Features Extraction from Point Clouds for Automated Detection of Deformations on Automotive Body Parts

Arjun Yogeswaran and Pierre Payeur
School of Information Technology and Engineering
University of Ottawa
Ottawa, ON, Canada
[ayoge099, ppayeur]@site.uottawa.ca

Abstract—This paper proposes an innovative solution to the problem of extracting feature nodes from a 3D model and grouping nearby feature nodes according to the likelihood that they belong to the same feature. The technique is designed specifically with the problem of detecting unwanted deformations on automotive body part in mind, where feature line detection will not always give the best results. Using an octree representation, the multiresolution method is able to analyze the model for features of various scales. It also uses the octree data structure for feature grouping, and provides an alternative to feature line extraction for connecting similar feature nodes. An existing technique is compared to the proposed approach for feature extraction, and results are presented for the feature grouping method using a point cloud of a miniature car model.

Keywords—feature extraction, surface map analysis, deformation detection, pattern recognition, quality control, automotive body parts.

I. INTRODUCTION

Feature extraction techniques can be very useful to gain meaningful information out of a dataset. The process can advantageously be automated in quality control applications where compliance of the shape of an object with some predefined standards is to be validated. In the context considered for this work, the objective is to automatically detect surface deformations over high resolution 3D point clouds of automotive body parts on an assembly line. The goal is to improve quality consistency over large production volumes. The surfaces considered are characterized by their smooth curvatures, along with some intentional sharper features, such as door handles, molding holes and aesthetic curves; and with some undesirable features, such as damages produced during manufacturing, which are to be detected. Most of the intentional features are larger, while the deformations represent rather small features compared to the overall size of the object under inspection.

For the purposes of this work, features are defined as meaningful variations over the surface of a 3D model. To determine which features are intentional and which features are unintentional deformations, some sort of classification must be performed. Extracting feature segments is the first phase. Being capable of identifying features at various scales is important, since the dimensions of the intentional features and deformations may vary. The second phase is known as feature grouping, and identifies which of the detected feature segments belong to the same physical feature. This stage is important to determine the size and shape of various features on the automotive parts to separate the intentional features from the undesired deformations.

Most methods detect meaningful variations over the surface of a 3D model directly from 3-dimensional datasets and build on volumetric representations. Many techniques can be applied to the extraction of features, but cannot be easily applied to grouping features [1,2]. Some methods propose feature extraction and identify feature lines of a mesh [3,4]. But since the application requires the detection of small features which may not always be connected it would be useful to identify relationships between seemingly different features that may belong to the same desired feature.

This paper proposes a new technique to extract meaningful features from a 3D mesh and to group them based on their proximity and similarity using a single octree structure. Though classification is beyond the scope of this paper, the method works at various scales and provides multiresolution information which can be useful in identifying and classifying important features. The method is divided into two stages: the Feature Extraction stage and the Feature Grouping stage.

The paper is structured as follows. Section II reviews state-of-the-art techniques which can be applied to the problem considered. Section III discusses the proposed feature extraction and feature grouping techniques. Section IV shows the results obtained with the proposed method. Finally, Section V presents future improvements and conclusions.

II. STATE-OF-THE-ART

With the growth of 3D imaging and modeling technologies, the problem of performing pattern recognition over surface shape models has recently received significant attention, but still remains an important field of investigation. Feature extraction has been well-researched recently, with several methods that deliver good results.

A. Feature Extraction from 2D Projections

Two-dimensional feature extraction methods applied on 3-dimensional data can be very effective under some circumstances. This strategy involves generating a 2D representation of 3D data and applying traditional 2D edge detection filters to properly identify features.

