to initial regions based on the reference plane and its normal direction. For each region of the data set a new subdivision plane (known as projection plane) passing the central position of the region is generated. Average projection error of the data points in the region to the projection plane (based on the total number of points in the region and the distance from the point in the region to the projection plane) is computed and if it is greater than a given subdivision error, the region is again subdivided into two halves from the central position of the region and the process is continued until all the projection errors is with in the desired subdivision error. The second stage- Data reduction- consists of considering the data points of each subdivided region as a grid structure in the project plane to get black and white points(based on the intersection between the grid and the projection plane). Data reduction is first employed to derive a skeletal curve from the black points in the plane by removing all the black points that are deletable and we get the skeletal form of the digital image. The center points of the black squares are selected and they are known as the Feature Points(FPs) of the region. MDE(Maximum Distance Error) algorithm is applied and all the redundant points are removed and a collection of regions of FPs are got. The third stage- the extraction of RP layer contours- is got by performing IPCM (Point-based Curve Model). IPCM starts from one of the end regions and is progressively growing process either across the direction of the regions or across the reverse direction across the regions. The RP layer contours are extracted by intersecting the constructed IPCM with a set of user defined planes.

A patch partition is constructed in [QiK11]. P denotes the 3D point cloud. C is the bounding cube of P. P is uniformly splitted into local points et in a subcube, according to the cell boundaries and this process of uniform partition is done iteratively until three condition based on local normal vector of the point and local centroid of the subcube, average normal vector of the partition, the number of points in the subcube and the threshold value K. A unique hierarchical octree structure is constructed using the above partition algorithm. Before grouping the similar patches, the similarity of theirs should be checked based upon the alignment of the patches and patch descriptors. Alignment is got by finding the supporting planes and align these planes. Patch descriptors are computed and compared by using a statistical descriptor and by getting the height and distance histograms.

A face based method for segmentation is proposed in [BV04]. The method is based on a hierarchy of tests. A test(sometimes known as filter) is a procedure to classify points as stable or unstable. Each test is based on an indicator- geometric or error or similarity indicator. An indicator is an estimated scalar or vector quantity assigned to each point within the point region being segmented. There are mainly four tests and segmentation method is based on hierarchy of tests and each test computes certain indicators and based on these triangle strips are filtered out along sharp or smooth edges. The resulting subregions are checked 1) whether they can be approximated by a single surface 2) they are linear extrusions or surfaces of revolution and 3) whether they are smooth, multiple surfaces, which need to be further segmented. The four test cases are : i) Apply edge/ blend filtering ii) To segment smooth multiple regions apply dimensionality test iii) For multiple ruled regions, apply direction filtering and iv) For multiple, doubly-curved regions, apply axis filtering. For all the above tests, normal vectors and curvatures are estimated. Segmentation is a multi step process- First, regions sharing sharp edges are separated, then regions sharing various types of smooth edges are split into components.

The segmentation method proposed in [YL99] is based on surface curvature properties. These properties are used to obtain boundaries present in the point cloud. The boundary detection technique is for extracting geometric information from a dense point cloud without smoothing. Detection of edge points in both step edge and crease edge exhibits surface curvature properties. At a step edge, the curvature exhibits a zero crossing and two peaks of opposite sign. At crease edge, the curvature exhibits a peak. After edge points are detected, edge-neighborhood chain-coding algorithm is used for forming continuous boundary curves. A boundary curve is assumed as an open curve segment. To build closed boundary loops, a boundary curve is connected to a near boundary curve. After getting the boundary loop, in the partitioning process(by scan line algorithm) the point set is broken down into subsets, where the subsets meet along the boundaries.

A method for segmentation based on fuzzy methods is proposed in [BL08a]. The segmentation is divided into four steps: classification of data points into surface categories, classification of points lying on planar features into different planes, refinement of planar surfaces and validation. First of all, find the neighborhood radius, the number of surface categories and fuzzy clustering parameters. For each data point, compute the relative height difference between the point and its neighbors. Classify data points into surface categories with Fuzzy C Means(FCM) algorithm. For planar category points: Compute unit normal vector of the tangent plane, Set the initial prototypes of unit normal vector of the tangent plane space with FCM algorithm,Find the valid prototypes with Possibilistic C-Means (PCM) mode seeking algorithm,Delete redundant prototypes, Cluster points around prototypes. For each primary cluster in the planar category points: Compute d (distance of the tangent plane to the origin) for all the points in the cluster, Set the initial prototypes of d-space, Find the valid prototypes, Delete redundant prototypes and Cluster points around prototypes. For each secondary cluster, fit a plane. Add unlabeled points to clusters. For non-planar category points, add points to clusters. For each data segment, refit a plane.

A method is proposed in [BF11] to extract roads from a dense point set. The method is divided into steps such as Road map registration, road points extraction, point cloud partition, attractor map construction, road boundary estimation and semantic tagging. We concentrate on point cloud partition for this write up. The point cloud partition is based on topology and geometry of the input map. Specifically, for spline curve S representing a road patch connecting two intersections of the given road network, a subcloud for S is proposed with the following steps: 1. For each sample s in S (spaced at 1 meter intervals), a local support plane $P(s)$ by fitting a plane to LIDAR points within a 4 meter radius of s is estimated. 2. For each spline sample s, a working set of points $WS(s)$ from the LIDAR point cloud is extracted that lie within a distance of 0:5 meters from the support plane $P(s)$ and within a 22 meter radius of s. 3. The working sets of all the spline samples are merged into one set of LIDAR points to form a subcloud for S.

Yu Liu et al.[LX08] present a non parametric clustering algorithm, called Cell Mean Shift(CMS) to extract clusters of a set of points on the Gaussian sphere. When a surface is represented by uniformly sampled point set, the distribution of points densities of the Gaussian image of the point set is a reflection of the curvature variation of the surface. Based on this observation, points on the Gaussian sphere are grouped into different clusters in the proposed segmentation approach. Suppose each triangle of a mesh is represented by a six dimensional vector composed of its centroid and normal, then an existing nonparametric clustering algorithm viz., Mean Shift(MS), is employed for clustering such vectors and mesh triangles are grouped based on the differences of normals between two adjacent triangles. Based on the MS procedure, CMS is developed to cluster points on the Gaussian sphere.

Two algorithms for segmentation and classification that form a processing pipeline which goes from raw 3D data, via 3D segmentation to classification is proposed in [DUV^{+}10]. To sieve the ground from objects, the segmentation pipeline combines a mean elevation map representation for the ground and a voxel based map for 3D objects in the scene. To get the mean elevation map, different terrain modeling methods such as elevation map mean, elevation map min-max , elevation map multi levels etc. are evaluated. The segmentation is composed of two processes- the extraction of the ground surfaces and the segmentation of the objects above the ground. Extraction of the ground surfaces, the data are stored in a database called voxel-grid and the data is separated in to ground and non ground categories. Points are considered in batches according to the voxel it belongs to. A voxel is considered to contain ground data if it is a member of the lowest set of adjacent non-empty voxels in a vertical column and if 3D points stored in that set of voxels has a low residual when fitted to a plane. Standard deviations are calculated for each point. Ground cells are clustered by 2D adjacency to form partitions. With the largest partition as the ground surface, the rest of the partitions are incorporated if they are within a height tolerance. All of the voxels in the database that contain data and not were considered to be part of the ground surface is considered to be non ground voxels. After segmentation, the individual segments are classified.

