2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

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

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

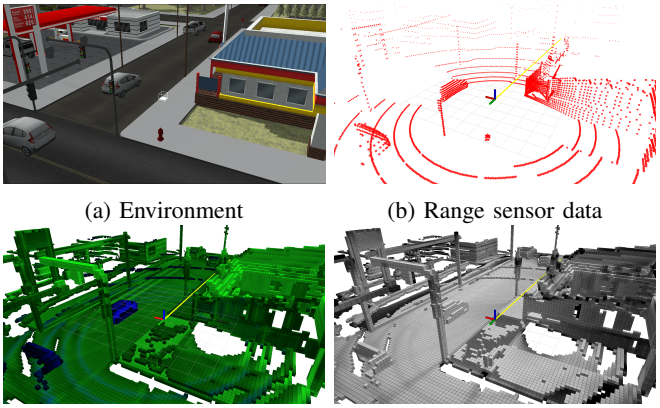
***Abstract*— Mapping three-dimensional (3-D) dynamic environments is essential for aerial robots but challenging to consider the increased dimensions in both space and time compared to 2-D static mapping. This paper presents a kernel-based 3-D dynamic occupancy mapping algorithm, K3DOM, that distinguishes between static and dynamic objects while estimating the velocities of dynamic cells via particle tracking. 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 mapping problem in a Bayesian framework and represent the map through Dirichlet distribution to update posteriors 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 map by estimating whether each discretized space, i.e., cell, 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 cell from online measurements stream. Related to this work, existing methods for static occupancy mapping focus mainly on considering spatial correlation among cells as sparse and noisy sensor measurements cause inconsistencies between environments and their occupancy maps. The 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]. Recently, [4] applied the Bayesian kernel inference [5] to 3-D mapping for efficient Bayesian updates of posteriors. Nonetheless, these methods are incapable of identifying moving objects and, thus, unsuitable for dynamic environments.

Further studies have been proposed to deal with dynamic environments. [6] introduced continuous dynamic occupancy mapping using Gaussian process regression, and [7] enabled faster estimation through stochastic variational inference. [8] also took a variational approach on Hilbert map to enable long-term dynamic occupancy mapping. Other mathematical formulation can be also found in [9], [10], [11], while they are shown not to be real-time capable. Nevertheless, [12] 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 in [13]. By the virtue of multi-hypothesis evidential representation, its following studies have shown promising results [14]. However, the aforementioned 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.



(a) Environment (b) Range sensor data

(c) Classification map (d) Uncertainty map

Fig. 1: Data flow and outputs of K3DOM in simulated urban environments. The algorithm classifies the two moving cars by representing dynamic cells as blue while static cells as green in (c). (d) indicates more uncertain estimation with darker color in gray scale.

In this paper, we present an efficient and robust 3-D dynamic occupancy mapping algorithm, called K3DOM. It distinguishes between static environments and moving objects, provide uncertainty estimates of the classification, and estimate the velocities of dynamic cells. 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 [4] 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 [4] to dynamic environments are twofold. First,

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)

dynamic occupancy can be inferred only through temporal correlation since range sensors such as LiDAR measure occupancy without dynamic states. Thus, we need to estimate whether the measurements are for dynamic or static objects. Second, it is hard to resolve the disparity between accumulated map and new measurement. In dynamic environments, such inconsistency happens frequently as dynamic objects move into vacant space that has not been occupied for a long time. In such case, the accumulated information overwhelms the new measurement so that the algorithm disregards the environmental change that the measurements tell.

II. Background - Bayesian Generalized Kernel Inference

The 3-D static occupancy mapping algorithm in [4] introduced Bayesian Generalized Kernel Inference (BGKI) by extending the discrete counting sensor model proposed in [15]. The counting sensor model represents the occupancy probability of each map cell through Bernoulli distribution and employs Beta distribution as its conjugate prior under conditionally independent measurement model. Then, the Bayesian update of the Bernoulli parameter is deduced to counting how often beams have ended in and passed through each cell. BGKI extends this discrete counting scheme to continuous space and imposes spatial correlation among occupancy predictions at nearby points.