978-1-4244-4778-7/09/$25.00 ©2009 IEEE

Two main strategies can be identified in this case: *i*) depth imaging and *ii*) surface normal imaging. The depth imaging method relies on projecting the 3-dimensional model onto a 2-dimensional map, where each pixel's intensity corresponds to a depth value. By examining these depth values, significant changes in the surface shape can be put in evidence, depending on the projection that is considered to create the map. This method results in a 2-dimensional array of depths, where traditional edge detection methods can readily be applied to determine the location of the features. Alternatively, the surface normal imaging method maps pixel elements of the 2-dimensional image to the orientation of surface normals. Significant changes in adjacent surface normals should indicate relevant features. The surface normal image can be analyzed with traditional edge detection techniques also.

These methods have been tested on models representing relatively flat surfaces, where desired features exist on only one side. They prove to be very effective when using good edge detection methods, such as the Canny edge detector [5]. However, with 2D-based methods, a reference perspective is required, and since a 3D model must be projected onto a 2D plane, some faces of the object are inherently missing in the intermediate representation, preempting the extraction of a full set of features. A more robust solution for 3D models, such as automotive body parts which may not be flat or one-sided, is a volumetric feature extraction method.

B. Feature Extraction using Mesh Aggregation

Feature extraction using mesh aggregation was proposed by Lee *et al.* [6], and classifies edges as sharp edges or smooth edges. Sharp edges are drastic changes in adjacent surface normals, while smooth edges are gradual variations between adjacent surface normals, as a smooth curve would produce.

Working on a 3D mesh, all of the adjacent triangular surfaces in the mesh that exhibit a variation in their normal orientation beyond a certain threshold are regarded as boundary meshes. A cost value is assigned to each of the faces, and they represent the likeliness that a face can be on a boundary or not. The area of the polygonal meshes is also considered to detect gradual rounded curves that correspond to smooth edges. A second threshold is used to define the maximum area of a group of polygons that will be merged to form a mesh. Adjacent polygons are merged together to form a larger mesh. When the cost value becomes too high, the curve is an edge. If the surface area becomes too high the mesh is not mapped as an edge.

This method is effective at identifying features at various scales, due to its ability to detect small sharp features as well as large curved features. However, it is still not truly a multiresolution method because it requires different thresholds at different scales. Also, a secondary technique is still required to reduce the density of the mesh.

C. Multi-scale Feature Extraction

A multi-scale feature extraction method was proposed by Pauly *et al.* [7] which allows feature extraction from a 3D object composed of surfaces at various detail levels.

It operates on point cloud data and detects deformations in the surfaces of the model. A weight is given to each point to represent the confidence that it belongs to a feature. A feature is defined by local variations in the surface normals, and for different scales, different neighbourhood sizes are used. Strong features are determined by selecting a threshold and considering variations over that threshold as feature nodes. If a feature persistently exceeds that threshold over multiple scales, it is classified as a strong feature. To extract feature lines from the feature nodes and weights discussed above, a minimum spanning tree is generated.

The results presented in the paper show that this method performs well at identifying prominent features of various scales. However, important features are not always a connected line of feature nodes, but rather correspond to disjointed feature nodes or lines. For the proposed application, they should be counted as a single feature group, but even with reasonable scale selection, this algorithm would cause disjointed feature lines to appear as multiple features.

III. Proposed Method

The proposed method can be separated into two components: feature extraction that searches for nodes of interest, and feature grouping that associates the detected nodes into larger structures.

A. Feature Node Extraction

Woo *et al*. introduced a 3D feature detection technique based on octree structures [8], and uses recursive subdivision of the volume of a 3D model to identify features. It requires a model with a reconstructed surface and partitions the model into subsections which represent varying levels, or scales, of features. Since the data being used is in the form of an ordered point cloud, surface reconstruction is a trivial task.

