PLASE : A Novel Planar Surface Extraction Method for the Autonomous Navigation of Micro-Air Vehicle

Rafid Siddiqui*, Mohammad Havaei*, Siamak Khatibi*, Craig Lindley*
*Blekinge Institute of Technology, 37179 Karlskrona Sweden.

***Abstract*— A planar surface extraction method is proposed for the indoor navigation of a micro-air vehicle (MAV). The algorithm finds planar clusters from the unorganized pointclouds. This is achieved by implementing a novel approach that first segments the data points into clusters and then each cluster is estimated for its planarity. The method is tested on indoor point cloud data obtained by 3D PrimeSense based sensor. In order to validate the algorithm, a simulated model containing a set of planes has been constructed, with noise injected into the model. The results of the empirical evaluation suggest that the method performs well even in the presence of the noise and non-planar objects, suggesting that the method will be a viable one for use in MAV navigation in the presence of noisy sensor data.**

I. INTRODUCTION

THE Efficient extraction of surfaces from the environment is a fundamental requirement for robust robotic perception, although this is a challenging task and an active research area. It is important to extract only those parts of the environment that are most relevant in the application domain, so the data can be processed effectively and efficiently. Selective data extraction can also result in improvement of the overall robustness of the algorithms used in various crucial areas, including robotic navigation, manipulation, object recognition, and scene analysis, etc. The success or failure of the particular solution in these areas is dependent on the application domain as well as the robotic architecture due to the different demands posed by each domain [1] and the challenges posed by the nature of the maneuverability of the robot. In case of navigation of MAVs (Micro Aerial Vehicles) the challenges are quite different from those faced by ground robots, since MAVs require faster decision making with minimal onboard resources. From this perspective, it is important to retrieve only the most important information from the raw data coming from sensors, and to process only the information that is most crucial to the application domain.

In general, navigation solutions can be divided into two categories: (a) Map-based navigation and (b) Model-based navigation. The former has been applied in ground robotics for the past decade and is robust when computational power is not an issue or decision making is not required so frequently. The approach is to build a map of the environment and update it continuously. The resulting maps are good for path planning and for supporting cognitive abilities of robots, but they require lot of computation and hence slower decision making since the refresh rate of a map is dependent on the scan rates of sensors. Recently, applying such methods for navigation has been a trend in flying robots. Keeping in view the context, we are only discussing the three map based techniques which differ from each other such that they can sub-divide the area. One technique describes a distributed approach while second one describes a conventional SLAM based approach using both laser and stereo vision sensors. The last one gives a semi-map based technique using occupancy grids.

In [2] a decentralized approach to map building is proposed in which one node utilizes a shared map from other nodes. It was observed that this method works well unless one node has a higher error rate than the others, in which case the error rate is reduced for other nodes while it increases for the node itself. A map based technique for GPS denied indoor environments was employed in [3] using a stereo vision camera system and a laser scanner. Obstacle avoidance in such systems is vulnerable since it is dependent on the scan rate of the laser sensor. A good utilization of maps for navigation is achieved by building occupancy grids for obstacle avoidance. In this context a Multi Volume Occupancy grid (MVOG) is an efficient method which divides the 3D space into occupied and vacant space [4].

In contrast to map-based navigation methods, model-based methods utilize primitive shape models that are common in the environment, or grab the structure that is generated from motion. In 2D visual navigation, a lot of work has been performed using optical flow for navigation of MAVs because of its simplicity and resemblance to biological vision systems of insects [5, 6]. In [7] a hybrid approach has been proposed that combines 2D video information from a camera and position information from a laser scanner. The Structure From Motion (SFM) was obtained from video camera images and the information was fused with GPS and inertial sensors in order to improve the accuracy of navigation. The extraction of most primitive features (e.g. plane, cone, sphere, and torus) is another efficient method that can be used for robust navigation of MAVs [8].

In the work reported in this paper, we intend to extract planar structures from the environment, since these are the most common ingredient in the man-made structures. Surfaces are extracted from the point cloud by segmenting it into clusters using Normalized Cuts [9] with an improved association function. The NCut is chosen as a segmentation technique for its broad applicability in the related context [10-11] because of being based on graph partitioning approach which is one of the robust methods used in image segmentation. The complete detail of the algorithm is described in the next section. The resulted planes will be used in ongoing work for the estimation of MAV location in the point cloud of an indoor scene.

This article is organized as follows: Section II provides details of the proposed algorithm. In Section III the architecture of the target MAV platform is described. Experimental results are presented in section IV and section V presents our conclusions.

II. METHODOLOGY

In this section, the methodology used for extracting the planar surfaces from the environment is described. First there is a brief description of NCut and then the improved weighting function is presented that is used in the study. The algorithm is given in the later part of the section.

