Kernel-Based 3-D Dynamic Occupancy Grid Mapping with Particle Tracking

Youngjae Min, Do-Un Kim, and Han-Lim Choi

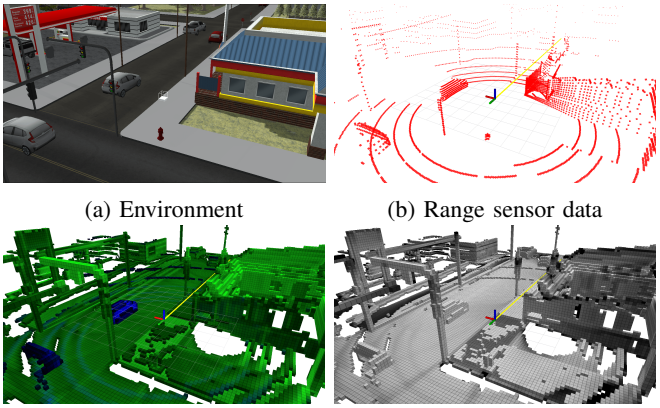
***Abstract*— Mapping dynamic and three-dimensional (3-D) environments is essential for robots and UAVs, but also a challenging task to consider the increased dimensions in both space and time compared to 2-D static mapping. This paper presents a kernel-based 3-D dynamic occupancy grid mapping algorithm, called K-DOGM, that distinguishes between static environments and moving objects while estimating the velocities of dynamic grids as well. The proposed algorithm brings the benefits of kernel inference such as its simple computation, consideration of spatial correlation, and natural measure of uncertainty to the domain of dynamic mapping. We formulate the dynamic occupancy grid mapping problem in a Bayesian framework and represent the map through Dirichlet distribution to apply kernel inference in a recursive way with intuitive heuristics. The proposed algorithm demonstrates its promising performance compared to baseline in diverse scenarios simulated in ROS environments.**

I. Introduction

Robots understand their states and surroundings through environment perception. Mapping three-dimensional (3-D) environments is especially essential for autonomous aerial vehicles (UAVs) moving in 3-D space to keep a safe space from terrain and obstacles with possible movements. Mapping local environments is often done with occupancy grid map by estimating whether each discretized space, i.e., grid, is occupied or not. In addition to the occupancy information, it is also necessary to distinguish dynamic obstacles from static environments and estimate their dynamic states, e.g., velocity, for vehicles to plan a collision avoiding maneuver.

Our goal is to efficiently build a 3-D map that estimates the occupancy and velocity of each grid from online measurements stream. Related to this work, existing methods that handle static occupancy mapping focus mainly on relaxing the assumption of grid independence by considering spatial correlation. The sparsity and noise of sensor measurements can make inconsistencies between environments and their occupancy maps. These discrepancies become more problematic in 3-D mapping because sensor rays in 3-D space are sparser than those in 2-D space. Several works have been proposed to incorporate such spatial correlation, including the methods based on Gaussian process regression [1] and logistic regression with hilbert maps [2], [3]. In recent studies [4], [5], they take advantage of Bayesian update to provide an uncertainty estimate of their prediction. Unfortunately, these methods are neither capable of identifying moving objects nor suitable for dynamic environments.

The authors are with the Department of Aerospace Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, 34141, Korea (email: yjmin313@kaist.ac.kr; dukim@lics.kaist.ac.kr; hanlimc@kaist.ac.kr)



(a) Environment (b) Range sensor data

(c) Classification map (d) Uncertainty map

Fig. 1: Data flow and outputs of K-DOGM in simulated urban environments. The algorithm classifies the two moving cars by representing dynamic grids as blue while static grids as green in (c). (d) indicates more uncertain estimation with darker color in gray scale.

On the other hand, methods that address the estimation of dynamic state pose the challenge of differentiating dynamic objects from the static environment. The sound mathematical formulation can be found in [6], [7], [8], but they are proven not to be real-time capable. Nevertheless, [9] proposed a primitive version of Sequential Monte Carlo Bayesian Occupancy Filter (SMC-BOF) exploiting a particle filter to make a real-time viable algorithm. Furthermore, this idea has been combined with the Dempster-Shafer theory of evidence by Tanzmeister et al. [10]. By the virtue of multi-hypothesis evidential representation, its following studies have shown promising results [11]. However, these works are focusing on 2-D environments, and simply extending them to 3-D environments requires extensive computing power and memory due to the higher degree of freedom of 3-D environments.

In this paper, we present an efficient and robust 3-D dynamic occupancy grid mapping algorithm, called K-DOGM. It distinguishes between static environments and moving objects, provide uncertainty estimates of the classification, and estimate the velocities of dynamic grids. The main contribution of our work is that we developed an algorithm that exploits both spatial and temporal correlation by representing the map with Dirichlet distribution and applying kernel inference to it with intuitive heuristics. The proposed algorithm extends the existing kernel-based static mapping algorithm [5] to dynamic environments and brings the benefits of kernel inference to the literature of dynamic mapping.