A method for extracting point cloud of useful main surface from 3D point cloud of an existing product or part is presented in Shoubin Liu et al.[Liu]. A traditional snake model is a curve $l_s, s \in [0, 1]$ moving over a 2D grey-level image $I(x, y)$ to minimize its energy E_{snake} which is represented in terms of strain energy of the curve, bend energy of the curve and a term which is in terms of parametric curve or snake. Gradient Vector Flow(GVF) model of snake was proposed by Xu and Prince to solve the problem of limited capture range of the traditional snake. A 3D point cloud is sliced in to 2D point cloud at different slice sections. A grey level image $I(x, y)$ is generated for each section. According to the effective vertical height and horizontal width of 2 D point cloud, a rectangular window is established. The rectangular window acts as the boundary of the image $I(x, y)$. GVF field is only applied with in the window. To determine dimensions of image $I(x, y)$, points of 2D points are projected to vertical axis and horizontal axis respectively. Each point of point cloud in one section is calculated to determine its match pixel. Once match pixel is found, this pixel is then assigned a grey-level value 255. Other pixels without match are assigned a grey-level value 0. Pixels with grey-level value 255 are edge points. Thus, the image $I(x, y)$ obtained from 2D point cloud is an edge map. Regulation of number of snake points is needed. A snake tracing algorithm is used to extract candidate points and to repair the gap points caused by mini-structures. In the first pass of the algorithm, a temporary point record is created. If there is no background point to match the snake point, the snake point itself will be recorded. This means a gap appearing. Two indices of Gap Start and Gap Stop are used to record the indices of gap points. During the second pass, the recorded start and stop points of each gap are revised for extending its range. During the third pass, points in the temporary point record are traced and repeated points are deleted. A complete point cloud belonging to main surface in a section is extracted. Gap points caused by mini-structures are repaired.

To recognize an object and grasp, a robot should know the object clearly well. But it is impossible to keep all the objects in database. So it is better to keep the primitive shapes in the database and the object is mapped to a combination of primitive shapes. A method is proposed for the above process in [Gar09]. The method scans the object and start processing the point cloud and divides the point cloud into its main parts, using the box decomposition algorithm. The box decomposition algorithm finds a root box which comprises of the whole point cloud, by Minimum Volume Bounding Box(MVBB) algorithm[BHp01]. MVBB algorithm is done by approximating the diameter of the given point set and then approximating the MVBB by checking the constraints on the volume. In [Gar09], after finding the primitive shapes in the point clouds, each division of the point cloud is fit into most similar primitive shape and got an output model composed of these primitive shapes.

Region Growing for segmentation: Region growing is a method for segmentation which comes under the Surface based method for segmentation and we have around eight papers in which segmentation is done by region growing.

Region growing is used in [NLZW10] selecting the seed point as the point with maximum curvature. Based on the minima rule the variation of normal vectors, it is decided whether the seed point and the neighboring points with in a distance constraint belong to the same region or not. If the point is unlabeled ie. not processed and if its variance of normals is less than a threshold variance, then the point will belong to the same region with the seed point. If the seed points and all its neighborhood are labeled, but not all points are labeled, then select a point with maximum curvature from the remaining points and continue the region growing.

In [JRBBK10], Markov Random Fields (MRFs) are viewed as graphical models that give a frame work for labeling problems in multi class segmentation. Points in the input point cloud are the vertices of the graph and the edges is the pairwise relationship between node i and node j. An energy function is described as the sum of unary potential function and the pairwise potential function. The two types of edges viz., t-links(terminal link) and n-link(neighborhood connections) are represented by the two energies.To find out the initial seeding, one of the methods used in Geometric Plane Seeding, in which the neighborhood of surrounding points are computed using kd-tree. In Geometric Plane Seeding the normal vectors for each point are found using a neighborhood of surrounding points computed with a kd-tree. After finding out the normals in Geometric Plane Seeding, the points with normals aligned, within some threshold, to the gravity vector are clustered and a RANSAC plane fit is performed on each cluster. After seed point generation for modeling color properties of an object hypothesis, Gaussian Mixture Models(GMM) are utilized. For each of K components in GMM for a label, the mean RGB value, the inverse covariance matrix and the component weight from the seed point are learned. The unary cost for a point taking a label is determined by the likelihood of the color of that point belonging to the GMM. The iterative energy minimization is done by iterative GMM learning until convergence is obtained. The largest cluster lowest in the plane is selected as the table. Any clusters within a threshold is also considered as part of this table. The points higher than the estimated table plane, are clustered based upon colour, normal and distance.

The second step in the two step process of segmentation proposed in [RVDHV06] is region growing. This stage uses the point normals and their residuals (got in the first step of the segmentation process), in accordance with the user specified parameters to group points belonging to the smooth surfaces. Region growing is based on two constraints: Local connectivity and Surface smoothness. Local connectivity is used to ensure that points in a segment should be locally connected. Surface smoothness is ensured by selecting the points whose normals won't vary too much from each other. A residual threshold and smoothness threshold are specified. The point with the minimum residual is selected as the current seed. Using KNN or FDN select the neighboring points of the seed and add the points to the current region if they satisfy the smoothness threshold condition between the normals of the current seed and its neighbors. The points which have less than the residual threshold are added in to the list of potential seed points. If that list is not empty, set the current seed to the next available seed and repeat the region growing process.

In the method proposed [DN07], for each seed cluster determined by using the clustering in feature space, a region growing is performed by assigning those points to the plane segment which are with in a normal distance band around the seed cluster's plane and if the distance between the points normals and the seed clusters regression plane is smaller than a predefined threshold. If no more seed-cluster can be determined, the remaining points are assigned to the rejection class. Later, connected component analysis in object space and a merging of similar components considering feature and object space are performed.

Zhan et al.[Zha] proposed a point cloud segmentation algorithm based on colorimetrical similarity and spatial proximity. The algorithm has three steps: region growing, region merging and refinement processes. The region growing process starts from growing the first point of the input point cloud. When the region growing process meets an unlabeled point, it initiates a new region and the point is added to the region and the point is pushed in to a stack. When the point is popped from the stack and finds its k nearest neighbors within a given distance and grows the current region on the basis of colorimetrical similarity. The colorimetrically similar points are added to the current region and added also to the stack and the current growing process terminates when the stack is empty. This process is repeated until all the points are labeled. The output of this process is roughly segmented regions to be used in the second step (region merging and refinement) of the proposed method. Using region-region threshold for colorimetrical similarity, two neighboring regions are checked and if they are colorimetrically similar, they are put in to homogeneous list. For each region, the region merging process checks whether the region has been in some homogeneous list. Using K-nearest neighbors and a distance constraint, the neighbors of each region is searched and they are compared with the region based on colorimetrical similarity. The similar neighbors are added to the homogeneous list of the current region. The regions in the same homogeneous list are merged. Using a threshold for minimal size of an acceptable region the segments are refined.

