Multitask Learning for Scalable and Dense Multilayer Bayesian Map Inference

Lu Gan, Youngji Kim, Jessy W. Grizzle, Jeffrey M. Walls, Ayoung Kim, Ryan M. Eustice, and Maani Ghaffari

***Abstract*—This article presents a novel and flexible multitask multilayer Bayesian mapping framework with readily extendable attribute layers. The proposed framework goes beyond modern metric-semantic maps to provide even richer environmental information for robots in a single mapping formalism while exploiting intralayer and interlayer correlations. It removes the need for a robot to access and process information from many separate maps when performing a complex task, advancing the way robots interact with their environments. To this end, we design a multitask deep neural network with attention mechanisms as our front-end to provide heterogeneous observations for multiple map layers simultaneously. Our back-end runs a scalable closed-form Bayesian inference with only logarithmic time complexity. We apply the framework to build a dense robotic map including metric-semantic occupancy and traversability layers. Traversability ground truth labels are automatically generated from exteroceptive sensory data in a self-supervised manner. We present extensive experimental results on publicly available datasets and data collected by a 3D bipedal robot platform and show reliable mapping performance in different environments. Finally, we also discuss how the current framework can be extended to incorporate more information such as friction, signal strength, temperature, and physical quantity concentration using Gaussian map layers. The software for reproducing the presented results or running on customized data is made publicly available.**

***Index Terms*—Bayesian inference, continuous mapping, multitask learning, robot sensing systems, semantic scene understanding, traversability estimation.**

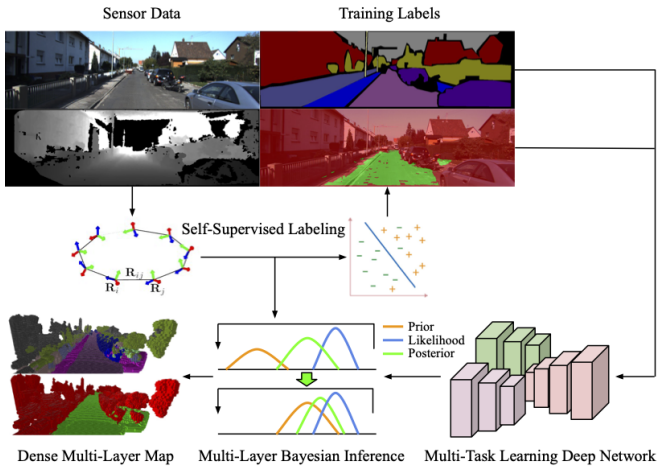


Fig. 1. Overview of our proposed multitask multilayer Bayesian mapping framework. We first employ exteroceptive sensory data and sensor trajectory to generate training data for map attributes other than semantics (traversability specifically) in a self-supervised manner. We then train a unified segmentation deep neural network for monocular image via MTL using available data. Predictions of the MTL network are processed through our multilayer Bayesian inference sequentially and efficiently to build dense maps. Inter-layer correlations are also leveraged to improve the map posterior. Our framework is designed to be easily extended to include additional layers.

I. Introduction

ROBOTIC mapping is the process of inferring a model of the environment from noisy measurements and is an essential task in the pursuit of robotic autonomy [1]. Over the recent decades of highly active research on robotic mapping, maps have used different representations, included different sensing modalities, and been applied to a variety of tasks such as localization [2], as a reference for navigation [3], and autonomous exploration [4]. Traditionally, maps built for robot localization contain geometric information of the environment. For example, occupancy grid maps employ voxel grids as the map representation and probabilistically assign a binary label to each grid cell to denote whether it is occupied [5, 6, 7]. The geometric structure of the scene is also encoded in dense maps where Truncated Signed Distance Fields (TSDFs) are used to represent a surface implicitly [8, 9]. For better visualization [10] and camera tracking [11], 3D maps are also textured with appearance (e.g., color) information.

Although classical robotic maps have reached a level of maturity for localization purposes, most practical robotic applications require more than just geometric information, e.g., high-level path planning and task planning. Semantic mapping timely extends the map representation to include semantic knowledge in addition to geometry and appearance, and becomes a highly active research area [12, 13, 14, 15, 16, 17, 18, 19]. In recent years, object instances, as another form of semantic knowledge, are also incorporated into robotic maps [20, 21, 22].

With the inclusion of scene understanding, semantic maps enable a much greater range of tasks for robots. For instance, domestic robots can fetch tableware from the kitchen, and a

Toyota Research Institute provided funds to support this work. Funding for M. Ghaffari was in part provided by NSF Award No. 2118818. Funding for J. Grizzle was in part provided by NSF Award No. 2118818. Y. Kim was supported by the International Research & Development Program of the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (Grant number: 2019K1A3A1A12069741). *(Lu Gan and Youngji Kim contributed equally to this work.) (Corresponding author: Lu Gan.)*

L. Gan, J. Grizzle, R. Eustice, and M. Ghaffari are with the Robotics Institute, University of Michigan, Ann Arbor, MI 48109, USA. {ganlu, grizzle, eustice, maanigj}@umich.edu.

Y. Kim is with the Department of Civil and Environmental Engineering, KAIST, Daejeon, S. Korea. youngjikim@kaist.ac.kr.

A. Kim is with the Department of Mechanical Engineering, SNU, Seoul, S. Korea. ayoungk@snu.ac.kr.

J. Walls is with Woven Planet Holdings, Inc. jeff.walls@woven-planet.global.

legged robot can walk on firm pavement instead of unstable vegetation. However, even with semantics, maps can still be limited for higher level planning. Planning systems may desire other environmental attributes for better decision making, such as the traversability of the ground surface, the affordance [23, Ch. 8] of surrounding objects, the outdoor temperature, or the concentration of some quantity in the air. These attributes form a high-level understanding of the environment, which is regarded as one of the key requirements of the *robust-perception age* in robotics [24].

At present, each of these attributes is often estimated independently and represented in a separate map. To perform a complex task involving multiple attributes, a robot needs to access and process information across different maps [25]. This not only complicates planning, but ignores the correlation among different sources of information. In this paper, we argue that instead of using an independent map for a specific task, information can be stored in one shared map with several correlated layers where each layer contains one attribute. We propose a multitask multilayer Bayesian mapping framework which

1) provides rich information such as semantics and traversability and is extendable to include more layers;
2) runs online using an arbitrary number of sensors (e.g., cameras and LiDARs);
3) unifies and simplifies the access to different map attributes for communication efficiency;
4) leverages inter-layer correlations to improve the map posterior distribution.

This work focuses on applying the proposed mapping framework to build a second *traversability* layer on top of the first *semantic* layer developed in our previous work [19]. Traversability, by definition in robotics, means "the capability of a ground robot to reside over a terrain region under an admissible state wherein it is capable of entering given its current state" [26]. We choose traversability as the second layer for two reasons. First, a traversability map is essential for a robot to safely and efficiently navigate within its environment and plan local motion policies with improved stability. Secondly, traversability is highly correlated with the semantic properties of an environment. For example, sky is non-traversable to a ground robot; road is the opposite; mud can either be traversable or untraversable.

An overview of the proposed framework is shown in Fig. 1. We first employ exteroceptive sensor data and robot trajectory to generate ground truth labels for image traversability segmentation in a self-supervised manner. We then design and train a multitask deep neural network as our front-end to provide multiple observations simultaneously. Our back-end runs a closed-form Bayesian inference for dense multilayer mapping where semantic information is leveraged in traversability estimation. The multitask learning (MTL) front-end not only achieves systematic efficiency, but also improves prediction performance over the Single-Task Learning (STL) equivalent. The multilayer Bayesian inference back-end is scalable and provides map uncertainty.

This work has the following contributions:

1) We propose a novel and extendable multitask multilayer Bayesian mapping framework to provide rich environmental information in a single mapping formalism.
2) We develop a multitask deep neural network for scene semantic and traversability segmentation.
3) We present a self-supervised approach to automatically generating training data for traversability segmentation.
4) We formulate a scalable and dense multilayer mapping algorithm via closed-form Bayesian inference that leverages inter-layer correlations.
5) We provide an open-source implementation of the proposed method and present extensive experiments using real and simulated data. We also discuss the reproducibility and extendability of the developed mapping framework.

The remainder of this paper is organized as follows. A review of the related work on multilayer robotic mapping, self-supervised learning for traversability estimation and deep MTL is given in Section II. Section III describes the design of our multitask deep network, and how we generate labeled traversability data for training without requiring manual annotation. Section IV formulates our multilayer Bayesian map inference in the semantic-traversability case. Experiments and discussions are presented in Section V and Section VI, and finally, Section VII concludes the paper.

II. Related Work

In this section, we review the existing literature on multilayer robotic mapping, self-supervised traversability estimation, and deep MTL.

A. Multilayer Robotic Mapping

Multilayer robotic mapping, also known as hybrid mapping, stems from the work of Kuipers and Byun [27] where a hierarchical map with a global (middle-level) *topological* layer and a local (low-level) *metric* layer is built in robot exploration. This layout has been widely used in early works of multilayer mapping [28, 29, 30]. These works rarely contain multiple attributes observed by different sensory modalities, but rather abstract topological relationships from geometric information (i.e., homogeneous). Later, a high-level *conceptual* layer is added to form the *spatial semantic hierarchy* [31, 32, 33], where semantic knowledge extracted from vision or dialogue is integrated. Therefrom, multilayer robotic maps include environmental attributes more than geometry (i.e., heterogeneous), and are built from multiple sensory modalities.

More recently, computer vision and machine learning advancements significantly increase the potential of multilayer robotic mapping. Pronobis and Jensfelt [34] develop a probabilistic framework based on chain graphs to build a four-layer map in three hierarchy levels using multi-modal sensory data: the low-level sensory layer, the middle-level place layer and categorical layer, and the high-level conceptual layer. Jiang et al. [35] propose a four-layer lane-level map model for autonomous vehicles, where each layer contains different types of data and is dedicated to different navigation tasks. For instance, the first road layer is used for static mission

planning and the fourth lane layer for reference trajectory planning. A multilayer High-Definition (HD) map consisting of a road graph, lane geometry and semantic features is also the convention in today's self-driving industry.

