2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Convolutional Bayesian Kernel Inference for 3D Semantic Mapping

Joey Wilson, Yuewei Fu, Arthur Zhang, Jingyu Song
Andrew Capodieci, Paramsothy Jayakumar, Kira Barton, and Maani Ghaffari

***Abstract*— Robotic perception is currently at a cross-roads between modern methods, which operate in an efficient latent space, and classical methods, which are mathematically founded and provide interpretable, trustworthy results. In this paper, we introduce a Convolutional Bayesian Kernel Inference (ConvBKI) layer which learns to perform explicit Bayesian inference within a depthwise separable convolution layer to maximize efficency while maintaining reliability simultaneously. We apply our layer to the task of real-time 3D semantic mapping, where we learn semantic-geometric probability distributions for LiDAR sensor information and incorporate semantic predictions into a global map. We evaluate our network against state-of-the-art semantic mapping algorithms on the KITTI data set, demonstrating improved latency with comparable semantic label inference results.**

I. Introduction

Robust world models are essential for safe and reliable autonomous robots. Within a world model, an autonomous robot can embed a high level of scene understanding through multiple modalities of information, such as semantic or motion labels. One common world model is a map, where a geometric framework models the world in a manner interpretable to both robots and humans, encouraging reliability and trust.

Although some works have proposed to discard maps in lieu of an end-to-end deep learning autonomous robot framework, a world model is still critical for safety and trustworthiness. Through a world model, robot failures can be safely diagnosed post-mortem and understood by humans due to the shared human-robot understanding.

Semantic mapping is a framework for robotic mapping which extends the geometric map to include scene ontology. Semantic labels incorporate a higher level of scene understanding by labeling the world with semantics, such as people and chairs. This information can be beneficial for robotic behavior planning.

Recently, works within mapping have explored learning-based neural implicit representations, which move beyond the structured geometric representations of earlier, probabilistic hand-crafted algorithms. Despite efficient latent operations, there is still a clear trade-off. While maps encoded in a latent space are argued to be more efficient, trainable, and faster, they lose the reliability and trustworthy behavior of hand-crafted mapping methods. In contrast, hand-crafted mapping methods are mathematically derived and can be understood with quantifiable uncertainty, which is necessary for predicting robot failures.

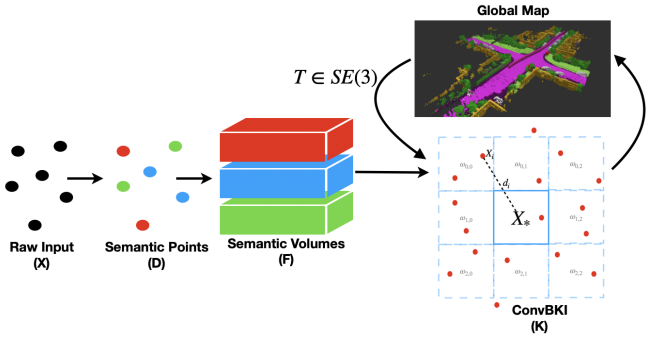


Fig. 1: Structural diagram of ConvBKI. 3D points are assigned semantic labels from off-the shelf semantic segmentation networks, and grouped into voxels by summing coinciding points. The constructed semantic volumes are convolved with a depthwise filter to perform a real-time Bayesian update on a semantic 3D map.

In this paper, we attempt to combine the advantages of deep learning-based approaches with the safe nature and predictability of hand-crafted approaches for 3D semantic mapping. Concretely, we demonstrate that a probabilistic Bayesian inference semantic mapping approach [1] can be written as a differentiable depthwise convolution [2] layer, thus enabling an end-to-end mapping framework with the efficiency, speed, and trainable nature of deep learning frameworks, while maintaining quantifiable uncertainty and reliability of a hand-crafted approach. Our main contributions are as follows.

i. Create a real-time 3D semantic mapping neural network layer, which finds middle-ground between classical robotic mapping and modern deep learning.
ii. Propose novel differentiable kernels for Bayesian semantic mapping, and demonstrate improved performance through optimization.
iii. Open source all software for future development at https://github.com/UMich-CURLY/NeuralBKI.

II. Literature Review

In this section, we review 3D semantic mapping and the trade-offs between learned and hand-crafted approaches.

J. Wilson, Y. Fu, A. Zhang, J. Song, K. Barton, and M. Ghaffari are with the University of Michigan, Ann Arbor, MI 48109, USA. {wilsoniv,ywfu,arthurzh}@umich.edu, {jingyuso,bartonkl,maanigj}@umich.edu
A. Capodieci is with Neya Systems Division, Applied Research Associates, Warrendale, PA 15086, USA. acapodieci@neyarobotics.com
P. Jayakumar is with the US Army DEVCOM Ground Vehicle Systems Center, Warren, MI 48397, USA. paramsothy.jayakumar.civ@army.mil

A. Learned vs. Hand-Crafted Mapping

Historically, most mapping methods were hand-crafted and mathematically derived. Early semantic mapping algorithms semantically labeled images, then projected to 3D and directly updated matching voxels through a voting scheme or Bayesian update [3]–[7]. Later semantic mapping algorithms applied further optimization through Conditional Random Fields (CRF), which encourages consistency between adjacent voxels [8]–[10]. Separately, continuous mapping algorithms estimate occupancy through a continuous non-parametric function such as Gaussian processes (GPs) [11], [12]. However, these methods suffer from a high computation load, rendering them impractical for on-board robotics. For example, GPs have a cubic computational cost with respect to the number of data points and semantic classes [13]. Other works have also explored semantic mapping with alternative data-efficient representations such as surfels [14], truncated signed distance functions [15], and meshes [16], [17].