The major challenges of extending the static mapping algorithm in [5] to dynamic environments are as follows. First, dynamic occupancy can be identified only through temporal correlation because common range sensors such as LIDAR provide occupancy information without classifying static and dynamic states. Therefore, the existing recursive update equation needs to estimate whether the occupancy measurements are for dynamic or static objects. Second, inconsistency between accumulated map and new measurement cannot be resolved well. In dynamic environments, such inconsistency happens frequently as dynamic objects move into vacant space that has not been occupied for a long time. In this situation, previously accumulated information overwhelms the current measurements, and thus, the algorithm disregards the environmental change that the measurements tell.

II. Background - Bayesian Generalized Kernel Inference

The static occupancy grid mapping algorithm in [4], [5] introduced Bayesian Generalized Kernel Inference (BGKI) by extending the counting sensor model proposed in [12]. The counting sensor model represents occupancy probability through a Bernoulli likelihood function and counts how often a beam has ended in and passed through each grid. BGKI adapts the discrete counting scheme to continuous space and imposes spatial correlation between occupancy predictions at nearby points by considering some neighborhood of a query point.

For the i^{th} grid, its occupancy probability θ^i is modeled with Beta distribution. The prior of θ^i is given by $Beta(\alpha_0)$ where $\alpha_0 := \{\alpha_{0,O}, \alpha_{0,F}\}$ shapes the occupancy probability θ^i and free probability $1-\theta^i$. When range sensor measurements $\mathcal{X} := \{X, Y\}$, a set of location and its occupancy state (0 for 'free' and 1 for 'occupied') pairs, are available, the posterior of θ^i is updated to follow $Beta(\alpha^i)$ with

$$\alpha^i_O = \alpha_{0,O} + \sum_{(x.y)\in\{X,Y\}} k(x, x^i)y \tag{1}$$

$$\alpha^i_F = \alpha_{0,F} + \sum_{(x.y)\in\{X,Y\}} k(x, x^i)(1-y) \tag{2}$$

where the kernel function is defined as

$$k(x,x') := \begin{cases} \sigma_0[\frac{1}{3}(2+\cos(2\pi\frac{d}{l}))(1-\frac{d}{l}) + \frac{1}{2\pi}\sin(2\pi\frac{d}{l})] & \text{if } d < l \\ 0 & \text{if } d \geq l \end{cases}. \tag{3}$$

σ_0 and l are the kernel scale parameter and length scale parameter, respectively. Since the update only requires the measurements within a distance of l from the query point, employing a $k-d$ tree enables efficient neighborhood search. The update can then be evaluated in $\mathcal{O}(\log M)$ time for M measurements, and updating the entire N grids takes $\mathcal{O}(N \log M)$ time.

Note that the neighborhood formed by the kernel function let each grid reflect measurements beyond its boundary. We adapt this scheme to consider spatial correlation among neighboring grids. Also, we deal with the distribution of occupancy probability parameters, θ^i for BGKI, instead of directly affecting the parameters as in the evidence-based algorithms [8], [11]. The indirect update of occupancy probability provides a natural way to measure the uncertainty of the estimation through evaluating variance (see Sec. III-B).

III. Dynamic Occupancy Grid Map Representation

In this section, we formulate dynamic occupancy grid mapping problem in a Bayesian framework. Then, we introduce our approach to represent the grid map in order to solve the estimation problem.

A. Problem Formulation

In 3-D dynamic occupancy grid mapping problem, the environments are represented by N grids $\mathcal{G} := \{g^1, g^2, \cdots, g^N\}$ with the center position of each i^{th} grid at $x^i \in \mathbb{R}^3$. The purpose of the problem is to estimate the occupancy state $\omega^i_t \in \Omega := \{F, S, D\}$ of each i^{th} grid at time-step t where F, S, and D represent that the grid is not occupied (free), statically occupied, and dynamically occupied, respectively. The state estimation at time-step t is based on accumulated range sensor measurements $\mathcal{X}_t := \{X_k, Y_k\}^t_{k=1}$ where X_k and Y_k represent locations of measurements and measured values at time-step k, respectively.