BGKI employs the kernel inference method proposed in [5]. With the Bernoulli likelihood model and the 'smooth' assumption with its extended likelihood, the conjugate prior of the occupancy probability θ_* of a query point x_* is given by $Beta(\alpha_0, \beta_0)$ which shapes the occupancy probability θ_* and free probability $1-\theta_*$. For range sensor measurement $\mathcal{X} := \{X, Y\}$ which is a set of positions and their corresponding occupancy states (0 for 'free' and 1 for 'occupied'), the posterior of θ_* is updated to follow $Beta(\alpha, \beta)$ with

$$\alpha = \alpha_0 + \sum_{(x.y)\in\{X,Y\}} k(x_*, x)y \tag{1}$$

$$\beta = \beta_0 + \sum_{(x.y)\in\{X,Y\}} k(x_*, x)(1-y) \tag{2}$$

where the sparse kernel function from [16] is employed:

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

$d := ||x - x'||_2$, and 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 query points takes $\mathcal{O}(N \log M)$ time.

Note that BGKI can be applied to any query point in the continuous space $\mathbb{R}^3$. When the query point is the center of each cell, the neighborhood formed by the kernel function could let each cell reflect measurements beyond its boundary unlike the discrete counting sensor model. We adapt this scheme to consider spatial correlation among neighboring cells. Also, we deal with the distribution of occupancy probability parameters as in BGKI, instead of directly affecting the parameters as in the evidence-based algorithms [11], [14]. 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 Map Representation

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

A. Problem Formulation

In 3-D dynamic occupancy mapping problem, the environments are represented by N cells where the center of each i^{th} cell locates at $x^i \in \mathbb{R}^3$. The purpose of the problem is to estimate the occupancy state $\omega_t^i \in \Omega := \{F, S, D\}$ of each i^{th} cell at time-step t where F, S, and D represent that the cell is not occupied (free), statically occupied, and dynamically occupied, respectively. The state estimation at time-step t is based on accumulated range sensor measurements $\mathcal{X}_t := \{X_k, Y_k\}_{k=1}^t$ where X_k and Y_k represent locations of measurements and measured values at time-step k, respectively. Note that, since free space information is indirectly available from range sensors, single free measurement is derived for each measurement ray as the closest point on the ray from the query point as in [4].

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

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

This factorization can be dealt with in a 2-step procedure similarly as common Bayesian filters such as Extended Kalman filter and particle filter [17]. The last term can be further expanded as

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

This formulation predicts the new state ω_t^i from the previous posteriors $\{p(\omega_{t-1}^j \mid x^j, \mathcal{X}_{t-1})\}_{j=1}^N$ without new observation. Then, the first term in (4) updates the predicted belief to the posterior with the new measurements $\{X_t, Y_t\}$. Our algorithm is heuristically developed upon this 'predict-update' framework.

B. Occupancy with Dirichlet Distribution

We represent the occupancy through categorical distribution over Ω and model the priors and posteriors of its parameters for the states F, S, and D as Dirichlet distribution, which is a multivariate generalization of Beta distribution used in [4]:

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

where $\boldsymbol{\theta}_t^i = \{\theta_{t,\omega}^i\}_{\omega\in\Omega}$ with $\theta_{t,\omega}^i := p(\omega_t^i = \omega \mid x^i, \mathcal{X}_t)$, and $\boldsymbol{\alpha}_t^i := \{\alpha_{t,\omega}^i\}_{\omega\in\Omega}$ are the concentration parameters. Similarly, we model the predicted belief (5) as

$$\boldsymbol{\theta}_t'^i := p(\omega_t^i \mid x^i, \mathcal{X}_{t-1}) \sim Dir(\boldsymbol{\alpha}_t'^i). \tag{7}$$

$\boldsymbol{\alpha}_t^i$ is used in color coding for map visualization with (9):

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

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

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

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

where the subscript $(\,\cdot\,)_\Omega := \sum_{\omega\in\Omega}(\,\cdot\,)_\omega$. 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} cell 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 cell, we estimate the velocity of each dynamic cell by incorporating a particle filter. We adopt the particle management scheme in [11], but utilize particles only for dynamic cells instead of the whole map. 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} cell is represented as

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

where $\mathcal{P}_t^i := \{p : x_p \in i^{th}\,\text{cell}\}$. Then, the velocity of the cell 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 representations of the posteriors introduced in Sec. III approximate the true posteriors. In this section, we present our prediction and update algorithm to keep the form of the approximation based on intuitive heuristics.

A. Prediction Step ($\boldsymbol{\alpha} \to \boldsymbol{\alpha}'$)

We compute $\boldsymbol{\alpha}_t'^i$ for the predicted belief in (7) from the previous posterior with parameters $\boldsymbol{\alpha}_{t-1}^i$ for each i^{th} cell. As movements of dynamic cells cause environmental changes, the main focus of the prediction step is reflecting the movements of particles that represents dynamic characteristics of each cell. 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} cell, we predict $\boldsymbol{\alpha}_t'^i$ as below (the cell index and the time subscript are dropped for notational simplicity).