Scene graph, a data structure commonly used for describing 3D environments in computer graphics, has been employed in robotics as another form of multilayer map. In a scene graph, each layer has a set of *nodes* representing entities and *edges* between nodes indicating entity relations; each node is also associated to some *attributes*. Armeni et al. [36] construct a four-layer graph that spans an entire building and includes semantics on objects, rooms and cameras, as well as the relationships among them. Rosinol et al. [37] further propose 3D dynamic scene graphs to handle dynamic entities and capture actionable information in the environment. Early works in this direction pose challenges to general robotic navigation and exploration as scene graphs are usually constructed offline and the actionability is at an abstractive topological level. However, recent works [38, 39] have shown promising results for online scene graph construction that is readily usable for path planning. The work of [40] is also a recent attempt at real-time 3D scene graph construction for indoor environments. The proposed work aims to improve the metric-semantic aspect of a more general robotic dense map.

DenseSLAM by Nieto et al. [41] is similar to our work as it also builds a dense grid-based multilayer map of the environment using a probabilistic framework. The fundamental difference is that it contains only geometric information acquired by range-finder sensors. The work by Nordin and Degerman [42] is also related to ours in which a multilayer map with a geometric traversability layer and a ground roughness layer is built for path planning (based on traversability) and velocity control (based on roughness) of ground vehicle systems. However, the map does not provide any semantic knowledge of the environment or explore inter-layer correlations. More recently, Zaenker et al. [43] present a hypermap framework for autonomous semantic exploration. The map includes a grid-based occupancy layer, a polygonal semantic layer, and an exploration layer indicating the areas have yet to be explored. Our work differs from it in that we do not need an extra hypermap interface to convert and unify the information in each layer. Instead, we use the same grid-based representation for each layer to achieve unification.

B. Self-Supervised Learning for Traversability Estimation

Traversability analysis/estimation of the environment is crucial for autonomous navigation in a non-end-to-end framework. In the survey done by Papadakis [26], the traditional approaches to estimating traversability are categorized into *geometry-based* analysis of digital elevation maps [44] and *appearance-based* classification of the terrain into a set of pre-defined classes using supervised learning [45]. However, supervised approaches are not robust to environmental changes and unsustainable for the widespread deployment of robots. They require a significant amount of manual labeling effort to adapt to different distributions. Self-supervised learning, in contrast, using a reliable module to generate supervisory signals for training another model by exploiting the correlations between different input signals, not only automates the labeling process but also gains flexibility in changing environments and platform variations (as the supervisory signals also vary accordingly under different conditions). We note that for some tasks of which the labels do not change much across environments and platforms, such as semantic segmentation, supervised learning approaches are still predominant.

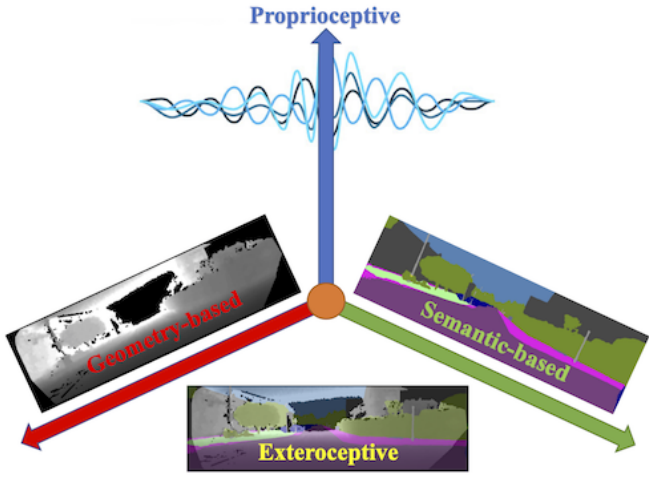


Fig. 2. Space decomposition of self-supervised traversability learning methodologies based on the type of supervisory signals employed, i.e., proprioceptive, geometry-based and semantic-based approaches, and the latter two comprising the domain of exteroceptive approaches.

Similar to [26], we can project these methodologies onto a space characterized in Fig 2, based on the type of supervisory signals employed in self-supervised traversability learning methods. We note that for traversability estimation problems, where we usually need to *predict* the traversability of the ground areas have yet to be traversed, no matter what type of supervisory signals is employed, the *deployed* sensory data tends to be exteroceptive.

Proprioceptive sensors that measure the internal states of the robot, e.g., acceleration [46, 47], force [48, 49], torque [49] and vibration [48], are commonly used to generate supervisory signals for traversability as they directly reflect the physical robot-terrain interaction. Stavens and Thrun [46] generate the label of terrain roughness automatically from a vehicle's inertial measurements while driving. Acceleration signals are used as labels for traversability cost regression based on image textures in [47]. Brooks and Iagnemma [48] train two proprioceptive classifiers based on a traction force model and wheel vibration. Wellhausen et al. [49] record force-torque signals from foot-mounted sensors and images from an onboard camera equipped by a legged robot. Label values generated from the force-torque signals are then projected onto the images to train a convolutional network for terrain property prediction. Recently, robot-terrain interaction sounds captured by microphones are also used as a proprioceptive modality to self-supervise a visual terrain classification network [50].

A common drawback of using proprioceptive data as supervisory signals is that labels can only be generated for the

areas directly interacted with the robot. Heuristics may be necessary for generating negative samples. Thus, exteroceptive sensor (e.g., camera, LiDAR) data is also used for self-supervision. Stanley won the 2005 DARPA Grand Challenge by employing a self-supervised road detection algorithm that uses laser range finders to identify and label drivable areas in the corresponding images [51]. Stereo images are used to self-supervise a vision-based traversability classifier in a near-to-far setting on a DARPA program LAGR platform [52]. To train a deep segmentation network for path proposals in urban environments, Barnes et al. [53] project vehicle future paths (positive) and obstacles detected by a LiDAR scanner (negative) onto images for labeling. Similarly, Broome et al. [54] input geometric features extracted from accumulated LiDAR point clouds to a fuzzy-logic rule set to label the traversability for radar frames automatically. However, all these works only use geometric information acquired by the exteroceptive sensors. We use both geometric and semantic information extracted from exteroceptive sensory data for self-supervised traversability labeling. In addition, we also exploit MTL and multilayer mapping to improve the traversability estimation accuracy.

C. Deep Multitask Learning

MTL shares the same philosophy as our multilayer map inference, aiming to improve the performance of multiple related learning tasks by leveraging useful information among them [55]. Theoretically, MTL models induce a preference for hypotheses that explain multiple tasks, thus leading to reduced overfitting and improved generalization capabilities. Although MTL emerged long before the advent of deep learning [56], it has received more attention recently due to more promising advantages in combination with deep learning, e.g., improved data/memory efficiency and learning/inference speed.

In the deep learning era, MTL equates to designing Deep Neural Networks (DNNs) capable of learning shared representations from multitask signals [57]. Particularly, three problems have been mainly studied in this field: network architectures, optimization strategies, and task relationships [57, 58]. For architectures, the key questions to answer are *what to share* and *how to share* parameters among tasks. Existing MTL methods are often categorized into *hard* and *soft parameter sharing* networks [55]. Hard parameter sharing networks share the exactly same model weights in shared layers [59], while in soft parameter sharing networks, tasks have separate weights but interact with each other through regularization [60], cross-talk [61, 62], or attention [63, 64] in recent works. A new taxonomy of MTL architectures is proposed in [57], grouping them into *encoder-focused* and *decoder-focused* networks based on where the task interactions take place. It is noteworthy that MTL network architecture can also be learned automatically and dynamically from data instead of being manually designed and fixed [65, 66].

However, with an increase in the number and gap of tasks being studied, a well-known issue arises in MTL as *negative transfer*. Negative transfer refers to the phenomenon of performance degradation when sharing information among unrelated or loosely-related tasks. Studies on optimization strategies and task relationships are the effort to tackle this problem. The main stream of works on optimization strategies treat MTL as a single-objective optimization problem with a weighted sum of task-specific losses, and focus on task balancing in terms of these weights. Uncertainty weighting [67] and Gradient Normalization (GradNorm) [68] are two popular approaches among them. Another stream of works formulate MTL as a multi-objective optimization problem and try to find a Pareto optimal solution among all tasks [69]. On the other hand, task relationship learning also directs a way to address negative transfer by leveraging task relatedness to choose tasks to be learned together [70], or to design network architectures. As this topic falls outside the scope of our work, we refer the interested reader to the more detailed discussion in [55].

Attention mechanism has often been used in deep learning to visualize and interpret the inner states of convolutional neural networks, and been successfully applied to natural language processing tasks. More recently, the usage of attention in MTL achieves promising performances [63, 64]. As soft parameter sharing methods, attention-based networks use soft and differentiable attention masks to select/modulate features for each task from a shared backbone. They allow each task to use the shared representation differently, which alleviates negative transfer in hard parameter sharing networks where the shared features are constrained to be the same. As attention modules are small compared to the network backbone, these methods also do not suffer from the scalability issue in common soft parameter sharing networks where each task owns a completely separate set of parameters. Furthermore, attention modules can be incorporated into any feed-forward backbone architectures and learned in an end-to-end manner.

III. Unified Segmentation Model for Multilayer Mapping via Multitask Learning

Robotic maps containing only geometric information usually directly take sensor (e.g., camera, LiDAR, radar, sonar) data as inputs to their mapping systems. However, with the emergence of semantic mapping, a reasoning block that can interpret the raw sensor data for a higher-level understanding of the scene has become essential to the mapping pipeline. This reasoning block is usually chosen as a deep neural network for place recognition, object detection, or semantic segmentation, depending on the specific mapping framework.

As most produced maps only contain one environmental attribute (e.g., room-level, object-level, or dense semantics), the employed DNN is trained for a single task. However, in a multilayer mapping system, it is inefficient or even unrealistic to have multiple DNNs, one per layer. Therefore, we propose to design a unified DNN model for different environmental/map attributes functioning as our reasoning block via MTL. This approach also justifies a practical and real-world application of MTL research in robotics.

In this section, a multitask DNN for image segmentation is presented. The MTL network takes monocular RGB images as inputs, and outputs several dense segmentation results to provide useful measurements for the following multilayer map

inference. We first describe the general architecture design and model objective of the network. Then, we present a specific example of training the network for scene semantic and traversability segmentation. Semantic ground truth labels are publicly available in common data sets, whereas traversability labels are rare and usually robot-dependent. An automated traversability labeling approach using exteroceptive sensory data is also proposed in this section to tackle the issue. Based on how the labels are generated, we regard semantic segmentation as a supervised learning procedure while traversability segmentation self-supervised. Furthermore, the training procedures are generalizable to other map attributes, given the corresponding ground truth labels.