We formulate the estimation problem in a Bayesian framework. The posterior for the i^{th} grid at time-step t is then expressed as below.

$$p(\omega^i_t \mid x^i, \mathcal{X}_t) \propto \underbrace{p(Y_t \mid \omega^i_t, x^i, X_t, \mathcal{X}_{t-1})}_{\text{update}} \underbrace{p(\omega^i_t \mid x^i, \mathcal{X}_{t-1})}_{\text{prediction}} \tag{4}$$

This factorization can be dealt with in a 2-step procedure similarly as common Bayesian filters such as Extended Kalman filter and particle filter [13]. The last term can be further expanded as

$$p(\omega^i_t \mid x^i, \mathcal{X}_{t-1}) = \int p(\omega^i_t \mid \omega^{1:N}_{t-1}, x^{1:N}, \mathcal{X}_{t-1}) \, p(\omega^{1:N}_{t-1} \mid x^{1:N}, \mathcal{X}_{t-1}) d\omega^{1:N}_{t-1}. \tag{5}$$

We can interpret this formulation as predicting the new state ω^i_t from the previous posteriors $\{p(\omega^j_{t-1} \mid x^j, \mathcal{X}_{t-1})\}^N_{j=1}$ without new observation. Then, the first term updates the predicted posterior to the true posterior with the new measurements $\{x_t, y_t\}$. Our algorithm is developed upon this formulation.

B. Occupancy with Dirichlet Distribution

We model the true posteriors as Dirichlet distribution, which is a multivariate generalization of Beta distribution used in [5], over the three probability values for the states F, S, and D:

$$\theta^i_t := p(\omega^i_t \mid x^i, \mathcal{X}_t) \sim Dir(\alpha^i_t) \tag{6}$$

where $\theta^i_t = \{\theta^i_{t,\omega}\}_{\omega\in\Omega}$ with $\theta^i_{t,\omega} := p(\omega^i_t = \omega \mid x^i, \mathcal{X}_t)$, and $\alpha^i_t := \{\alpha^i_{t,\omega}\}_{\omega\in\Omega}$ are the concentration parameters.

Similarly, we model the predicted posteriors as Dirichlet distribution:

$$\theta_t'^i := p(\omega_t^i \mid x^i, \mathcal{X}_{t-1}) \sim Dir(\alpha_t'^i) \tag{7}$$

When visualizing a map, we use the parameters in color coding by

$$RGB = (0,\ \mathbb{E}[\theta_{t,S}^i],\ \mathbb{E}[\theta_{t,D}^i]) \tag{8}$$

An advantage of using Dirichlet distribution is the simple computation of its mean and variance. They are easily calculated as below.

$$\mathbb{E}[\theta_{t,\omega}^i] = \frac{\alpha_{t,\omega}^i}{\alpha_{t,A}^i} \quad \forall \omega \in \Omega, \tag{9}$$

$$Var[\theta_{t,\omega}^i] = \frac{\mathbb{E}[\theta_{t,\omega}^i](1 - \mathbb{E}[\theta_{t,\omega}^i])}{1 + \alpha_{t,A}^i} \quad \forall \omega \in \Omega \tag{10}$$

where $\alpha_A := \sum_{\omega \in \Omega} \alpha_\omega$. The mean value is utilized as an estimation of the posterior $\theta_{t,\omega}^i$, while the variance represents the uncertainty of the estimation. For the i^{th} grid at time-step t, we classify its occupancy state as D (dynamically occupied) when $\mathbb{E}[\theta_{t,D}^i]$ is above a threshold and as O (occupied either statically or dynamically) when $\mathbb{E}[\theta_{t,S}^i + \theta_{t,D}^i]$ is above a threshold.

C. Velocity with Particle Tracking

In addition to classifying the occupancy state of each grid, we estimate the velocity of each dynamic grid by incorporating a particle filter similarly as proposed in the literature [ref, steyer p.4]. A particle p is represented by a position $x_p \in \mathbb{R}^3$, a velocity $v_p \in \mathbb{R}^3$, and a weight $w_p \in \mathbb{R}^+$, so that the dynamic parameter of the i^{th} grid is represented as:

$$\alpha_{t,D}^i = \sum_{p \in \mathcal{P}_t^i} w_p \tag{11}$$

where $\mathcal{P}_t^i := \{p : x_p \in g^i\}$. Then, the velocity of the grid is estimated as:

$$v_t^i = \frac{1}{\alpha_{t,D}^i} \sum_{p \in \mathcal{P}_t^i} w_p v_p. \tag{12}$$

Note that this velocity information is useful in diverse applications such as path planning with collision avoidance.

IV. Prediction and Update of Map

The Dirichlet representation of the posteriors introduced in Sec. III is an approximation to the true posteriors. In this section, we present our prediction and update algorithm to keep the approximation close to the true posteriors based on intuitive heuristics.

