2012 12th International Conference on Control, Automation, Robotics & Vision
Guangzhou, China, 5-7th December 2012 (ICARCV 2012)

An Optimization Approach for 3D Environment Mapping using Normal Vector Uncertainty

Sheraz Khan[1],Nikos Mitsou[2], Dirk Wollherr[1], Costas Tzafestas[2]
[1]Institute for Automatic Control Engineering
Technische Universität München (TUM), 80290 München, Germany
sheraz.khan,dw@tum.de
[2]School of Electrical and Computer Engineering, Division of Signals, Control and Robotics,
National Technical University of Athens (NTUA), Greece
nmitsou@mail.ntua.gr, ktzaf@softlab.ntua.gr

***Abstract*—In this paper a novel approach for 3D environment mapping using registered robot poses is presented. The proposed algorithm focuses on improving the quality of robot generated 3D maps by incorporating the uncertainty of 3D points and propagating it into the normal vectors of surfaces. The uncertainty of normal vectors is an indicator of the quality of the detected surface. A controlled random search algorithm is applied to optimize a non-convex function of uncertain normal vectors and number of clusters in order to find the optimal threshold parameter for the segmentation process. This approach leads to an improved cluster coherence and thus better maps.**

I. Introduction

During the last few years, the increase in computational power and sensor technology has shifted the focus of research in robotic mapping towards the generation of 3D environment representations. Since 3D representations hold more information with regard to scene understanding, they outperform the more common 2D maps. Sensors such as the Microsoft kinect and high resolution stereo cameras can generate accurate 3D point clouds and a number of algorithms have been proposed to extract semantics from this raw spatial data [1], [2].

The environment representation consists of a set of planes that exist in the robot environment and have been captured in a 3D point cloud. The process of building such a representation implies the segmentation of point clouds into surfaces. Different lines of research, such as Markov Networks [3] and the RANSAC algorithm [4] have been proposed to tackle this problem. Our approach is closer to the ones presented in [5], [6] and [7] where region growing methods are used for plane extraction. In our work, registration of the point clouds is not needed since the assumption that the poses of the robot are known is followed.

The proposed approach partitions the environment into 3D cells where each grid cell center is a random variable dependent on the distribution of the points in each cell as well as their inherent uncertainty. No specific assumptions about the error distribution of the raw points are made; rather we adopt a more generic model of uncertainty in which only the standard errors of the points are available. The uncertainty is propagated from the grid level to the level of surface orientation and then utilized in an uncertain clustering algorithm to group cells with similar normal vectors into planes. However, instead of a simple plane extraction, we incorporate the notion of uncertainty of the 3D points in order to extract both planes and their corresponding orientation uncertainty. This uncertainty is later on used as an estimation of the accuracy of the segmentation process. The quality of planes with high uncertainty is improved through an optimization approach by dynamically adjusting the segmentation threshold parameter thus improving the quality of the overall map. Experiments with real robots which produces dense point clouds of indoor environments demonstrate that the algorithm can extract important spatial relations required to develop better 3D maps.

The rest of the paper is organized as follows: A brief literature review is provided in Section II. The uncertain 3D grid and essential concepts are presented in Section III. In Section IV-A, our algorithm for uncertain clustering is presented. The optimization approach is present in Section V. In Section VI, we report our results obtained by conducting different experiments on synthetic and actual sensor data. Conclusions and future work are presented in Section VII.

II. Related work

Initially, the majority of the work in the domain of mobile robot mapping have been focused on building 2D spatial maps [8] [9], [10]. All approaches incorporate uncertainty in the problem description thus leading to probabilistic mapping algorithms. Since the last few years, developing 3D spatial models has gained major interest in the robotics field [1]. Only a few algorithm extend 2D occupancy grid formulation directly to 3D [11]. Most 3D mapping techniques rely on segmentation, clustering and plane fitting techniques. In many of these works, similar approaches on surface detection and modeling have been followed. In [12], the authors use techniques such as outlier removal, resampling, segmentation and model fitting in order to reconstruct an indoor environment, in particular a kitchen. Their data scans refer to the whole environment, and their methods are applied over this static dataset. In [13], the incremental nature of data acquisition has been considered. The authors use an incremental update of their representation by taking into account only the points that do not overlap with existing models. In [2], an incremental segmentation algorithm for 3D points has been proposed. For every point in the new scan its neighbours are found. If any of these points are already assigned to a cluster, the new point is also assigned to this cluster. If more than one clusters are candidates for assignment, then a merging of these clusters is performed. In [14], [15], two approaches for 3D semantic mapping of urban environments are presented. The received point cloud is segmented and planes are extracted. In [16], the surface extraction is followed by classification. A set of hard-coded rules based on the position and size of the surfaces is additionally used to classify them into common labels of walls, floors, ceilings and doors. In [17], a normal vector estimation upon the original points and a voxel growing technique are presented.