A. Multitask Network with Attention Mechanisms

1) Architecture Design: Inspired by recent works on MTL using attention [63, 64], our network consists of two components: a single shared encoder with task-specific attention modules, and K task-specific decoders for K tasks. This design combines hard and soft parameter sharing strategies: each task has its individual decoder and an attention-modulated shared encoder. It reduces the risk of overfitting by encouraging the encoder to learn to extract features that can fit all tasks. In the meantime, it leverages attention mechanisms to alleviate the negative transfer issue when the encoder is shared among less related or even conflicting tasks.

We adopt two attention mechanisms following the work of Maninis et al. [64]: Squeeze-and-Excitation (SE) blocks and Residual Adapters (RA). Squeeze-and-excitation block is proposed by Hu et al. [71] to adaptively recalibrate channel-wise feature responses by explicitly modeling interdependencies between channels. In other words, it is a lightweight gating mechanism in channel-wise relationships. SE block has a *squeeze* operator (usually an average pooling per channel) which aggregates feature maps into a channel descriptor, followed by an *excitation* operator (two fully-connected layers around non-linearities) that maps the descriptor to a set of channel weights. Finally, the original feature maps are reweighted with the produced channel weights.

The idea of residual adapters is proposed by Rebuffi et al. [72] for multi-domain learning. A small number of domain-specific parameters are added to the adapters to tailor the network to diverse visual domains, while a high percentage of parameter sharing among domains is still maintained. As multidomain learning can be considered as a specific MTL problem, RAs have been applied to MTL for adapting and refining the shared features of each task [64]. RA module is essentially a 1×1 filter bank in parallel with a skip connection, introducing a small amount of computational overhead while substantially improving accuracy and alleviating overfitting [72]. We choose a more efficient variant with parallel adapters [73] in our network. We set the SE and RA parameters to be task-dependent to allow each task to learn its own channel-wise relationships, as well as utilize the shared encoder differently.

Different from [64], we implement the task-specific SE and RA modules inside the *bottleneck* blocks (one 3×3 convolution in the middle of two 1×1 convolutions) of a shared residual encoder, as shown in Fig. 3. To better utilize the shared features extracted by other residual blocks, we add a task-specific element-wise Attention Mask (AM) to those blocks. The attention mask is chosen to be light-weighted, consisting of a 1×1 convolutional filter, a batch normalization layer, and a sigmoid activation to ensure the output belongs to $[0, 1]$. We note that both RA and AM are implemented using convolutions, but the output of RA is added to, whereas the output of AM is element-wise multiplied by the shared features. The following equations represent the difference:

$$F_k(x) = \mathrm{RA}_k(x) + F(x), \tag{1}$$

$$F_k(x) = \mathrm{AM}_k(x) \odot F(x), \tag{2}$$

where $F(x)$ and $F_i(x)$ represent the original shared features and the modified task-specific features of task k, respectively, and $\odot$ denotes element-wise multiplication.

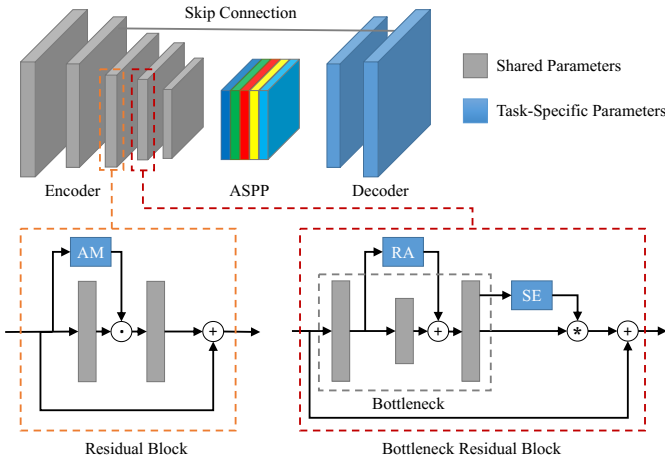


Fig. 3. Network architecture of each task using an illustrative backbone, where $\odot$, $+$ and $*$ denote operations of element-wise multiplication, addition, and matrix multiplication, respectively. The network has task-specific decoders and task-specific attention modules attached to the shared encoder. Attention modules consist of Residual Adapter (RA) and Squeeze-and-Excitation (SE) in the bottleneck residual blocks and Attention Mask (AM) in all other residual blocks. This attentive multitasking architecture can be built upon any feed-forward neural network with residual blocks and can be trained end-to-end.

Fig. 3 illustrates the general architecture of each task in our network. Each task has its dedicated SE, RA and AM modules in the shared encoder to select and refine the shared parameters in a way more favorable to the specific task, with a relatively low parameter overhead. The modified features output from the entire shared encoder are fed into task-specific decoders. In our specific implementation, we have an Atrous Spatial Pyramid Pooling (ASPP) [74] segmentation module between the shared encoder and task-specific decoders for feature resampling, but this MTL architecture can be built on any feed-forward neural network with residual blocks. All task-shared and task-specific parameters are learned in an end-to-end manner.

2) Model Objective: In an MTL setting, the optimization objective for K tasks is usually formulated as a weighted sum

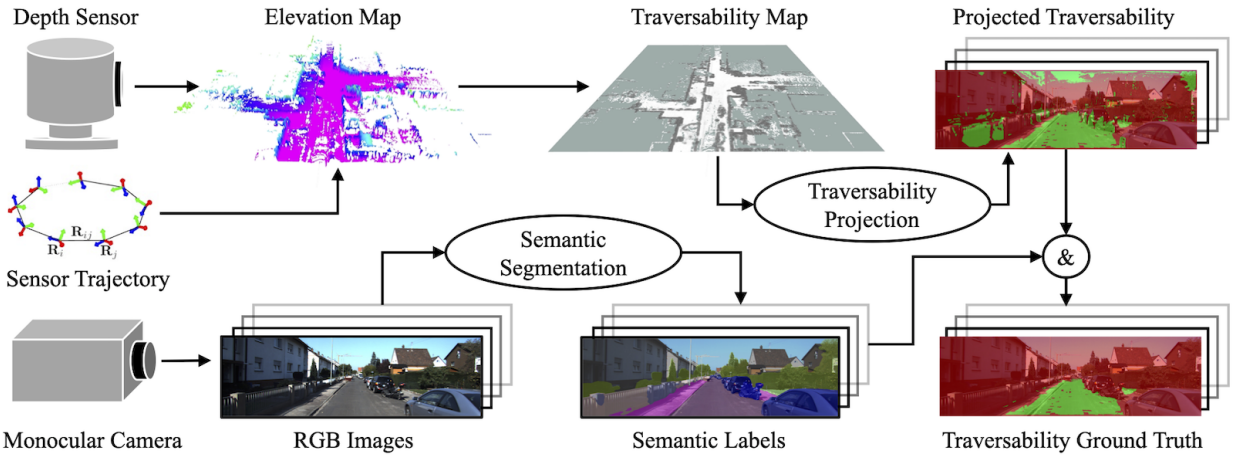


Fig. 4. Our self-supervised traversability labeling pipeline in which exteroceptive sensory data is used as supervisory signals. The upper part concerns geometry; An elevation map is built using point cloud data captured by a depth sensor (LiDAR, depth camera or stereo camera) and the sensor trajectory. A 2D traversability map is then computed from the elevation map and is projected onto images. The lower part considers semantics; We use the predicted semantic labels to filter out (false positive) noises in the projected traversability images. Combining both geometry-based and semantic-based self-supervised approaches, the final traversability ground truth labels can be generated automatically and accurately (green is for traversable area and red for untraversable area).

of all task-specific losses:

$$\mathcal{L}_{\texttt{MTL}}(\mathbf{X}, \mathbf{Y}_{1:K}) = \sum_{k=1}^{K} w_k \mathcal{L}_k(\mathbf{X}, \mathbf{Y}_k), \tag{3}$$

where $\mathbf{X}$ is the input, $\mathbf{Y}_k$ and w_k are the ground truth label and task-specific loss weight of task k, $k = 1, 2, ..., K$, respectively. For *heterogeneous* MTL, where tasks include classification and regression problems, the task-specific loss $\mathcal{L}_k$ has different formulation according to the type of problem. As we are considering a *homogeneous* MTL problem that consists of image segmentation tasks (i.e., classification), the task-specific loss function is chosen to be class-weighted one-hot label cross-entropy loss:

$$\mathcal{L}_k(\mathbf{X}, \mathbf{Y}_k) = -\sum_{c=1}^{C_k} w_{k,c} \mathbf{Y}_{k,c} \log \hat{\mathbf{Y}}_{k,c}, \tag{4}$$

where $\mathbf{Y}_{k,c}$ is a binary variable indicating whether the ground truth label $\mathbf{Y}_k$ is class c, and $\hat{\mathbf{Y}}_{k,c}$ is the network predicted likelihood of $\mathbf{X}$ belonging to class c, $c = 1, 2, ..., C_k$. In order to compensate for unbalanced training data, we also weight the loss according to the number of samples in each class [75], i.e., class weight $\lambda_{k,c}$ will be higher if class c has fewer samples. Precisely, $\lambda_{k,c}$ is set to be the reciprocal of the frequency of class c in training data, and is not part of the network parameters.

B. MTL for Semantic and Traversability Segmentation

The previous subsection proposed a generic MTL network with attention mechanisms that can be used as a unified segmentation model in our multilayer mapping system. This subsection describes a specific implementation of the network for scene semantic and traversability segmentation to provide the following mapping module with necessary measurements.

For segmentation tasks, we choose DeepLabv3+ architecture [76] with WideResNet38 backbone [77] as this combination is used in [78] and achieves the state-of-the-art semantic segmentation performance on KITTI dataset [79]. In our MTL setting, the semantic/traversability segmentation task has its individual decoder and its own set of attention modules attached to the shared encoder (the shared layers and task-specific layers are shown in different colors in Fig. 3). The design of task-specific modules is the same for all tasks, except that the number of output channels of the last convolutional layer is set to be the number of classes in that task. We consider the traversability segmentation task as a pixel-wise binary classification problem (i.e., traversable vs. untraversable); thus, the number of classes in this task is two.

During training, all network parameters (shared and task-specific) are trained together using input images, semantic and traversability ground truth labels simultaneously. The following subsection describes an approach to automatically generating traversability ground truth labels used for training and evaluation in our experiments.