A. *Normalized Cuts*

The segmentation of a point cloud can be modeled as a graph partitioning problem. Each point in 3D space is modeled as a node V_i and an edge E_{ij} is formed by connecting two nodes in a weighted undirected graph G(V,E). The weight for each edge is the function of the distance from one point to the other point in a pair with small distance resulting in larger weights. This represents a function of similarity between the nodes/points. We want to segment the point cloud into disjoint set of points or clusters V_1, V_2, V_3,…,V_m such that the similarity of nodes in V_i is maximized and similarity across the nodes of V_j is minimized. According to the NC (Normalized Cut) algorithm, the optimal bipartition of a graph into two sub-groups A and B is obtained by minimizing the Ncut value as follows:

$$(1) \qquad Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

where cut(A,B) = $\sum_{(u\in A, v\in B)} w(u,v)$ represent the dissimilarity between A and B and w(i, j) is the similarity between the node i and j. assoc(A,V) is the total connection from nodes of A to all the other nodes and assoc(B,V) is the total connection from nodes of B to all other nodes. Let x be an N=|V| dimensional vector for a partition that divides graph V into two sets A and B such that x_i = 1 if i is in A and -1 otherwise. Let $d_i = \sum_j w(i,j)$ be the total connections from node i to all other nodes. According to [9] an approximate discrete solution to minimize Ncut(A,B) can be obtained by solving the following equation:

$$(2) \qquad \arg min_x Ncut(x) = min_y \frac{y^T(D-W)y}{y^T D y}$$

Where D= diag(d_1, d_2, d_3,…,d_n), $d_i = \sum_j w(i,j)$, W = [w_{ij}], and y = (1+x) – (1-x) $\sum_{x_i>0} d_i \Big/ \sum_{x_i<0} d_i$. Using $y \in \Re$ (2) can be solved by the general Eigen value system:

$$(3) \qquad (D-W)y = \lambda Dy$$

B. *Planar Dissimilarity*

The core of NCut is the representation of the association among the points. This is done using a weighting function describing the similarity/dissimilarity of the data. As mentioned in [9], this function is application specific; hence it needs to be modified in order to work on the target point cloud data. The association function in its original form representing the weights for each edge is given by [9]:

$$(4) \qquad W = e^{-\left(\frac{d_k}{\sigma}\right)^2}$$

where d_k is the distance from node i to node j and σ is the scale factor taken as 5-10 percent of the maximum distance in the original work. After analyzing the histograms of distances for some points it becomes clear that 'w' needs to give more weight to the closer points than the points that are far away. This requires a modified association function with adaptive for each node. Sample histograms showing the distances from a point to all other points are depicted in the figure 1.

In order to get better clustering for 3d pointcloud, an updated weigtening funtion 'w_p' is used which can emphasize closer points more than ones that are far away. This improves the effect of minimization in the Ncut technique. The new weighting function with median $\boldsymbol{\mu}$ taken as the center of gravity of each node is given in the (5).

$$(5) \qquad W_p = e^{\left(\frac{-\mu(2d_k-\mu)}{\sigma_p^2}\right)}$$

where $_p = k$ with 'k=0.25' is used in this study. This makes adaptive for each node, which helps in reshaping the weighting function according to the input data.

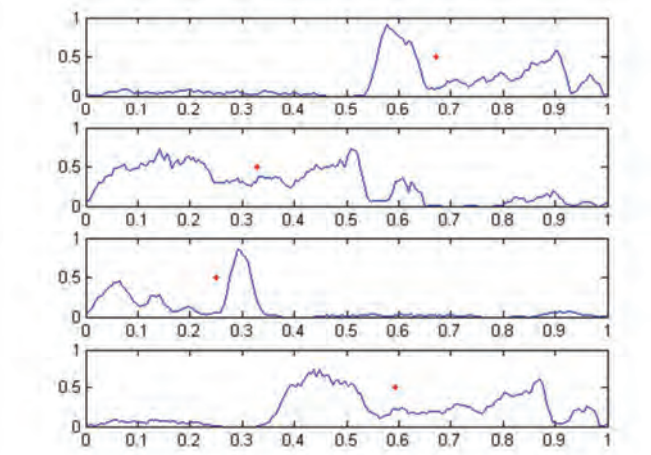


Figure 1: Normalized histograms of some sample graph nodes with respective center of gravities.

The normalized histogram of distances between one node to the other nodes can be seen as indicator of node's association to the other nodes. The center of gravity of the normalized histogram also indicates the center of point clouds mass seen from a node position. The desired weight function would emphasize the importance of the nodes belonging to a cluster and also would separate the outliers nodes. On the other hand the range of distances for a given node to other nodes varies due to the change in the respective neighborhood. For such a variation in the data nodes, a fixed function as was being used earlier, can fail to

distinguish between smooth edges thus resulting in incorrect segmentation of the interconnected or occluded objects in the scene.