In comparison to the above mentioned techniques, we utilize uncertain clustering [18] that takes into account the uncertainty that is introduced during the sampling process. Moreover, instead of applying a filtering algorithm such as RANSAC for outlier removal, we apply an optimization step to improve the quality of the extracted planes based on the propagated uncertainty.

III. Uncertain 3D grid and Mapping

In our scenario, a mobile robot autonomously navigates in an indoor environment without any prior knowledge. The robot perceives the world through its sensors by continuously scanning the environment. Each scan produces a point cloud and as the robot navigates in the environment a collection of point clouds is generated. Starting from the first scan, the goal is to dynamically detect objects that span more than one consecutive scans in an efficient way. The basic components of the grid formulation are mentioned below.

A. Density grid

The collection of 3D scans S_1, S_2, ..., S_t that have been collected during the robot exploration can be considered as a set of 3D points, $S = \{\mathbf{p_1}, \mathbf{p_2}, \ldots, \mathbf{p_n}\}$. There is an inherent uncertainty associated with the location of each point $\mathbf{p_i}$, due to measuring device limitations. A common approach for handling this uncertainty is by assuming knowledge of the underlying probability density function (PDF) that generates this uncertainty. However, PDFs are usually introduced as part of the modeling assumptions and roughly approximate the actual distribution. Therefore, we adopt a more flexible model of uncertainty where we assume that the standard error of the sensor is available [19]. In our case, this error is the standard deviation of the sensor. Thus, an uncertain point is represented as a tuple $(\mathbf{p_i}, \varepsilon(\mathbf{p_i}))$, where $\mathbf{p_i}, \varepsilon(\mathbf{p_i}) \in \mathbb{R}^\mathbf{3}$. In particular, $\mathbf{p}_i = [x_i, y_i, z_i]^T$ are the values in the X, Y, Z dimensions, $\varepsilon(\mathbf{p_i})$ represents the error in the point which is a random variable with zero mean and standard error σ along each dimension. Although the error along different dimensions vary, we make a simplistic assumption. We assume that σ is the same and is equal to the maximum error in any dimension. Intuitively, we can imagine an uncertain point as a sphere of radius σ around $\mathbf{p_i}$.

Due to the huge amount of data that has been accumulated over time, it is inefficient to work upon the original raw data points. Therefore, we partition the data space into a 3D *grid* $\mathcal{U}$ consisting of *cells* $\{u_i\}$ of size ξ along all dimensions and work on the grid instead of the original raw data. In this paper the grid cell size is always bigger than the standard error of the points. The grid itself is dynamic and expands as new data scans are accumulated. Thus, memory is reserved only for areas where objects are detected. The number of points falling into a cell u comprise the *density* of the cell. Since a lot of cells might be empty or contain only a few points, we employ a *density threshold* τ to distinguish between *dense* and *sparse* cells.

For each cell, we maintain a set of statistics.

Definition 1 (Cell statistics): Let u be a cell in the grid and let $\{\mathbf{p_1}, \ldots, \mathbf{p_n}\}$ be the set of uncertain points mapped to u. The cell statistics contains the following entries explained below:

- The linear sum of the points for each dimension, $L \in \mathbb{R}^3$: $\sum_{i=1}^{n} \mathbf{p_i}$.
- The square sum of the points for each dimension, $S_{sq} \in \mathbb{R}^3$: $\sum_{i=1}^{n} \mathbf{p_i^2}$.
- The square sum of the errors for each dimension, $E_{sq} \in \mathbb{R}^3$: $\sum_{i=1}^{n} \varepsilon(\mathbf{p_i})^\mathbf{2}$.
- The density of u.