In the method proposed in Mingrui DAI et al.[DZZJ09], the techniques used to segment points from different branches are in two steps: region growing and region merging. The procedure of region growing is a 4 step process as follows:(1) Select an unclassified point as seed point p; (2) Find the neighbor point set

$B = \{b_0, , b_n\}$ of p. Compute the angle between p's principal direction and the principal direction of (b_i) for each b_i in B. If the angle is less than a given threshold, b_i will be classified to the same group of p, and all these b_i constitute a new group Q. If Q is empty, go to step 1;(3) Set each member of Q as a new seed point, go to step 2; (4) Stop if all the points have been classified. As the coarse segmentation result has many over-segmented region, these regions are merged based on their features and spatial relationship. In the first step of the merge process, the angle of any two neighbor groups are calculated and stored them in an angle table. The pair of groups with the minimum angle is merged together if they satisfy axial restriction. The group directions of new groups are recomputed and angle table is updated. These steps are processed repeatedly until a maximum threshold for the angle is reached.

Vasiliki Stamati and Ioannis Fudos propose a method to partition a given point cloud to components based on the concavity intensity of each point using region growing method in [SF07].Concavity intensity is the smallest distance of a point of the cloud from the convex hull of the cloud. To derive concavity intensity, it is assumed that the point cloud has been pre-processed to remove duplicate points. 3D convex hull of the point cloud is computed using the Quick hull algorithm.P_f denotes the track of the point on the convex hull on some face f of the convex hull. The concavity intensity of p is the smallest $|pp_t|$ such that the line segment $|pp_t|$ does not cross the cloud point. After getting the concavity intensity of all the points, a region growing method applied to divide the point cloud into components. Before the region growing, a preprocessing of point cloud to calculate the variations in concavity intensity values for all the points in the point cloud for coordinate directions x, y, and z, by examining the concavity intensity values of the neighboring points.Seed points for region growing are chosen as the points with concavity intensity values as constant or almost constant. Region growing are carried out by adding points to the region if the points hold the following conditions: 1) the angle formed by the normal vector of a point and the mean normal vector of the region should be smaller than a threshold t and 2) the approximate gradients of the concavity intensity function in directions x, y and z for neighboring points of the same region should maintain the same sign value.

An algorithm for segmentation of point cloud based on a differential surface characteristic known as slippage signature is proposed in [Gef]. The slippage signature of a shape consists of the set of rigid motions that, when applied to the shape, slide the transformed copy along the original without forming any gaps. So the surface descriptor is based on how the surface behaves under different kinds of rigid motions. Given a surface S, a rigid motion M is a slippable motion of S if if the velocity vector of each point $x \in S$ is a tangent to S at x. Surfaces which are invariant under rigid motions is known as kinematic surfaces. The input point set is divided into component point sets such that each component can be well approximated by a connected piece of a single kinematic surface. A set of points P is known as slippable if it can be approximated by some kinematic surface. Segmentation algorithm is based on breaking the point set into slippable components. The segmentation is a bottom up approach which has three phases: Initialization, Patch growing and Termination. In the initialization phase, a similarity score between each pair of adjacent patches. In the patch growing, a pair of adjacent patches that are the most similar collapse into a single patch. The new covariance matrix is computed from the covariance matrices of the two patches to obtain the slippage signature for the merged patch. Also update the similarity score between the new patch and its neighbors. When the similarity score of the pair of patches at the top of the queue drops below a threshold, the growing of a patch is stopped.

2.1.4 Hybrid approach for segmentation

Segmentation problem of a 3D point cloud can be broadly classified in to three categories: region based, edge based and hybrid approaches. [YK08] present a hybrid method of segmentation by using fixed size LGPs(Local Geometric Pattern) in a discrete space. LGP is nothing but a set of neighboring points in a 3D point cloud. LGPs appearing on the linear plane are called linear LGPs. In the two-step segmentation method proposed: first reject the non-linear points from a point cloud(edge based segmentation), and then merge non rejected points whose LGPs have common normal vectors(region- based segmentation). Input to the hybrid method is a range image represented by a 2D digital image each of whose pixels (x, y) has a depth information $d(x, y)$ from a 3D scanner to an object surface. This input is transformed into a 3D triple valued image by quantizing the depth. Thus the input is transformed to a grid point set. Checking the LGP linearity, reject non-linear points from the grid point set. A linear LGP is a discrete plane patch $D(P)$ in a bounded space $Q(x)$, denoted by $D_{Q(x)}(P)$. Find a set of Euclidean planes such that the digitization of these planes is equal to $D_{Q(x)}(P)$. Three parameters α, β, and γ indicate the normal vector of the plane P. A set of feasible normal vectors are calculated from each linear LGP. The feasible region of each LGP is given as a set of normal cells that constitute a convex polygon in the space (α, β). The normal cells are embedded in to the 3D space (α, β, γ) , with γ =1. After the normal cells are embedded into the space, make the congruous ones by applying to them 48 transformations of rotations and symmetries of a cube of edge length 2, centered at the origin of the 3D space. Such a cube is called a cubical Gaussian sphere. The normal cells tiled on the cubical Gaussian sphere are projected onto a unit sphere centered at the origin. The unit sphere separated by projected normal cells are called the discrete Gaussian sphere. By using the discrete Gaussian sphere, a discrete version of extended Gaussian images are created, called as unified discrete Gaussian images. Using unified discrete Gaussian image $u(c)$ and the point sets $R(c)$, an algorithm is proposed for planar surface segmentation from a locally linear point set. The largest connected grid point set S_i, whose points have a common normal cell using $u(c)$ and $R(c)$, is looked for. In other words, each S_i is a maximally connected point set, whose points have a common normal cell. After finding S_i, check the size of S_i and if $S_i \geq s$, remove all points of S_i from every $R(c)$ and also modify $u(c)$. After modification and incrementing i, select a new S_i. To reduce the frequency of calculation of connected components, a priority queue, D_k, is made of normal cells c with their $u(c)$ that are not less than s. Repeatedly dequeue a normal cell h from D_k to obtain the maximum connected component C of $R(h)$. Obtain the currently maximum point set S_l by comparing the size of C with the maximum among those of other normal cells that are already dequeued from D_k. The loop is repeated until $u(h)$ is not more than the size of S_l. The maximum connected component of $R(h)$ is got, by a simple method based on a depth-first strategy by using a queue.

2.2 Methods for feature extraction

Given a point cloud, the methods for feature extraction are proposed in around twelve papers we have read. The methods used for feature extraction can be classified in to Graph based and Surface based.

2.2.1 Graph based methods for feature extraction

A method is presented for extracting sharp features on point-sampled surfaces in Christopher Weber et al.[WHH10]. The feature extraction method proposed consists of three steps: First, the data structure for the points set is built. In the second step, local neighborhood in the point set is created. In the third step, these neighborhood are analyzed to identify sharp features ie. to detect the sharp features in the point cloud, at first every point in the cloud for being part of a sharp feature or not is checked by analyzing the neighborhood of each single point. As neighborhood k-nearest is used. As a pre-processing step in the algorithm, kd-tree implementation is performed. The result is stored in a Riemannian graph. In this graph, the minimum spanning tree connecting the point is expanded to connect each point with its k-nearest neighbors.