Many modern approaches to mapping take advantage of neural networks to learn an efficient, implicit approximation of the world in a lower dimensional latent space [18]. Some approaches include applying recurrent neural networks [19], [20] or spatio-temporal convolution networks [21], [22] to model spatio-temporal dynamics. Other recent works have explored approximating continuous geometry implicitly with Neural Radiance Fields (NeRF) [23], [24] or occupancy networks [25]–[27], in order to negate the expensive memory of voxels.

While learning-based approaches have succeeded in minimizing memory or accelerating inference, they still encounter significant challenges. By implicitly approximating functions, there is no notion of when a network will fail, as provided by variance or the ability to diagnose an error. On the other side of the spectrum, mathematical hand-crafted approaches provide reliability and trustworthiness at the cost of efficiency.

B. Bayesian Kernel Inference

Semantic Bayesian Kernel Inference (S-BKI) [1] is a 3D continuous semantic mapping framework which builds on the work of [28] and [29]. BKI is an efficient approximation of GPs, requiring $\mathcal{O}(\log N)$ operations and $\mathcal{O}(N)$ memory instead of $\mathcal{O}(N^3)$, where N is the number points. In contexts such as mapping, there may be hundreds of thousands of points, rendering GPs impractical.

For supervised learning problems, our goal is to identify the relationship $p(y|x_*, \mathcal{D})$ given a sequence of N independent observations $\mathcal{D} = \{(x_1, y_1), ..., (x_N, y_N)\}$, where x_* is a query point. y represents observation values drawn from set Y corresponding to input values x drawn from set X. In 3D semantic mapping, the likelihood represents a distribution of semantic labels Y over geometric positions X.

Vega-Brown et al. [28] introduce a model and constraints which generalize local kernel estimation to Bayesian inference for supervised learning. They show that the maximum entropy distribution g satisfying $D_{KL}(g||f) \geq \rho(x_*, x)$ has the form $g(y) \propto f(y)^{k(x_*,x)}$. In this case, $\rho(\cdot,\cdot) \rightarrow \mathbb{R}^+$ [**Maani:** $\rho : X \times X \rightarrow \mathbb{R}^+$] is some function which bounds information divergence between the likelihood distribution $f(y_i) = p(y_i|\theta_i)$ and the extended likelihood distribution $g(y_i) = p(y_i|\theta_*, x_i, x_*)$. Functions k and ρ have an equivalence relationship, where each is uniquely determined by the other. The only requirements are that:

$$k(x, x) = 1 \forall x \quad \text{and} \quad k(x, x') \in [0, 1] \forall x, x', \tag{1}$$

where k is the kernel function. This formulation is especially useful for likelihoods $p(y|\theta)$ chosen from the exponential family, as the likelihood raised to the power of $k(x_*, x)$ is still within the exponential family.

Doherty et al. [29] then apply the BKI kernel model to the task of occupancy mapping. In occupancy mapping, occupied points are measured by a 3D sensor such as LiDAR, and free space samples can be approximated through ray tracing. Measurement $x_i \in \mathbb{R}^3$ then represents a 3D position with corresponding observation $y_i^c \in \{0, 1\}$, either indicating free space ($y_i^0 = 1$) or occupied space ($y_i^1 = 1$). In this case, $c \in \mathcal{C}$ is a binary variable indicating whether the point is occupied ($c = 1$) or free ($c = 0$). Adopting a prior distribution Beta(α_0^0, α_0^1) over θ_0 yields a closed-form update equation at each time step t, such that:

$$\alpha_{*,t}^c = \alpha_{*,t-1}^c + \sum_{i=1}^{N_t} k(x_*, x_i) y_i^c, \tag{2}$$

where * is the query voxel with centroid x_* and parameters θ_*. The equation provides a closed-form method for updating the belief that voxel * is occupied or free, given observed measurements and samples of free space. The kernel depends on distance of observed points to the centroid of each voxel, providing more weight to close points. Gan et al. [1] show that the same approach can be applied to semantic labels by adopting a Categorical likelihood and placing prior distribution Dir(C, α_0) over θ_*. Semantic labels y_i are obtained as estimations from state-of-the-art neural networks. This model is also calculated using Eq. (2), where the variable c is no longer binary, but represents one of C labels. y_i is again a Categorical distribution, representing the probability of each semantic category. From the Dirichlet distribution concentration parameters α_*, the expectation and variance of voxel $*$ is calculated as:

$$\eta_*^c = \sum_{j=1}^{C} \alpha_*^j, \quad \mathbb{E}[\alpha_*^c] = \frac{\alpha_*^c}{\eta_*^c}, \quad \mathbb{V}[\alpha_*^c] = \frac{\frac{\alpha_*^c}{\eta_*^c}(1 - \frac{\alpha_*^c}{\eta_*^c})}{1 + \eta_*^c}. \tag{3}$$

Although Semantic BKI has succeeded in 3D mapping, it is still limited in a few key ways. Firstly, the kernel is hand-crafted, and kernel parameters must be manually tuned. As a result, a single spherical kernel is shared between all semantic classes. Second, the update operation has a slow inference rate, as the kernel [**Maani: evaluation**] requires a nearest neighbor operation.

III. Method

We propose a novel neural network layer, Convolutional Bayesian Kernel Inference (ConvBKI), which is intended

to accelerate and optimize S-BKI. Compared to S-BKI, ConvBKI *learns* a unique kernel for each semantic class, and generalizes to 3D ellipsoids instead of restricting distributions to spheres. We demonstrate how to train the layer and incorporate it into an end-to-end deep neural network for updating semantic maps in static environments.