When a new point is added to the cell, the cell statistics are updated by summing up the point coordinates and errors to the corresponding entries of the cell. With these statistics, we maintain the virtual center as well its uncertainty for each cell. The virtual center $\mathbf{C_u}$ is a random variable given by the current instantiation of the center and the mean of the errors associated with the points in the cell:

$$C_u = \frac{\sum_{i=1}^{n} \mathbf{p_i}}{n} + \frac{\sum_{i=1}^{n} \varepsilon(\mathbf{p_i})}{n} \tag{1}$$

Note that the error is a random variable with zero mean. Therefore, $E[\varepsilon(\mathbf{p_i})] = \mathbf{0}$. This means that by considering the expected value of the center, the error factor is eliminated. To account for the error factor, we estimate the square of the Euclidean norm.

The expected distance between an uncertain point and the uncertain virtual center of a cell is defined as follows:

Lemma 3.1 (Expected distance): Let u be a cell in the grid containing the points $\{\mathbf{p_1}, \mathbf{p_2}, \ldots, \mathbf{p_n}\}$ and let $\mathbf{C_u}$ be its virtual center. Let $\mathbf{p_j}$ be an uncertain point, such that $\mathbf{p_j}$ is not assigned to any cell in $\mathcal{U}$. The expected value of the square of the distance between $\mathbf{p_j}$ and $\mathbf{C_u}$ is given by:

$$E[||p_j - C_u||^2] = p_j^2 + \varepsilon(p_j)^2 + \frac{(\sum_{i=1}^{n} p_i)^2}{n^2} + \frac{(\sum_{i=1}^{n} \varepsilon(p_i))^2}{n^2} - 2p_j \frac{\sum_{i=1}^{n} \mathbf{p_i}}{n} \tag{2}$$

Equation (2) describes the analytical case, i.e when the raw data points of the cell are available. In our case, though only the statistics are available for each cell. We can re-phrase (2) based on the cell statistics as follows:

$$E[||p_j - C_u||^2] = p_j^2 + \varepsilon(p_j)^2 + \frac{L^2}{d(u)^2} + \frac{E_{sq}}{d(u)^2} - 2p_j \frac{L}{d(u)} \quad (3)$$

Equation (3) defines the expected square distance between a point and the virtual center of a cell. A point is assigned to the cell in its neighbourhood based on the smallest expected square distance. In [19] further details regarding derivation of this expression are presented.

IV. Surface clusters

For the robot, point clouds do not offer any direct information about the spatial structure of the environment. To allow for geometric interpretation and abstraction, the robot should be able to process this data and extract patterns of spatial information. Typically, surfaces are used for the description of an environment; walls, doors, tables consist of (or can be partitioned into component) surfaces. Thus, the patterns we are focusing on in this work are *surfaces*. However, 3D point clouds represent only a noisy sampling of surfaces that exist in the real world and the explicit information about the *orientation* and *curvature* of the surfaces is lost during the sampling process. Normal vector estimation aims at restoring this information for every sampled point by constructing a vector that is orthogonal to the tangent plane of that point. To do so, existing methods compute the least square plane fitting in the neighborhood of the given point [20].

In our case, the notion of normal vector for points is extended to grid cells and also considers the uncertainty of the points. For the normal vector computation, the cell neighborhood should first be defined.

Definition 2 (Cell neighborhood):
Let u be a cell. Let d be the depth parameter, $d \geq 1$. The neighborhood of u in depth d, denoted by $N_d(u)$, consists of: *i)* all cells u' that are directly connected to u, and *ii)* all cells u'' for which there exists a path of cells $\langle u_1, u_2, \ldots, u_d \rangle$, $u_1 = u$, $u_d = u''$ such that u_{i+1} is directly connected to u_i, $1 \leq i \leq d$.

The normal vector of a cell is estimated by computing the least square plane fitting its neighborhood. Recall that each cell is represented in terms of a virtual center. To account for the uncertainty of the virtual centers and estimate its effect on the normal vectors, we employ a *min-max approach* that computes the maximum variation in the parameters of the plane. In particular, we perturb each virtual center proportionally to its uncertainty based on the center of the plane and calculate the best fitting plane in each case. The variation in the plane parameters is a direct indicator of the uncertainty in the normal vectors.

