CLASSIFICATION AND SEGMENTATION OF TERRESTRIAL LASER SCANNER POINT CLOUDS USING LOCAL VARIANCE INFORMATION

David Belton and Derek D. Lichti

Cooperative Research Centre for Spatial Information
Department of Spatial Sciences, Curtin University of Technology
Perth WA, Australia
beltondm@vesta.curtin.edu.au, d.lichti@curtin.edu.au

Commission V, WG 3

KEY WORDS: 3D laser scanning, point cloud, classification, segmentation

ABSTRACT

With the use of terrestrial laser scanning, it is possible to capture thousands of 3-dimensional points on the surface of an object. The problem is that the vast quantity of data that needs to be manipulated sometimes hinders its application. There exist several techniques for the semi-automatic extraction of low-level features that help compensate for this problem. In addition to reviewing some of these techniques, some modifications will also be discussed, along with their use in developing a covariance based procedure to preform classification and feature extraction for terrestrial laser scanning point clouds using local neighbourhoods. Results from this procedure are then used to segment the surface features of the point cloud. The application of this is demonstrated on several captured data sets. Additional information such as methods for defining local surface intersections and direction of principle curvature will also be detailed based on using local information.

1 INTRODUCTION

Terrestrial laser scanning (TLS) is still a relatively new innovation with regards to its application in industry areas. Recently it has seen use in monitoring structural deformation (Gordon *et al.*, 2003), modelling of industrial plants, recording and cataloguing of historical and cultural heritage sites (Langer *et al.*, 2000), landslide mapping (Ono *et al.*, 2000) and it has also been integrated with traditional surveying practices. It has the advantage over traditional surveying and photogrammetric techniques of being able to capture large volumes of 3-dimensional point cloud data efficiently without further processing. Some systems are quoted to be able to capture up to 400 points/cm^2 resulting in gigabytes of data.

While the short capture time may make TLS attractive, the vast quantity of data has meant that the bottleneck in work flow has been shifted from the data acquisition stage in the field, to the processing stage back at the office. Often weeks can be spent on data that took a couple of hours to capture with most of the data comprising unnecessary or redundant information. While some software exists to help the user by semi-automating some of the processes, a more automated approach would help alleviate this bottleneck and remove one of the major drawbacks of TLS.

A variety of techniques from traditional photogrammetric, computer vision and signal processing fields have been applied to the classification of point clouds. Some of these have included transformation into parameter space such as the Hough transform and the Gaussian sphere (Vosselman *et al.*, 2004) which try to group common elements together based on the surface parameters and surface normal information respectively. Techniques such as tensor voting (Tang *et al.*, 2004) and region growing (Besl and Jain, 1988) have been applied which segment the data based on localised information, Morphological approaches such as medial axis and skeletonisation have also been used by utilising diffusion equations, radial basis function and grass-fire techniques (Gorte and Pfeifer, 2004) (Ma *et al.*, 2003).

The main problem encountered with TLS point clouds is that it is a fully 3-dimensional data set that, while the sampling interval is consistent in the scanner's spherical domain, translates into an uneven distribution of points in the 3-dimensional Cartesian domain. Also, unlike airborne laser scanning (ALS) data (which are usually treated as being 2.5-dimensional for this reason), in most instances it is impossible to map the point cloud into a generic space. This is because this space usually takes the form of a complex manifold which cannot be easily unwrapped into 2-dimensions (without taking into account hyper-dimensional constraints). Local methods must also be able to take into account the changing distribution throughout the point cloud, as well as the varying error tolerances based not only on distance, but surface orientation with respect to the scanner (Bae *et al.*, 2005). These problems make application of processing techniques on TLS point clouds more difficult than for most point clouds.

This paper will outline a method to classify points by using the variance of the curvature in a local neighbourhood. A modified region growing algorithm will be given that uses the classification information to segment surfaces. It will then be illustrated how to use the local point information to define points of intersection between adjacent surfaces, as well as a method for determining the principle directions of change and some applications for this. The results of these methods used on practical data sets will be displayed along with future considerations.