A. Convolutional BKI

We build a faster, trainable version of Semantic BKI based on the key observation that Eq. (2) can be rewritten as a depthwise convolution [2]. We find that the kernel parameters are differentiable with respect to a map loss function and are therefore learnable. Learning the kernel parameters enables more expressive geometric-semantic distributions and improved semantic mapping performance.

The update operation in Eq. (2) performs a weighted sum of semantic probabilities over the local neighborhood of voxel centroid x_*. This operation can be directly interpreted in continuous space with radius neighborhood operations such as in PointNet++ [30], DGCNN [31], or KPConv [32]. However, we found that in practice, these operations are much too slow to compute for hundreds of thousands of camera or LiDAR points due to an expensive k-Nearest Neighbor operation. Instead, we perform a discretized update, where the geometric position of each local point is rounded to the position of the map voxel it falls in. Approximation through downsampling is already performed in Semantic BKI [1], and is a common step in real-time mapping literature [17], [33].

Given the prior local map of dimension $\mathbb{R}^{D_C \times D_X \times D_Y \times D_Z}$ and a labeled input point cloud, we first group points within corresponding voxels. D represents the dimension of the semantic channel (C) and Euclidean (X, Y, Z) axes. Let $I(*, i)$ be an indicator function representing whether point x_i lies within voxel $*$. From the semantic predictions over each point cloud, we compute input semantic volume $\mathbf{F} \in \mathbb{R}^{D_C \times D_X \times D_Y \times D_Z}$ as follows, where input $\mathbf{F}_*$ is the sum of all point-wise semantic predictions contained in voxel $*$.

$$\mathbf{F}_*^c = \sum_{i=1}^{N} I(*, i) y_i^c. \tag{4}$$

For each voxel, the Bayesian update can be calculated as the sum of the prior semantic map and a depthwise convolution over input $\mathbf{F}$. Let h, i, j be the discretized coordinates of voxel $*$ within $\mathbf{F}$, and k, l, m be indices within discretized kernel $\mathbf{K} \in \mathbb{R}^{D_C \times f \times f \times f}$ where f is the filter size. Then, we can write the update for a single semantic channel of voxel $*$ as

$$\alpha_{*,t}^c = \alpha_{*,t-1}^c + \sum_{k,l,m} \mathbf{K}_{k,l,m}^c \mathbf{F}_{h+k,i+l,j+m}^c, \tag{5}$$

where indices $k, l, m \in [-\frac{f-1}{2}, \frac{f-1}{2}]$. Note that this is the equation for a zero-padded depthwise convolution, where dense 3D convolution is performed at each voxel in the feature map, with a unique kernel $\mathbf{K}^c$ for semantic category c. As a result, this operation can be accelerated by GPUs and optimized through gradient descent.

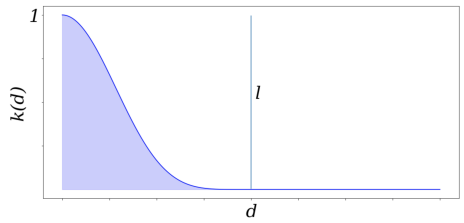


Fig. 2: Sparse Kernel Function. $k(d)$ has a maximum value of 1 at $d = 0$, and decays to 0 by $d = l$. Applied to semantic mapping, points proximal to the voxel centroid have more influence over the semantic label of the voxel.

Following [1] and [29], we use a sparse kernel [34] as our kernel function since the sparse kernel fulfills the requirements listed in Eq. (1). Additionally, the sparse kernel is differentiable so that a partial derivative of the loss function with respect to the kernel parameters can be calculated. The sparse kernel is shown in Eq. (6), where the parameters are kernel length l, and signal variance σ_0. Note that for Eq. (1) to remain valid, σ_0 must be 1, leaving only one tune-able parameter for the kernel function. For two points x and x', let $d := \|x - x'\|$. The sparse kernel is calculated as

$$k(d) = \begin{cases} \sigma_0[\frac{1}{3}(2 + \cos(2\pi\frac{d}{l})(1 - \frac{d}{l}) + \frac{1}{2\pi}\sin(2\pi\frac{d}{l}))], & \text{if } d < l \\ 0, & \text{otherwise} \end{cases}. \tag{6}$$

Effectively, kernel $\mathbf{K}$ is a weight matrix where each weight represents a semantic and spatial likelihood of correlated points. For example, if a point has semantic class road, then points nearby along the X or Y axes are also likely to have semantic class road, and would have a high weight. In contrast, a point labeled as pole would have more influence over points nearby vertically rather than horizontally.