Feature elements such as crease pattern and border pattern are extracted by a method in [GWM01]. The process is mainly divided into two phases: analysis phase and feature extraction phase. In analysis phase, a neighbor graph(Reimannian graph) is constructed from the input points. Since the Reimannian graph does not minimize the number of edges, a Delauny filtering approach is done , in which a Delauny tetrahedralization is computed by Qhull tool. Another filtering is followed which filtered triangles of the Gabriel complex. Two more triangle filters are used by which the holes in the surface are removed. In feature detection phase, by a three step process the crease and border patterns are found out. The first step analyses the neighborhood of each point and penalty functions are computed that describe how far the point is off a crease, corner or border point. The second step transfers the point penalty functions to the edges of the neighbor graph and combines the edge penalty functions with the edge length s to edge weights. Using the weights a modified Minimum Spanning Graph(MSG) is got. In the third step, the short branches of the MSG are pruned away.

A method is proposed in [DVVR07] to find closed sharp features from a given point cloud. First a region growing method is applied with normal estimation to cluster the point cloud to reduce the point cloud size. The normal vector is calculated as locally as possible. The k- nearest neighbors are got by generating a sphere with the point p as midpoint and radius such that the k- nearest neighbors are inside the sphere. The normal in p is estimated as the normal of the least squares plane through the 1-ring neighborhood of the point p. A region growing is performed to cluster the points based on the sharp edges using normal estimation. A connected graph G_{all} is constructed , where each edge represents a cluster and each edge connects two clusters that contain at least one point with overlapping 1-ring neighborhood.G_{all} has many unwanted gaps between small clusters. So the graph is extended by adding edges which connect between two small clusters that share two large neighboring clusters. The extended graph involving two small clusters give an idea on the location of the sharp feature lines. To remove many cycles from the extended graph, Minimum spanning Tree (MST) is constructed. Still if the MST has many short branches, and they are removed because they don't correspond to the actual features.

In Ruwen Schnabel et al.[SWWK08] algorithm for shape detection is a two step algorithm. In the second step, the topology graph describes the neighborhood relations between the primitive shapes detected in the point-cloud data. For each detected shape a vertex is inserted into the graph. Two shapes are considered connected if their supports are neighboring in space. The vertices associated to connected shapes are joined by an edge in the graph. Thus, to find the graph edges, the spatial proximity between the support of all detected shapes has to be determined. A 3D grid is used for this task . In order to avoid discretization dependencies due to the location of the cells, eight shifted and overlapping grids are used. All points are sorted into the grid and for all grid cells that contain points belonging to two different shapes, an edge is added to the graph connecting the graph vertices corresponding to these two shapes. The width of the grid cells defines how far apart shapes are allowed to be in order to still get connected in the topology graph.

2.2.2 Surface based methods for feature extraction

Ivo Milev et al.[MG06] present two methods for feature lines extraction of point clouds: i) automatic extraction of sharp edges and ii) Interactive extraction of sharp edges which is a semi-automatic method. The common algorithm used for both the above methods find out the neighbors of each point in the point cloud with in a specified radius and best fit plane in each point depending on the neighbors of the point is computed. The normal vector of the plane is interpreted as the normal approximation as the normal surface vector of the measured part. Fitting planes to point data is solved using least squares method. After the normal vector computation, it is checked whether the normal vectors of neighbored points are homogeneous and if so the point curvature values are determined. Two strip end lines for each region, a theoretical edge line - which specifies the position of the intersection curve of the extended surfaces and an edge line on the part (all three lines in NURBS representation) are computed. After the above computations, two or more adjacent surfaces can be constructed using the theoretical edge line as a common boundary. The segment strips can be done using the two strip end lines as boundaries with existing strip generation and calculation tool. In the second method viz. the interactive extraction of strip end lines, the difference is that the user will be requested to mark a region on the point cloud, where the algorithm will extract the feature lines.

Westkamper et al.[Wes] describes methods for the automated extraction of feature lines from measured or scanned point clouds. The methods attempt to find locations where surface normal vectors suddenly change direction. The change in direction of normal vectors represents the curvature of the part surface, so the feature lines extraction will be based on these point curvature values. To get the point curvature values, the points immediately adjacent to each point only are considered. A data structure known as a special version of 3D hashing table is used to compute only few distances for each point and thus the method of computation is known as 3D grid method. As mentioned in Ivo Milev et al.[MG06], the computation of point curvature values is done. A pre-segmentation is performed on the got point curvature values applying a simple threshold operation. Edge tracking is done on the pre-segmented values, by which the points with maximum curvature values are got. The neighbors of the last tracked edge point within a search radius is found out using the 3D grid method. The forbidden area ie., the area which includes all points with in a tube consisting of cylinders and sphere- are excluded. For each edge line detected, one cylinder is computed with this edge line as the centerline of the cylinder. For each edge point except the first and the last edge point, one sphere is computed with the edge point as the center of the sphere.

A method is presented for extracting sharp features on point-sampled surfaces in Christopher Weber et al.[WHH10]. In that method, feature detection is performed in two steps. First step discards all points belonging to a planar region with a simple flatness test. The flatness test consists in computing the standard deviation of the angles, i.e. the distance that means the angle to the median value of the directions is checked. If it is lower than a given threshold, the surface in this neighborhood is assumed to be flat or nearly flat without having a sharp feature. But the inverse is not true. A high deviation at a point does not imply a sharp feature. It can also appear in a high curvature region without a sharp feature. Having the local neighborhood of the k-nearest points constructed for a point, it is analyzed to decide if the sample point belongs to a sharp feature or not. For that a Gauss map clustering is applied. Feature detection is performed in two steps. The first step discards all points belonging to a planar region with a simple flatness test. The remaining feature candidate points undergo then in a second step a more precise selection process, called Gauss map clustering. A hierarchical agglomerative (bottom-up) clustering method is used. It begins with each point as a separate cluster and merges them successively into larger clusters. During analysis, all clusters containing only a few points are discarded, since they correspond to the noisy points. The remaining clusters are analyzed as follows: Opposite clusters on the sphere are considered as one cluster. If in the end a single cluster remains, it is decided that the current point does not belong to a feature. If two, three or four clusters remain, it is decided that the point belongs to a sharp feature. If more than four clusters remain, it is decided that the current point does not belong to a feature.