Given a 3D mesh, surface normal vectors can be calculated for all parts of the surface. They define the orientation of each segment of the mesh. Variations in the orientation of surface normals within a given region are estimated from the standard deviation of normal vectors within that region. This method facilitates the partitioning process. All of the triangles that make up the surface of the object are initially added to the root of the tree structure. The standard deviation of their surface normals is calculated, and compared to a threshold. First the mean normal is computed:

$$\bar{N} = \sum_{i=0}^{n} N_i / n \tag{1}$$

where n is the number of triangles at the node, $\overline{N}$ is the mean normal, and N_i is the unit normal of each of the triangles. Then the standard deviation, σ, of the normals can be estimated as:

$$\sigma = \sqrt{\frac{\sum_{i=0}^{n}(N_i - \bar{N})^2}{n}} = \sqrt{\frac{\sum_{i=0}^{n}(x_i - x)^2 + \sum_{i=0}^{n}(y_i - y)^2 + \sum_{i=0}^{n}(z_i - z)^2}{n}} \tag{2}$$

A threshold must be defined for the subdivision as the maximum standard deviation allowed in a volume before it should be further divided. If the standard deviation is larger than the threshold, the volume represented by the root is divided into 8 octants. 8 children are added to the root and each of those children represents each of the 8 octants that the volume is divided into. The triangles from the root are redistributed based on their spatial location into each of the 8 octants, and thus into each of the 8 children of the root. This process is repeated recursively for each of the children, and their children, and so on until either there are no children remaining, or a sufficient level of feature details is discovered. The depth of the tree determines how detailed the feature level is. Fig. 1 details the recursive process of the feature segmentation at various scales.

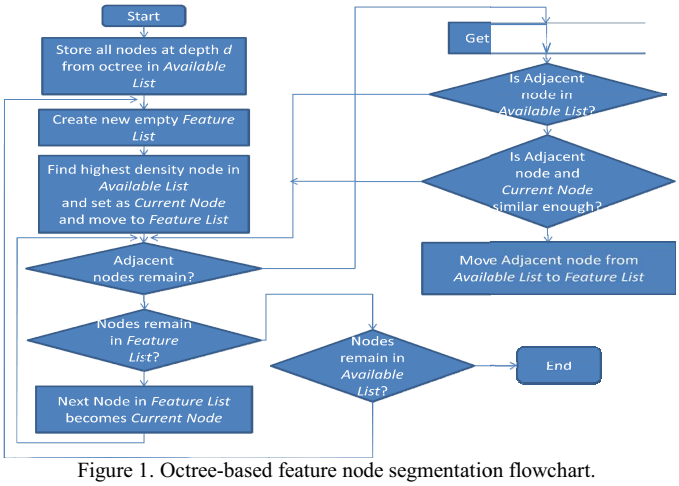


Figure 1. Octree-based feature node segmentation flowchart.

The final structure provides a tree where the surface triangles are distributed amongst the tree nodes. Leaves at greater depths represent finer detailed features of the mesh contained in smaller volumes. Leaves at lesser depths represent larger scale features contained in larger volumes.

The method is very useful in generating a multiresolution representation of the features in a mesh. However, because it uses a single threshold value throughout the entire tree, its ability to detect features can be unpredictable. A feature has to sufficiently affect the standard deviation of the surface normals across the selected volume for the method to investigate the mesh at a higher resolution. Sometimes this is not the case and the feature is not identified. The criteria for setting the threshold is that it must be high enough such that smooth curvatures and noise are not detected as features, but low enough such that the desired features are detected. This remains a subjective criterion that varies with the point cloud. Since standard deviation is used for this distinction, it can be hard to find values which meet the defined criteria.

To improve performance, a modification of the algorithm is proposed that uses a variable threshold for different levels of the tree. When higher threshold values are used at depths closer to the root, small features tend to be ignored given their small contribution to the standard deviation value compared to the entire volume of the object. At lower threshold values, nodes farther from the root have more children because variations in smaller volume give large standard deviation values beyond the threshold, resulting in excess features, such as noise and smooth curves, and a larger tree. To preserve small variations while reducing unnecessary features, and to minimize the complexity of the tree, the threshold should increase at greater depths of the tree, such that the smallest threshold remains at the root, and the largest threshold appears at the leaves. For this experimentation, the relationship between the depth of the node in the tree structure and the threshold has been modeled as a linear relationship:

$$t = a + ix \tag{3}$$

where t is the threshold at tree level x, a is the threshold at the root node, and i is the desired depth increment.