C. Self-Supervised Traversability Labeling

This subsection presents a self-supervised approach to generating vast quantities of labeled traversability training data using exteroceptive sensing without manual annotation. Specifically, our method uses image data, 3D sensing data acquired by the robot platform, and robot trajectories as supervisory signals for traversability. The main idea is to obtain the traversability labels of the environment from 3D geometry and transfer them to the 2D image domain for time-coherent annotation. The overview of our self-supervised traversability labeling process is illustrated in Fig. 4. We first determine the pixel-wise traversability labels of an image based on its

corresponding 3D geometry and robot capabilities, and then use available semantic information to filter out the label noises.

1) 3D Geometry-based Traversability Computation: We employ a geometry-based traversability estimation method with low complexity proposed by Wermelinger et al. [44]. The traversability is estimated based on terrain characteristics and the traversing capabilities of the robot. A robot-centric *elevation map* of the terrain is built from 3D measurements and 6-DoF poses of an exteroceptive sensor (such as a laser scanner, a depth camera, or a stereo camera). The elevation map is a 2.5D grid map in which each cell stores a height estimation and its variance [80].

Three typical local terrain characteristics, i.e., the *slope* s, *roughness* r, and *step height* h, are computed for each map cell by applying different filters to the elevation map. The local traversability score $t_{\texttt{score}} \in [0, 1]$ of that cell, with higher value meaning more traversable, combines these characteristics as follows:

$$t_{\texttt{score}} = 1 - w_1 \frac{s}{s_{\texttt{crit}}} - w_2 \frac{r}{r_{\texttt{crit}}} - w_3 \frac{h}{h_{\texttt{crit}}}, \tag{5}$$

where w_1, w_2 and w_3 are the weights that add up to 1. The $s_{\texttt{crit}}$, $r_{\texttt{crit}}$ and $h_{\texttt{crit}}$ are the robot-specific maximum allowed value for these characteristics. The traversability score is set to be 0 if one of s, r and h exceeds its critical value.

To generate dense labels for an image at timestamp t, we first build a local elevation map using successive frames of point cloud data from timestamp t to $t + l$ and the corresponding sensor poses, where l is the time window length chosen based on data acquisition rate. Essentially, we build a local elevation map that aligns with the camera view at timestamp t. We then compute the traversability score of each cell using (5), and result in a 2D local traversability map at timestamp t that has the same size and resolution as the input elevation map.

2) Map-to-Image Projection: To project the traversability map onto the image plane, we use each pixel's depth value to retrieve the corresponding traversability score from the map and associate it with that pixel. In this way, we generate an image where each pixel has a traversability score, except those with invalid depth values. We then set a threshold for the projected score to get a binary label. An example of the projected traversability labels is given in Fig. 4.

3) Semantics-based Noise Filtering: The traversability labels generated so far are purely from geometric considerations, which can be quite noisy due to errors in sensor reading or depth/pose estimation. As shown in Fig. 4, the projected traversability mislabels some sky pixels as traversable. To correct these false positive labels, we use semantic segmentation of the input image to filter out noises present in semantically untraversable areas, such as sky, buildings, and humans. After semantics-based noise filtering, the final traversability labels in Fig. 4 do not contain these false positives.

IV. Multilayer Bayesian Map Inference

Multilayer mapping aims to use the semantic occupancy map as *a priori* knowledge or complementary information in modeling other environmental attributes, such as traversability, friction, and concentration of a physical quantity, by considering their correlations with occupancy and semantics. To this end, we extend our previous work for semantic mapping [19] to a closed-form Bayesian inference method for multilayer mapping. This section first briefly introduces the semantic layer construction, and then describes how it can be leveraged in traversability layer inference. Extension of the method to more map layers is discussed in Section VI-A.

A. Continuous Semantic Mapping

Assuming the 3D map cells are indexed by $j \in \mathbb{Z}^+$, the j-th map cell with semantic probability θ_j can be described by a Categorical distribution (i.e., the *likelihood*):

$$p(y_i|\theta_j) = \prod_{k=1}^{K} \left(\theta_j^k\right)^{y_i^k}, \tag{6}$$

where $\theta_j = (\theta_j^1, ..., \theta_j^K)$, $\sum_{k=1}^{K} \theta_j^k = 1$ denote the probability of the j-th map cell taking the k-th category from a set of semantic categories $\mathcal{K} = \{1, 2, ..., K\}$. The one-hot encoded semantic measurement $y_i = (y_i^1, ..., y_i^K)$ is from the semantic segmentation result of the MTL network.

Let $\mathcal{X} \subset \mathbb{R}^3$ be the map spatial support, i.e., the Cartesian coordinates in 3D Euclidean space, the training set (data) for semantic map inference can be defined as $\mathcal{D}_y := \{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathcal{X}$ is the position of the i-th measurement and N is the number of training points. In Bayesian semantic mapping, we seek the *posterior* over θ_j; $p(\theta_j|\mathcal{D}_y)$.

For a closed-form solution, we place a Dirichlet distribution over θ_j as the conjugate *prior* of the Categorical likelihood, denoted by $\text{Dir}(K, \alpha_0)$, where $\alpha_0 = (\alpha_0^1, ..., \alpha_0^K)$, $\alpha_0^k \in \mathbb{R}^+$ are the concentration parameters. By applying Bayes' rule and Bayesian kernel inference [81], the posterior $\text{Dir}(K, \alpha_*)$, $\alpha_* = (\alpha_*^1, ..., \alpha_*^K)$ can be computed as follows:

$$\alpha_*^k = \alpha_0^k + \sum_{i=1}^{N} k(x_*, x_i) y_i^k, \tag{7}$$

where $k(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \to [0, 1]$ is a kernel function operating on the map query point x_* and the i-th training point x_i. To exactly and efficiently evaluate the kernel over relevant measurements (i.e., the measurements falling into the neighborhood of a query point), a sparse kernel [82] is employed with the k-d tree data structure to achieve map continuity and $\mathcal{O}(\log N)$ computation time for a single map query, where N is the number of training points. For details, interested readers can refer to [19, and references therein].

B. Semantic-Traversability Mapping

Bernoulli distribution is a natural and common choice for modeling binary variables such as occupancy [7], and terrain traversability [83]. For the j-th map cell, we treat its traversability as a Bernoulli distributed random variable which takes the value 1 (i.e., traversable) with probability ϕ_j. The measurement likelihood is described as:

$$p(z_i|\phi_j) = \phi_j^{z_i} (1 - \phi_j)^{1 - z_i}, \tag{8}$$

Algorithm 1 Semantic Traversability Bayesian Inference

Input: Training data: $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$; Query point: x_*
Initialize: semantic prior parameters $\alpha_*^k \leftarrow 0.001$
traversability prior parameters $\alpha_*, \beta_* \leftarrow 0.001$
for each $(x_i, y_i) \in (\mathcal{X}, \mathcal{Y})$ **do**
  $k_i \leftarrow k(x_*, x_i)$
  for $k = 1, ..., K$ **do**
    $\alpha_*^k \leftarrow \alpha_*^k + k_i y_i^k$
  end for
end for ▷ Semantic mapping
for each $(x_i, z_i) \in (\mathcal{X}, \mathcal{Z})$ **do**
  if x_i in cell j **then**
    $\alpha'_j, \beta'_j \leftarrow 0$
    for $k = 1, ..., K$ **do**
      if k-th category is traversable **then**
        $\alpha'_j \leftarrow \alpha'_j + \alpha_j^k$
      else if k-th category is untraversable **then**
        $\beta'_j \leftarrow \beta'_j + \alpha_j^k$
      end if
    end for ▷ Convert Dirichlet to beta distribution
    $\phi'_j \leftarrow (\alpha'_j - 1)/(\alpha'_j + \beta'_j - 2)$ ▷ Compute mode
    $u \sim \mathcal{U}(0, 1)$ ▷ Draw a uniformly distributed random number
    if $u \leq \phi'_j$ **then**
      $y'_i \leftarrow 1$
    else
      $y'_i \leftarrow 0$
    end if ▷ Sample from Bernoulli distribution
  end if ▷ Generate semantic-traversability measurements
  $k_i \leftarrow k(x_*, x_i)$
  $\alpha_* \leftarrow \alpha_* + k_i(y'_i + z_i)$
  $\beta_* \leftarrow \beta_* + k_i(2 - y'_i - z_i)$
end for ▷ Semantic-traversability mapping
return $\alpha_*^k, \alpha_*, \beta_*$

where the binary measurements $\mathcal{Z} = \{z_1, ..., z_N | z_i \in \{0, 1\}\}$ indicate whether the map cell containing the corresponding position in $\mathcal{X} = \{x_1, ..., x_N | x_i \in \mathbb{R}^3\}$ is classified as untraversable or traversable by the neural network. Given training set $\mathcal{D}_z := \{(x_i, z_i)\}_{i=1}^N$, Bayesian traversability mapping is thus formulated as seeking the posterior distribution $p(\phi_j | \mathcal{D}_z)$ of each map cell.

Semantics of the environment can provide an important insight into its traversability. For instance, we know the sky and buildings are untraversable to ground robots, while the road is traversable to an autonomous vehicle. Therefore, we argue that semantics can be used as another source of observation available for traversability mapping, and a better traversability inference performance is expected when the correlations are leveraged.

In semantic-traversability mapping, we first shrink the Dirichlet posterior distribution $\text{Dir}(K, \alpha_j)$, $\alpha_j = (\alpha_j^1, ..., \alpha_j^K)$ obtained from continuous semantic mapping to a beta distribution $\text{Beta}(\alpha'_j, \beta'_j)$, where α'_j is the sum of concentration parameters α_j^k for all traversable categories k, and β'_j is the sum of α_j^k for all untraversable categories. As Dirichlet distribution is a multivariate generalization of the beta distribution, $\text{Beta}(\alpha'_j, \beta'_j)$ can be proved to be a valid beta distribution.

From the beta distribution, we can deduce a Bernoulli distribution $\text{Bernoulli}(\phi'_j)$ to describe the traversability uncovered by the semantics, where $\phi'_j = (\alpha'_j - 1)/(\alpha'_j + \beta'_j - 2)$, $\alpha'_j, \beta'_j > 1$ is a maximum a posteriori (MAP) estimate of ϕ'_j. We then sample from this Bernoulli distribution to generate semantic-traversability measurements $\mathcal{Y}' = \{y'_1, ..., y'_N | y'_i \in \{0, 1\}\}$. Specifically, for each training point x_i, we first find the corresponding map cell j that contains x_i, and then generate a binary measurement y'_i according to the deduced $\text{Bernoulli}(\phi'_j)$ of that cell. We refer to the generated $\mathcal{Y}'$ as the semantic-traversability measurements because they are indirect observations of traversability through semantics.