The main improvement of the weight function lies in the scale invariant property. It can be observed from the figure 1, that the range of a distances for a given node varies due to the change in the respective neighborhood. However, the weighting function itself is scalable function which is adaptively scaled depending on the center of gravity. For example, 'wp' would generate more amplified response for node 1 than it would generate for node 3 of the data. This results in an overall segmentation that is more sensitive to the variation in the small changes by enhancing the effect of minimization in the NCut which is dependent on the individual weights.

C. *Surface Estimation*

In this section we describe the process of estimating the planarity of each cluster. For '*K*' clusters obtained by NCut we identify those clusters that have a planar tendency. In order to achieve this, Least-Square Plane (LSP) fit is applied to the point normals which are calculated for each point by taking 'K' nearest neighbors and finding a surface normal made by the point with its neighbors. Suppose the normal of the LSP is $\overrightarrow{P} = (p_x, p_y, p_z)$ and the error associated with plane fitting is $E = \sqrt{\sum_{i=1}^{N} d_i / N}$ where '*N*' is the number of points in the cluster and d_i is the distance of the point $p(x_i, y_i, z_i)$ and the plane. The minimization of the error can then be done by using Singular Value Decomposition (SVD) of the following matrix '*S*':

$$S = NN^T \quad (6)$$

where $N = (\boldsymbol{x} - \mu_x, \boldsymbol{y} - \mu_y, \boldsymbol{z} - \mu_z)$ with $\boldsymbol{x} = (x_i)^T, \boldsymbol{y} = (y_i)^T, \boldsymbol{z} = (z_i)^T$ being the x, y and z coordinates of the point normal of the data and $\boldsymbol{\mu}=(\mu_x, \mu_y,, \mu_z,)^T$ is the centroid of the data normals. The eigenvectors with Eigen values $\lambda_1, \lambda_{2,...} \lambda_N,$ are then calculated. The normal vector $\overrightarrow{P}$ can be equated with the Eigenvector with the minimum eigenvalue and the plane fit error can be equated with $\sqrt{\lambda_{min}/N}$. The value of E gives a fast estimate of whether a given cluster is planar or not. If E ≤ then the cluster is considered to be planar, where is taken sufficiently close to zero.

D. *Post Processing*

The planar clusters can have some part of the noise that passes through the plane extraction threshold. As it can be seen in the figure 5(c).These scattered noisy points in the planar clusters should be removed so that they might not affect later processing. In order to remove these points, a local variance based pruning method is implied. Each point is classified as data or noise based on its spread from the 'k' nearest neighbors. The result of this pruning method is shown in the figure 5 (e).

E. *Planar Surface Extraction Procedure*

In this section we describe our novel PLASE(PLAnar Surface Extractor) method. In order to extract planar clusters from the point cloud, the cloud is segmented using Normalized Cuts [9] with a new weighting function as described in the previous section. Each cluster is processed for identification of its planarity nature and candidate clusters are obtained. This planar data is filtered for possible noise, which then would provide the input to RANSAC (RANdom Sample Consensus) that would fit a planar model within the selected clusters although application of RANSAC is not being done in this study.

- Set k=2
- While(true)
  - Extract 'k' clusters C_i for i=1,2,…k
  - For each C_i form a set of planar segments $P = \{C_1, C_{2,...}, C_m\}$ such that $E_i <$ where $m \leq k$.
  - Apply post processing filteration for each P_j $j=1,2,\ldots,m$ in order to remove the noisy data points.
  - $\Delta s = s_{k+1} - s_k$ where $s_k = \sum var(C_i)$
  - If($\Delta s \leq t$) terminate else k = k+1

III. SYSTEM SPECIFICATION

The proposed algorithm is designed to be used on a MAV platform that has been customized as part of the study. The MAV system consists of 6-rotor copter with obstacle avoidance and vision processing capabilities. The architecture of the system is depicted in figure 2. A PrimeSense 3D camera has been mounted on the MAV to provide point cloud data. An on-board CPU with 1.3GHz processing speed is responsible for the vision processing tasks. Lower level obstacle avoidance is performed by a microcontroller that is responsible for generation of he control signals to the MAV's flight controller. The microcontroller uses input from 6 ultrasonic sensors chained together to generate the appropriate control (Yaw, Pitch, Roll) commands to change the state of the MAV.

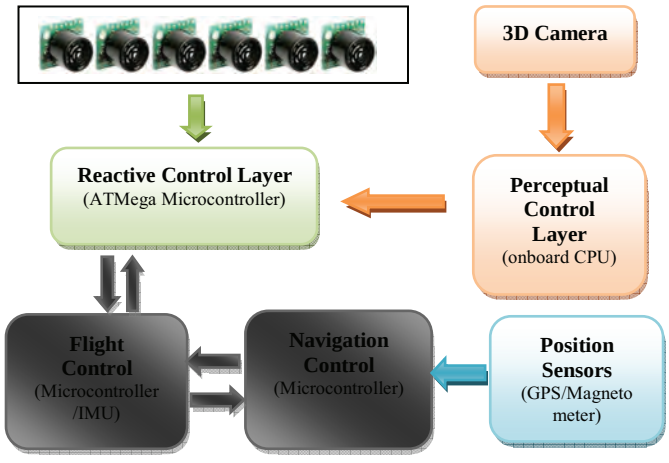


Figure 2: Architecture of the MAV platform with sensor integration and decision making layers.