A. Prediction Step ($\alpha \to \alpha'$)

We compute $\alpha_t'^i$ for the predicted posterior in (7) from the previous posterior with parameters α_{t-1}^i for each i^{th} grid. As movements of dynamic grids cause environmental changes, the main focus of the prediction step is reflecting the movements of particles that represents dynamic characteristics of each grid. We employ a constant velocity model with process noise for the motion of each particle. When new measurements arrive after dt seconds from the previous measurements, the states of a particle p are updated as

$$x_p = x_p + v_p dt + n_x \tag{13}$$

$$v_p = v_p + n_v \tag{14}$$

with process noises $n_x \sim \mathcal{N}(0, \sigma_x I)$ and $n_v \sim \mathcal{N}(0, \sigma_v I)$.

When the moved particles form a new set $\mathcal{P}_t^i$ for each i^{th} grid, we predict $\alpha_t'^i$ as below (the time subscript is dropped for notational simplicity).

$$\alpha'_S = \gamma^{dt} \alpha_S \tag{15}$$

$$\alpha'_D = \min(\sum_{p \in \mathcal{P}_t} w_p, \max(0, \gamma^{dt}(\alpha_F + \alpha_D) - \alpha'_S)) \tag{16}$$

$$\alpha'_F = \gamma^{dt}(\alpha_F + \alpha_D) - \alpha'_D \tag{17}$$

where γ is a decaying factor that increases the uncertainty of the previous estimation by decreasing the concentration parameters. Note that the variance (10) increases while the mean (9) remains the same when the parameters are decreased by the same ratio. Newly incoming particles form α'_D while it cannot pass over $\gamma^{dt}(\alpha_F + \alpha_D)$. As dynamic objects move into and out of free space, we impose the complementary relation by preserving the (decayed) summation of the free and dynamic parameters. We additionally penalize the dynamic parameter by decreasing its upper bound with α'_S to prevent particles falling into static area.

B. Update Step 1: Rebalancing ($\alpha' \to \alpha''$)

We aim to perform a kernel-based update similar to (1) to estimate the posterior (6) from the new measurements $\{X_t, Y_t\}$. Although the kernel inference in (1) benefits from its simple and recursive computation as well as its intuitive interpretation, there are two major challenges to apply it directly to dynamic environments. First, it does not differentiate old and new measurements so that new measurements hardly turn over accumulated disparities among the parameters. For instance, when a dynamic object occupies a grid that has been unoccupied for a long time, sum of new occupancy measurements near that grid is far smaller than the accumulated free parameter of the grid. Thus, the operation of simple addition cannot make the dynamic parameter larger than the free parameter.

In this step, we solve the issue by re-balancing the parameters based on new measurements. For each i^{th} grid, we first evaluate occupancy measurements for the two classes, 'free'

and 'occupied', using the kernel function as in (1):

$$\Delta\alpha_O = \sum_{(x,y)\in\{X_t,Y_t\}} k(x^i, x)y \tag{18}$$

$$\Delta\alpha_F = \sum_{(x,y)\in\{X_t,Y_t\}} k(x^i, x)(1-y). \tag{19}$$

where the free measurements are derived one for each measurement ray as the closest point from the query point as in [5]. We rebalance α' based on $\Delta\alpha$ and α' itself by computing rebalancing ratios among each of α'. However, we cannot trust the ratios when α' or $\Delta\alpha$ has small values with insufficient measurements. Thus, we set a credit function

$$C(\alpha) = \tanh(\alpha/\alpha_c) \tag{20}$$

to evaluate the credit of the parameters by

$$c = C(\sqrt{\Delta\alpha_A \alpha'_A}). \tag{21}$$

.

The rebalancing ratios are calculated as follows:

$$\lambda_{D\to F} = c\frac{\Delta\alpha_{F-O}}{\Delta\alpha_A}\frac{\alpha'_D}{\alpha'_A} \tag{22}$$

$$\lambda_{S\to FD} = c\frac{\Delta\alpha_{F-O}}{\Delta\alpha_A}\frac{\alpha'_{O-F}}{\alpha'_A} \tag{23}$$

$$\lambda_{F\to D} = c\frac{\Delta\alpha_{O-F}}{\Delta\alpha_A}\frac{\alpha'_{F-O}}{\alpha'_A} \tag{24}$$

where $\alpha_O := \alpha_S + \alpha_D$ and $\alpha_{\omega-\omega'} := \max(0, \alpha_\omega - \alpha_{\omega'})$. When free measurements dominate occupancy measurements, $\lambda_{D\to F}$ depresses erroneously located particles by reducing their total sum of weights, while $\lambda_{S\to FD}$ corrects the incorrectly assigned portion to α_S in (28). Here, we give the half of $\lambda_{S\to FD}$ to D as the state change from O to F indicates that a dynamic object has just moved out from the space. On the other hand, when occupancy measurements dominate free measurements in area predicted as free, it is an evidence of a dynamic object moved into that region. Thus, $\lambda_{F\to D}$ keeps the complementary relationship between F and D by transferring the parameters.