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

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

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

where γ is a decaying factor that increases the uncertainty of estimation by decreasing the total amount of accumulated concentration parameters. Note that the variance (10) increases while the mean (9) remains the same when the parameters are decreased by the same ratio. New 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 ($\boldsymbol{\alpha}' \to \boldsymbol{\alpha}''$)

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 (1) benefits from its simple and recursive computation as well as its intuitive interpretation, we can't directly apply it to dynamic environments as it does not differentiate old and new measurements. For instance, when a dynamic object occupies a cell that has been unoccupied for a long time, new occupancy measurements near that cell hardly turn over the accumulated free parameter of the cell. In this step, we solve the issue by rebalancing the parameters based on new measurements.

For each i^{th} cell, we first evaluate occupancy measurements for two classes, 'free' and 'occupied', using the kernel function as in (1):

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

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

Then, we compute rebalancing ratios among each of $\boldsymbol{\alpha}'$ based on $\Delta\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}'$ itself. However, when those ingredients have small values with insufficient measurements, their information is unreliable. Thus, we set a credit function

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

that evaluates the reliability of a parameter. The credit scale α_c is empirically chosen in the experiments. The joint credit of the ingredients are assessed with their geometric mean as

$$c = C(\sqrt{\Delta\alpha_\Omega \alpha'_\Omega}). \tag{21}$$

We calculate the rebalancing ratios as follows:

$$\lambda_{D\to F} = c\,\frac{(\Delta\alpha_{F-O})^+}{\Delta\alpha_\Omega}\frac{\alpha'_D}{\alpha'_\Omega} \tag{22}$$

$$\lambda_{S\to FD} = c\,\frac{(\Delta\alpha_{F-O})^+}{\Delta\alpha_\Omega}\frac{(\alpha'_{O-F})^+}{\alpha'_\Omega} \tag{23}$$

$$\lambda_{F\to D} = c\,\frac{(\Delta\alpha_{O-F})^+}{\Delta\alpha_\Omega}\frac{(\alpha'_{F-O})^+}{\alpha'_\Omega} \tag{24}$$

where $(\cdot)^+ := \max\{0, (\cdot)\}$, $(\cdot)_{\omega-\omega'} := (\cdot)_\omega - (\cdot)_{\omega'}$, and $(\cdot)_O := (\cdot)_S + (\cdot)_D$. Note that $((\cdot)_{\omega-\omega'})^+/(\cdot)_\Omega$ is non-negligible only when $(\cdot)_\omega$ dominates $(\cdot)'_\omega$. When free measurements dominate occupancy measurements, $\lambda_{D\to F}$ depresses erroneously located particles by reducing their total sum of weights, while $\lambda_{S\to FD}$ adjusts wrong portion of α_S. Here, we purposely 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 and locates nearby. 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. Finally, 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 ($\boldsymbol{\alpha}'' \to \boldsymbol{\alpha}$)

Another limitation of the kernel inference (1) in dynamic environments is the combined measurement of static and dynamic states under the single state 'occupied'. Hence, it is hard to choose the true state that prompted the measurement and, thus, to add $\Delta\alpha_O$ to the true state. Our solution is to distribute $\Delta\alpha_O$ into S and D using a ratio $\beta \in [0, 1]$:

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

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

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

We choose β based on the rebalanced $\boldsymbol{\alpha}''$ as below:

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

TABLE I: Hyperparameters for all experiments

Hyperparameter	Symbol	Value
Kernel length scale	l	0.5 m
Kernel scale	σ_0	0.1
Dirichlet prior	$\alpha_{0,F}, \alpha_{0,S}, \alpha_{0,D}$	0.001
Decaying factor	γ	0.99
Credit scale	α_c	2.5

For small α''_O with insufficient confidence on occupancy, β is set close to 1 with low credit value. Otherwise, we set β close to the ratio of $\boldsymbol{\alpha}''$ to follow our rebalanced prediction. We give large portion of $\Delta\alpha_O$ to S rather than D in the little confidence case to focus the limited number of particles to certainly dynamic area. If the true state was D, the wrongly assigned portion to S would be adjusted through $\lambda_{S\to FD}$ term when the dynamic object leaves the cell.