While it is possible to learn an individual weight for each position and semantic category in filter $\mathbf{K}$, we found that restricting the number of parameters through a kernel function increases the ability of the network to learn generalizable semantic-geometric distributions quickly. Therefore, we learn a sparse kernel $k^c(\cdot)$ for each semantic category, and assign kernel values to $\mathbf{K}$ at each filter index, where distance depends on the resolution Δr of the voxel map. For a filter of dimension f and resolution Δr, the kernel weights $\mathbf{K}_{k,l,m}^c$ at filter indices k, l, m are calculated by evaluating kernel function k^c at the offset of position k, l, m from the centroid as follows.

$$\mathbf{K}_{k,l,m}^c = k^c(\|\Delta r \cdot (\frac{f-1}{2} - \begin{bmatrix} k \\ l \\ m \end{bmatrix})\|_2) \tag{7}$$

A plot of the sparse kernel function is included in Fig. 2 for reference. To accommodate complex geometric structures of real objects, we also propose a compound kernel [35, Ch. 4] computed as the product of a kernel over the horizontal

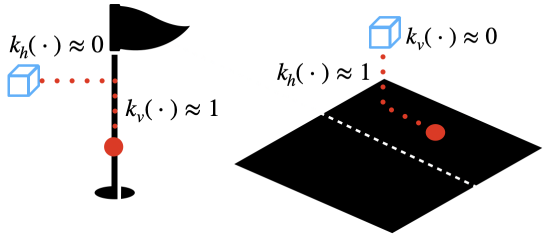


Fig. 3: Illustration of compound kernel motivation. ConvBKI learns a distribution to geometrically associate points with voxels. Whereas a point (red) labeled pole suggests a vertically adjacent voxel (blue) may also be a pole, it does not imply the same for a horizontally adjacent voxel. Likewise, a point labeled as road suggests horizontally adjacent voxels are also road but not vertically located voxels. Hence, a compound kernel enables ConvBKI to learn more expressive semantic-geometric distributions.

plane (k_h) and vertical axis (k_v) as

$$f(\begin{bmatrix} x \\ y \\ z \end{bmatrix}, \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}) = k_h(\| \begin{bmatrix} x - x' \\ y - y' \end{bmatrix} \|) k_v(\| \begin{bmatrix} z - z' \end{bmatrix} \|). \quad (8)$$

Intuitively, ConvBKI treats the output of a semantic segmentation neural network as sensor input, and learns a geometric probability distribution over each semantic class. Semantic classes have different shapes, where classes such as poles are more vertical and classes such as road have influence horizontally. The motivation behind a compound kernel for each semantic class is visualized in Fig. 3.

B. Global Mapping

Next, we apply our ConvBKI layer to the task of global mapping. In global mapping, sensor input is used to construct a full map of the environment, maintaining all past information.

At initialization, the map consists of an empty set of voxels. Concentration parameters of new voxels in the local region of the ego vehicle are assigned to prior, which is a small non-zero value for each semantic channel. At each time step, the input to our network is a global pose $T_t \in \text{SE}(3)$, and 3D data $\mathcal{X}_t$ in the form of a point cloud, stereo image, or both. From our prior global map G_{t-1}, we query position of T_t to identify the nearest voxel $*$ to the ego position and the local set of voxels with the same shape as $\mathbf{F}$. The local voxels serve as a prior local map L_{t-1} for the Bayesian update.

Once we have obtained prior local map L_{t-1}, a semantic segmentation network predicts labels $\mathcal{Y}_t$ for 3D points $\mathcal{X}_t$. Next, we apply Eq. (4) to calculate the input to our ConvBKI layer, aligning points $\mathcal{D}_t$ with local map L_{t-1}. Finally, we apply the 3D depthwise convolution from Eq. (5) to update local map L_{t-1} and obtain updated states L_t. The updated voxels from L_t are transferred back to CPU and replace their prior states in G_{t-1} to form G_t.

To accelerate computation and maximize efficiency for the global mapping operation, we make a couple of design choices. First, the global map G is stored on CPU memory and the local update is performed on GPU due to restricted GPU memory. For efficient retrieval of the local map, the global map is stored in a matrix where each row contains a key and value. The key is the voxel discretized indices, and the value is the semantic concentration parameters. The local map can be obtained in real-time by batch querying all voxels within local boundaries. We also accelerate runtime by applying garbage collection, where voxels which have not been updated recently (10 frames) are removed from memory to reduce the search space.

C. Training

We train the kernel functions separately from the respective mapping algorithms for memory efficiency and speed. For static data, applying ConvBKI over each time step individually is equivalent to applying ConvBKI once over all points since the operation is merely a weighted sum. Therefore, when training, we load the past $\mathcal{T}$ point clouds with predicted semantic labels $\mathcal{D}_{t-\mathcal{T}:t}$ and transform all points to the current frame T_t. All semantically labeled 3D points are then used to create input encoding $\mathbf{F}_t$ through Eq. (4), so that only one convolution is performed instead of a convolution at each time step.

ConvBKI learns a probabilistic distribution of semantic segmentation labels over geometrically neighboring predictions. Therefore, ConvBKI must be trained on noisy semantic segmentation predictions similar to the test test. Since semantic segmentation networks achieve higher performance on data they have been trained on, ConvBKI must be trained on a held out set, such as a validation set. Empirically, training on the validation set instead of training set results in a nearly 4% improvement in mean Intersection over Union (mIoU) on the test set of Semantic KITTI [36].

IV. Results

We perform ablation studies on hyper-parameters of ConvBKI, then compare performance with previous 3D semantic mapping baselines. Lastly, we visualize the semantic-geometric distributions learned by the ConvBKI layer. For each set of results, we compare ConvBKI with a single kernel shared between all semantic classes (ConvBKI Single), ConvBKI with one kernel for each semantic category (ConvBKI Per Class), and ConvBKI with a compound kernel for each semantic category (ConvBKI Compound).

We compare against two versions of S-BKI to enable direct comparison. S-BKI with 0.2 m resolution and discretization is a direct comparison to our work, equivalent to ConvBKI Single without optimization at a kernel length of 0.3 m. We also compare against the reported results of S-BKI from [1], which runs without discretization at a voxel resolution of 0.1 meters and with tuned thresholding. We refer to the S-BKI baselines as S-BKI (0.2m) and S-BKI (fine) where fine indicates a 0.1 m resolution compared to our 0.2 m resolution without discretization. S-BKI reports a latency of 2 Hz with downsampling and 0.6 Hz without. In contrast, our network runs at a quicker inference rate of *37* Hz (27 ms) to perform the Bayesian update, and 13.2 Hz (76 ms) to query the map.