Then, the final rebalancing equations are as below:

$$\alpha''_F = \alpha'_F + \frac{\lambda_{S\to FD}}{2}\alpha'_S + \lambda_{D\to F}\alpha'_D - \lambda_{F\to D}\alpha'_F \tag{25}$$

$$\alpha''_S = \alpha'_S - \lambda_{S\to FD}\alpha'_S \tag{26}$$

$$\alpha''_D = \alpha'_D + \lambda_{F\to D}\alpha'_F + \frac{\lambda_{S\to FD}}{2}\alpha'_S - \lambda_{D\to F}\alpha'_D. \tag{27}$$

C. Update Step 2: Kernel Inference ($\alpha'' \to \alpha$)

The other challenge for applying kernel inference to dynamic environments is the combined measurement of static and dynamic states as a single state 'occupied'. This integrated measurement makes it hard to choose the true state that prompted the measurement and, thus, to add $\Delta\alpha_O$ to the corresponding state. Our solution is to split $\Delta\alpha_O$ into the classes S and D using a ratio $\beta \in [0, 1]$:

$$\alpha_S = \alpha''_S + \beta\Delta\alpha_O \tag{28}$$

$$\alpha_D = \alpha''_D + (1-\beta)\Delta\alpha_O \tag{29}$$

$$\alpha_F = \alpha''_F + \Delta\alpha_F \tag{30}$$

TABLE I: Hyperparameters for all experiments

Hyperparameter	Symbol	Value
Kernel length scale	l	0.5 m
Kernel scale	σ_0	0.1
Dirichlet prior	α_0	0.001
Decaying factor	γ	0.99
Credit scale	α_c	2.5

We choose β based on the re-balanced predicted parameters α'' as below:

$$\beta = (1 - c(\alpha''_O)) + c(\alpha''_O)\frac{\alpha''_S}{\alpha''_O}. \tag{31}$$

When α'' is small, i.e., there is not enough knowledge on the occupancy states, β is set close to 1 with low credit value. Otherwise, we set β close to the ratio of α'' to follow our re-balanced prediction.

We give large portion of $\Delta\alpha_O$ to S rather than D in little knowledge case to focus the limited number of particles to certainly dynamic area. If the true class was D, the wrongly assigned portion to S would be adjusted when the dynamic object leaves the grid through $\lambda_{S\to FD}$ term.

V. Experimental Results

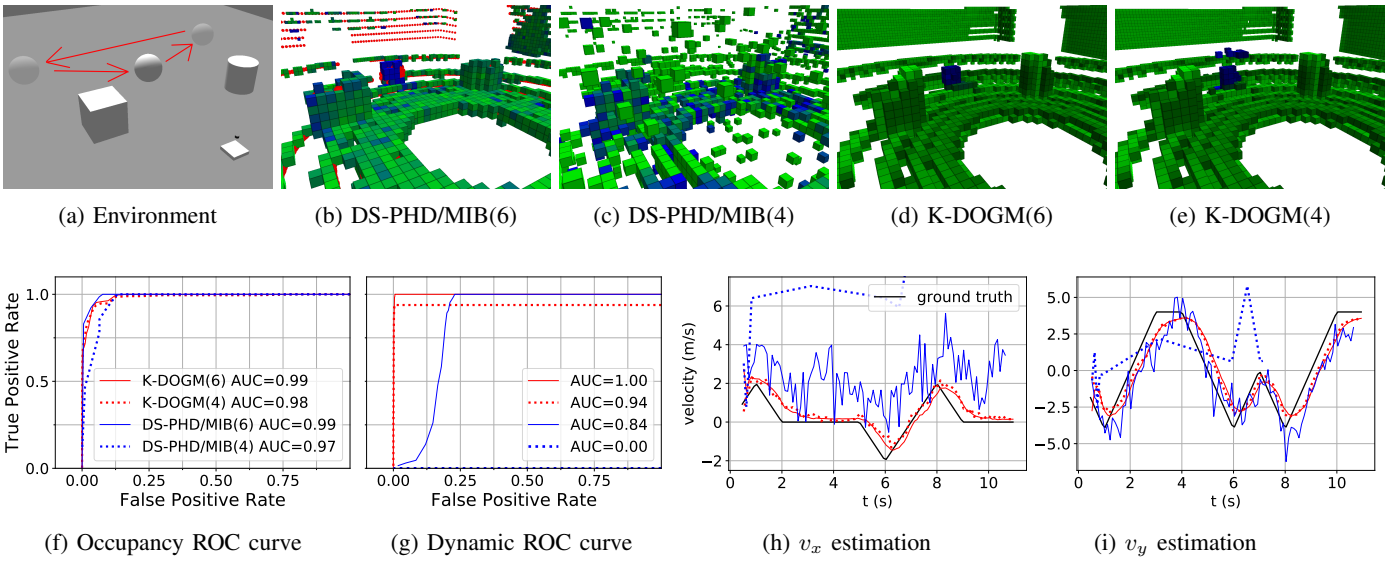
The proposed algorithm is evaluated through experiments in simple environment, complex environment, and urban environment simulated through *gazebo*. The experiments are processed in *ROS* (Robot Operating System) with a desktop with i7-7700K quad-core CPU and RTX 2060 Super GPU. We implemented the algorithm using CUDA parallel computing and was able to process 5Hz LiDAR data stream in real-time. We make use of the VLP 16 model of *velodyne_simulator* package to generate realistic LiDAR measurement data. Hyperparameters throughout the experiments are listed in Table I. The result is compared with the baseline method, DS-PHD/MIB filter [8]. For the easiness of comparison, we follow the same particle management algorithm with a fixed number of particles as in [8].