Definition 3 (Normal vector of a cell):
Let u be a cell and let $N_d(u)$ be its neighborhood in depth d. The normal vector $\overrightarrow{u}$ of cell u is a random variable with an expected value given by the best plane fitting $N_d(u)$ and a deviation approximated by the min-max approach.

Based on the vicinity of the cells in the grid and their normal vectors denoting the surfaces they belong to, we can now define the notion of surface clusters.

Definition 4 (Surface cluster):
A surface cluster is a maximal set of connected dense cells $\{u_1, u_2, \ldots, u_k\}$ belonging to the same surface.

In order to extract the surface clusters, we use an uncertain version of the clustering algorithm presented in [21] explained in the next section.

A. Uncertain 3D clustering

1) Extraction of planes: We first map S to the grid structure and extract the cell statistics according to Definition 1. Then, we find the dense cells based on the density threshold τ. The dense cells that have normal vector uncertainty lower than a certain threshold ω are considered for clustering with a density based uncertain algorithm. Neighboring cells are grouped into clusters if their normal vectors have similar orientation.

In particular, a new cluster C is created starting from a random cell u: the normal vector of C is initialized to the normal vector of u. The algorithm tries to expand the cluster based on the directly connected dense cells u' from u. The expansion is possible iff u' and C belong to similar surfaces, i.e. if the cosine similarity of the normal vectors is less then the orientation distance threshold ϕ. Note that both $\overrightarrow{u'}$ and $\overrightarrow{C}$ are uncertain normal vectors, thus we compute their expected distance similarly to (3). If the addition is possible, the statistics of C are updated so as to consider the effect of u. We maintain cluster statistics similarly to unit statistics (Definition 1). The algorithm restarts at some other dense cell u that has not been visited yet and continues until no more unvisited dense cells exist.

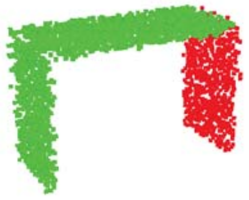
The output of the algorithm is a set of clusters/planes; each cluster is represented in terms of its component dense cells and of a normal vector describing the orientation of each plane.

V. Optimization over the initial clustering

In this section, we discuss an optimization approach which improves the quality of the map based on the notion of uncertainty of each cluster. As mentioned in the previous step, the clustering algorithm utilizes a global orientation threshold ϕ for the whole point cloud. However, the algorithm is sensitive to the value of this threshold and areas with different densities and uncertainties cannot utilize the same threshold. As shown in the experimental section, this would lead to erroneous results such as two different planes being grouped into one.

In order to improve the quality of the clustering step, we apply an additional step of optimization over the extracted clusters. This optimization is done only for clusters with high uncertainty in the surface orientation vectors. Given the map built by the robot consisting of clusters, normal vectors and their respective uncertainities, the optimization approach aims at modifing the initial clusters based on the orientation threshold to minimize the uncertainty in each cluster while maintaining an appropriate number of clusters. To find the optimal orientation threshold for each cluster C, we apply the clustering algorithm described in the previous section over all points of C for different orientation thresholds ϕ:

$$[s_x^i, s_y^i, s_z^i, N_c] = f_c(\phi, C)$$


(a) $Threhold = 0.97$

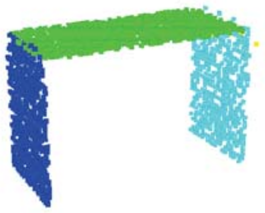

(b) $Threshold = 0.7$

Fig. 1. Results on the synthetic dataset. The datasets (a) Wrong Clusters (threshold 0.97), (b) Clusters after optimization (threshold 0.7)