The reactive control layer is a PID controller which is responsible for the low level decision making for the absolute obstacle avoidance. It also has a higher priority over perceptual control layer. The auto pilot commands are generated as the result of visual algorithm and are sent to the reactive layer which makes decision of forwarding

these commands to the flight controller by observing the signals received from the distance sensors. In case of near obstacle, reactive control generates its own pilot commands to keep the vehicle at a safe margin.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

The proposed algorithm is applied on a set of point cloud data obtained from a 3D PrimeSense camera and the results are visualized. The results of clustering with original and improved weighing functions are depicted in the figure 3 with each cluster in distinct color. These clusters are then tested for their planarity nature using the method explained in the section II and the results can be visualized in the last row of the figure 3. The clustering result has been improved by the use of W_p which enhances the cuts in the graph of the nodes. Adaptivity in the calculation helps to better represent the locality of each node and hence ensures the relative cluster spread. This also makes the method invariant to changes in data due to view point change or sensor noise, although it has been observed that the association function is not invariant across different scales of the data. In this study, each point cloud is down sampled to the same degree and hence every point cloud has the same data size.

In order to validate the algorithm, a simulation model containing a set of planes and noise has been constructed. The planar point cloud model data is injected with Gaussian noise and some random structures to mimic the natural scene as shown in figure 4. This point cloud model is used to generate various indoor scenes containing different numbers of planes and noisy objects, and the performance of the algorithm is measured. The scenes created from the model along with their respective results can be visualized in the figure 5.

For each P_j $j=1,2,\ldots,m$ cluster that results from the application of the algorithm, an error rate $_j$ is measured that reflects the clustering performance of the proposed method over multiple scenes. Each cluster of the ground truth is assigned to a class that is most frequent in the resultant cluster. Then the error rate of this assignment is measured by counting the number of correctly assigned points:

$$(7) \qquad \epsilon(\Omega, \mathbb{C}) = 1 - \left[\frac{1}{n}\sum_k \max_j \left|\omega_k \cap c_j\right|\right]$$

where $\Omega = \{\omega_1, \omega_2, \omega_3, \ldots \omega_k\}$ is the set of clusters and $\mathbb{C} = \{c_1, c_2, c_3, \ldots c_j\}$ is the set of classes.

The error for every scene is calculated 10 times with different spread of the noise each time and the average error for each scene is taken. The evaluation results of each scene with different weighting functions are depicted in the table 1. This reflects the average error rate over multiple runs of the same scene with varying noise ( ) density of the scene. The effect of sigma on error rate for each scene is depicted in the Appendix A1.

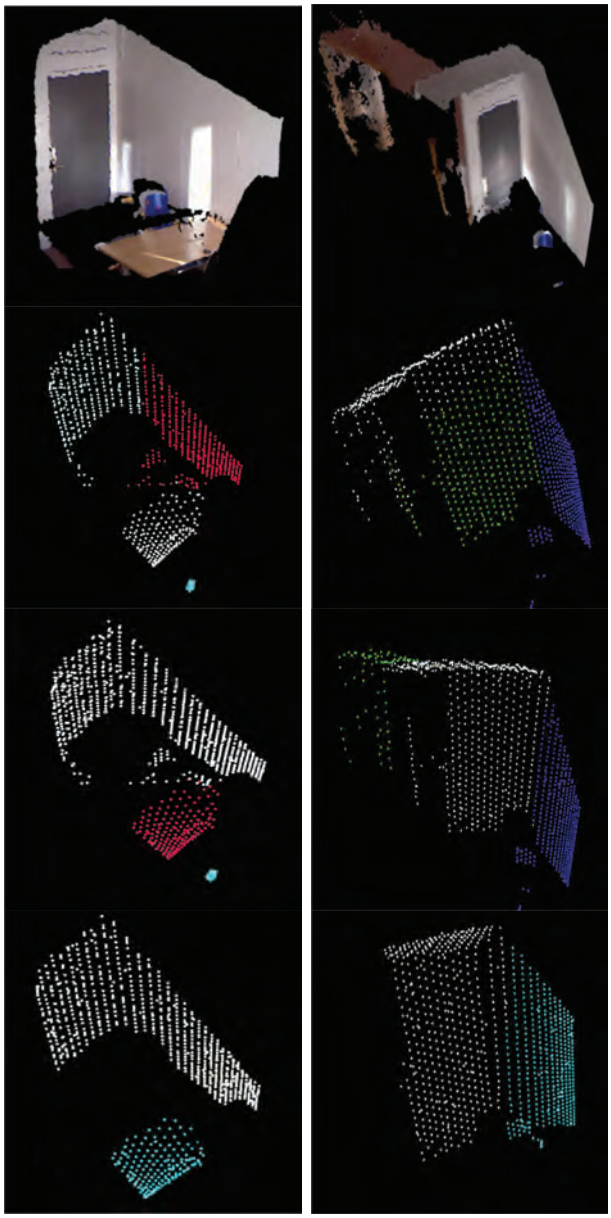


Figure 3: First Row: High resolution point cloud of original scenes. Second Row: clustering using 'W' Third Row: Clustering using 'W_p' and Fourth Row: The clusters from the third row are then tested for their planarity nature to obtain planar clusters as mentioned in section IIC.