The evaluation consists of two parts: classification and velocity estimation. In classification evaluation, the state of each grid is determined by its internal parameters (i.e. α). It is worthwhile to note that the grid with high uncertainty such as one in occluded region is excluded in evaluation. Because K-DOGM and DS-PHD/MIB use different sets of parameters to represent the state of a grid, we use different exclusion rules. We filter out the i^{th} grid when the following condition is met.

$$\begin{cases} \Sigma(\alpha^i_F + \alpha^i_D + \alpha^i_S) < \zeta_0 & \text{for K-DOGM} \\ m^i_O + m^i_F < \zeta_1 & \text{for DS-PHD/MIB} \end{cases} \tag{32}$$

where m^i_ω is the evidence of the hypothesis ω (O for occupied and F for free). We empirically use $\zeta_0 = 0.5$ and $\zeta_1 = 0.1$ throughout the experiments. After the exclusion, a grid is classified as D if

$$\begin{cases} \mathbb{E}[\theta^i_{t,D}] > \zeta_2 & \text{for K-DOGM} \\ \|V_i\|^2 > \zeta_3 & \text{for DS-PHD/MIB} \end{cases} \tag{33}$$



(a) Environment (b) DS-PHD/MIB(6) (c) DS-PHD/MIB(4) (d) K-DOGM(6) (e) K-DOGM(4)

(f) Occupancy ROC curve (g) Dynamic ROC curve (h) v_x estimation (i) v_y estimation

Fig. 2: Experiments in simple environment. (a) describes the simulated *gazebo* environment. (b-e) represents the classification results of different algorithms with varying particle numbers while (b) includes LiDAR data as red dots. (f-g) show ROC curve of each classification. (h-i) plot velocity estimation results along with the ground truth. (6) and (4) indicate that the algorithm is evaluated with the particle numbers of 10^6 and 10^4, respectively.

where $\|V_i\|$ is the Euclidean norm of the mean velocity of the grid. The grid is correctly classified if its center is inside a dynamic object. A grid is classified as O if

$$\begin{cases} \mathbb{E}[\theta^i_{t,S} + \theta^i_{t,D}] > \zeta_4 & \text{for K-DOGM} \\ m^i_O + \frac{1}{2} m^i_\Omega > \zeta_5 & \text{for DS-PHD/MIB} \end{cases} \tag{34}$$

and correct when its center is inside an object. Note that a grid can be classified as D and O simultaneously. By adjusting the value of $\zeta_2, \ldots, \zeta_5$ in a wide range, the ROC (Receiver Operating Characteristic) curves are obtained. Then, AUC (Area Under the Curve) is computed which is a common criterion for the evaluation of classification algorithms.

In velocity evaluation, the mean velocity of each grid is estimated first with (12). Then, the velocity of each object can be estimated as well provided that the exact pose and geometry of each object is known.

$$v_t^{Obj} = \frac{\sum_{x^i \in Obj} v_t^i \, \mathbb{E}[\theta^i_{t,D}]}{\sum_{x^i \in Obj} \mathbb{E}[\theta^i_{t,D}]} \tag{35}$$

The velocity estimate of each object is compared to the ground truth value.