We train ConvBKI with the Adam optimizer [37] at a learning rate of 0.007 for one epoch using the weighted negative log likelihood loss. We initialize the kernel length parameter to $l = 0.5\,\text{m}$ and train ConvBKI with the last

TABLE I: Ablation study of voxel resolution on Semantic KITTI sequence 8 for compound ConvBKI with filter size $f = 5$.

Resolution	mIoU (%)	Latency (ms)	Mem. (GB)
N/A (Input)	54.6	n/a	n/a
0.4 m	58.2	**8.5**	**2.4**
0.2 m	**59.3**	11.1	2.7
0.1 m	59.0	30.1	5.0

TABLE II: Ablation study of filter size on Semantic KITTI sequence 8 for compound ConvBKI with resolution 0.2 m.

Filter Size	mIoU (%)	Latency (ms)
N/A (Input)	54.6	n/a
$f = 3$	59.0	**9.5**
$f = 5$	59.3	11.1
$f = 7$	59.5	13.5
$f = 9$	**59.6**	17.6

$\mathcal{T} = 10$ frames, as we found 10 frames to optimally balance performance and training time.

A. Ablation Studies

We perform a series of ablation studies over filter size and voxel resolution of ConvBKI Compound on Semantic KITTI [36] sequence 8. Sequence 8 is part of the validation set and therefore has not been previously seen by the semantic segmentation network during training. We train and test on a voxel grid with bounds of [-20, -20, -2.6] to [20, 20, 0.6] m along the (X, Y, Z) axes, where points outside of the voxel grid are discarded and not measured in the results. Average latency of the ConvBKI layer is measured on an NVIDIA RTX 3090 GPU over 100 repetitions, with standard deviation < 0.4ms.

First, we study the effect of voxel resolution on the inference time and mIoU of ConvBKI. We compare ConvBKI with resolutions 0.1, 0.2 and 0.4 m, and a constant filter size $f = 5$ for all models. Table I indicates that a finer resolution can increase performance, however the segmentation difference between 0.2 and 0.1 m resolution is marginal at the cost of greater memory and slower inference. For real-time driving applications, this suggests that 0.2 m resolution may be a strong middle ground. Note that the optimal resolution will vary between applications.

Next, we study the effect of the filter size on inference time and performance. While a larger filter size increases the receptive field of the kernel and potentially improves the predictive capability as a result, filter size also cubically increases computation cost. Therefore, identifying a balance between filter size and computational efficiency is important for real-time application. We study filters of size $f = 3$, 5, 7, and 9 at a resolution of 0.2 m. Table II demonstrates that filter sizes can improve segmentation accuracy, however quickly increase run-time. In practice a filter size of 5 or 7 may be optimal, as a filter size of 9 offers little improvement with a large increase in computational cost.

B. KITTI Dataset

Following [1], we evaluate on the KITTI dataset [42] with semantically labeled images from [38] as there exist semantic mapping benchmarks for comparison. We follow the same process as [1], where depth is estimated from ELAS [43], pose is estimated from ORB-SLAM [44], and semantic labels are estimated from the deep network dilated CNN [39]. We compare against a CRF-based semantic mapping system [40], BGKOctoMap-CRF [1], [29], S-BKI [1], and S-CSM [1], which all have previously established baselines. Images are projected to 3D and updated by ConvBKI with bounds [-40, -40, -5.0] to [40, 40, 5.0] m, a resolution of 0.2 m, and a filter size of $f = 5$. Semantic segmentation performance is calculated for all image points within 40 m of the ego vehicle.

TABLE III: Semantic results on KITTI Odometry sequence 15 [38].

Method	Building	Road	Vege.	Sidewalk	Car	Sign	Fence	Pole	Average
Segmentation [39]	92.1	93.9	90.7	81.9	94.6	19.8	78.9	49.3	75.1
Yang et al. [40]	**95.6**	90.4	**92.8**	70.0	94.4	0.1	**84.5**	49.5	72.2
BGKOctoMap-CRF [29]	94.7	93.8	90.2	81.1	92.9	0.0	78.0	49.7	72.5
S-CSM [1]	94.4	95.4	90.7	84.5	95.0	22.2	79.3	51.6	76.6
S-BKI (fine)	94.6	95.4	90.4	84.2	95.1	**27.1**	79.3	51.3	77.2
S-BKI (0.2m)	92.6	94.7	90.9	84.5	95.1	21.9	80.0	52.0	76.5
ConvBKI Single	92.7	94.8	90.9	84.7	95.1	22.1	80.2	52.1	76.6
ConvBKI Per Class	94.0	95.5	91.0	87.0	95.1	22.8	81.8	52.9	77.5
ConvBKI Compound	94.0	**95.6**	91.0	**87.2**	**95.1**	22.8	81.9	**54.3**	**77.7**

Table III details the performance of the semantic segmentation input, each baseline, and each variation of ConvBKI. We find that the optimized ConvBKI Single performs slightly better than its direct comparison S-BKI (0.2m). Likewise, more expressive kernels increase performance as ConvBKI Compound has a higher mIoU than ConvBKI Per Class, which has a higher mIoU than ConvBKI Single, as expected. ConvBKI Compound with 0.2 m resolution and discretization can also improve upon the mIoU and latency of S-BKI (fine), which has a finer 0.1 m resolution without discretization. The improvement of ConvBKI Compound is due to optimization, a more expressive kernel, and hardware acceleration on GPU.