Quentin Merigot et al. [MOG11]presents a method for extracting curvature information, sharp features, and normal directions of a piecewise smooth surface from its point cloud sampling in a unified framework. To every point cloud C (or surface S) its Voronoi covariance measure (VCM) $V_{C,R}$ (respectively, $V_{S,R}$), is associated, from which one can extract information on the feature and curvature of the underlying surface. This measure is defined, following the Voronoi-based normal estimation technique, by considering the covariance matrices of Voronoi cells. Two algorithms for computing the Voronoi covariance measure are presented: a Monte-Carlo method, and a method based on tessellation. Voronoi cell of a point p in a point cloud C is the set of points of the space, which are closer to p than to any other point in C. Given a point cloud C and an offset radius R, the Voronoi Covariance Measure (VCM) of a point $p \in C$ at scale R is a symmetric matrix that captures information about the shape of the Voronoi cell of p. Voronoi cells and as a result VCM can be unstable under noisy sampling. Thus, the convolved Voronoi covariance measure at a point x (not necessarily in C) is defined by summing the covariance matrices defined above among all points $p \in C$ in a neighborhood of x. The convolved VCM of a point near the underlying surface S provides information about the normals and curvature of S. The edge detection method using VCM proposed is as follows, which needs three parameters, the offset radius R, the convolution radius r, and a threshold T :
1) Compute the Voronoi covariance measure $V_{C,R}$. 2) For every point $p \in C$ do the following steps:
2a) Compute the estimation between VCM and the curvature,$V_{C,R}(B(p,r))$ and diagonalize it.
2b)Sort the eigen pairs $(\lambda_i(p), e_i(p))_{i\in\{1,2,3\}}$, by decreasing order of eigenvalues. 2c) Compute the ratio $r(p) = \lambda_2(p)/\Sigma_i\lambda_i(p)$. Then the point p is on a sharp edge if $r(p) \geq T$ and the edge direction is given by $e_3(p)$.

An algorithm for curve skeleton extraction from incomplete point data is proposed in [TZCo09]. In this paper a curve skeleton is thought of as a generalized ROtational Symmetry Axis(ROSA)(which is a 1D structure) of a shape. To compensate missing data, orientation information is effectively utilized for ROSA computation. It is assumed that the shapes are covered by cylindrical regions except at their joints. The curve skeleton extraction has three steps- ROSA construction, thinning, re-centering of the resulting skeletal cloud and the extraction of the complete 1D skeleton. In ROSA construction, a cutting plane is considered through the point cloud with an orientation that minimizes the variance of the angles between normals of the point cloud and a normal of a given point. A relevant neighborhood of point clouds is identified, using the concept of Mahalanobis distance and a graph is obtained by taking the connected component. The optimal cutting plane is obtained by solving the non-linear optimization problem by an iterative approach. After getting the optimal cutting plane, the centre of local rotational symmetry(ROSA point) is computed. The computed ROSA points collectively form the initial point skeletal cloud. Joint regions of the given shape is non-cylindrical and do not have optimal cutting plane, and so the ROSA points are scattered. Spatial coherence by Laplacian smoothing is applied to remedy this problem. After thinning process the spatially coherent skeletal cloud converges to a 1D structure. If the centered-ness of the cloud is not distorted, re-centering of the skeletal cloud is done.

A process for extracting compact shape descriptor for point cloud data is proposed in [ARC$^+$04]. The process depends on the concept of persistent homology and tangents. Homology is a an algebraic invariant that counts the topological attributes of a given shape X that is embedded in R^3, in terms of its Betti numbers. Homology gives finite compact description of connectivity of a shape. Persistent homology is an algebraic invariant that identifies the birth and death of each topological attribute of a shape from its growth from a scratch. Persistent homology describes the connectivity of the evolving shape via a multiset of intervals in each dimension. Geometry of the shape is examined by looking at the tangents of each point of the shape. A compact descriptor, ie. barcode is got by applying persistent homology to the filtered tangent complex of the shape.

A method is proposed to extract lines of curvature from a given noisy point cloud using normals and curvature information in [KNSS09].A statistical approach is proposed to compute surface curvature and this method denoise the point cloud also. The statistical estimation of curvature has two steps. In the first step, minimum neighborhood of each point p_i is found out by finding the closest points after projecting the points to the local tangent plane of p_i. A guess on curvature tensor values are done and after that, the Iteratively Reweighted Least Squares(IRLS) process begins. This process refines the shape and size of each neighborhood around every point by weighting samples appropriately based on fitting error.As a next step, point cloud correction ism done which includes outlier rejection, point position correction and principal direction smoothing. After the point cloud correction, tracing of lines of curvature is done.

A multilevel approach to edge detection is proposed in [GP05]. The input point cloud is tesselated and filtered to obtain a manifold Solid-to-Layer(STL) model. The STL model is decimated using a chordal deviation filter to obtain two models with different Levels Of Details(LOD). The points of two models are then classified and ordered into a hierarchical structure, indicating the relationships between points by triangular connections. After the above mentioned clustering, the edge detection process is done on the low LOD model. The edge points are got by proceeding with the assumption that each edge point is located near other edge points and surrounding points near one point with a higher curvature could have similar characteristics.

2.3 Methods for normal and curvature estimation

Given a point cloud, the methods for normal and curvature estimation are mentioned in around seven papers we have read. The methods used for normal and curvature are surface based method. Also, in some of the methods even though the objective of the method is segmentation or feature line extraction, the methods for normal and curvature estimation are extensively used.

The approach proposed by Pinghai Yang et al.[YQ07] to compute curvatures of a surface falls into the categories of surface-based method of curvature estimation, but differs from other surface-based methods in that it is based on point set surfaces. An implicit surface form to compute derivatives is used. The moving least-squares (MLS) surface is the underlying representation of the point set and it forms the basis of the differential geometric analysis of point-set surfaces. Two key points in the projection procedure of MLS surfaces are concentrated on: (a) Evaluating the normal direction through a vector field $n(x)$, (b) Searching for the local minimum of an energy function $e(y,n(x))$. When evaluating the normal vector, it is assumed that the normal information at each input point data is available. If the normal information is not readily available as in some applications, a statistical analysis of the neighboring samples is applied to estimate the normal vectors, e.g., an eigen analysis of the covariance matrix of the point positions. A normal vector for any point with the normals of the nearby sample points is computed. A normal vector field is the normalized weighted average of the normals at the sample points. The local minimum is searched in one of the iteration of the overall projection process. The Gaussian and mean curvatures of this implicitly defined MLS surface is got by applying the curvature formulas for implicit surfaces. To calculate the principal curvatures of an MLS surface, the native form of MLS is converted into an implicit form and the principal curvatures are derived from the Gaussian curvature and mean curvature.

The algorithm for segmentation of point cloud, proposed in [NLZW10] finds out the normal vector of a given point. As a second step, curvature for each point by cubic surface fitting is also got. Contour points of a point cloud is extracted by projecting the original points in to its respective plane and by using boundary extraction algorithm. Convex hull of the set of contour points is computed.

One of the steps in the two step segmentation process proposed in [RVDHV06], is normal estimation. The normal of each point is found out by fitting a plane to some neighborhood points. The neighborhood points are got by either K Nearest Neighbors(KNN) method or by Fixed Distance Neighborhood(FDN) method. Plane fitting problem is reduced to eigen value problem. The plane can be parameterized with its normal and its distance from the origin, viz., Hesse normal form of the plane. The residual plane fitting is used to find areas of high curvature.

Ou Yang et al. [OF05] present a method to estimate normal vectors of a point cloud data. A normal vector is a local geometric property of a 3D surface and specific to a given point. In general, the following steps should be performed for normal vector estimation: 1) Identify the applicable neighboring points for estimating the normal vector, 2) Estimate the normal vector based on points in the local neighborhood, and 3) Establish the inner and outer directions of the normal vector.