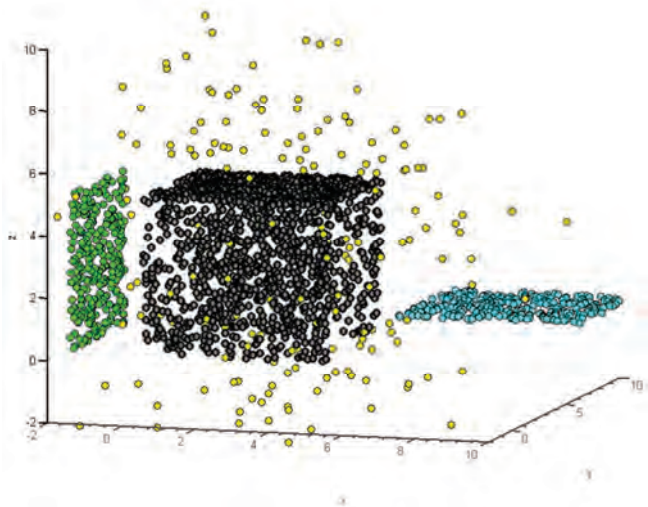


Figure 4: A simulation model of the environment.

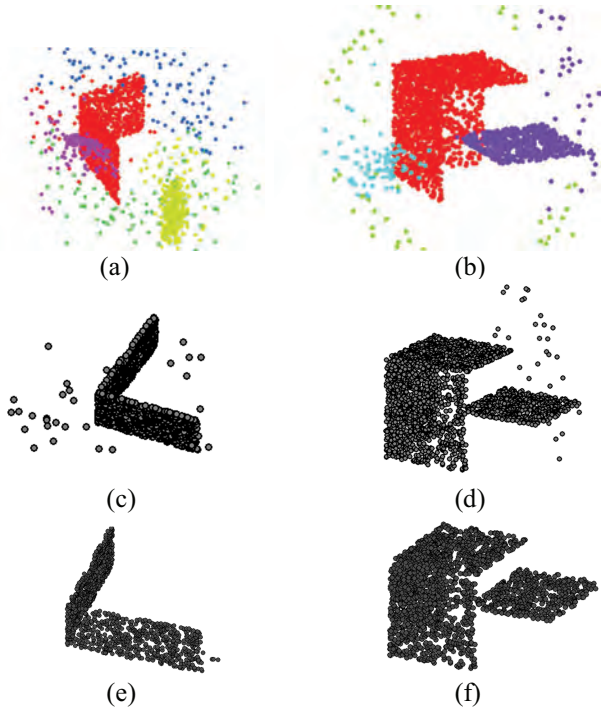


Figure 5: (a)-(b) clustering using simulated scenes. (c)-(d) Plane extraction result on different scenes, (e)-(f) Result of pruning..

The precision and recall are widely used performance metrics for evaluating the correctness of a detection algorithm. They can be considered as an extended version of accuracy that is a simple metric which computes the fraction of instances for which correct result is obtained. In order to compute the precision and recall of the proposed system, labels for a given point is divided into two classes: planes and, non-planes or noise. Recall is then computed as the fraction of correctly classified points as planes among all points that actually belong to the planar class. While precision is the fraction of correctly classified planar points among those that the algorithm believes to belong to the planar class. The relationship of precision and recall can be observed in the figure 6. A high recall means that we haven't missed any plane in the scene but we might have unwanted points in the planar clusters. High precision means that everything detected was planar but we might not have covered all the planar points. Formally, the precision for a given class is the ratio of number of true positive to the total number of elements labeled as belonging to the planar class. Recall in this context can be defined as the ratio of true positives to the total number of elements that actually belong to planar class. So, the precision and recall of the system can be written as following:

$$(8) \qquad P = \frac{\tau^+}{\tau^+ + \psi^+}, \; R = \frac{\tau^+}{\tau^+ + \psi^-}$$

where τ^+ is the number of planar points that are correctly classified as planes and ψ^+ is the number of non-planar points that are classified as planes. Whereas the ψ^- is the number of planar points that are classified as non-planes.