TABLE I
AVERAGE UNCERTAINITIES OF NORMAL VECTORS FOR SYNTHETIC DATA

	s_x	s_y	s_z
Without Optimization	0.34	0.18	0.34
With Optimization	0.09	0.13	0.15

```
**Algorithm** ControlledRandomSearch()
**Input:** Orientation Threhold $\phi$,
        Maximum Iterations $\zeta$,
        Cluster $C$,
**Output:** $\hat{\phi}$ , Optimal orientation threshold
        $[\hat{s}_x, \hat{s}_y, \hat{s}_z, \hat{N}_c]$,
        $\hat{s}_x, \hat{s}_y, \hat{s}_z$ Minimum normal vector uncertainty,
        $\hat{N}_c$ Optimal number of clusters,

1. Randomly sample based on constraints $0 \leq \phi \leq 1$:
      $\phi_s = \{\phi_1, \phi_2, \ldots, \phi_n\}$ ;
2. Evaluate cost function $\beta$ at $\phi_{samples}$:
      $\beta_s = \{\beta_1, \beta_2, \ldots, \beta_n\}$;
3. while(counter $\leq \zeta$)
4.       Find $\beta_{max}$ = max($\beta_s$)
5.       Generate $m$ samples from $\phi_s : m \leq n$;
6.            $\phi_m = \{\phi_1, \phi_2, \ldots, \phi_m\}$
7.       Find centroid $\overline{\phi}_{1:m-1}$;
8.       Generate new sample $\phi_{new} = 2\overline{\phi}_{1:m-1} - \phi_m$;
9.       Evaluate cost function: $\beta_{new}$ given $\phi_{new}$;
10.      If($\beta_{new} < \beta_{max}$);
11.           replace $\phi_{max}$ with $\phi_{new}$ ;
12.      **increment** counter;
13. **end**;
```

Fig. 2. The pseudocode of controlled random search algorithm.

where s_x, s_y and s_z are the uncertainties in the normal vectors for all newly generated clusters (in case the cluster is split up) and N_c represents the number of these clusters. For every cluster C we keep the threshold that optimizes the cost function given the contraints:

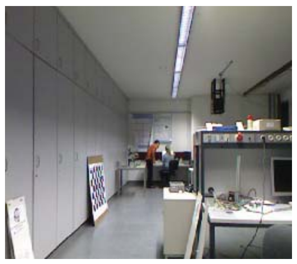
$$\beta(s_x, s_y, s_z, N_c) = w_{nv} \sum_{i=0}^{N_c} (s_x^i + s_y^i + s_z^i) + w_{nc} N_c$$

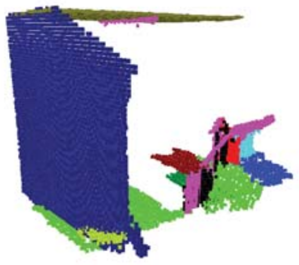
$$0 \leq \phi \leq 1,$$

where w_{nv}, w_{nc} represent the weighting factors for the normal vectors and number of clusters respectively. The addition of N_c in the cost function is essential otherwise the optimization would generate a large number of clusters with very small uncertainties.

An analytical evaluation of the cost function shows that it exhibits a staircase property. The cost function can remain constant for a certain range of orientation threshold values, however a change in number of clusters can cause a huge increase/decrease in the cost function. These sudden changes attribute to staircase property or step behaviour in the cost function. This property of the cost function makes all gradient based optimization algorithms not applicable for our case. Thus, in order to find an optimal configuration for each cluster, we implement a "controlled random search" algorithm [22] over the search space of the orientation threshold ϕ. The optimization is done for all uncertain clusters and since the search space is not high dimensional, the algorithm converges to an optimal configuration with respect to the ground truth.

The pseudocode of the controlled random search optimization is given in Fig 2. The input parameters to this function are the initial value of the orientation threshold and the maximum number of iterations ζ required for the optimization. The outputs of the algorithm are the optimal (minimum) uncertainities of the normal vectors and number of clusters based on the trade off between these two elements. A large increase in the number of clusters can cause the overall uncertainty to reduce however may not represent the optimal configuration and vice versa. The algorithm begins by generating n samples, $\phi_s = \{\phi_1, \phi_2, \ldots, \phi_n\}$ from the search space based on constraints and evaluating the cost function at each of these samples (line 1-2). The algorithm finds the value ϕ_{max} with the maximum cost β_{max} (lines 4) and then tries to replace this maximum value by generating a new sample using 'm' random samples from the set ϕ_s (line 5-6). The new sample ϕ_{new} (line 8) is calculated by using the centroid of $\overline{\phi}_{1:m-1}$ (line 7) and sample ϕ_m. If the cost function value


(a) Actual Environment


(b) Without Optimization

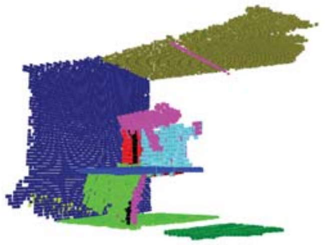
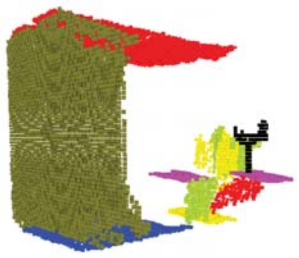