To incorporate both measurements into traversability inference, we assume that measurements $\mathcal{Y}'$ also have the same likelihood as in (8):

$$p(y'_j | \phi_j) = \phi_j^{y'_j} (1 - \phi_j)^{1 - y'_j}. \tag{9}$$

We are now concerned with the posterior over possible ϕ_j; $p(\phi_j | \mathcal{D}_{y'}, \mathcal{D}_z)$. Assuming $\mathcal{D}_{y'}$ and $\mathcal{D}_z$ are independent, we have:

$$p(\phi_j | \mathcal{D}_{y'}, \mathcal{D}_z) \propto p(\phi_j | \mathcal{D}_{y'}) p(\phi_j | \mathcal{D}_z), \tag{10}$$

where $\mathcal{D}_{y'}$ comes from semantic segmentation and mapping, while $\mathcal{D}_z$ is directly obtained by traversability segmentation.

For a closed-form formulation, we adopt a conjugate prior of the Bernoulli likelihood as $\phi_j \sim \text{Beta}(\alpha_0, \beta_0)$, in which $\alpha_0 > 0$ and $\beta_0 > 0$ are the shape parameters. According to Bayes' rule, we have $\phi_j | \mathcal{D}_{y'} \sim \text{Beta}(\alpha_{y'}, \beta_{y'})$, where $\alpha_{y'}$ and $\beta_{y'}$ are defined as follows:

$$\alpha_{y'} := \alpha_0 + \sum_{i,\ x_i \text{ in cell } j} y'_i \tag{11}$$

$$\beta_{y'} := \beta_0 + \sum_{i,\ x_i \text{ in cell } j} (1 - y'_i). \tag{12}$$

Then we have $\phi_j | \mathcal{D}_{y'}, \mathcal{D}_z \sim \text{Beta}(\alpha_j, \beta_j)$, where:

$$\alpha_j := \alpha_{y'} + \sum_{i,\ x_i \text{ in cell } j} z_i \tag{13}$$

$$\beta_j := \beta_{y'} + \sum_{i,\ x_i \text{ in cell } j} (1 - z_i). \tag{14}$$

Applying Bayesian kernel inference [81], the final traversability posterior $p(\phi_j | \mathcal{D}_{y'}, \mathcal{D}_z)$ can be obtained as $\text{Beta}(\alpha_j, \beta_j)$ with α_j and β_j defined as:

$$\alpha_j := \alpha_0 + \sum_{i=1}^{N} k(x_j, x_i)(y'_i + z_i) \tag{15}$$

$$\beta_j := \beta_0 + \sum_{i=1}^{N} k(x_j, x_i)(2 - y'_i - z_i). \tag{16}$$

The MAP estimate of ϕ_j then has the closed-form solution:

$$\hat{\phi}_j = \frac{\alpha_j - 1}{\alpha_j + \beta_j - 2} \text{ and } \alpha_j, \beta_j > 1. \tag{17}$$

The expected value and variance of ϕ_j can also be computed in closed form:

$$\mathbb{E}[\phi_j] = \frac{\alpha_j}{\alpha_j + \beta_j} \text{ and } \mathbb{V}[\phi_j] = \frac{\alpha_j \beta_j}{(\alpha_j + \beta_j)^2(\alpha_j + \beta_j + 1)}. \tag{18}$$

An algorithmic implementation of the semantic traversability Bayesian inference procedure is provided in Algorithm 1. Lines 4-9 are the inference procedure for semantic mapping and lines 10-31 are for semantic-traversability mapping. We use a simple sampling strategy (lines 21-26) to generate semantic-traversability (pseudo-)measurements from the Bernoulli distribution deduced from semantic posterior. In this way, we are able to leverage more accurate and up-to-date semantic posteriors in traversability inference, instead of the noisy single-frame semantic measurements.

The computational complexity for building the semantic layer is $\mathcal{O}(M \log N)$, where M and N is the number of test points and training points, respectively. The semantic-traversability mapping retains the same $\mathcal{O}(M \log N)$ time complexity as the semantic mapping, with small additional memory expenditures to store the deduced Bernoulli distributions and traversability posteriors. However, as we use the posterior distribution of one layer to build another layer, the computational complexity of the multilayer mapping grows linearly with the number of map layers.

Remark 1. *The traversability layer inference is similar to the continuous occupancy mapping approach of [7]. While the mathematical derivations are the same, the traversability layer in this work performs two updates per iteration; one essentially from the inferred semantic layer and one directly from a neural network.*

V. Experimental Evaluation

In this section, we evaluate our proposed method using two publicly available datasets for robotic applications: the KITTI dataset [79] and the TartanAir dataset [84]. For both datasets, qualitative and quantitative results are provided with discussion. At the end of this section, we further validate the proposed method using data collected by our bipedal robot platform, Cassie Blue.

A. Implementation Details

For the MTL network, we use the architecture shown in Fig. 3 based on DeepLabv3+ and WideResNet38 backbone. During training, we use an SGD optimizer with initial learning rate 0.001, momentum 0.9 and weight decay 0.0001. A polynomial learning rate policy is also employed with the power of 1.0. In addition, we make the Synchronized Batch Normalization (SBN) in [78] task-specific, with a batch size of 8 distributed over two NVIDIA TITAN RTX GPUs. We use the pretrained semantic segmentation weights on Cityscapes [85] for model initialization in all experiments. The traversability-specific decoder is also initialized with the pretrained weights considering the two tasks are closely related. All task-specific attention modules are randomly initialized. Our network implementation is based on [78] using PyTorch. We provide our code at https://github.com/ganlumomo/mtl-segmentation.

The multilayer Bayesian mapping algorithm is implemented in C++ with the Robot Operating System (ROS) and uses the Learning-Aided 3D Mapping (LA3DM) library [7]. Kernel length-scale and kernel scale of the sparse kernel are set to 0.3 m and 10, respectively, and remain fixed throughout the experiments. Dirichlet concentration parameters and beta shape parameters are initialized to 0.001. All mapping experiments are conducted on an Intel i7 processor with 8 cores and 32 GB RAM. The implementation of our multilayer mapping algorithm is available at https://github.com/ganlumomo/MultiLayerMapping.

For self-supervised traversability labeling, we use the robot-centric elevation mapping library [80] and the traversability estimation package [44]. Dataset loaders for KITTI and TartanAir are provided, along with the code for projecting traversability scores onto the image plane. Elevation maps are built at a resolution of 0.1 m. The $s_{\texttt{crit}}$, $r_{\texttt{crit}}$ and $h_{\texttt{crit}}$ in (5) are set to 1.0, 0.05 and 0.12, respectively. The weights are set to $1/3$, and the threshold for $t_{\texttt{score}}$ is 0.5. The whole labeling pipeline is implemented in ROS, and available at https://github.com/ganlumomo/traversability_labeling_ws.

B. Ablation Study

To better understand the effect of the task-specific attention modules, we first perform an ablation study of our MTL network with and without attention modules and compare the performance with the corresponding single-task learning baseline. For a fair comparison, we use the same architecture (DeepLabv3+ with WideResNet38 backbone) and pretrained model (on Cityscapes) for the following settings:

1) *STL*: The vanilla DeepLabv3+ architecture with WideResNet38 backbone for single-task learning. This model is proposed by Zhu et al. [78] for semantic segmentation, and we also train it for traversability segmentation.
2) *MTL w/o Attentions, Fixed*: The standard MTL baseline with a shared encoder and task-specific decoders [57], trained with the shared parameters fixed.
3) *MTL w/o Attentions*: The same architecture as above [57] with all parameters trained.
4) *MTL w/ Attentions*: The same architecture as above with additional task-specific attention modules and task-specific SBN operators attached to the shared encoder. All parameters are trained.

As for evaluation criterion, in addition to the standard mean Intersection over Union (mIoU) for segmentation accuracy, we also use a multitask performance metric defined in [57]:

$$\Delta_{\texttt{MTL}} = \frac{1}{K} \sum_{i=1}^{K} (-1)^{l_i} (M_{m,i} - M_{s,i}) / M_{s,i}, \tag{19}$$

where $M_{m,i}$ and $M_{s,i}$ are the performance of the MTL and STL model on task i, respectively, and K is the number of tasks. $l_i = 0$ if a higher value means a better performance for metric M_i, and 1 otherwise. A higher $\Delta_{\texttt{MTL}}$ means a better multitask performance w.r.t the same STL baseline.

We train these four models using the same hyperparameters on KITTI dataset [79]. For semantic segmentation, we use the KITTI pixel-level semantic dataset consisting of 200 annotated

TABLE I
ABLATION STUDY OF DEEP MTL NETWORKS FOR SEMANTIC AND TRAVERSABILITY SEGMENTATION. MEAN INTERSECTION OVER UNIONS (MIOUS) OF 19 SEMANTIC CLASSES AND 2 TRAVERSABILITY CLASSES ARE REPORTED ON A TEST SET FROM KITTI ODOMETRY SEQUENCE 10. MULTITASK PERFORMANCE $\Delta_{\mathtt{MTL}}$ INDICATES THE AVERAGE PER-TASK PERFORMANCE GAIN W.R.T THE SINGLE-TASK BASELINE (SEE (19)).

Setting	Model	Training	Segmentation mIoU (%)		$\Delta_{\mathtt{MTL}}$ (%)
			Semantic	Traversability	
1	STL	Individually for each task	83.08	79.80	+ 0.00
2	MTL w/o Attentions, Fixed	Simultaneously with shared parameters fixed	83.08	77.06	- 1.72
3	MTL w/o Attentions	Simultaneously	83.24	78.38	- 0.79
4	MTL w/ Attentions	Simultaneously	**86.51**	**83.91**	**+ 4.64**

training images. We randomly select 20 images for testing and exclude them from the training process. For traversability segmentation, we automatically generate traversability labels for 200 images from sequence 06 and 07 of the KITTI odometry dataset for training, and another 200 images from sequence 10 for testing. The ablation study results are reported in Table I.

From the results in Table I, the standard MTL baselines (setting 2 and 3) fail to outperform their single-task equivalent (setting 1) for traversability segmentation on the KITTI dataset while attain or slightly surpass the single-task semantic segmentation performance. In a word, the two MTL models degrade the single-task performance (indicated by the negative $\Delta_{\mathtt{MTL}}$ values). Similar observations are also reported by Vandenhende et al. [57] on the PASCAL dataset, where the standard MTL baseline underperforms the STL network on tasks such as semantic segmentation, human part segmentation, and saliency detection.