This modification provides more precise results in extracting features, as well as reducing the size of the tree. The relationship can be further optimized to improve performance on various models.

B. *Feature Node Grouping*

Though the previous section proposed a technique to identify features, in order to classify features or gain more information about them, seemingly adjacent features can be grouped together. The detected features are actually segments of a larger feature. By grouping these feature segments, the method aims to define a collection of segments for each major feature in the mesh. The method is therefore extended to group feature nodes such that these groupings can provide information about the size, shape, and other characteristics of the particular feature and aid in classification to distinguish an intentional feature from an unwanted deformation.

The octree generated with the proposed feature node extraction algorithm contains very useful information that can be used to group feature segments. Different features can be at different scales, and since the generated octree is a multiresolution representation, different depths of the octree can be selected as the appropriate scale to group the desired features. The grouping begins by using only the extracted features, that is by evaluating the remaining triangles in the nodes at the deepest level of the tree. Those triangles correspond to feature triangles. All triangles that are not contained in nodes at the deepest level of the tree, defined as non-feature triangles, are removed from nodes at all depths. Information such as the relative occupancy of each node and the standard deviation of the surface normals contained in each node is also important. This can aid in determining which nodes should be connected into forming a larger feature. The standard deviation of the unit normals has already been calculated for each node, since that value was required to generate the octree in the first place. Moreover, since each node represents a certain volume (a cell of the octree) it may contain some of the triangles that define the surface mesh of the object. Therefore, the relative occupancy can be defined as the surface of those triangles with respect to the volume of the node that contain them. The relative occupancy, ρ, can be calculated as follows:

$$A_i = \frac{1}{2}|v_{i1} \times v_{i2}| \quad (4)$$

$$\rho = \sum_{i=0}^{n} \frac{A_i}{v} \quad (5)$$

where v_{i1} and v_{i2} are any two of the edge vectors that define triangle T_i, and v is the volume represented by the node. The triangles used to estimate these values are the feature triangles that remain in the node after removing non-feature triangles.

Two thresholds are required to measure the likelihood that adjacent volumes should be connected. The first threshold defines the maximum allowable percentage difference in the standard deviation of unit normals between the two volumes. The second threshold defines the maximum allowable percentage difference in relative occupancy between the two volumes. It is best to model these thresholds as percentages, rather than as absolute values, since over different levels of the tree the standard deviation values and relative occupancy values will differ greatly. The thresholds are selected such that nodes containing parts of the same feature can be grouped without connecting unrelated nodes together.

With the appropriate values calculated for each node, the feature grouping algorithm can begin. At a given tree level, which indicates the resolution for which the feature grouping will take place, all nodes that were extracted in the previous phase at that level are retrieved and added to a list of available nodes. In the segmentation technique proposed by *Woo et al.*, these nodes were removed to leave only homogenous surfaces [8]. Here, the opposite is done by using these nodes to determine connected features.

The node with the highest relative occupancy in the available list makes the starting node. It is added to the empty feature list and removed from the list of available nodes to define the current node. Each spatially adjacent node is retrieved by traversing the octree to find which nodes are spatially adjacent to the current one [9]. Then each of these nodes is compared to the current node by using relative occupancy and standard deviation of unit normals as metrics. The criteria for determining the similarity of adjacent nodes are that: *i*) the difference in relative occupancy must meet the relative occupancy threshold, and *ii*) the difference in standard deviation of the normals must meet the orientation variation threshold. In practical terms, this means that adjacent feature nodes are populated with a similar amount of feature triangles and exhibit a similar difference in their unit normals orientation. If a node meets these criteria and is on the available list, it is added to the feature list and removed from the available list. The next node in the feature list becomes the current node, and the process repeats until all nodes in the feature list are exhausted or a maximum list size is reached.