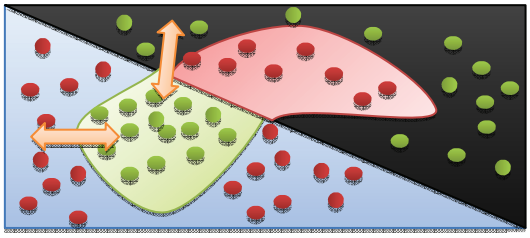


Figure 6: Plane: blue triangle, Noise: grey triangle, τ^+ : green polygon in blue triangle, τ^-:red polygon in grey triangle, ψ^+ : green points in grey triangle, ψ^-:red points in blue triangle, Precision: vertical arrow, Recall: horizontal arrow.

A measure that combines the precision and recall is called harmonic mean or the F-measure. We used F-score as a combined measure in order to calculate the plane extraction performance of the algorithm. The general F_α measure can be given as following:

$$(9) \qquad F_\alpha = \frac{(1+\alpha^2).P.R}{\alpha^2.P+R}$$

Where α is the weight of precision to recall. F_1 is the balanced F-score because precision and recall are equally weighted.

Table 1: Performance of the algorithm using surfaces from simulated scenes. Clustering error is calculated as according to equation 7 and planar cluster estimation column depicts the F-score.

Scene	**Avg Clustering Error** $(\sum \epsilon \; / n)$		**PLASE(PLAnar Surface Extractor)** $F_\alpha = \frac{(1+\alpha^2).P.R}{\alpha^2.P+R}$		
	$(D - W_p)y = \lambda Dy$	$(D - W \;)y = \lambda Dy$	$F_{0.5}$	F_1	F_2
1	0.25	0.44	0.93	0.87	0.82
2	0.12	0.46	0.86	0.86	0.87
3	0.03	0.38	0.97	0.95	0.94
4	0.10	0.25	0.40	0.36	0.30
5	0.05	0.37	0.85	0.80	0.77

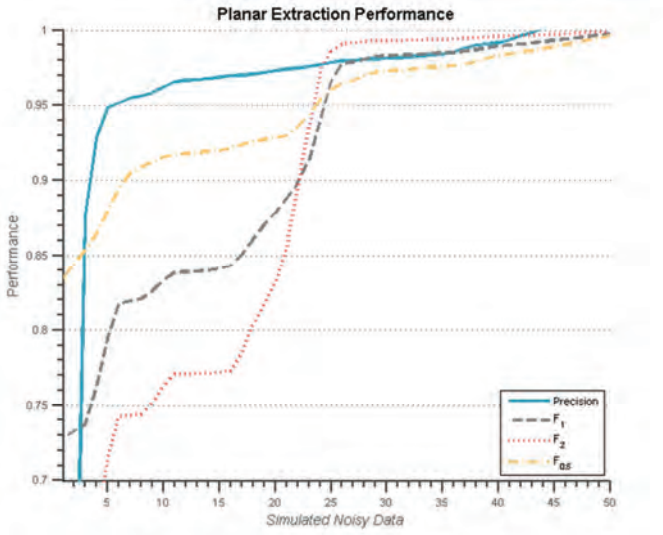


Figure 7: Performance of planar surface extraction. Horizontal axis represents the individual instances of the data for each scene with varying noise and Vertical axis represents the classification performance.

The results indicate the tendency of the algorithm for successfully finding planes in the environment even in the presence of noise and other non-planar objects. As it can be observed in the figure 7 which depicts the performance of the proposed method on the simulated data over multiple runs of the five scenes, that for most of the data instances, the performance (F-score) of the method remains above ninety percent. The proposed plane extraction method is also able to classify the scenes consisting of planar surfaces with good precision rate, as can be observed in the figure 7.

V. CONCLUSIONS AND FUTURE WORK

In this study we proposed a novel plane extraction method (PLASE) that can be used to localize a MAV in an indoor environment. The results suggest that this method performs well even in the presence of noise and non-planar objects, and that it should be effective for indoor navigation of a MAV. The performance of the algorithm is dependent on the success of the individual steps, which means that wrongly segmented data can lead to bad planes. This can be a potential problem that could be overcome by using adaptive clustering in NCut. Due to time constraints, a simple iterative cluster number estimation method was used; this will be improved in future work.

The method is implemented in MATLAB and is not real time; the method will be re-implemented in C and for the MAV platform as part of the future work.

REFERENCES

[1] D. Kragic and M. Vincze, "Vision for Robotics," *Foundations and Trends in Robotics*, vol. 1, 2010, pp. 1-78.

[2] J. Kim, L.O. Lee, E. Nettleton, and S. Sukkarieh, "Decentralized approach to unmanned aerial vehicle navigation: Without the use of the global positioning system and preloaded maps," Professional Engineering Publishing Ltd., 2004, pp. 399-416.

[3] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments," *Unmanned Systems Technology XI, 14-17 April 2009*, USA: SPIE - The International Society for Optical Engineering, 2009, p. 733219 (10 pp.).