Mingrui DAI et al.[DZZJ09] propose a technique of segmenting 3D points of a real tree to distinguish its crown and all branches. Given a single-scanned point cloud of a tree, the principal direction and principal curvature of each point are computed firstly. Then an energy function that presents the possibility that a point is on branches is defined. The energy of each point is estimated and the segmentation of leaves and branches is determined by the energy. Then, points from different branches are divided into different groups based on principal directions. Branches of a tree are approximately cylinders; therefore points from branches have the feature of cylinders in shape. The following two aspects should be specially considered: (1)local geometric features at point p is nearly parallel to the direction of cylindrical axis. (2) Points on branches distribute nearly uniformly in its axial direction. These properties are applied to distinguish points sampled from branches and those from the leaves. The neighbor points of each point considered help to determine the local shape of a surface. This neighborhood is selected by K-nearest neighbor (KNN) algorithm. To compute the principal curvature, local normal for each point must be estimated first, by plane fitting. Then a local coordinate system with point as the origin and normal of the point as the z axis is established. To obtain the principal curvature, a quadric surface is fitted to approximate the local point cloud surface.

Given an unstructured point cloud sampled from a smooth curve in R^2 or a smooth surface in R^3, a method is proposed to estimate the normal at each point of the cloud, in [MNG04]. For each point O, find all the points in the point cloud inside a circle of radius r centered at O and then compute the total least square line fitting those points. The same concept is applied for computing normals from a smooth curve in R^2 or a smooth surface in R^3.

A method is proposed to extract lines of curvature from a given noisy point cloud using normals and curvature information in [KNSS09].A statistical approach is proposed to compute surface curvature and this method denoise the point cloud also. The statistical estimation of curvature has two steps. In the first step, minimum neighborhood of each point p_i is found out by finding the closest points after projecting the points to the local tangent plane of p_i.

3 Observations on the existing methodologies

3.1 Sources of input point sets

For the existing methodologies we referred in the last sections, the input point clouds are mainly acquired by scanning the image.

The point set is got by using laser scanner in [RVDHV06, DZZJ09, Zha, HjY11] and [MG06], by using stereo camera in [Lie07], by CT scanner in [Liu] and by using 3D scanner in [IG07] and [YK08]. The input models are also taken from qslim[YNBH10], AIM@SHAPE [KNSS09, FR06], INRIA and MPII [FR06], Rapid form, Princeton segmentation bench mark, Stanford university, Microsoft Research [OF05] and City University of Hong Kong. LIDAR data[GF09], Langweil model of Prague, Riegl data and Velodyne data [DUV$^+$10] are some other sources of point sets. Point clouds are also generated by random sampling[DN07] or near uniform sampling[YQ07]. The input objects are taken from CVAP database in [Gar09]and defined as inventor (.iv) files and later converted into coordinate (.crd)files to make it a point cloud. The input models for [KTB07]is taken from Digital Michaelangelo Project 3D model Repository and the Stanford 3D scanning repository. The two mobile point clouds and the brick cloud used as input in [DVVR07]are taken from Metris N.V.Belgium.The input data was collected by Neptec with one airborne scanner and four car mounted TITAN scanners facing left, right, forward-up and forward-down in [GKF09, BF11].

3.2 Data structures used in existing methods

Data structures used in general, are stack, queue, graph(Reimannian graph, topology graph), tree (minimum spanning tree, octree, kd-tree) and hash table. The structures used are Voronoi diagram/ mesh, voxel point and Gaussian map/image.

3.3 Common concepts of existing methods

The most common sub methods used in majority of the existing methods for segmentation are plane fitting, normal estimation and curvature estimation. Segmentation is viewed as a graph cut problem in many existing methods. Segmentation is also done by region growing in some of the existing methods.

3.3.1 Plane fitting:

In some of the existing methods, plane fitting is done before segmentation either for removing closer points or for normal estimation.

Before segmentation, iterative plane fitting is done in [GF09, DUV$^+$10] and in [GKF09] to estimate the ground plane and the points which are close to the ground plane are removed. In [IG07],and in [RVDHV06] before segmentation, normal vectors are computed by plane fitting.In [BL08a], plane fitting is iteratively done after clustering. A local support plane is estimated by fitting a plane in [BF11]. Plane fitting is employed in [JRBBK10] during the process of geometrical plane seeding. Dorninger et al.[DN07] uses plane fitting in terrain modeling and in [MG06] least squares method is used to perform plane fitting. In all these works, plane fitting is a pre-process to segmentation or feature extraction. In [BL08a] plane fitting is done after the primary clustering step to refine clustering further.

3.3.2 Normal and curvature estimation

Normal and curvature estimations are pre-steps to segmentation in some existing methods. Normal estimation is done in any of the following methods: (i) surface fitting (ii) plane fitting (iii) cubic surface fitting (iv)least square fitting (iv) Delauny triangualtion or (v) PCA. Normal or curvature variance is considered as one of the criteria for classifying points in to different partitions.

Normal estimation by surface fitting or Delauny triangulation is done in [WKWL02]. Normal vectors of points are estimated before segmentation in [NLZW10, OF05, MOG11, MNG04], and curvatures are estimated by cubic surface fitting in [NLZW10]. Normals are computed fitting plane to some neighborhood planes in [RVDHV06, IG07, DZZJ09]and curvature is estimated by normal vector distribution of local neighborhood on the surface in [IG07]. Principal curvature is computed by local normal of points in [DZZJ09]. In [SF07], normal vector is computed from the adjacent facets of a the point. In [BV04], curvature is estimated by least square fitting. In [LX08], unit normal is estimated by least square fitting. In [YL99], surface curvatures are computed from locally least square approximated curvatures. A set of analytical equations for computing surface curvatures is proposed in [YQ07]. Normal is estimated by PCA in [BF11]. The normal vector estimation or curvature estimation is one way or other followed in[YQ07, NLZW10, JRBBK10, DN07, MG06, Wes, JH09, OF05, Yan10, WHH10, Lie07] and [MOG11].

3.3.3 Region growing

Region Growing is an important method by which segmentation is done in many existing methods. A seed point is selected either randomly or with optimum values based on maximum curvature or minimum residual value or best similarity score or with minimum concavity intensity in different existing methods. Growing is done by the distance criteria or by principal direction criteria, or difference in angle formed by mean vector criteria. In almost all the cases, growing stops either all the nodes are processed or when a pre specified condition is met.

In [HjY11], after the octree construction, the octree seed is pushed in to the stack and based on the geometric similarity, the octants are processed and the process is repeated until the stack is empty. In [DN07], the region growing is performed for taking each point as seed cluster and points are added to the cluster based on the distance between the point's plane and seed cluster's plane and stopped the growing process until all seed clusters are done. In [NLZW10], region growing is performed using nearest neighboring clustering, which is based on variation of normals, taking the point with maximum curvature as the seed point and the growing stops when all the seed points and their neighbors are done. In [RVDHV06], region growing is performed taking the point with minimum residual value as the seed, the points in one segment should not have much larger variance the normals and the region growing is done until all the seed points are processed. In [Yan10], region growing is performed to merge the neighboring planes. The pair of points with best similarity score is the seed points. Patch growing is done by selecting the adjacent points with most similarity score and merging into one patch. When the similarity score is below a threshold, then the growing process is stopped.Taking first point of the input cloud and colorometrical similarity as the seed and the constraint respectively, the growing process continues until all points are labeled in [Zha]. In [DZZJ09], region growing and region merging are done, taking the seed point randomly and growing is done based on the principal directions and neighbor relations as criteria. In [SF07], region growing is done based on the angle formed by the normal vector of a point and the mean normal vector and the approximate gradients of the concavity intensity function in directions, the seed points are the points with the concavity intensity values are constant or almost constant. Gelfand and Guibas propose a segmentation method which is a patch growing method ,in [Gef].