A. Simple Environment

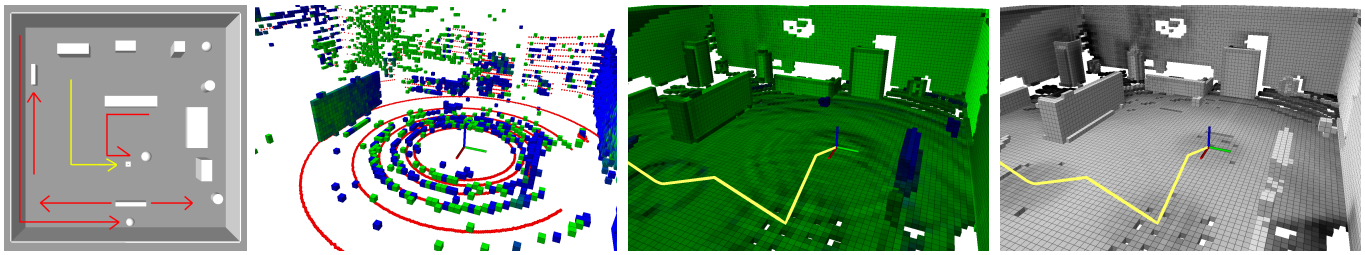
The simple environment is composed of two static objects, one dynamic object, and the sensor that stays at a single location. The dynamic object is moving only in the XY plane. The overall configuration of the environment is shown in Figure 2a. To see whether the algorithm is using the particles efficiently, we vary the number of particles and see the effect. The results are shown in Figure 2.

In the classification results (Figure 2b to 2e), blue block represents dynamic grid, and green block represents static grid. With a large number of particles ($\nu = 10^6$), the proposed algorithm shows more neat classification results and less noisy velocity estimates than the baseline algorithm, but both algorithms perform well. However, with a small number of particles ($\nu = 10^4$), DS-PHD/MIB totally loses the ability to estimate dynamic occupancy as shown in Figure 2g, 2h, and 2i. Meanwhile, there is little degradation of the performance for K-DOGM. The result shows that K-DOGM use particles efficiently and is capable of representing the map with much smaller number of particles.

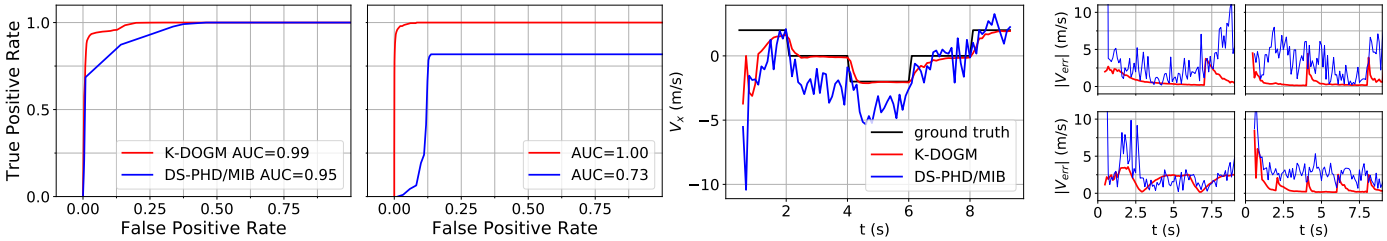
B. Complex Environment

The complex environment is composed of multiple static objects and four dynamic objects which provide a more challenging scenario compared to the simple environment. Static objects are arranged so that some of them partially hide the others. Dynamic objects not only move freely including in z-direction but also move with piece-wise continuous velocities. Furthermore, the sensor is moving around rather than fixed at the same location. We indicate the trajectory of the dynamic objects by the red arrows, and that of the sensor by the yellow arrow as in Figure 3a. Both algorithms are using $\nu = 10^6$ particles in this experiment.

As shown in Figure 3b and 3c, DS-PHD/MIB fails to separate dynamic objects from static objects while K-DOGM successfully does. This result may be occurred as DS-PHD/MIB represent both the static and dynamic objects with particles. In other words, K-DOGM utilizes the particle more efficiently. Moreover, K-DOGM provides an explicit expression of uncertainty which is visualized in Figure 3d. In Figure 3f, note that the true positive rate of DS-PHD/MIB does not converges to 1. This result can happen when the dynamic objects go through the previously vacant space with high evidence on free with zero evidence on occupancy. When a



(a) Environment (b) DS-PHD/MIB (c) K-DOGM (d) Uncertainty map



(e) Occupancy ROC curve (f) Dynamic ROC curve (g) v_x estimation (h) $\|V_{err}\|$ estimation for each obj

Fig. 3: Experiments in complex environment. (a) describes the simulated *gazebo* environment. (b-c) represents the classification results while (b) includes LiDAR data as red dots. (e-f) show ROC curve of each classification. (g-h) plot velocity estimation results.

grid belong to a dynamic object is occluded in occupancy measurement because of the volume of the object. the free evidence is reduced but still remains while dynamic evidence is zero. In this case, this grid is neither excluded from the evaluation nor classified as a dynamic grid.