Now that a feature has been determined by connecting previously detected nodes in the tree structure associated with the 3D point cloud, the next feature must be detected. The remaining node with the highest relative occupancy in the available list is added to a new empty feature list.

The process described above is repeated until no more nodes remain in the available list. Fig. 2 details the feature grouping process.

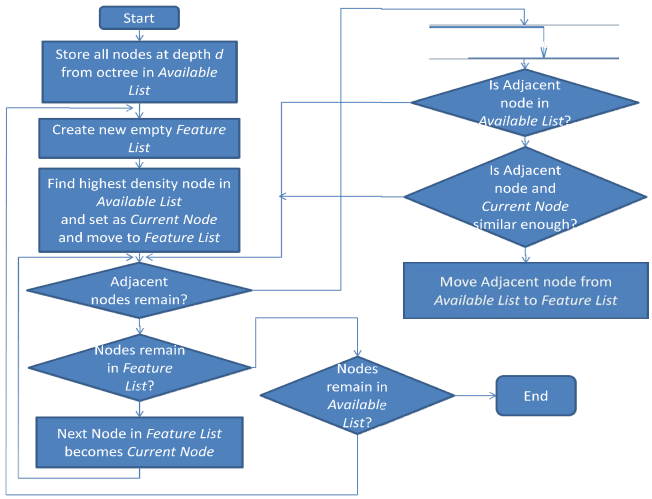


Figure 2. Octree-based feature nodes grouping flowchart.

The result of this algorithm is several lists, each containing one or more nodes. Each list represents a different feature which can then be classified as intentional or undesirable in the surface inspection task considered. Small lists can be removed to avoid noise or unwanted small features.

IV. Experimental Results and Comparison

The model that is used to validate the proposed technique is a 3D point cloud of a miniature car door and side panel. It contains 1024x1024 3D measurements collected with a high resolution laser range finder over a surface of about 50cmx50cm. A simulated deformation was manually created by manipulating the point cloud to introduce a region of 40x40 points where the depth was modified following a step function with an approximate magnitude of 1cm. The relative size of the deformation represents about 0.15% of the surface of the object. Fig. 3 shows a point-base rendering of the high-density 3D point cloud dataset, with the deformation appearing as the narrow darker region on the lower left side of the surface. The surface also exhibits intentional features, such as the door knob, the gap between the door and the side panel, as well as the smooth aesthetic curvature visible along the vertical axis.

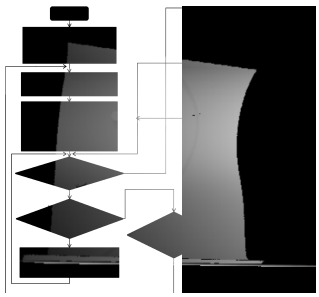
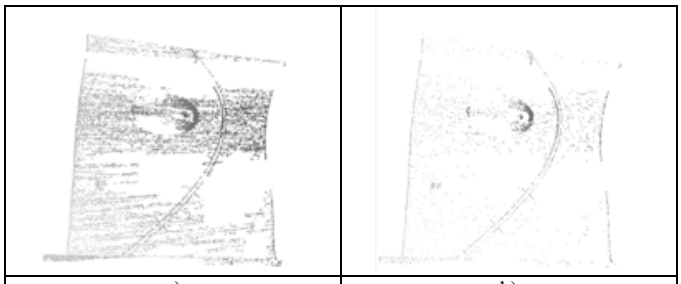


Figure 3. Point-base rendering of 3D point cloud dataset.