C. Semantic KITTI

We perform quantitative analysis on the Semantic KITTI [36] data set. We train a ConvBKI filter with bounds [-40, -40, -2.6] to [40, 40, 2.6] m, resolution 0.2 m, and filter size 5 following the results of the ablation studies. We compare again against S-CSM, and S-BKI [1] with Darknet53-kNN [41] as semantic segmentation input. For evaluation we increase the bounds to [-50, -50, -2.6] to [50, 50, 2.6] m and assign points outside the map to the semantic segmentation network predictions, since only local points within the boundaries are updated in the map.

Table IV details the results of ConvBKI trained on the validation set of Semantic KITTI, compared to the baselines and input semantic segmentation network on both the validation and test set. Similar to Table III, on the validation set, ConvBKI Single achieves a higher mIoU than un-optimized S-BKI (0.2m) and has a lower mIoU than more expressive ConvBKI Per Class, which has a lower mIoU than ConvBKI Compound. ConvBKI Compound achieves a higher mIoU than S-BKI (0.2m) on both the validation and test set; however has a lower mIoU than S-BKI (fine) on the test set. The discrepancy is likely due to the combination of a difference in resolution, variation in the test and validation

TABLE IV: Semantic results on Semantic KITTI [36] validation and test set.

Data Split	Method	Car	Bicycle	Motorcycle	Truck	Other Veh.	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other Gr.	Building	Fence	Vegetation	Trunk	Terrain	Pole	Sign	Average
Val	Segmentation [41]	91.0	25.0	47.1	40.7	25.5	45.2	62.9	0.0	93.8	46.5	81.9	**0.2**	85.8	54.2	84.2	52.9	72.7	53.2	40.0	52.8
	S-BKI (0.2m)	92.6	30.3	55.3	43.1	25.0	51.9	69.9	0.0	93.6	46.8	81.9	0.1	87.9	57.5	86.0	59.8	74.0	60.0	43.2	55.7
	ConvBKI Single	92.0	29.8	57.4	44.4	25.2	53.1	72.1	0.0	93.1	45.8	80.9	0.1	88.2	57.8	86.1	61.2	74.0	59.7	44.4	56.1
	ConvBKI Per Class	92.6	34.5	59.2	34.6	39.4	58.6	73.5	0.0	93.0	47.2	80.9	0.1	88.4	58.3	86.4	61.7	74.2	58.4	**47.4**	57.3
	ConvBKI Compound	**94.0**	**37.5**	**60.0**	33.3	**40.5**	**59.4**	**74.4**	0.0	93.3	49.0	81.2	0.1	88.5	59.5	86.8	62.2	75.0	59.9	46.5	**58.0**
	S-BKI (fine)	93.5	33.5	57.3	**44.5**	27.2	52.9	72.1	0.0	**94.4**	**49.6**	**84.0**	0.0	**88.7**	**59.6**	**86.9**	**62.5**	**75.3**	**63.6**	45.1	57.4
Test	Segmentation [41]	82.4	26.0	34.6	21.6	18.3	6.7	2.7	0.5	91.8	65.0	75.1	27.7	87.4	58.6	80.5	55.1	64.8	47.9	55.9	47.5
	S-BKI (0.2m)	**84.0**	28.5	39.9	25.2	19.7	7.9	3.3	0.0	92.3	**67.5**	76.5	28.5	89.1	61.5	82.3	61.6	66.5	55.3	64.4	50.2
	ConvBKI Compound	83.8	**32.2**	**43.8**	**29.8**	**23.2**	8.3	3.1	0.0	91.4	62.6	75.2	27.5	89.1	61.6	81.6	62.5	65.2	53.9	63.0	50.4
	S-BKI (fine)	83.8	30.6	43.0	26.0	19.6	**8.5**	**3.4**	0.0	**92.6**	65.3	**77.4**	**30.1**	**89.7**	**63.7**	**83.4**	**64.3**	**67.4**	**58.6**	**67.1**	**51.3**

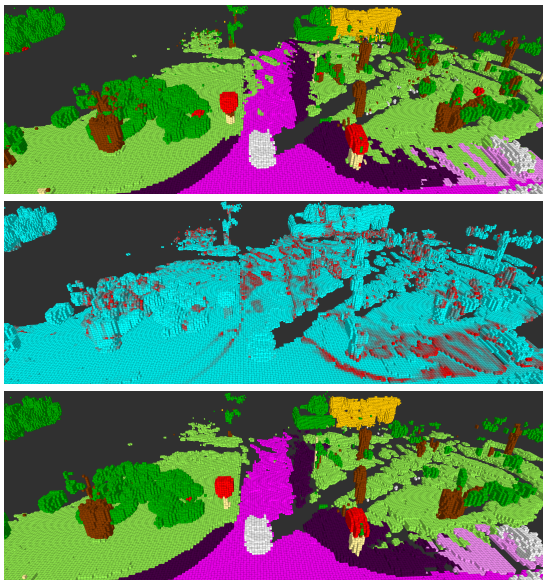
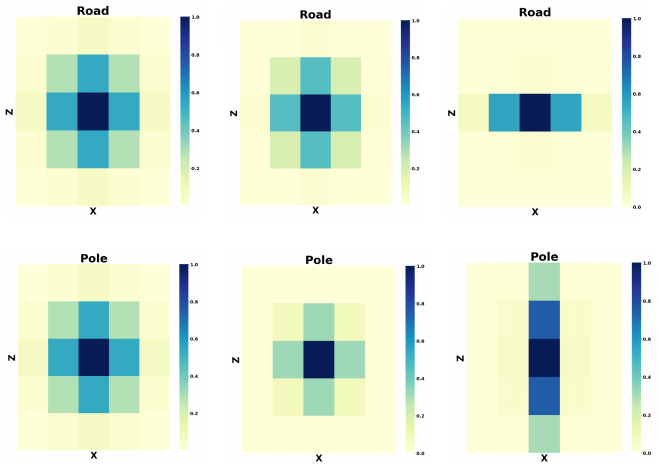