V. Experimental Results

The proposed algorithm is evaluated in various environments simulated through *gazebo* with comparison to the 3-D extended DS-PHD/MID filter [11]. For the easiness of comparison, we follow the same particle management algorithm with a fixed number of particles as in [11]. 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[1] using CUDA parallel computing and were able to process 5Hz LiDAR data stream in real-time. We employed the VLP-16 model of *velodyne_simulator* package to generate realistic LiDAR measurement data. Hyperparameters throughout the experiments are listed in Table I.

The evaluation consists of two parts: classification and velocity estimation. In classification evaluation, the state of each cell is determined by the concentration parameters (i.e. $\boldsymbol{\alpha}$). It is worthwhile to note that the cell with high uncertainty such as one in occluded region is excluded in evaluation. As K3DOM and DS-PHD/MIB use different sets of parameters to represent the state of a cell, we employ different exclusion rules. We filter out the i^{th} cell if

$$\begin{cases} \alpha^i_\Omega < \zeta_0 & \text{for K3DOM} \\ m^i_O + m^i_F < \zeta_1 & \text{for DS-PHD/MIB} \end{cases} \tag{32}$$

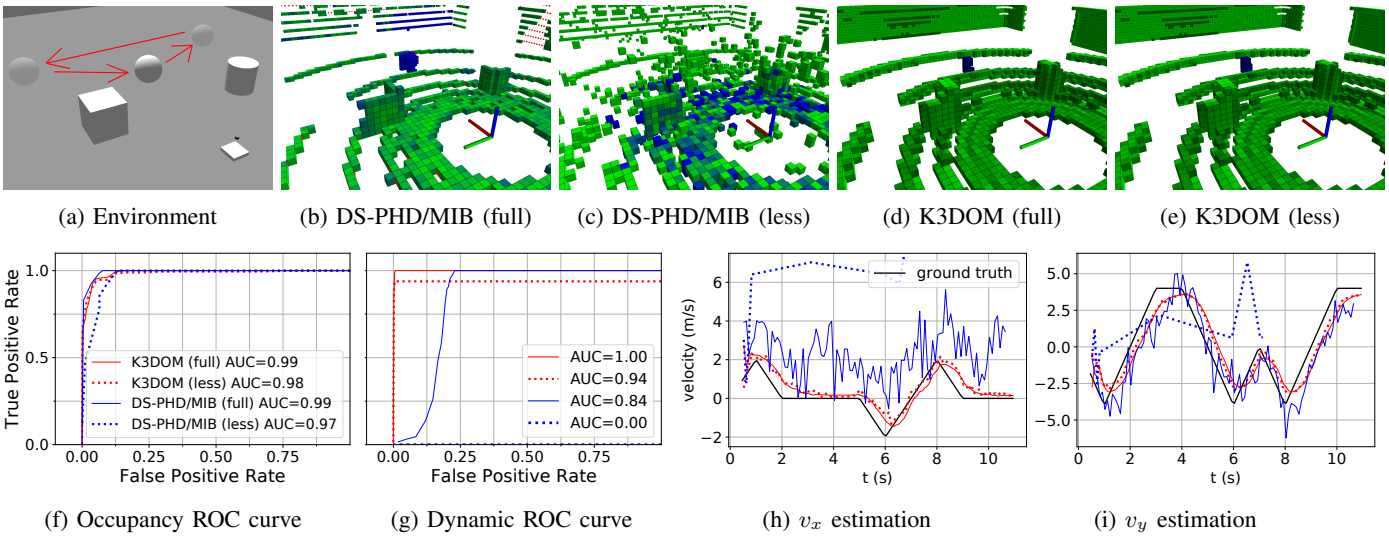
where m^i_ω is the evidence of the hypothesis ω (O for occupied and F for free). We empirically use $\zeta_0 = 0.5$ and $\zeta_1 = 0.1$ throughout the experiments. After the exclusion, a cell is classified as D if

$$\begin{cases} \mathbb{E}[\theta^i_{t,D}] > \zeta_2 & \text{for K3DOM} \\ \|V_i\|^2 > \zeta_3 & \text{for DS-PHD/MIB} \end{cases} \tag{33}$$

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

$$\begin{cases} \mathbb{E}[\theta^i_{t,S}] + \mathbb{E}[\theta^i_{t,D}] > \zeta_4 & \text{for K3DOM} \\ m^i_O + \frac{1}{2}m^i_\Omega > \zeta_5 & \text{for DS-PHD/MIB} \end{cases} \tag{34}$$

[1]The code is available online: https://github.com/youngjae-min/k3dom



(a) Environment (b) DS-PHD/MIB (full) (c) DS-PHD/MIB (less) (d) K3DOM (full) (e) K3DOM (less)

(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) represent the classification results of each algorithm with varying particle numbers while (b) includes LiDAR data as red dots. (f-g) show ROC curves of occupancy and dynamic classification. (h-i) plot velocity estimation results along with the ground truth.

and correct when its center is inside an object. Note that a cell 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 cell is estimated first with (12). Then, the velocity of each object is estimated as well given the exact pose and geometry of each object.