2 CLASSIFICATION

2.1 Covariance Analysis

Covariance analysis is often used as a starting point in classification of 3D point clouds based on geometric properties (Pauly *et al.*, 2002). This is performed by determining the covariance matrix for a local neighbourhood surrounding the point of interest referred to as the index point, defined as:

$$\Sigma = COV(X) = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_z^2 \end{bmatrix} \quad (1)$$

with the entities in the covariance matrix for a neighbourhood of size k defined as:

$$\sigma_x^2 = var(x) = E(x^2) - E(x)^2 = \frac{1}{k}\sum_{i=1}^{k}(x_i - \bar{x})^2 \quad (2)$$

$$\sigma_{xy} = cov(x,y) = E(xy) - E(x)E(y) = \frac{1}{k}\sum_{i=1}^{k}(x_i - \bar{x})(y_i - \bar{y})^2 \quad (3)$$

where $E(x)$ is the expected value for an axis ($E(x) = \bar{x}$), and $var(x)$ and $cov(x,y)$ denoting the variance and covariance respectively.

Covariance analysis then proceeds by examining the eigenvalues (λ_i) and eigenvectors (e_i) of the covariance matrix. The eigenvectors will correspond to the principle components (or directions) of the neighbourhood with the eigenvalues denoting the variance in the direction of their corresponding eigenvector. The orientation or the local surface normal for the neighbourhood can then be approximated by the eigenvector with the smallest corresponding eigenvalue. This is because it is equivalent to the formulation of the least squares plane fitting problem (Shakkarji, 1998). The only difference is that in the entries in matrix Σ are not divided by the k and the smallest eigenvalue is equal to the sum of the residuals squared.

By using the variance in the normal direction as a indicator of the quality of fit, we can determine if the neighbourhood is comprised of surface points by setting an adequate threshold. This relies on the assumption that the neighbourhood has the same density and distribution as the entire point cloud i.e. that each neighbourhood is comparable to another. The problem with this is that if the density and distribution is not always consistent, it becomes difficult to set a threshold, especially with point clouds captured using TLS where attributes such as distance from scanner to surface, incident angles and error properties often result in each point having a unique variance value for a surface. Work done by (Bae *et al.*, 2005) does allow the approximate calculation of the unique theoretical threshold value for each point based on the scanners origin and attributes. If the point cloud contains multiple scans, then the problem becomes more complex because of the inclusion of variance summation, ray tracing and combined distribution of points to accurately determine the appropriate threshold, and the scanner information (especially the origin) is not always retained.

Another method to counter this often used in computer graphics is to use the variance in the normal direction over the total variance in the neighbourhood, which is also known as the percentage of population variance or surface variance:

$$\kappa(pt) \approx \sigma_n^2(p) = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \quad (4)$$

which may be considered to be comparable to a normalised plate tensor used in tensor voting (Tang *et al.*, 2004). This value is used to approximate the level of curvature for the neighbourhood and is more consistent with different neighbourhood densities and sizes. The method does have a shortcoming since the value may be attributable to noise or texture (surface properties such as roughness) instead of curvature of the surface. This means that the threshold must also allow for this. Also, while this curvature approximation is not directly effect by changes in neighbourhood size (because the denominator and numerator change proportionally to each other as size of neighbourhood changes), the size should be chosen based on the resolution of the scan and the level of detail. This way, there is a sufficient sample to preform the calculations, but small details are not smoothed over. The next section will detail a measure for classification of surfaces that is less sensitive to some of the effects outlined.

2.2 Surface Detection

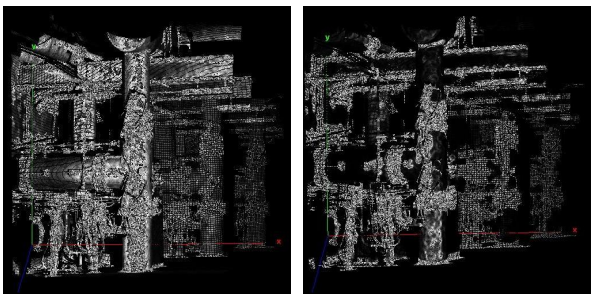
If we examine the geometric properties of a point cloud, then each point can be loosely categorised as belonging to a surface, being on or near an edge or intersection of two or more surfaces, lying on the boundary of a surface, belonging to a line or a singular point. These are generally considered as the lowest level of features, with higher level features comprising of one ore more of these low level features. The main focus in this paper is to label the points as belonging to a surface, edge or boundary group, with the emphasis on surface points.