Fig. 4: Example map produced by ConvBKI Compound on the validation set of Semantic KITTI. The expected semantic map is shown in the top image, and the variance is shown in the middle, where red indicates high variance and blue indicates low variance. Removing voxels with high variance or uncertainty (e.g., $\mathcal{V}[\alpha_*^c] > 0.01$) improves the quality of the robotic map.



(a) Single (b) Per Class (c) Compound

Fig. 5: Illustration of kernels learned by ConvBKI on the road and pole semantic classes, plotted at $\Delta Y = 0$. Adding degrees of freedom increases expressivity by allowing the kernel to learn class-specific geometry.

set, and threshold tuning of S-BKI [1].

Overall, ConvBKI Compound achieves a higher mIoU than direct comparison S-BKI (0.2m) on all data sets due to optimization and a more expressive kernel. While S-BKI (fine) at a finer 0.1 m resolution without discretization achieves higher performance on the test set of Semantic KITTI, ConvBKI Compound achieves higher mIoU on the other two data sets with lower latency. For a voxel grid with bounds [-40, -40, -2.6] to [40, 40, 2.6] m, resolution 0.2 m, and filter size 5, ConvBKI updates the map at 37 Hz and queries local voxels from the global map at 13.2 Hz. In contrast, S-BKI (fine) reports an inference rate of 0.6 Hz.

D. Qualitative Results

Lastly, we present qualitative results illustrating the distributions learned by the ConvBKI layer, and the generated global map. A video of online mapping can be found in the supplementary material.

We include an example map in Fig. 4 of the Semantic KITTI validation set produced by ConvBKI Compound. The top image demonstrates the expected semantic label produced by the network. As can be seen, there is still noise present, especially around the road. Removing voxels with high variance calculated by Eq. (3) yields the bottom image, which is improved qualitatively.

Fig. 5 demonstrates the kernels learned by variations of the ConvBKI layer for single, per class, and compound kernels. Each variation of ConvBKI improves potential semantic-geometric expressiveness. ConvBKI Single learns a spherical semantic-geometric distribution shared between all classes. However, semantic classes do not share the same geometry in the real world. ConvBKI Per Class adds the capability to learn a unique distribution for each semantic category, but is still limited by spherical geometry. ConvBKI Compound learns a 3D ellipsoid which can be more expressive for classes such as pole or road.

V. Conclusion

In this paper, we introduced a differentiable 3D semantic mapping algorithm which combines reliability and trustworthiness of classical probabilistic mapping algorithms with the efficiency and differentiability of modern neural networks. We demonstrated that our network can achieve improved results compared to previous 3D mapping approaches, with real-time inference rates. For future work we intend to investigate the ability of ConvBKI to extend to other data sets and real world mobile robots, propagation of dynamic objects within the BKI framework [45], and other methods to accelerate mapping.

REFERENCES

[1] L. Gan, R. Zhang, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Bayesian Spatial Kernel Smoothing for Scalable Dense Semantic Mapping," *IEEE Robot. Autom. Letter.*, vol. 5, no. 2, pp. 790–797, 2020.

[2] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *ArXiv*, vol. abs/1704.04861, 2017.

[3] J. Stückler, N. Biresev, and S. Behnke, "Semantic mapping using object-class segmentation of RGB-D images," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2012, pp. 3005–3010.

[4] H. He and B. Upcroft, "Nonparametric semantic segmentation for 3D street scenes," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2013, pp. 3697–3703.

[5] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2017, pp. 4628–4635.

[6] S. Sengupta, E. Greveson, A. Shahrokni, and P. H. S. Torr, "Urban 3D semantic modelling using stereo vision," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2013, pp. 580–585.

[7] A. Cramariuc, F. Tschopp, N. Alatur, S. Benz, T. Falck, M. Brühlmeier, B. Hahn, J. Nieto, and R. Siegwart, "Semsegmap – 3d segment-based semantic localization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2021, pp. 1183–1190.

[8] S. Sengupta and P. Sturgess, "Semantic octree: Unifying recognition, reconstruction and representation via an octree constrained higher order MRF," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2015, pp. 1874–1879.

[9] Z. Zhao and X. Chen, "Building 3D semantic maps for mobile robots using RGB-D camera," *Intell. Service Robot.*, vol. 9, 10 2016.

[10] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg, "Joint Semantic Segmentation and 3D Reconstruction from Monocular Video," in *Proc. European Conf. Comput. Vis.*, 2014, pp. 703–718.

[11] J. Wang and B. Englot, "Fast, accurate gaussian process occupancy maps via test-data octrees and nested Bayesian fusion," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2016, pp. 1003–1010.

[12] S. T. O'Callaghan and F. T. Ramos, "Gaussian process occupancy maps," *Int. J. Robot. Res.*, vol. 31, no. 1, pp. 42–62, 2012.