The observed performance degradation can be attributed to the negative transfer issue in which the gradient of the traversability segmentation loss to the shared encoder conflicts with that of the semantic segmentation loss during training. Without any task-specific capabilities, the shared encoder tends to be undertrained for an individual task. This issue is alleviated by adding attention modules to modulate the shared encoder in a task-specific manner: in setting 4, a performance gain is obtained for both semantic segmentation ($+3.43\%$) and traversability segmentation ($+4.11\%$). According to the ablation study, we use our MTL network with task-specific attentions in the following experiments.

C. KITTI Dataset

KITTI dataset [79] is a real-world benchmark for a variety of computer vision and robotic tasks ranging from visual odometry, object detection and tracking. It is collected by an autonomous driving platform equipped with stereo cameras, a LiDAR scanner, IMU and GPS, etc., in accord with the common setup of a modern robotic system. Recently, KITTI also released its semantic and instance segmentation benchmark. However, none of these benchmarks exactly match this work. Accordingly, we quantitatively evaluate our system using the ground truth data we generated.

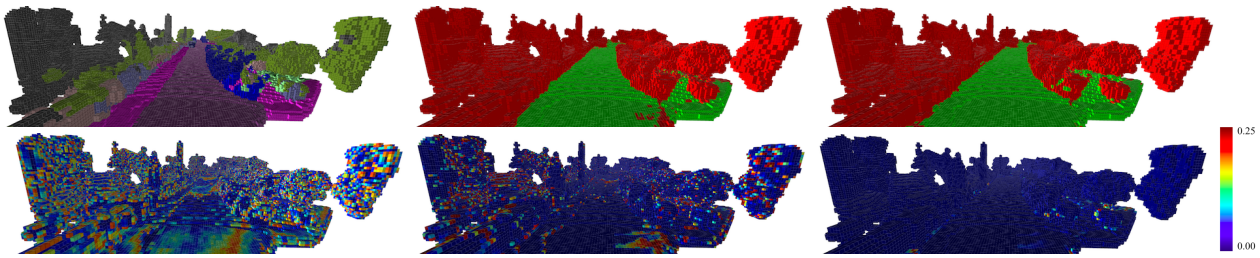
For training our MTL network, we use the KITTI pixel-level semantic segmentation dataset which consists of 200 training images. It shares the same 19 semantic class definition with Cityscapes [85], including *road*, *sidewalk*, *building*, *car*, etc. For traversability segmentation, we generate ground truth labels for the KITTI odometry dataset sequence 06 and 07, and randomly select 200 training images for task balancing. We first use ORB-SLAM2 [86] to estimate the 6-DoF camera poses from stereo measurements, and transform them to get the LiDAR poses using camera-LiDAR extrinsics. We then use the LiDAR point clouds and the corresponding LiDAR poses to generate traversability scores, as described in Section III-C. To project the scores onto images, we use the depth values estimated by stereo matching [87].

TABLE II
QUANTITATIVE RESULTS OF OUR SYSTEM ON KITTI DATASET USING INTERSECTION OVER UNIONS (IOUS) FOR TRAVERSABILITY (TRAV.) CLASSIFICATION.

Method	Untraversable (%)	Traversable (%)	Mean (%)
STL Trav. Segmentation	92.02	75.34	83.68
MTL Trav. Segmentation	96.42	91.42	93.92
Trav. Mapping	96.81	92.34	94.58
Semantic-Trav. Mapping	**97.62**	**94.46**	**96.04**

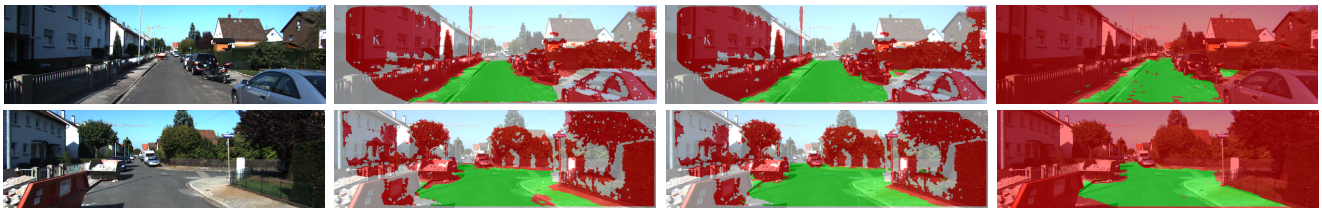
We test our framework on KITTI odometry sequence 15, and provide the quantitative results in Table II. As the ground truth semantic labels for the sequence are unavailable (KITTI does not provide semantic labels for its odometry data; however, we need the sequential information for mapping), we only compare the traversability segmentation performance between the STL baseline and our MTL network. As shown in Table II, our MTL network improves about 10% mIoU for traversability segmentation, and especially improves 16.08% IoU for the traversable class. Two reasons might justify this: First, 200 training images are insufficient to well train an STL network that is pretrained for another task; Second, traversability is highly correlated with semantics in an on-road environment, and the MTL network is more data efficient for those tasks.

The performance of our multilayer Bayesian mapping is also evaluated in Table II, where traversability mapping refers to traversability layer inference without semantic information, and semantic-traversability mapping uses Algorithm 1 that leverages inter-layer correlations. For quantitative comparison, we project the 3D maps onto 2D images and compare them with the ground truth images pixel-wisely. To this end, for each pixel in the test image, we use its depth value to query the inferred traversability probability from the corresponding



(a) (b) (c)

Fig. 5. 3D qualitative results of our multilayer Bayesian mapping algorithm on KITTI dataset: (a)-(c) are respectively the semantic layer, traversability layer and semantic-traversability layer with its corresponding uncertainty map. The uncertainty map shows the variance of the posterior distribution for each voxel, using Jet colormap for the range of $[0, 0.25]$ (for semantic maps, it shows the variance of the Dirichlet posterior for the predicted class, and for traversability maps, it shows the variance of the beta posterior). We can see the improved estimation results and reduced estimation variances in (c) compared with (b) by leveraging semantic posteriors in traversability inference. Relatively high variances are also observed for the voxels with estimation error.



(a) (b) (c) (d)

Fig. 6. 2D quantitative results of our multilayer Bayesian mapping algorithm on KITTI dataset. (a) Input color images. (b) Projected images from the traversability maps (the pixels with invalid depth are excluded). (c) Projected images from the semantic-traversability maps. (d) Traversability ground truth labels generated automatically using the method in Section III-C. We can see the improved estimation in (c) compared with (b) based on the ground truth.

map cell and convert it to a binary prediction by thresholding. We exclude all pixels with invalid depth from both result and ground truth images before computing the metrics. The qualitative results of 3D multilayer maps and 2D projected images are shown in Fig. 5 and 6, respectively.

The results show that through sequential Bayesian inference, traversability mapping alone can improve the classification accuracy (line 3 compares to line 2 in Table II), which is beneficial from multi-frame measurements. However, leveraging semantic posteriors in traversability inference achieves the best performance (line 4 in Table II). This is also shown in Fig. 5, where the semantic-traversability mapping not only corrects some misclassifications, but also reduces the map uncertainty.

D. TartanAir Dataset

As KITTI dataset aims for on-road autonomous vehicle applications, the environments are semblable and relatively simple. To further test our framework in other environments, we employ a new challenging dataset, the TartanAir [84]. TartanAir is mainly introduced as a benchmark for visual SLAM algorithms, but it is also suitable for other robotic applications such as robotic mapping. It is collected in photo-realistic simulation environments and thus able to provide multi-modal sensor measurements (including stereo and depth images, LiDAR point clouds) and precise ground truth data (such as semantic labels and camera poses).

Unlike Cityscapes or KITTI collected by ground robots from consistent camera viewpoints (pointing to the front from the same height), TartanAir has various camera viewpoints and diverse sensor motion patterns due to its exemption from platform constraints, posing a challenge for the segmentation network. It is also collected in a variety of simulation environments including indoor, outdoor, underwater and sci-fi scenes. Without loss of generality, we choose two representative outdoor environments that are close to the domain of our application for our experiments, i.e., *abandoned factory* and *neighborhood*. For both environments, we use a long sequence for training and several short sequences for testing. Specifically, sequence P000 of 2176 frames is used as training data for the abandoned factory environment, and sequence P000 of 4204 frames for the neighborhood environment.

TABLE III
QUANTITATIVE EVALUATION OF OUR SELF-SUPERVISED TRAVERSABILITY LABELING COMPARED WITH THE SIMULATED PRECISE TRAVERSABILITY LABELS ON TARTANAIR TRAINING SETS USING INTERSECTION OVER UNIONS (IOUS).

Environment	Untraversable (%)	Traversable (%)	Mean(%)
Abandoned Factory	93.42	81.80	87.61
Neighborhood	91.83	85.30	88.57

In addition, as the TartanAir dataset provides precise ground truth data for its simulation environments, we can also use these data to evaluate the performance of our self-supervised traversability labeling. To this end, we compare the traversability labels we generated using estimated camera poses and semantic segmentation results with the precise traversability labels generated using the simulated ground truth camera poses (i.e., precise geometry) and semantic labels (i.e., precise semantics) provided by the dataset. The qualitative comparison of both traversability labels is shown in Fig. 7, where the upper side shows the best examples and the lower side presents the worst examples. From the worst cases, we can clearly see

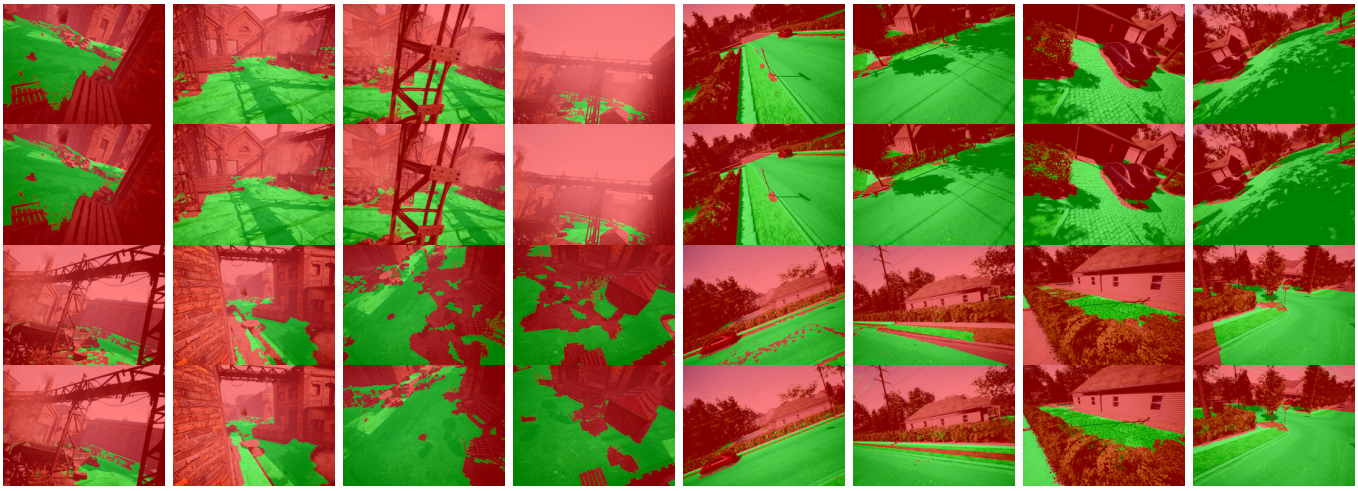


Fig. 7. Qualitative evaluation of our self-supervised traversability labeling proposed in Section III-C compared with the simulated precise traversability labels on TartanAir training sets. The left side is generated from sequence P000 of the abandoned factory environment, and the right side is from sequence P000 in the neighborhood environment. The top two rows are the selected best examples of the generated (first row) and simulated precise (second row) traversability labels, and the bottom two rows show the worst cases in the same order. The color images are overlaid with the labels for visualization.