A. Feature Node Extraction Results

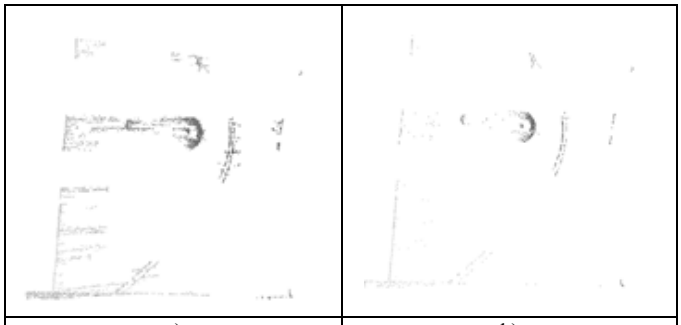
To evaluate the quality of feature node extraction, the original octree-based method of Woo *et al.* is compared to the proposed approach. Fig. 4 and Fig. 5 show the features extracted with the original method applied on the dataset, respectively with a threshold on the standard deviation of the

normals set at 0.50, and set at 0.55. The depth levels illustrated correspond respectively to the 10th level and 16th level in the octree structure, the latter being associated with finer features.



a) b)

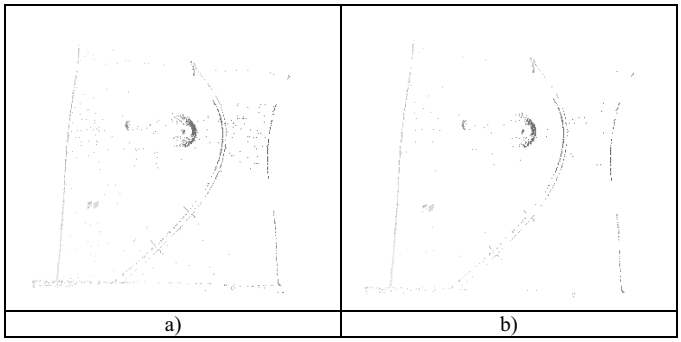
Figure 4. Original method, threshold= 0.50: a) depth 10, and b) depth 16.



a) b)

Figure 5. Original method, threshold=0.55: a) depth 10, and b) depth 16.

Fig. 6 shows the features extracted with the proposed method applied on the same data set with the linear threshold defined in Eq. (3), using a root threshold of a=0.45, and a tree depth increment of i=0.01. These parameters cause the threshold to start at 0.45 at the root node, and increment by 0.01 for every level of the tree, until it ends at 0.61 at the 16th level. The data shown are the features extracted respectively at the 10th and 16th level of the tree, as in Fig. 4 and Fig. 5.



a) b)

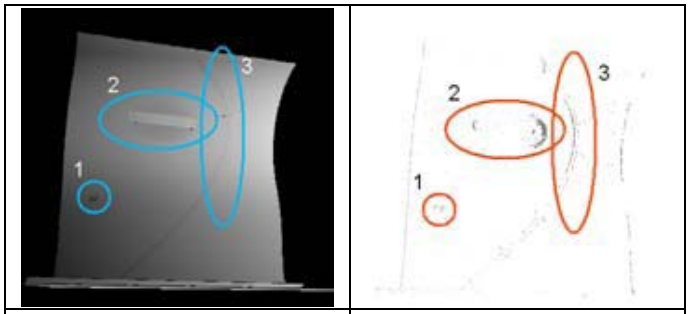
Figure 6. Proposed method, a= 0.45, i= 0.01: a) depth 10, and b) depth16.

It is visible that even with well-selected thresholds, too many features are ignored or too much of the model is identified as features, when using the original method. With a threshold that was too high, at a tree depth of 10, as in Fig. 5a, several important features have already been removed, such as part of the aesthetic curve, the borders of the door, and the manually created deformation. With a threshold that was too low, at a tree depth of 16, as in Fig. 4b, excess features and noise had still not been removed.