[13] M. G. Jadidi, L. Gan, S. A. Parkison, J. Li, and R. M. Eustice, "Gaussian Processes Semantic Map Representation," *ArXiv*, vol. abs/1707.01532, 2017.

[14] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss, "SuMa++: Efficient LiDAR-based Semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2019, pp. 4530–4537.

[15] L. Schmid, J. Delmerico, J. L. Schönberger, J. Nieto, M. Pollefeys, R. Siegwart, and C. Cadena, "Panoptic Multi-TSDFs: a Flexible Representation for Online Multi-resolution Volumetric Mapping and Long-term Dynamic Scene Consistency," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2022, pp. 8018–8024.

[16] M. Herb, T. Weiherer, N. Navab, and F. Tombari, "Lightweight Semantic Mesh Mapping for Autonomous Vehicles," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2021, pp. 6732–3738.

[17] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2020, pp. 1689–1696.

[18] V. Cartillier, Z. Ren, N. Jain, S. Lee, I. Essa, and D. Batra, "Semantic MapNet: Building Allocentric SemanticMaps and Representations from Egocentric Views," in *Proc. AAAI Nat. Conf. Artif. Intell.*, February 2021.

[19] L. Sun, Z. Yan, A. Zaganidis, C. Zhao, and T. Duckett, "Recurrent-OctoMap: Learning State-Based Map Refinement for Long-Term Semantic Mapping with 3-D-Lidar Data," *IEEE Robot. Autom. Letter.*, vol. 3, no. 4, pp. 3749–3756, 2018.

[20] Y. Xiang and D. Fox, "DA-RNN: Semantic Mapping with Data Associated Recurrent Neural Networks," in *Robotics. Sci. Sys.*, vol. 13, 2017.

[21] P. Wu, S. Chen, and D. N. Metaxas, "MotionNet: Joint Perception and Motion Prediction for Autonomous Driving Based on Bird's Eye View Maps," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 11 382–11 392.

[22] J. Wilson, J. Song, Y. Fu, A. Zhang, A. Capodieci, P. Jayakumar, K. Barton, and M. Ghaffari, "MotionSC: Data Set and Network for Real-Time Semantic Mapping in Dynamic Environments," *IEEE Robot. Autom. Letter.*, vol. 7, no. 3, pp. 8439–8446, 2022.

[23] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, "Block-NeRF: Scalable Large Scene Neural View Synthesis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, June 2022, pp. 8248–8258.

[24] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," in *Proc. European Conf. Comput. Vis.*, 2020, pp. 405–421.

[25] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy Networks: Learning 3D Reconstruction in Function Space," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, June 2019.

[26] S. Lionar, L. Schmid, C. Cadena, R. Siegwart, and A. Cramariuc, "NeuralBlox: Real-Time Neural Representation Fusion for Robust Volumetric Mapping," in *Proc. IEEE Int. Conf. 3D Vis.*, 2021, pp. 1279–1289.

[27] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional Occupancy Networks," in *Proc. European Conf. Comput. Vis.*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., 2020, pp. 523–540.

[28] W. R. Vega-Brown, M. Doniec, and N. G. Roy, "Nonparametric Bayesian inference on multivariate exponential families," in *Proc. Advances Neural Inform. Process. Syst. Conf.*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27, 2014.

[29] K. Doherty, T. Shan, J. Wang, and B. Englot, "Learning-Aided 3-D Occupancy Mapping with Bayesian Generalized Kernel Inference," *IEEE Trans. Robot.*, vol. 35, no. 4, pp. 953–966, 2019.

[30] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," in *Proc. Advances Neural Inform. Process. Syst. Conf.*, 2017, pp. 1–10.

[31] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic Graph CNN for Learning on Point Clouds," *IEEE Trans. Graph.*, vol. 38, no. 5, oct 2019.

[32] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and Deformable Convolution for Point Clouds," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 6410–6419.

[33] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2017, pp. 1366–1373.

[34] A. Melkumyan and F. Ramos, "A Sparse Covariance Function for Exact Gaussian Process Inference in Large Datasets," in *Proc. Int. Joint Conf. Artif. Intell.*, 2009, p. 1936–1942.

[35] C. Rasmussen and C. Williams, *Gaussian processes for machine learning*. MIT press, 2006, vol. 1.

[36] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.

[37] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proc. Int. Conf. Learning Representations*, 2015.

[38] S. Sengupta, E. Greveson, A. Shahrokni, and P. H. S. Torr, "Urban 3D semantic modelling using stereo vision," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2013, pp. 580–585.

[39] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions," in *Proc. Int. Conf. Learning Representations*, 2016.

[40] S. Yang, Y. Huang, and S. Scherer, "Semantic 3D occupancy mapping through efficient high order CRFs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 09 2017, pp. 590–597.

[41] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2019, pp. 4213–4220.

[42] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 3354–3361.

[43] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Proc. Asian Conf. Comput. Vis.*, 2011, pp. 25–38.

[44] R. Mur-Artal, J. Montiel, and J. Tardos, "ORB-SLAM: a versatile and accurate monocular slam system," *IEEE Trans. Robot.*, vol. 31, pp. 1147 – 1163, 10 2015.

[45] A. Unnikrishnan, J. Wilson, L. Gan, A. Capodieci, P. Jayakumar, K. Barton, and M. Ghaffari, "Dynamic semantic occupancy mapping using 3D scene flow and closed-form Bayesian inference," *IEEE Access*, vol. 10, pp. 97 954–97 970, 2022.