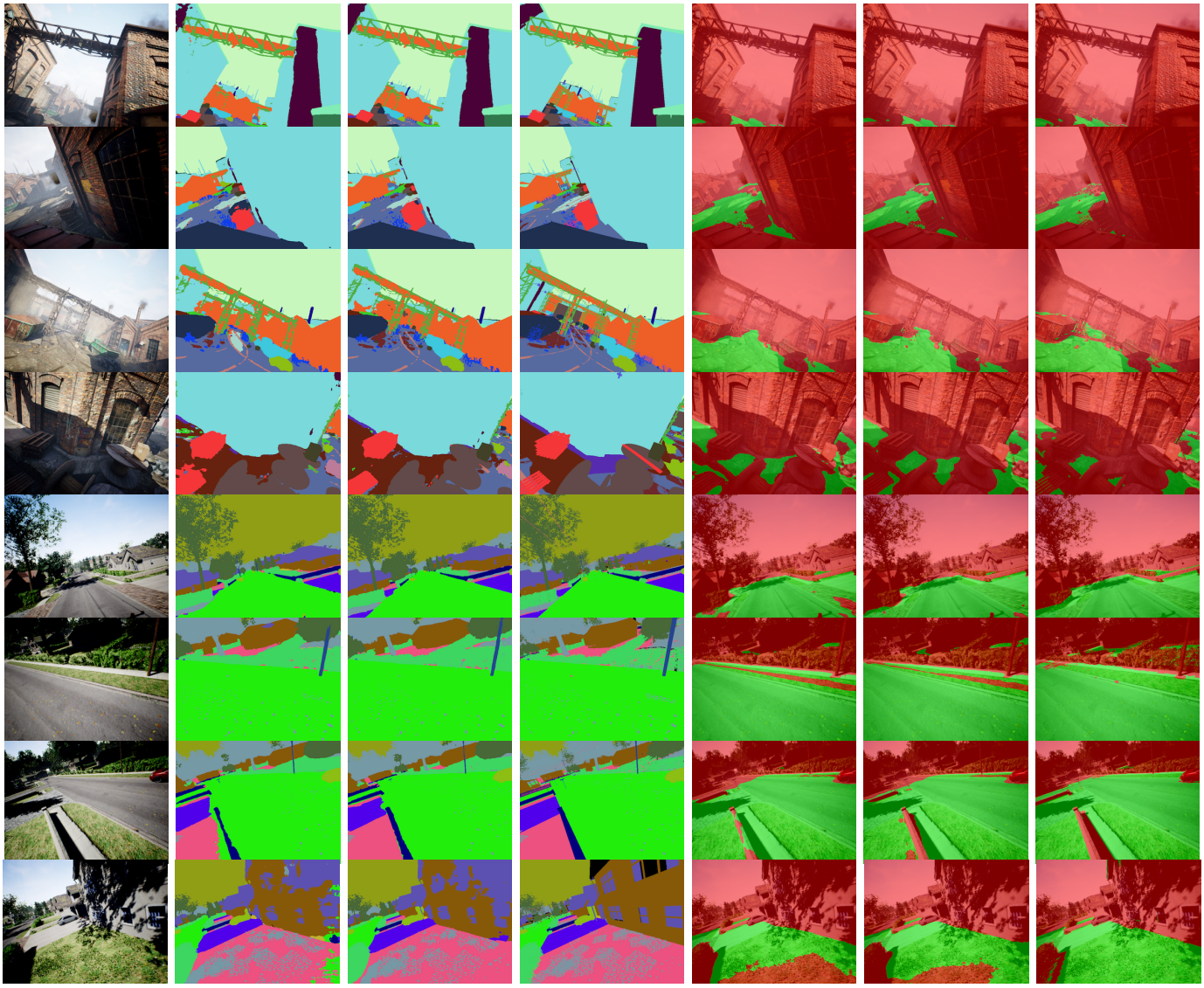
TABLE IV
QUANTITATIVE RESULTS OF OUR MTL NETWORK ON TARTANAIR TEST SETS. MEAN INTERSECTION OVER UNIONS (MIOUS) OF 30 SEMANTIC CLASSES FOR THE ABANDONED FACTORY ENVIRONMENT, 17 SEMANTIC CLASSES FOR THE NEIGHBORHOOD ENVIRONMENT, AND 2 TRAVERSABILITY CLASSES ARE REPORTED WITH THE MULTITASK PERFORMANCE (Δ_{MTL}). THE STL AND MTL BASELINE MODELS ARE DEFINED IN SETTING 1 AND 3 IN SECTION V-B.

Environment	Sequence (No. of images)	Method	Semantic Seg. (mIoU)	Traversability Seg. (mIoU)	Δ_{MTL} (%)
Abandoned Factory	P001 (434)	STL	47.77	85.78	+ 0.00
		MTL Baseline	50.05	86.90	+ 3.04
		MTL	**53.16**	**87.01**	**+ 6.36**
	P002 (927)	STL	42.61	**81.56**	+ 0.00
		MTL Baseline	42.54	79.25	- 1.50
		MTL	**46.50**	80.37	**+ 3.84**
	P009 (339)	STL	44.85	82.41	+ 0.00
		MTL Baseline	45.33	81.78	+ 1.53
		MTL	**47.40**	**84.82**	**+ 4.31**
	Average	STL	45.08	83.25	+ 0.00
		MTL Baseline	45.97	82.64	+ 0.62
		MTL	**49.02**	**84.06**	**+ 4.86**
Neighborhood	P002 (523)	STL	49.74	75.36	+ 0.00
		MTL Baseline	48.92	**76.82**	+ 0.14
		MTL	**52.00**	75.60	**+ 2.43**
	P005 (724)	STL	47.22	77.63	+ 0.00
		MTL Baseline	**51.12**	77.15	**+ 3.82**
		MTL	50.37	**77.83**	+ 3.46
	Average	STL	48.48	76.50	+ 0.00
		MTL Baseline	50.02	**76.99**	+ 1.91
		MTL	**51.19**	76.72	**+ 2.94**

the errors caused by noisy semantic segmentation (e.g., lower column 1), the drift in pose estimation (e.g., lower column 6) and their combinations.

The quantitative evaluation is also given in Table III. It can be observed that our self-supervised traversability labeling is able to provide reasonably accurate image traversability labels with mean IoUs 87.61% and 88.57% compared with the simulated precise traversability labels. Therefore, we argue that the proposed self-supervised traversability labeling is an effective and inexpensive method to generate image traversability labels for real environments where the geometric and semantic ground truth is usually unavailable. The traversability labeling evaluation is conducted on the training sequences of the TartanAir dataset. In the following experiments, we consistently use the traversability labels generated by our self-supervised method as the traversability ground truth for training and testing.

As TartanAir only provides raw mesh labels randomly assigned by the AirSim simulator, we manually group the mesh labels into semantic labels, resulting in 30 semantic classes for the abandoned factory environment and 17 semantic classes for the neighborhood. Next, we use the semantic labels and our



(a) (b) (c) (d) (e) (f) (g)

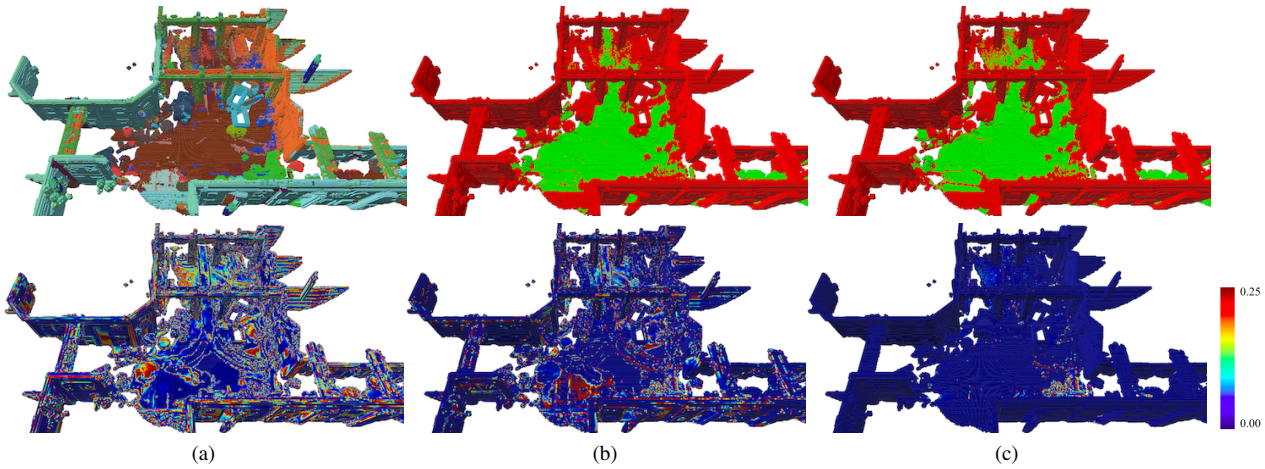
Fig. 8. Qualitative results of our MTL network compared with the STL network on TartanAir dataset. The first four rows are from the abandoned factory environment, and the rest are from the neighborhood environment. For each environment, we show selected examples from the best cases to the worst cases. (a) Input color images. (b)-(d) are STL, MTL, and ground truth semantic segmentation, respectively. (e)-(g) are STL, MTL, and ground truth traversability segmentation, respectively. Note that (g) are automatically generated using the method in Section III-C. We can see that (c) and (f) have better segmentation results compared with (b) and (e). The performance improvement obtained for the same input image on both tasks also indicates the correlation between the two tasks.

generated traversability labels to train our MTL network. We evaluate the segmentation performance on the test sequences and compare with the corresponding STL network and the standard MTL baseline in [57] (setting 3 in Section V-B). The quantitative results are listed in Table IV.