As mentioned in the previous section, the curvature approximation can be used to determine if a point is such that its neighbourhood comprises of a smooth, near planar surface. However, due to its sensitivity to highly curved surfaces and, more importantly, textured surfaces, it will sometimes misclassify surfaces regions as containing edges. To rectify this, instead of looking for a point with a surrounding neighbourhood having a low curvature value, we look for a point with a surrounding neighbourhood that has a consistent curvature level. This means for surfaces such a pipes or rough surfaces such as corrugated roofs which may have points that do not have a low curvature value, the curvature will still be consistent throughout the region. For points that are near an edge, the neighbourhood will have range of curvature from low to high.

To see if the curvature is consistent, we examine the variance of the curvature throughout the neighbourhood.

$$var(\kappa(pt)) = E(\kappa^2) - E(\kappa)^2 = \frac{1}{k}\sum_{i=1}^{k}(\kappa_i - \bar{\kappa})^2 \quad (5)$$

If the point is on a smooth surface, then the variance should be nominally zero. If we examine a point cloud that contains highly textured or curved surfaces, we can see that the variance of curvature dampens the effects these have compared to just using curvature, as seen in the difference between Figure 1a and Figure 1b. We can also see how the incident angle effects the approximate curvature value on areas of piping and how the variance of curvature is not as significantly effected.



(a) (b)

Figure 1. (a) Gray scale image of the curvature index vs (b) variance of curvature index applied to an industrial scene. Notice contrast between curvature and variance of curvature for the pipes

A drawback to using variance of curvature is that it increases the computation time (but not the complexity). However, it can be

used in conjunction with a curvature threshold, so it only verifies or invalidates edges that are first detected by the curvature measure, and doesn't recheck the majority of smooth planar surfaces that can be detected by just using the curvature threshold. Another drawback to using the variance of curvature is finding a threshold value. Although it is possible to use information to calculate a theoretical value as with curvature, it is a much more complex process, meaning that trials are normally used determine a threshold (although it has been observed to remain relatively fixed for different point clouds). As with curvature, a neighbourhood size should be used so that there is a sufficient sample, but small features are not smoothed over. The neighbourhood size for curvature is normally the same as that used for variance of curvature.

2.3 Boundary Detection

Points that lie on the boundary of a surface (but not near the intersection of two surfaces present in the scan cloud) will initially be classed as surface points since their neighbourhood will have the same properties as that for a interior surface point. However, because the index point lies on the boundary, when the neighbourhood is projected onto the local best fit plane, the distribution of the neighbourhood takes on a more elliptical shape when compared with an interior point which has a distribution with a circular shape. This can also be seen in the difference between the two largest eigenvalues, since they represent the variance in the principle directions on this plane. A small difference will represent an interior point, and a large difference represents a boundary point (Tang *et al.*, 2004).

However since the distribution of the point cloud will most probably not be consistent, this method may be too sensitive. A simpler and more robust technique is to examine the position of the index point in relation to the centroid of the neighbourhood. If there is a large difference, then it is because the point is close to the boundary and there is no points to one side of it. This can be determined by setting a confidence region around the centroid of the neighbourhood and testing to see if the projected index point is outside the confidence region. This is done by using the eigenvalues and the chi-squared test applied to the equation 6 (Johnson and Wichern, 2002).

$$\frac{(u_i)^2}{\lambda_1} + \frac{(v_i)^2}{\lambda_2} \leq \chi^2_2(\alpha) \tag{6}$$

with u and v being the project coordinate system with

$$u_i = e_1 \bullet (pt_i - pt_c) \tag{7}$$
$$v_i = e_2 \bullet (pt_i - pt_c) \tag{8}$$

and e_1 and e_2 being the two eigenvalues with the largest eigenvalues, pt_i as the vector containing the xyz-coordinate data for a point and pt_c being the centroid of the neighbourhood.

2.4 Classification Results

The results of application to two buildings in Figure 2, and an industrial plant in Figure 3 are shown. Both used a constant neighbourhood size of fifty points due to good average resolution of the point clouds. The surface and boundary detection were applied to every point with a variance tolerance of 0.0001 (from user trials). The major problems occur when the density of the point cloud changes or is too low. This can be seen in some of the structures in Figure 3 and with the recessed doorways and windows in Figure 2. One solution is to thin the point cloud so that the distribution becomes more consistent. Care must be taken as this can result in a loss of information that may be detrimental to the results.