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

This estimate from each algorithm is compared to the ground truth value.

A. Simple Environment

As described in Figure 2a, a stationary sensor in the lower right corner observes two static objects and a dynamic object moving along the triangular path in the XY plane. We compare the proposed algorithm to the baseline while varying the number of particles to check the efficiency of particle usage. The classification results in Figure 2b-2e are visualized with the color code (8) where blue and green represent dynamic and static cells, respectively.

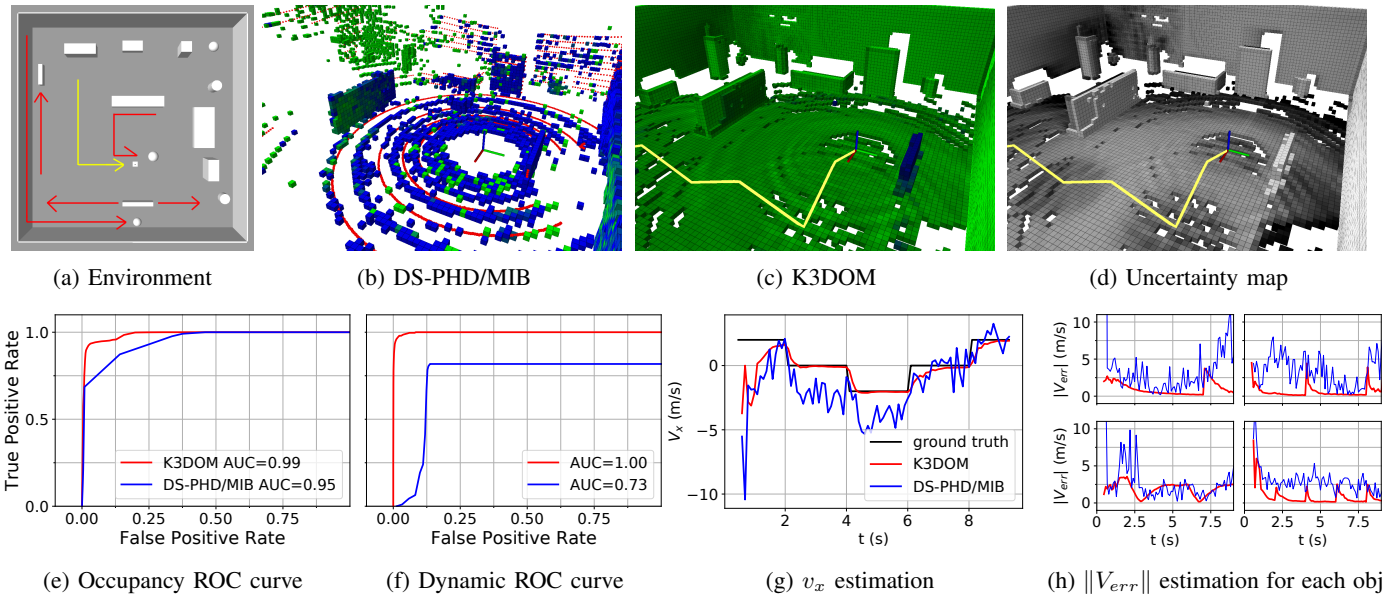
With 10^6 particles, denoted as 'full', both algorithms perform well while the proposed algorithm shows better performance over the baseline in dynamic classification with lower false positive rate as shown in Figure 2g. In terms of velocity estimation, K3DOM also provides more accurate and consistent results as shown in Figure 2h-2i. Their performance gaps get distinct with 10^4 particles, denoted as 'less'. DS-PHD/MIB totally loses the ability to estimate dynamic occupancy while there is little degradation of the performance for K3DOM. The result shows that K3DOM is capable of representing the map efficiently with much smaller number of particles.