(c) Without Optimization

Fig. 3. Results on the actual dataset.(a) Actual Environment (b) Without Optimization, (c) Without Optimization


(a) With Optimization

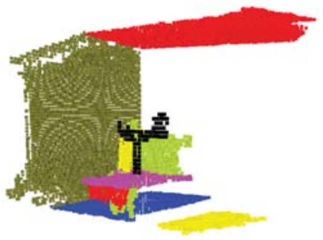

(b) With Optimization

Fig. 4. Results on the actual dataset.(a) Clusters after Optimization , (b) Clusters after Optimization

TABLE II
Average Uncertainities of Normal Vectors for Real Data

	s_x	s_y	s_z
Without Optimization	0.32	0.36	0.18
With Optimization	0.22	0.25	0.13

at the new sample ϕ_{new} is less then ϕ_{max} (line 10-11), then values are swapped, otherwise the iterations continue until the variance in the sample set falls below a certain threshold or the maximum number of iterations are reached.

VI. Experiments

In this section, an experimental evaluation of the proposed optimization technique on a synthetic as well as a real dataset is presented. The synthetic dataset consists of 3 planes forming a table-like structure. In case the clustering algorithm fails due to incorrect clustering threshold (leading to higher cluster uncertainities) the optimization approach can improve the quality of those clusters. Similarly we also evaluate the optimization approach on a real dataset and present results.

A. Results on synthetic data

The synthetic dataset is presented in Figure 1. During the generation of the 3D point clouds for these planes, white noise has been added to the points in order to simulate the uncertaint data collection of real devices. In order to demonstrate the applicability of our approach, we initialize the clustering procedure with a very high orientation distance threshold (0.97 rad). In Figure 1(a), it is shown that the clustering algorithm merged two planes into one. However after applying the controlled random search optimization approach, the optimal orientation distance threhold for this specific cluster is found (0.7 rad) leading to the correct representation of the actual environment. These results on sythetic data set has shown that it is possible to improve cluster quality based on the normal vector uncertainty for each cluster. Table I shows the average uncertainties in the normal vectors with and without optimization. It can be seen that the optimization approach clearly minimizes the average uncertainties of the normal vectors.

B. Results on real dataset

An evaluation of this optimization approach is also presented in an actual scenario, in which a robot is present in an indoor environment. The experimental environment consists of several objects (e.g. walls, doors, tables and chairs). The scans are collected through a kinect device mounted on the robot. The actual environment is depicted in Figure 3(a). The results without optimization are shown in Figure 3(b) and Figure 3(c). It can be seen that in flat regions such as the wall, ceiling and floor the planes are estimated with high certainty. However, at cluttered regions that contain small objects such as the monitor and table etc, the clustering algorithm does not perform well. Due to high uncertainty in the planes, small surfaces are merged together (such as the *pink* cluster which is tilted at a certain angle) that do not correspond to the real environment.

The evaluation of the same scenario with the controlled random search optimization approach is presented in Figure 4(a) and Figure 4(b). It can be seen that the optimization approach splits up regions with higher uncertainty. For example, the *pink* cluster is split up into multiple planes (such as the *yellow* and the *black* cluster) orthogonal to each other representing the actual scenario in a more accurate manner.

Table II shows the average uncertainities of normal vectors corresponding to all the clusters. It can be seen that the uncertainities are higher when the optimization approach is not used and this increase can be attributed to clusters which have been merged incorrectly as shown in Figure 3(b). In retrorespect to this case the average uncertainities of all clusters are lower since all incorrectly merged clusters have been split up as shown in Figure 3(c).

VII. Conclusions and Outlook

In this paper, we present a new approach for optimization of point cloud segmentation. Our approach propagates the uncertainty on the spatial attributes of the points in the point clouds to the normal vectors of the 3D cells. An optimization approach is used to improve the quality of clusters based on the normal vector uncertainty. The extraction of the surfaces and their uncertainties is a process that does not require intensive calculations. It is based on the update of the statistics of the 3D cells and the min-max approach and can be executed online. On the other hand, the cluster improvement technique is a slow iterative process that can be applied once a complete map of the environment is present to ensure that all clusters are optimized with respect to the uncertainty criterion.