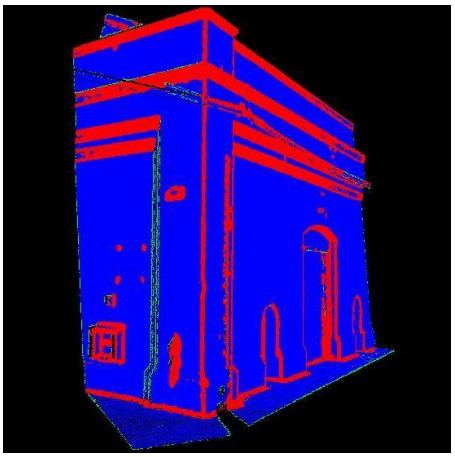


Figure 2. Point cloud from a section of the Midland rail yards in Perth, Western Australia. The classification results are shown with blue representing surfaces, red representing edges and green representing boundaries.

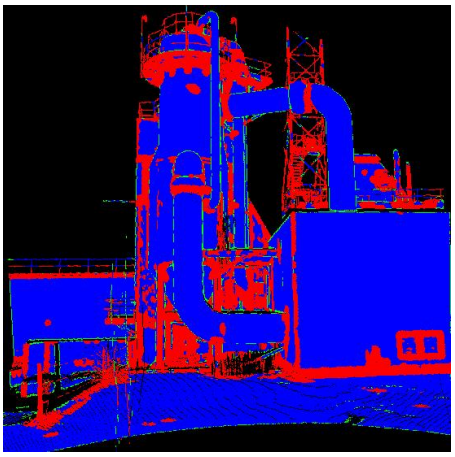


Figure 3. Leica test data of an industrial plant. The classification results are shown with blue representing surfaces, red representing edges and green representing boundaries.

3 SEGMENTATION

Segmentation is a straight forward (although not always easy) process to group common surface points. Often region growing is used on the available information channels to segment the surfaces based on whether they are not significantly dissimilar to its neighbouring points. Since the edge and boundary points have already been identified by classification, this can be used to easily segment the point cloud.

Region growing starts by selecting an arbitrary point that has been classified as a surface point to use as a seed point and labelling it with a new surface label (if it does not already have one). The neighbouring points are then examined, starting from the closest point, and these points are added sequentially to the surface group until a non surface point is found or the distance from the seed point exceeds some limit. If the point being added has been labelled as belonging to a different surface group, then both surface groups are merged. This process is repeated with each surface point as the seed point until all points have been examined.

If a surface group contains to few points, then it may have to be ignored as having too few points to be useful.

```
1:  procedure SEGMENTATION(Points X_i)
2:      for i = 1 to n do
3:          if X_i is a surface then
4:              if X_i = Unlabelled then
5:                  S_i ← new label
6:              end if
7:              Get k nearest neighbours (N_j) for X_i
8:              for j = 1 to k do
9:                  if N_j is a surface then
10:                     Label N_j the same as X_i
11:                 else if N_j is a surface and labelled then
12:                     For all points labelled the same as N_j, label the same surface as X_i
13:                 else
14:                     Break from loop since this is a edge/boundary point
15:                 end if
16:             end for
17:         end if
18:     end for
19:     return
20: end procedure
```

Figure 4. The segmentation algorithm.

The segmentation algorithm (Figure 4) uses the classified edge and boundary points (found by the methods previously outlined) as a crude form of cut plane to stop the region growing crossing from one surface to another. Once the points are segmented, a surface may be fitted to groups so that intersection between surface can be determined and near edge points can be reintegrated into the surfaces.

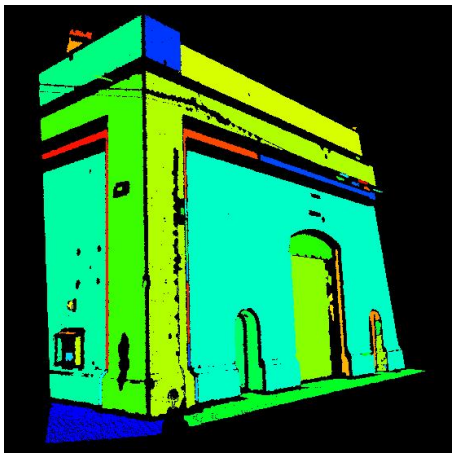


Figure 5. The segmentation results of the Midland rail yards in Perth, Western Australia, using the classification results.