[4] I. Dryanovski and W. Morris, Jizhong Xiao, "Multi-volume occupancy grids: An efficient probabilistic 3D mapping model for micro aerial vehicles," *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010), 18-22 Oct. 2010*, Piscataway, NJ, USA: IEEE, 2010, pp. 1553-9.

[5] F. Kendoul, Z. Yu, and K. Nonami, "Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles," *Journal of Field Robotics*, vol. 27, 2010, pp. 311-334.

[6] A. Beyeler, J.-C. Zufferey, and D. Floreano, "Vision-based control of near-obstacle flight," Van Godewijckstraat 30, Dordrecht, 3311 GZ, Netherlands: Springer Netherlands, 2009, pp. 201-219.

[7] T. Kanade, O. Amidi, and Q. Ke, "Real-time and 3D vision for autonomous small and micro air vehicles," *2004 43rd IEEE Conference on Decision and Control (CDC), 14-17 Dec. 2004*, Piscataway, NJ, USA: IEEE, 2004, pp. 1655-62.

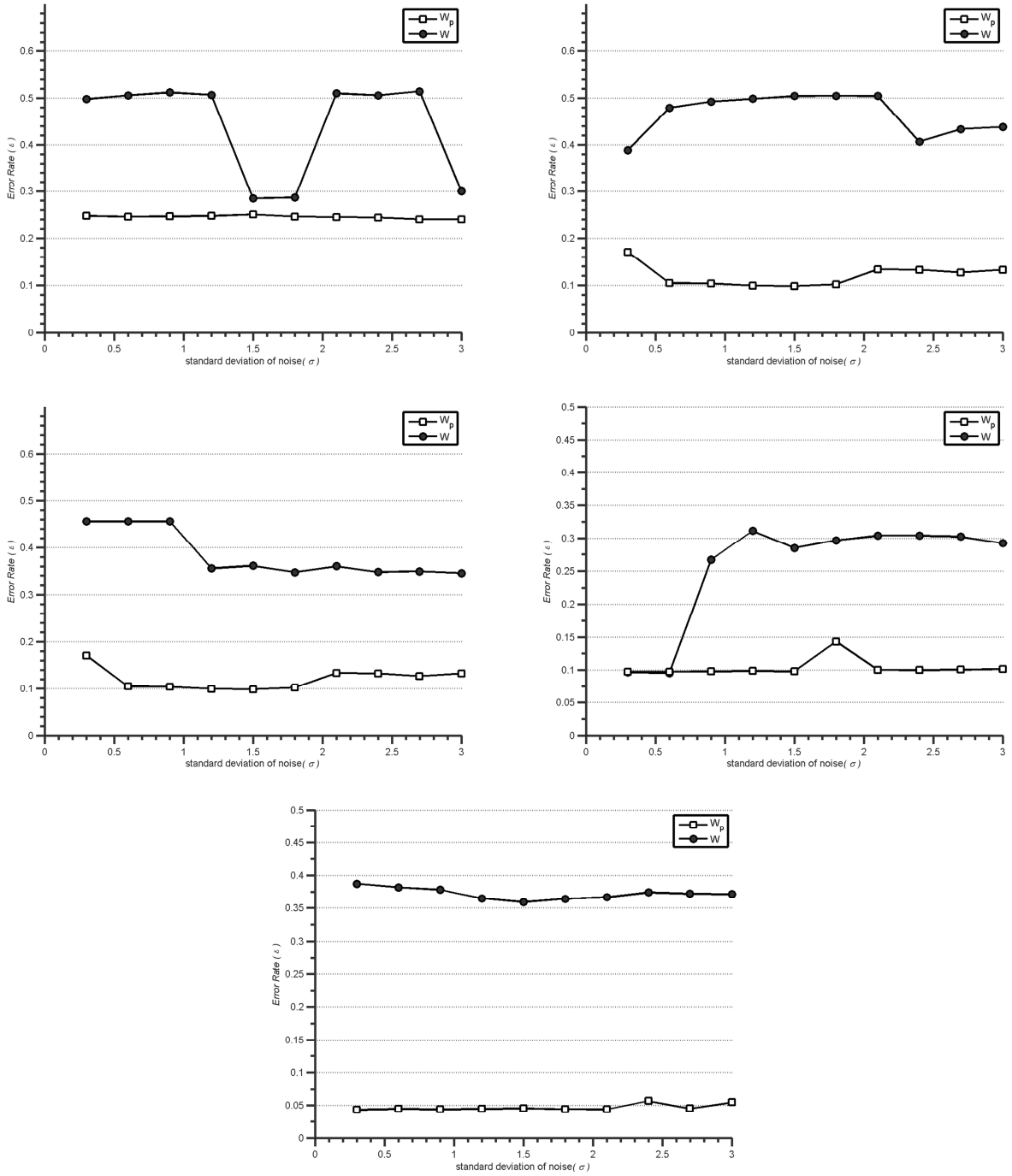
[8] G.-P.M. Hegde, Cang Ye, "Extraction of planar features from Swissranger SR-3000 range images by a clustering method using normalized cuts," *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009), 11-15 Oct. 2009*, Piscataway, NJ, USA: IEEE, 2009, pp. 4034-9.