Meanwhile, the occupancy classification results show small gaps in terms of AUC in Figure 2f whereas the baseline explicitly fails to map the environment with less particles in Figure 2c. This discrepancy occurs since the true positive rate of DS-PHD/MIB quickly increases along with the false positive rate when the classification threshold in (34) decreases. Also, numerous free cells are included in the evaluation so that just 5-10% false positive rate severely degrades the quality of the constructed map. Nevertheless, the proposed algorithm shows better classification results and more accurate and consistent velocity estimates than the baseline.

B. Complex Environment

The complex environment consists of multiple static objects and four dynamic objects so is more challenging compared to the simple environment. Static objects are arranged so that the sensor cannot observe all objects simultaneously. Dynamic objects move with piece-wise continuous velocities spanning the z-direction as well as the XY plane. 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 in Figure 3a. Both algorithms utilize 10^6 particles in this experiment.

As shown in Figure 3b-3c, K3DOM successfully separates dynamic objects from static objects unlike DS-PHD/MIB. This result may be occurred as DS-PHD/MIB represents both the static and dynamic objects with particles. In other words, K3DOM utilizes the particle more efficiently. Moreover, K3DOM provides an explicit expression of uncertainty which is visualized in Figure 3d. Note that the true positive rate of DS-PHD/MIB does not converges to 1 in Figure 3f. This result can happen when a dynamic object goes through



(a) Environment (b) DS-PHD/MIB (c) K3DOM (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) represent the classification results while (b) includes LiDAR data as red dots. (e-f) show ROC curves of each classification. (g) plots velocity estimation results of a moving object. (h) plots norm of velocity estimation errors for each moving object

the previously vacant space with high evidence on 'F' and zero evidence on 'O'. When a cell belong to a dynamic object is occluded in measurements because of obstacles, the free evidence is reduced but still remains nonzero while occupancy evidence is zero. In such case, the cell is neither excluded from the evaluation nor classified as a dynamic cell.

The velocity evaluation results also clearly show that K3DOM 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 K3DOM rapidly chases the ground-truth value even it changes abruptly. As shown in Figure 3h, K3DOM also estimates velocity more consistently with less error than the baseline.

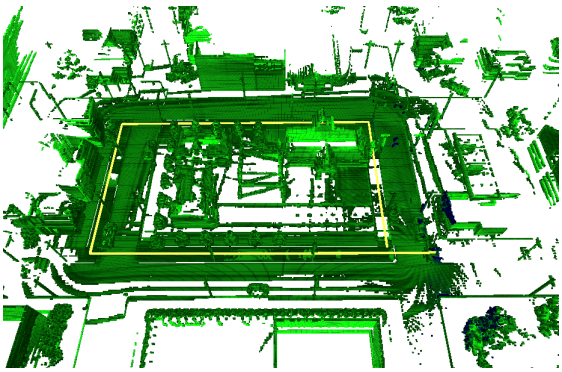


Fig. 4: Constructed map with classification for the simulated urban environment after the sensor moves along the periphery of a block.

C. Urban Environment

The simulated urban environment consists of various realistic objects with complex geometry, rather than a simple box or cylinder. We employed 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 K3DOM can classify the dynamic objects well while capturing the detailed shape of the static environment. Furthermore, Figure 4 demonstrates that K3DOM 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 mapping algorithm, K3DOM, by adapting kernel inference for dynamic environments with particle tracking. We overcame the inherent challenges of applying kernel inference in dynamic environments through the 2-step estimation algorithm developed 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 function 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 in part by the Agency for Defense Development under contract UC190028RD and in part 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, 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.

[5] W. Vega-Brown, M. Doniec, and N. Roy, "Nonparametric bayesian inference on multivariate exponential families," in *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2*, pp. 2546–2554, 2014.

[6] S. T. O'Callaghan and F. T. Ramos, "Gaussian process occupancy maps for dynamic environments," in *Experimental Robotics*, pp. 791–805, Springer, 2016.

[7] R. Senanayake, S. O'Callaghan, and F. Ramos, "Learning highly dynamic environments with stochastic variational inference," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2532–2539, IEEE, 2017.

[8] R. Senanayake and F. Ramos, "Bayesian hilbert maps for dynamic continuous occupancy mapping," in *Conference on Robot Learning*, pp. 458–471, PMLR, 2017.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] A. Melkumyan and F. Ramos, "A sparse covariance function for exact gaussian process inference in large datasets," in *Proceedings of the 21st international jont conference on Artifical intelligence*, pp. 1936–1942, 2009.

[17] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.