Results of the segmentation algorithm applied to the data sets used in the previous section are given in Figure 5 and Figure 6. One of the biggest problems is the fact that some of the surfaces that we know to exist have be dropped out or ignored. This can be seen in places such as the doorways and wall recesses. This is due to having to few points on these surfaces classified as surface points. However, this problem comes more from the sampling process than the segmentation process since only a few points are taken across these surfaces (such as around the doorways) due to the high incident angle (often seen in ALS point clouds). However, it is not always possible to get an ideal sampling, so other methods need to be developed to deal with this, such as up-

Figure 6. The segmentation results of the Cyclone test data of an industrial plant using the classification results.

sampling (Attene *et al.*, 2003) and edge/step function fitting at discontinuities (Vosselman and Dijkman, 2001).

4 LOCAL INTERSECTION OF SURFACES

Often the intersection of surfaces are the only information needed. Such instance are when the point clouds are being used to generate vector models or 2D floor or elevation plans. To find this, a surface must be fitted to the segments and the intersection between the surfaces calculated. While this is a trivial problem for planar surfaces, it becomes increasingly complex with non planar surfaces. In this case, a local-based method to determine the intersection of surfaces is often used (Attene *et al.*, 2003) (Cooper and Campbell, 2004).

The first step is to determine which points for a surface are adjacent (or close to) another point on another surface. One method is to apply the boundary detection method outlined earlier to only the edge points to determine if the points could be positioned near an edge. Then these points can be checked to see if another boundary point for a surface is within a certain distance. In practice however, it is just as computationally efficient to check each surface point to see if another point belonging to a different surface can be reached.

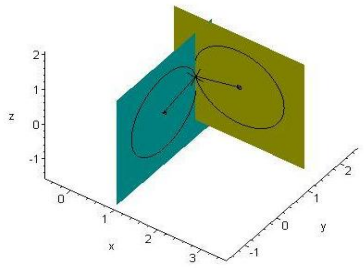


Figure 7. The local intersection point of two surface points defined by where the circles met

The pair of points found are then used to determine the local *optimal* edge point between the two based on a specified cost function. The idea is to extend the boundary of the surface along the local plane to define the intersection (analogous to first order approximation of the surface), and to grow two circles along these planes and set the edge point at where they met (Figure 7). This assumes that the distance to the intersection from the surface

boundary is small so that any effect of surface curvature is negligible. The first step is to get the approximate normal direction for each point and check to make sure that an intersection is possible by checking the angular difference.

$$\theta = \arccos(\hat{n}_1 \bullet \hat{n}_2) \tag{9}$$

where $\hat{n}_1$ and $\hat{n}_2$ are the normal vectors for the pair of points p_1 and p_2. If the angular difference is too small, then the surfaces will be considered parallel and therefore will not have any intersection close to the points. This occurs when there is a jump discontinuity between two surfaces. The next step is to fit a plane through each point based on their normal vectors. These planes may be improved by fitting them through the centroid of neighbourhoods surrounding the each point to reduce errors from noise. The equation for the line of intersection for these planes may be found by using an appropriate method such as:

$$\mathbf{M} = [\hat{n_1}\hat{n_2}] \tag{10}$$

$$b = -\begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \tag{11}$$

$$\mathbf{M}x_0 = b \tag{12}$$

with d_1 and d_2 coming from the plane equations:

$$\hat{n_1} \bullet x_i + d_1 = 0 \tag{13}$$

$$\hat{n_2} \bullet x_i + d_2 = 0 \tag{14}$$

so that x_0 being a point on the intersection and the direction of the line a given by finding the solution to the null space of $\mathbf{M}$ to give the parametric equation for the line of intersection as

$$l(t) = x_0 + t * a \tag{15}$$

To give an approximate point on the intersection of the two surfaces, we can select some value for t. However, the further away from the two surface points it is, the less reliable the intersection approximation of the surfaces will be. Therefore, we should use a value of t such that the approximation point is as close to the surface points as possible.

The best method is maximise the cost function

$$\underset{t}{MAX} \left\{ \arccos\left(\frac{(l(t) - p_1) \bullet (l(t) - p_2)}{\|l(t) - p_1\| \|l(t) - p_2\|} \right) \right\} \tag{16}$$

which is the angle between the line $\overrightarrow{p_1 l(t)}$ and $\overrightarrow{p_2 l(t)}$. This is because as the angle gets smaller, then the distance from the surface points increases and the less reliable the intersection is. However, this is a complex equation to find the optimal closed form solution to (which takes up several pages). A simpler objective function that will produce similar results is to minimise the sum of the squared distances between the surface points to the intersection point.

$$\underset{t}{MIN} \{\|l(t) - p_1\|^2 + \|l(t) - p_2\|^2\} \tag{17}$$

Care must be taken in case the point that is returned is not close to the surface points. This may indicate the parallel surface or poor normal approximations. The usefulness will depend on how good the normal vector approximations are.