3.3.4 Graph cut problem

Graph cut is the common method used for segmentation and some form of distance is the measure used in cutting the graphs. In some of the cases, octree is constructed first and MST cut is being done.

A k- nearest graph is constructed using geodesic distance in [YNBH10] and a normalized graph cut is done to segment the point set. A weighted graph where the weight function is based on Euclidian distance is constructed and the minimal cut is performed in [SZ09]. A 4- nearest graph is constructed in [GF09], (where the weight function of the edges is based on the distance) and a min cut is performed on the graph to segment the point set. Nearest neighbor graph is constructed and a min cut is performed in [GKF09] to segment the given cloud.

PCA on an octree of input points is performed to extract the planar parts and an MST (whose weight function is calculated based on average linkage distance and Mahalanobis distance) is constructed using the planar parts and MST is cut for getting different segments in [FR06]. An octree is constructed and subdivided based on the number of nodes incident on the octree cell and an octree graph is formed and the cycles in the graph are removed which will result in skeletons and segments in [SWWK08]. Point cloud is partitioned in to octree in [HjY11] until there is no node to be processed, PCA is done to extract planar properties of the cloud. Octree is constructed and octree decomposition is done based on standard deviation in [WKWL02]. In [QiK11], a hierarchical octree structure is constructed after the initial partition and before the actual segmentation.

In [SWWK08], from the point cloud, primitive shapes are detected using RANSAC (points inside a diameter is considered to belong to one shape) algorithm and a topology graph is constructed and the edge between the vertices(vertices are the primitive shapes identified) are made if they are connected shapes.

Even though graph cut problem is the common method used for segmentation, here is an exception for that, where graph matching problem is used. Pairing up of points based on the distance using minimum weighted perfect matching in graphs is done in [SN06] and the points are grouped in to two halves and reordering of points are done iteratively to decompose the point set.

3.4 Limitations of the existing methods

Discussing about the limitations of the proposed methods, more computing time is one of the limitations pointed out in [Lie07] and [LX08], less resulting accuracy in [YQ07] and [Wes] . Under segmentation is there in [RVDHV06] and mis-segmentation as well as over segmentation are displayed in [Zha]. Convergence errors are present in [Liu]. [SWWK08] and [DZZJ09] are not able to deal with all the input cases. In [LWZL03], if the user defined subdivision error is too small, the number of subdivided regions may be many and some regions may not have enough points. The extreme input cases- such as highly inaccurate normals and point cloud by under sampling by one scan- fails the method proposed for curve skeleton extraction, in [TZCo09], also when the cut profile is highly concave, ROSA points are not got properly which will affect the curve skeleton extraction. In [KNSS09], computational cost and memory requirement of the proposed method to compute lines of curvature from the given point cloud is high. For face based segmentation based on tests, in [BV04], the setting of various thresholds and diameters of neighborhoods highly depend on user experience.

4 Conclusion and future work

In this report we have concentrated on the existing methods which segment a given point set. As a future work, we are trying to apply variations of some of the concepts from the existing methods.

In [SMK$^+$10], Prominent Cross Section(PCS) was computed from a given mesh model and part based segmentation was shown to be an application. Instead of a mesh model as input, we can try to get PCS from a given point set. For getting PCS, we need to get a primary plane from the given points. For getting a primary plane, normals can be constructed for all the points as specified in [MNG04]. Using the normals, plane and sectional Gauss map, PCS can be computed, which can be used for segmentation.

Another alternative for segmentation is 1) construct a k-nearest graph as per[YNBH10] 2) compute the normals of all points 3) cosine similarity of each pair of normals can be found out 4) assign the cosine similarity as a weight function of the edge in the graph 5) perform min cut [HP] on the graph which may result in segmentation of the point set.

A feasibility study has to be done and more refinements have to be done on the above mentioned methods.

References

[ARC$^+$04] M. Alexa, S. Rusinkiewicz, Anne Collins, Afra Zomorodian, Gunnar Carlsson, and Leonidas Guibas. A barcode shape descriptor for curve point cloud data, 2004.

[BF11] Aleksey Boyko and Thomas Funkhouser. Extracting roads from dense point clouds in large scale urban environment. *ISPRS Journal of Photogrammetry and Remote Sensing*, October 2011.

[BHp01] Gill Barequet and Sariel Har-peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *J. Algorithms*, 38:82–91, 2001.

[BL08a] J Biosca and J Lerma. Unsupervised robust planar segmentation of terrestrial laser scanner point clouds based on fuzzy clustering methods. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63(1):84–98, 2008.

[BL08b] A Bucksch and R Lindenbergh. Campino a skeletonization method for point cloud processing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63(1):115–127, 2008.

[BV04] Pal Benko and Tamas Varady. Segmentation methods for smooth point regions of conventional engineering objects. *Computer-Aided Design*, 36(6):511–523, May 2004.

[DN07] P. Dorninger and C. Nothegger. 3d segmentation of unstructured point clouds for building modelling. In Uwe Stilla, Helmut Mayer, Franz Rottensteiner, Christian Heipke, and Stefan Hinz, editors, *Photogrammetric Image Analysis (PIA07)*, volume 35, pages 191–196. ISPRS, 2007.

[DUV$^+$10] B. Douillard, J. Underwood, V. Vlaskine, A. Quadros, and S. P. N. Singh. A pipeline for the segmentation and classification of 3D point clouds. In *Experimental Robotics: The 12th International Symposium on Experimental Robotics*. Springer, 2010.

[DVVR07] Kris Demarsin, Denis Vanderstraeten, Tim Volodine, and Dirk Roose. Detection of closed sharp edges in point clouds using normal estimation and graph theory. *Computer-Aided Design*, 39(4):276–283, 2007.

[DZZJ09] Mingrui Dai, Xiaopeng Zhang, Yi-Kuan Zhang, and Marc Jaeger. *Segmentation of Point Cloud Scanned from Trees*, pages 1–12. 2009.

[FR06] Jan Fransens and Frank Van Reeth. Hierarchical pca decomposition of point clouds. *3D Data Processing Visualization and Transmission, International Symposium on*, 0:591–598, 2006.

[Gar09] Sergio Garcia. *Fitting Primitive Shapes to Point Clouds for Robotic Grasping*. PhD thesis, School of Electrical Engineering, Royal Institute of Technology, 2009.

[Gef] Guibas Gefland. Shape segmentation using local slippage analysis.

[GF09] Aleksey Golovinskiy and Thomas Funkhouser. Min-cut based segmentation of point clouds. In *IEEE Workshop on Search in 3D and Video (S3DV) at ICCV*, September 2009.