One possible future research direction could be to improve the computation time for the optimization algorithm. Further improvements are needed in order to generate planes with better quality in an online fashion with real time constrains. Additionally, a more complex optimization over different thresholds such as the depth parameter and the cell size used by the clustering algorithm will be considered and is expected to further improve the resulting map.

VIII. Acknowledgments

This work is part of the EU-STREP project IURO (Interactive Urban Robot), supported by the 7th Framework Programme of the European Union, ICT Challenge 2 Cognitive Systems and Robotics, contract number 248317, see also www.iuro-project.eu.

References

[1] J. Weingarten and R. Siegwart, "Ekf-based 3d slam for structured environment reconstruction," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 3834–3839.

[2] K. Klasing, D. Wollherr, and M. Buss, "Realtime segmentation of range data using continuous nearest neighbors," in *ICRA*, 2009.

[3] R. Shapovalov and A. Velizhev, "Cutting-plane training of non-associative markov network for 3d point cloud segmentation," in *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*. IEEE, 2011, pp. 1–8.

[4] B. Oehler, J. Stueckler, J. Welle, D. Schulz, and S. Behnke, "Efficient multi-resolution plane segmentation of 3d point clouds," in *Proceedings of the 4th international conference on Intelligent Robotics and Applications - Volume Part II*, ser. ICIRA'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 145–156. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-25489-5_15

[5] R. Rusu, Z. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3d point cloud based object maps for household environments," *Robotics and Autonomous Systems*, vol. 56, no. 11, 2008.

[6] A. Zavodny, P. Flynn, and X. Chen, "Region extraction in large-scale urban lidar data," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 1801–1808.

[7] T. Rabbani, F. van Den Heuvel, and G. Vosselmann, "Segmentation of point clouds using smoothness constraint," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, no. 5, pp. 248–253, 2006.

[8] S. Thrun, "Exploring artificial intelligence in the new millennium," G. Lakemeyer and B. Nebel, Eds. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, ch. Robotic mapping: a survey, pp. 1–35. [Online]. Available: http://dl.acm.org/citation.cfm?id=779343.779345

[9] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *Robotics, IEEE Transactions on*, vol. 23, no. 1, pp. 34–46, 2007.

[10] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *International Joint Conference on Artificial Intelligence*, vol. 18, 2003, pp. 1151–1156.

[11] K. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems," in *Proc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation*, 2010.

[12] R. B. Rusu, Z. C. Marton, N. Blodow, A. Holzbach, and M. Beetz, "Model-based and learned semantic object labeling in 3D point cloud maps of kitchen environments," in *IROS*, 2009.

[13] Z. C. Marton, R. B. Rusu, and M. Beetz, "On fast surface reconstruction methods for large and noisy datasets," in *ICRA*, 2009.

[14] D. Wolf, A. Howard, and G. Sukhatme, "Towards geometric 3d mapping of outdoor environments using mobile robots," in *IROS*. IEEE, 2005.

[15] R. Valencia, E. Teniente, E. Trulls, and J. Andrade-Cetto, "3d mapping for urban service robots," in *IROS*. IEEE, 2009.

[16] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 11, 2008.

[17] J. Deschaud and F. Goulette, "A fast and accurate plane detection algorithm for large noisy point clouds using filtered normals and voxel growing," in *5th International Symposium on 3D Data Processing, Visualization and Transmission, Espace Saint-Martin, Paris, France*, 2010.

[18] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed. Academic Press, 2006.

[19] C. C. Aggarwal and P. S. Yu, "A framework for clustering uncertain data streams," in *Proc. ICDE*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 150–159.

[20] K. Klasing, D. Althoff, D. Wollherr, and M. Buss, "Comparison of surface normal estimation methods for range sensing applications," in *ICRA*, 2009.

[21] N. Mitsou, I. Ntoutsi, D. Wollherr, C. Tzafestas, and H. Kriegel, "Revealing cluster formation over huge volatile robotic data," in *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*. IEEE, 2011, pp. 450–457.

[22] W. Price, "A controlled random search procedure for global optimisation," *The Computer Journal*, vol. 20, no. 4, pp. 367–370, 1977.