With the proposed method, the results shown in Fig. 6 are more accurate. Features of interest are identified with few excess features. At a depth of 10, there are very few excess features, while all major features are still intact. At a depth of 16, more unwanted noisy features have been removed. With the settings considered for the linear threshold, the threshold values range from 0.45 to 0.61, which represents a proper overlap with that of the original method. The proposed approach results in a larger portion of wanted features to be detected while eliminating more unwanted features. The proposed method is also more efficient, as shown by the size of the tree generated with the respective methods. At a tree depth of 16, there are 17718 nodes containing features with the original method with a threshold of 0.50. There are 5656 nodes containing features at a tree depth of 16 with a threshold of 0.55. With the proposed approach, there are only 6714 nodes at a tree depth of 16, which is much less than the original method at a threshold of 0.50, and only slightly greater than the original method at a threshold of 0.55, yet all major features are still intact. Overall, this experimentation demonstrates that the proposed method preserves more relevant data within a smaller data structure. Generating the smaller data structure reduces computation time, and requires fewer resources, which is beneficial, as the end goal is to obtain results in real time on an assembly line.

B. Feature Node Grouping Results

To evaluate the performance of the feature grouping component of the proposed algorithm, three feature groups over the course of various resolutions are examined. The important features are shown and labeled in Fig. 7.



a) b)

Figure 7. a) Important features on point cloud dataset (1:deformation; 2:door knob; 3:aesthetic curve), and b) important features extracted.

The percentage difference threshold on the standard deviation of unit normals is set at 0.70, while the percentage difference threshold on the relative occupancy is set at 0.75. Feature node grouping is experimented and evaluated at tree depths 4 to 8. Feature node grouping is only evaluated at these depths because at greater depths the resolution is very fine and connectivity is determined for very small regions, and at lesser depths, the resolution is so shallow that everything has high connectivity. These depths provide the proper scale for features that are important for the application considered.

Fig. 8 shows a relative occupancy map at each depth, as the resolution increases, where pixel intensity corresponds to the relative occupancy of each node volume as defined in Eq. (5). This figure illustrates how the octree represents resolutions at various depths, as well as showing the relative occupancy values of each node. As resolution increases, the relative occupancy of feature nodes concentrates on finer details, which provides an important mechanism to detect features of various scales within the same point cloud.

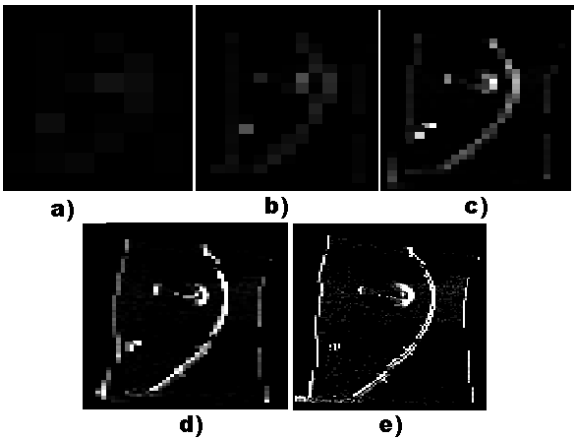


Figure 8. Node relative occupancy as resolution changes, for: a) depth 4, b) depth 5, c) depth 6, d) depth 7, and e) depth 8.

Fig. 9 presents the results of the feature node grouping phase when applied on the nodes extracted in Fig. 6b.

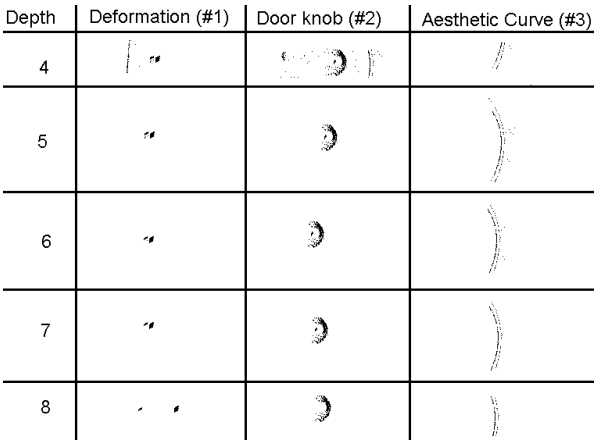


Figure 9. Feature groups obtained at various depths.