The velocity evaluation results also clearly show that K-DOGM outperforms DS-PHD/MIB. Figure 3g shows the x-component of velocity of the dynamic object whose trajectory has C shape in the middle of Figure 3a. The figure shows that the velocity estimate of K-DOGM rapidly chases the ground-truth value even it changes abruptly. Figure 3h shows the norm of velocity error for each dynamic object. K-DOGM estimates velocity more consistently with less error than the baseline.

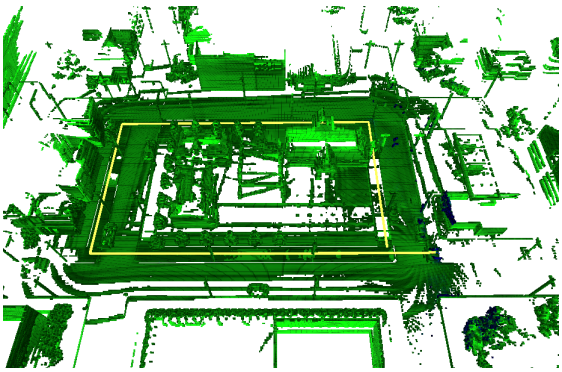


Fig. 4: Classification map in the simulated urban environment after the sensor moves along the periphery of a block.

C. Urban Environment

The simulated urban environment comprises of various objects with complex geometry, rather than a box or cylinder. More specifically, we utilize the various models provided in the *gazebo*, such as buildings, vehicles, trees, and so forth. In this experiment, we qualitatively evaluate our proposed algorithm. Figure 1c shows that K-DOGM can classify the dynamic objects well while capturing the detailed shape of the static environment. Furthermore, Figure 4 demonstrates that K-DOGM is capable of scale-up to large data while preserving the details of the map.

VI. Conclusion

This work proposed the 3-D dynamic occupancy grid mapping algorithm, K-DOGM, using kernel inference and particle tracking. We overcame the inherent challenges of applying kernel inference in dynamic environments through the 2-step estimation algorithm developed in a Bayesian framework with intuitive heuristics. The proposed algorithm have shown promising performance compared to the baseline with real-time processing capability.

While the algorithm considers spatial correlation through kernel inference on a query point with nearby measurements in the update step, the prediction and rebalancing steps are not considering their neighborhood. Further developing the algorithm to fully reflect the spatial correlation would be an interesting future work with expected performance improvements.

Acknowledgment

This research was supported by Unmanned Vehicles Core Technology Research and Development Program through the National Research Foundation of Korea(NRF), Unmanned Vehicle Advanced Research Center(UVARC) funded by the Ministry of Science and ICT, the Republic of Korea (2020M3C1C1A01082375)

REFERENCES

[1] S. T. O'Callaghan and F. T. Ramos, "Gaussian process occupancy maps," *The International Journal of Robotics Research*, vol. 31, no. 1, pp. 42–62, 2012.

[2] K. Doherty, J. Wang, and B. Englot, "Probabilistic map fusion for fast, incremental occupancy mapping with 3d hilbert maps," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1011–1018, IEEE, 2016.

[3] F. Ramos and L. Ott, "Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1717–1730, 2016.

[4] K. Doherty, J. Wang, and B. Englot, "Bayesian generalized kernel inference for occupancy map prediction," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3118–3124, IEEE, 2017.

[5] K. Doherty, T. Shan, J. Wang, and B. Englot, "Learning-aided 3-d occupancy mapping with bayesian generalized kernel inference," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 953–966, 2019.

[6] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière, "Bayesian occupancy filtering for multitarget tracking: an automotive application," *The International Journal of Robotics Research*, vol. 25, no. 1, pp. 19–30, 2006.

[7] M. Tay, K. Mekhnacha, M. Yguel, C. Coue, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessiere, "The bayesian occupation filter," in *Probabilistic Reasoning and Decision Making in Sensory-Motor Systems*, pp. 77–98, Springer, 2008.

[8] D. Nuss, S. Reuter, M. Thom, T. Yuan, G. Krehl, M. Maile, A. Gern, and K. Dietmayer, "A random finite set approach for dynamic occupancy grid maps with real-time application," *The International Journal of Robotics Research*, vol. 37, no. 8, pp. 841–866, 2018.

[9] R. Danescu, F. Oniga, and S. Nedevschi, "Modeling and tracking the driving environment with a particle-based occupancy grid," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1331–1342, 2011.

[10] G. Tanzmeister and D. Wollherr, "Evidential grid-based tracking and mapping," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1454–1467, 2016.

[11] S. Steyer, G. Tanzmeister, and D. Wollherr, "Grid-based environment estimation using evidential mapping and particle tracking," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 3, pp. 384–396, 2018.

[12] D. Hahnel, R. Triebel, W. Burgard, and S. Thrun, "Map building with mobile robots in dynamic environments," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 2, pp. 1557–1563, IEEE, 2003.

[13] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.