[GKF09] Aleksey Golovinskiy, Vladimir G Kim, and Thomas Funkhouser. Shape-based recognition of 3d point clouds in urban environments. *2009 IEEE 12th International Conference on Computer Vision*, 2(Iccv):2154–2161, 2009.

[GP05] L Galantucci and G Percoco. A multilevel approach to edge detection in tessellated point clouds. *CIRP Annals Manufacturing Technology*, 54(1):127–130, 2005.

[GWM01] Stefan Gumhold, Xinlong Wang, and Rob Macleod. Feature extraction from point clouds. In *In Proceedings of the 10 th International Meshing Roundtable*, pages 293–305, 2001.

[HjY11] Zhi-yi ZHANG Xin WANG Hui-jun YANG, Dong-jian HE. Acm siggraph vrcai'2011, a novel algorithm for segmenting fruit from unorganized point clouds. 2011.

[HP] Sariel Har-Peled. Minimum cut in a graph.

[IG07] Cheuk Yiu Ip and S.K. Gupta. Retrieving matching cad models by using partial 3d point clouds, 2007.

[JH09] Beatriz Marcotegui Jorge Hernndez. Point cloud segmentation towards urban ground modeling. In *5th GRSS/ISPRS Joint workshop on remote sensing and data fusion over urban areas, Shangai, China*, May 2009.

[JRBBK10] Matthew Johnson-Roberson, Jeannette Bohg, Mårten Björkman, and Danica Kragic. Attention-based active 3d point cloud segmentation. In *IROS*, pages 1165–1170, 2010.

[KNSS09] Evangelos Kalogerakis, Derek Nowrouzezahrai, Patricio Simari, and Karan Singh. Extracting lines of curvature from noisy point clouds. *Comput. Aided Des.*, 41:282–292, April 2009.

[KTB07] Sagi Katz, Ayellet Tal, and Ronen Basri. Direct visibility of point sets. *ACM Trans. Graph.*, 26, July 2007.

[Lie07] Jyh-Ming Lien. Approximate star-shaped decomposition of point set data. In *SPBG*, pages 73–80, 2007.

[Liu] *Point Cloud Segmentation Using Gradient Vector Flow Snake.*

[LWZL03] G. H. Liu, Y. S. Wong, Y. F. Zhang, and Han Tong Loh. Error-based segmentation of cloud data for direct rapid prototyping. *Computer-Aided Design*, 35(7):633–645, 2003.

[LX08] Yu Liu and Youlun Xiong. Automatic segmentation of unorganized noisy point clouds based on the gaussian map. *Computer-Aided Design*, 40(5):576–594, 2008.

[MG06] Ivo Milev and Lothar Gruendig. Geometrical approximation and segmentation of laser scanning point clouds geometrical approximation and segmentation of laser scanning point clouds. *Data Processing*, pages 1–8, 2006.

[MNG04] N. J. Mitra, A. Nguyen, and L. Guibas. Estimating surface normals in noisy point cloud data. In *special issue of International Journal of Computational Geometry and Applications*, volume 14, pages 261–276, 2004.

[MOG11] Quentin Mérigot, Maks Ovsjanikov, and Leonidas J. Guibas. Voronoi-based curvature and feature estimation from point clouds. *IEEE Trans. Vis. Comput. Graph.*, 17(6):743–756, 2011.

[NLZW10] Xiaojuan Ning, Er Li, Xiaopeng Zhang, and Yinghui Wang. Shape decomposition and understanding of point cloud objects based on perceptual information. In *Proceedings of the 9th ACM SIGGRAPH Conference on Virtual-Reality Continuum and its Applications in Industry*, VRCAI '10, pages 199–206, New York, NY, USA, 2010. ACM.

[OF05] Daoshan OuYang and Hsi-Yung Feng. On the normal vector estimation for point cloud data from smooth surfaces. *Comput. Aided Des.*, 37:1071–1079, September 2005.

[QiK11] *A stegnographic scheme for 3D point cloud models.* Elsevier, October 2011.

[RVDHV06] T Rabbani, F Van Den Heuvel, and G Vosselmann. Segmentation of point clouds using smoothness constraint. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 36(5):1–6, 2006.

[SF07] Vasiliki Stamati and Ioannis Fudos. A feature based approach to re-engineering objects of freeform design by exploiting point cloud morphology. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, SPM '07, pages 347–353, New York, NY, USA, 2007. ACM.

[SMK$^+$10] Subramani Sellamani, Ramanathan Muthuganapathy, Yagnanarayanan Kalyanaraman, Sundar Murugappan, Manish Goyal, Karthik Ramani, and Christoph M. Hoffman. PCS: Prominent cross-sections for mesh models. *Computer-Aided Design and Applications*, 7(4):601–620, 2010.

[SN06] Jag Mohan Singh and P. J. Narayanan. Progressive decomp osition of point clouds without local planes. In *In Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, 2006.

[SWWK08] Ruwen Schnabel, Raoul Wessel, Roland Wahl, and Reinhard Klein. Shape recognition in 3d point-clouds. In V. Skala, editor, *The 16-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2008*. UNION Agency-Science Press, February 2008.

[SZ09] David Sedlacek and Jiri Zara. Graph cut based point-cloud segmentation for polygonal reconstruction. In *Proceedings of the 5th International Symposium on Advances in Visual Computing: Part II*, ISVC '09, pages 218–227, Berlin, Heidelberg, 2009. Springer-Verlag.

[TZCo09] Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-or. Curve skeleton extraction from incomplete point cloud, 2009.

[Wes] Schuhmann Westkamper, Peter Knorpp. Feature-line extraction from point clouds.

[WHH10] Christopher Weber, Stefanie Hahmann, and Hans Hagen. Sharp feature detection in point clouds. In *Shape Modeling International*, pages 175–186. IEEE Computer Society, 2010.

[WKWL02] H Woo, E Kang, Semyung Wang, and Kwan H Lee. A new segmentation method for point cloud data. *International Journal of Machine Tools and Manufacture*, 42(2):167–178, 2002.

[Yan10] Foerstner-W. Yang, M. Y. Plane Detection in Point Cloud Data. Technical Report TR-IGG-P-2010-01, Department of Photogrammetry, University of Bonn, 2010.

[YK08] Akihiro Sugimoto Ikuko Shimizu Yukiko Kenmochi, Lilian Buzer. Discrete plane segmentation and estimation from a point cloud using local geometric patterns. *International Journal of Automation and Computing*, 5(3):246, 2008.

[YL99] Min Yang and E. Lee. Segmentation of measured point data using a parametric quadric surface approximation. *Computer-Aided Design*, 31(7):449–457, 1999.

[YNBH10] Ichitaro Yamazaki, Vijay Natarajan, Zhaojun Bai, and Bernd Hamann. Segmenting point-sampled surfaces, 2010.

[YQ07] Pinghai Yang and Xiaoping Qian. Direct computing of surface curvatures for point-set surfaces. In Mario Botsch, Renato Pajarola, Baoquan Chen, and Matthias Zwicker, editors, *SPBG*, pages 29–36. Eurographics Association, 2007.

[Zha] Xiao Zhan, Liang. Color-based segmentation of point clouds.