The results show consistency over various resolutions by showing the same features being grouped together, such as features 1 and 3 in depths 5 to 7, and feature 2 in depths 4 to 8. Feature 1 gets broken up into multiple groups at the higher resolution of depth 8. When reaching a resolution that is too high for the scale of the feature, the latter may be separated into multiple groups due to a lack of connectivity between adjacent nodes. Unwanted features being grouped with feature 1 at the lower resolution of depth 4 are due to the resolution being too low for the scale of the feature, which results in too much connectivity between adjacent nodes. Similarly, feature 2 is grouped with other nodes at depth 4, but its extraction is immediately refined at depths 5 and beyond, and the dominant piece of feature 2 shows up consistently in depths 5 to 8. The capability to analyze the point cloud at various scales provided by the proposed 2-step approach is very promising. Consistency in the appearance of features over multiple scales demonstrates the algorithm's reliability.

V. Conclusions and Future Work

For the purposes of automotive body parts inspection, the experimental results show that the proposed feature extraction method works very well by providing accurate results on the location of features of interest and by generating a more compact octree data structure than with the original approach. Taking advantage of the octree structure which is inherently a multiresolution representation of the feature extraction, the feature grouping stage can be performed at various resolutions, revealing features of various scales. With the proposed solution, feature node extraction and feature grouping are integrated to perform over a single data structure, rather than necessitating the use of two different frameworks. This proved to be an efficient strategy, with numerous features successfully revealed over the course of several resolutions.

Future work will investigate how the consistency throughout several resolution levels can be exploited to further automate the deformation detection process by reducing dependency on threshold selection. Automatic feature classification will also be integrated.

Acknowledgment

The authors acknowledge the financial support of Precarn Inc., and the collaboration of Neptec Design Group Ltd and Honda Canada to this research.

References

[1] B. Mederos, L. Velho, and L.H. Figueiredo, "Moving Least Squares Multiresolution Surface Approximation", *Proceedings of SIBGRAPHI 2003 – XVI Brazilian Symposium on Computer Graphics and Image Processing*, p. 19.

[2] O. Schall, A. Belayev, and H. Seidel, "Robust Filtering of Noisy Scattered Point Data", *Point-Based Graphics Eurographic/IEEE VGTC Symposium Proceedings*, 2005, p.71-144.

[3] A. Hubeli and M. Gross, "Multiresolution Feature Extraction from Unstructured Meshes", *Visualization, 2001, Proceedings of VIS' 01*, 2001, p. 287-294.

[4] S. Gumhold, X. Wang, and R. MacLeod, "Feature extraction from point clouds", *10th International Meshing Roundtable, Sandia National Laboratories*, 2001, p. 293-305.

[5] J. Canny, "A Computational approach to edge detection", *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 8, 1986, p. 679-698.

[6] Y. Lee, S. Park, Y. Jun, and W.C. Choi, "A Robust Approach to Edge Detection of Scanned Point Data", *International Journal of Advanced Manufacturing Technology*, Volume 23, 2004, p. 263-271.

[7] M. Pauly, R. Keiser, and M. Gross, "Multi-scale Feature Extraction on Point-Sampled Surfaces", *Computer Graphics Forum*, Volume 22, Issue 3, 2003, p. 281-289.

[8] H. Woo, E. Kang, S. Wang, K., and H. Lee, "A New Segmentation Method for Point Cloud Data", *International Journal of Machine Tools and Manufacture*, Volume 42, Issue 2, 2002, p. 167-178.

[9] H. Samet, "Neighbor Finding Techniques for Images Represented by Quad-trees", *Computer Graphics and Image Processing 18*, 1982, p. 37-57.