Also, this method can be used to incorporate near edge points into the surfaces. This is done by examining the near edge points and determining which surface they could belong to based on the local surface approximation, and if they are within the circles defined on the local surface approximations. There is no guarantee that all the edge points will be integrated with a surface or that they will not be deemed to belong to more than one surface.

Some examples of finding the intersection points using this method are given in Figure 8 and Figure 9. While this works for clean data sets, it is susceptible to the noise and complex structures shown in Figure 8 and Figure 9. However it was able to determine the intersection point of the pipe to the surface in Figure 9 and the majority of the large surface intersections in Figure 8.

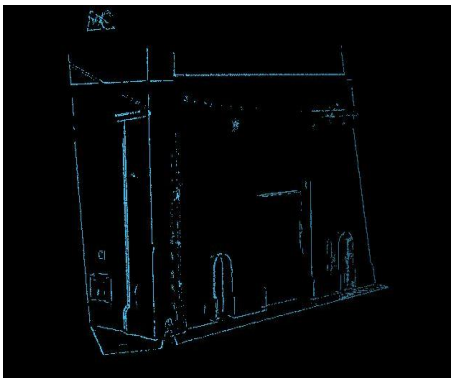


Figure 8. The local intersection applied to the Midland rail yards data set as well as the boundary points to define the extents of the segmented surfaces.

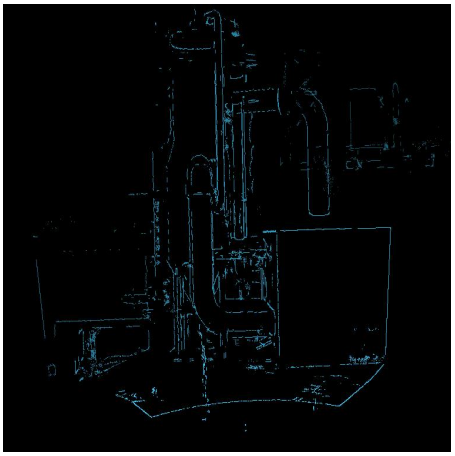


Figure 9. The local intersection applied to the cyclone test data set as well as the boundary points to define the extents of the segmented surfaces.

From these intersection points, techniques such as 3D Hough transforms (Katsoulas, 2003), edge tracking or path following, using edge direction (Tissainayagam and Suter, 2004), region growing and clustering can be used to fit and define edges.

5 PRINCIPAL CURVATURE DIRECTION

Another piece of useful information to have is the principle curvature directions. This can be used to determine the direction of an edge or pipe axis, as well as for finding cross sections of more complex structures. While the two largest eigenvectors may correspond to these directions, they are often too sensitive to the

distribution of the neighbourhood and noise to be consistently reliable. A more reliable method to use is to fit a second order polynomial surface to the neighbourhood of the points (based on a given a local reference plane) and use this to determine curvature direction. Another method that does not require the fitting of a surface uses the normals of the points in the neighbourhood.

The normals in the neighbourhood are the projected onto the plane, as well as their negative image The negative image is projected as well since we do not have a orientation for the normals since they were found by covariance analysis. The covariance matrix for these projected points are then found for the projected axes and decomposed to find the eigenvalues and vectors. These 2D principal directions (eigenvectors) can then be transformed back into the 3D coordinate system of the point cloud. This directions will approximate the minimum and maximum direction of curvature change and the corresponding eigenvalues will approximate the amount of change.

This information will be used in future applications for finding the direction of edges and pipes, as well as the orientation of cut planes to create cross sections of complex structures such as steel flanges. This may also be used to recognise structures by fitting their 2-dimensional profile, tracking edges and path following, or cleaning up the classified edge points by determining the best edge points.

6 CONCLUSIONS AND FUTURE WORK

A classification technique using covariance analysis to approximate curvature and to determine if the neighbourhood surrounding a point was a smooth surface by ensuring that the curvature was consistent throughout (i.e. an almost zero level of variance in curvature for the neighbourhood) was outlined . The boundaries were detected by a chi-squared test on the distance of the indexed point to the centroid of the neighbourhood. A rudimentary segmentation algorithm was given that used the classification information. A localised method for determining the intersection of surfaces was also outlined along with a method for determining the principle curvature directions from surface normals. The general workflow for processing the points is as follows; First the user specifies the tolerance for classification and the neighbourhood size. Then every point in the point cloud is classified based on this information and then segmented base on the outcome. Only near edge points are then re-examined using the local intersection method.