[9] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, 2000, pp. 888-905.

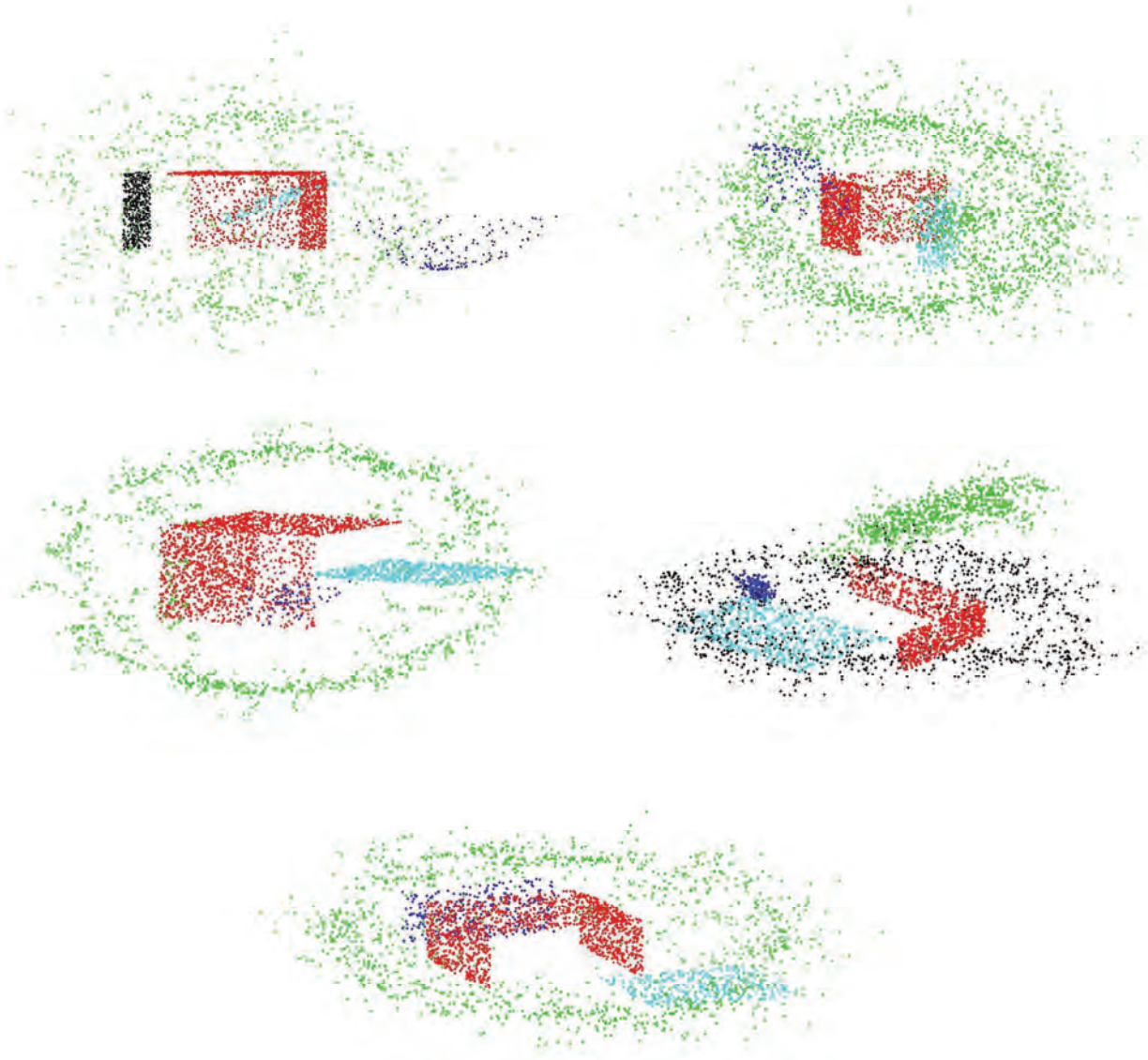
[10] C. Cigla and A. A. Alatan, "Object segmentation in multi-view video via color, depth and motion cues," in *2008 IEEE International Conference on Image Processing, ICIP 2008, October 12, 2008 - October 15, 2008*, San Diego, CA, United states, 2008, pp. 2724-2727.

[11] J. G. Rogers and H. I. Christensen, "Normalized graph cuts for visual SLAM," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009), 11-15 Oct. 2009*, Piscataway, NJ, USA, 2009, pp. 918-23.

APPENDIX



A1: A comparison of error rates of clustering performance using weight functions (W,W_p) for five scenes while each scene has been repeated ten times with varying noise in the simulation model.



A2: Simulated scenes depicting the construction of planar models containing Gaussian noise with =2.1 and some random non-planar objects.