Further work will focus on a more detailed interrogation of the index points based with regards to the principle curvature directions, not just the normal directions. While the local surface intersection is a simple means of finding intersections of surfaces, a more rigorous method is needed to handle discontinuities in surfaces that do not intersect, as well as utilising all the surface information.

ACKNOWLEDGEMENTS

This work has been supported by Curtin University of Technology and the Cooperative Research Centre for Spatial Information, whose activities are funded by the Australian Commonwealth's Cooperative Research Centres Programme. We would also like to thank McMullen Nolan and Partners Surveyors Pty Ltd for data sets of the Midland rail yards.

References

Attene, M., Falcidieno, B., Rossignac, J., and Spagnuolo, M., 2003. Edge-shaperener: Recovering sharp features in triangulations of non-adaptively re-meshed surfaces. In *Eurographis Symposium on Geometry Processing*.

Bae, K. H., Belton, D., and Lichti, D. D., 2005. A framework for position uncertainty of unorganised three-dimensional point clouds from near-monostatic laser scanners using covariance analysis. In *Proceedings of the ISPRS Workshop Laser scanning 2005*, Enschede, Netherlands.

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

Cooper, O. and Campbell, N., 2004. Augmentation of Sparsely Populated Point Clouds using Planar Intersection. In *Visualisation, Image and Image Processing (VIIP)*, Marbella, September, Spain, 359–364.

Gordon, S. J., Lichti, D. D., and Stewart, M. P., 2003. Structural deformation measurement using terrestrial laser scanners. In *Proceedings of 11th International FIG Symposium on Deformation Measurements*, Santorini Island, 25 - 28 May, Greece [CD-ROM].

Gorte, B. and Pfeifer, N., 2004. Structuring laser-scanned trees using mathematical morphology. In *Proc. 20th ISPRS Congress, Istanbul, Turkey*, pp. 929–933. ISPRS.

Johnson, R. A. and Wichern, D. W., 2002. *Applied Multivariate Statistical Analysis* (5 ed.)., Chapter Principal Components, pp. 426–476. New Jersey, USA: Prentice Hall.

Katsoulas, D., 2003. Robust Extraction of Vertices in Range Images by Constraining the Hough Transform. In: *Lecture Notes in Computer Science*, Volume 2652, pp. 360–369. Springer-Verlag GmbH.

Langer, D., Mettenleiter, M., Hrtl, F., and Frhlich, C., 2000. Imaging ladar for 3-D surveying and cad modeling of real-world environments. *International Journal of Robotics Research* 19(11), pp 1075–1088.

Ma, W. -C., Wu, F. -C., and Ouhyoung M., 2003. Skeleton extraction of 3d objects with radial basis functions. *International Conference on Shape Modeling and Applications 2003*, 207.

Ono, N., Tonoko, N., and Sato, K., 2000. A case study on landslide by 3D laser mirror scanner. *International Archives of Photogrammetry and Remote sensing* 35(B5), pp 593–598.

Pauly, M., Gross, M., and Kobbelt, L. P., 2002. Efficient simplification of point-sampled surfaces. In *VIS '02: Proceedings of the Conference on Visualization '02*, Washington, DC, USA, pp. 163–170. IEEE Computer Society.

Shakkarji, C. M., 1998. Least-squares fitting algorithms of the nist alogrithm testing system. *Journal of Research of the National Institute of Standards and Technology* 103(6), 633–641.

Tissainayagam, P. and Suter, D., 2004. Assessing the performance of coner detectors for point feature tracking applications. *Image and Computer Vision* 22, 663–679.

Tang, C. K., Medioni, P. Mordohai, P., and Tong, W. S., 2004. First order augmentation to tensor voting for boundary inference and multiscale analsis in 3D. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(5), 594–611.

Vosselman, G. and Dijkman, S., 2001. 3d building model reconstruction from point clouds and ground plans. *International Archives of Photogrammetry and Remote Sensing* XXXIV(3/W4), pp 37–44.

Vosselman, G., Gorte, B. G. H., Sithole, G., and Rabbani, T., 2004. Recognising structure in laser scanner point clouds. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 46(8/W2), 33–38.