We obtain a lower semantic segmentation mIoU on the TartanAir dataset than that on KITTI. This is reasonable as KITTI has a similar environment and the same semantic class definition as Cityscapes on which the network is pretrained. By contrast, TartanAir is more challenging due to the domain gap brought by the differences in environment, class definition and camera viewpoint. In this case, our MTL network still outperforms the STL network in both tasks, and has a better overall performance than the MTL baseline. However, we notice that the mIoU improvement for traversability segmentation is only less than 1%. We conjecture that this is because the correlation between the two tasks in TartanAir might not be as high as in KITTI. Some qualitative results of our MTL network are also presented in Fig. 8, where we can see a

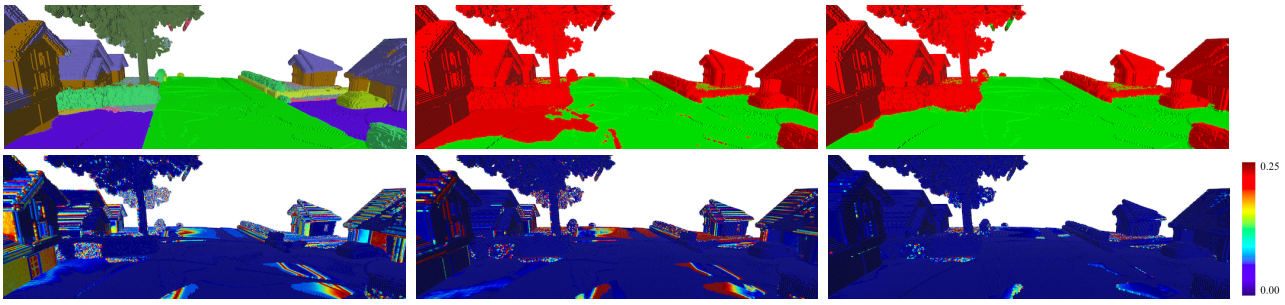
TABLE V
COMPARISON OF THE AVERAGE INFERENCE TIME PER IMAGE PER TASK AND THE SIZE OF MODEL PARAMETERS PER TASK AMONG THREE NETWORKS. THE EXPERIMENTAL SETUP IS GIVEN IN SECTION V-A

Method	Average Inference Time (sec)	Model Parameters (M)
STL	0.0296	137.1
MTL Baseline	**0.0185**	**85.80**
MTL	0.0199	92.05



(a) (b) (c)

Fig. 9. Qualitative results of our multilayer Bayesian mapping algorithm on TartanAir dataset abandoned factory environment (sequence P001): (a)-(c) are respectively the semantic layer, traversability layer, and semantic-traversability layer with its corresponding uncertainty map. The variance of each voxel is shown using Jet colormap for the range of $[0, 0.25]$.



(a) (b) (c)

Fig. 10. Qualitative results of our multilayer Bayesian mapping algorithm on TartanAir dataset neighborhood environment (sequence P005): (a)-(c) are respectively the semantic layer, traversability layer, and semantic-traversability layer with its corresponding uncertainty map. The variance of each voxel is shown using Jet colormap for the range of $[0, 0.25]$.

TABLE VI
QUANTITATIVE RESULTS OF OUR MULTILAYER BAYESIAN MAPPING ALGORITHM ON TARTANAIR DATASET USING INTERSECTION OVER UNIONS (IOUS) FOR TRAVERSABILITY (TRAV.) CLASSIFICATION.

Method	Untraversable (%)	Traversable (%)	Mean (%)
MTL Trav. Segmentation	98.63	74.71	86.67
Trav. Mapping	98.81	77.43	88.12
Semantic-Trav. Mapping	**99.06**	**80.70**	**89.88**

performance improvement is obtained for both semantic and traversability segmentation compared with single-task learning. The statistics of the inference time and model parameters of three networks are listed in Table V. We argue that the main advantage of using an MTL network in our multilayer mapping framework is saving computational resources for robotic applications. Even when the performance improvement of the MTL network is not substantial, STL network does not scale with the number of tasks (map layers).

The quantitative results of our multilayer Bayesian mapping algorithm are given in Table VI. Following the same trend in KITTI experiments, our traversability mapping improves the classification accuracy of single-frame segmentation by inference on multi-frame measurements (line 2 compares to line 1 in Table VI). Leveraging semantic posterior in traversability inference further improves the mapping performance. It is worth mentioning that our continuous semantic mapping also improves the mIoU of semantic segmentation from 52.30% to 60.30%. Although it is not the contribution of this work, our semantic-traversability mapping can benefit from this improvement by using the inferred semantic posterior instead of the noisy semantic segmentation (measurements). Figure 9 and 10 qualitatively show those map layers with the corresponding uncertainty map for the abandoned factory and neighborhood environment, respectively.

E. Cassie Robot Data

Lastly, we test our multilayer mapping system on real-world data collected using our Cassie Blue robot platform on the University of Michigan - North Campus. Cassie Blue is a bipedal robot equipped with customized sensor suite including an Intel RealSense depth camera D435 and a Velodyne LiDAR VLP-32C, as shown in Fig. 11. The extrinsics of camera and LiDAR are calibrated using [88]. A contact-aided invariant EKF [89] is running on-board for high-frequent robot pose estimation during data collection. The experiment setup is the same as in [19], except that we build multilayer map offline using recorded data at the recording speed.

For training the MTL network, we extract images and generate traversability labels from a recorded training sequence

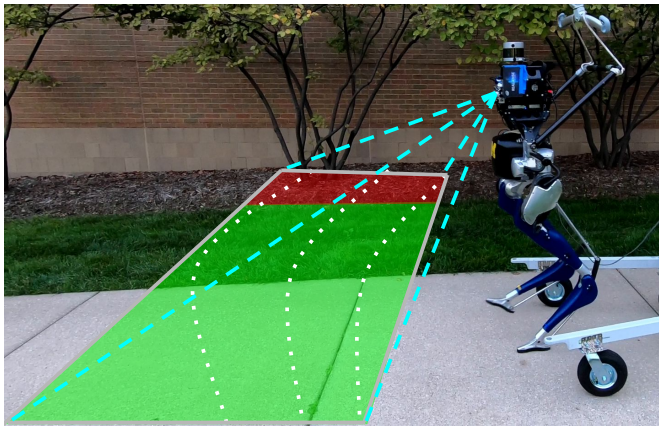


Fig. 11. The Cassie Blue robot platform equipped with a depth camera and a LiDAR scanner. We use the LiDAR point clouds and the estimated robot poses to build a 2D local traversability map based on geometric information. We then project the traversability scores onto the color image plane using the depth values from the camera to generate traversability ground truth labels automatically.

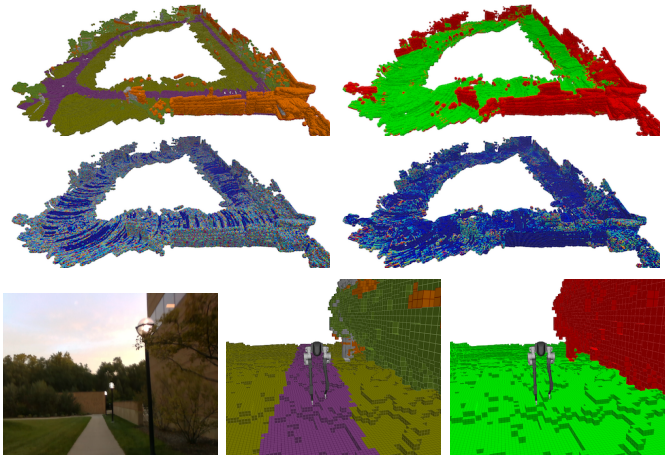


Fig. 12. Top block: Qualitative results of our multilayer Bayesian mapping algorithm on data collected with Cassie Blue on the University of Michigan - North Campus. From the left to right column are the semantic layer and semantic-traversability layer with its corresponding uncertainty map. A closed-up view of both map layers are shown in the bottom block with the corresponding input color image captured by the on-board camera shown on the left.

(ROS bag). We use LiDAR point clouds and the corresponding robot poses from invariant EKF to build a 2D traversability map, and project the traversability scores onto the color image plane using the depth values from the RealSense camera. The self-supervised traversability labeling process is illustrated in Fig. 11. For semantic labels, we use our manually annotated NCLT dataset collected on campus [90].

Due to the heavy computational requirement (especially GPU memory) of the MTL network, we are currently unable to run the network on the onboard computing device, which is the bottleneck that prevents running our framework fully online. This can be addressed by more efforts on optimizing the network for embedded systems with a performance trade-off, as in our previous work [19]. In this experiment, we run the MTL network offline to generate the semantic and traversability segmentation results for the recorded data.

Multilayer mapping algorithm takes the offline semantic and traversability segmentation for map building. It needs to be mentioned that the mapping module is running in an online manner: building the map when data is replayed from a ROS bag. The average runtime of the multilayer mapping is 1.83 sec/scan for both layers.

The qualitative results in Fig. 12 are visually correct and pleasing. Most ground areas are mapped as traversable (green) with some untraversable voxels (red) caused by pedestrians moving in the environment. It is noteworthy that the mapping algorithm processes the recorded data at the original collecting speed, which shows that our multilayer Bayesian mapping is efficient, and ready for online navigation and exploration applications.

VI. Discussion on Multitask Multilayer Mapping Framework

In this section, we discuss the generalizability of the proposed multitask multilayer Bayesian mapping framework and explain why it has the potential of being widely adopted within the robotics community. We first discuss the extension to more map layers via a constructive example. Next, we discuss significant problems that must be addressed for real-time applications of the proposed framework in autonomous navigation. The latter is an indispensable challenge for safe navigation, especially in unknown unstructured environments, and deserves more in-depth studies as future work.

A. Extensions to More Layers

The current framework can be easily extended to include more map layers. As a constructive example, this subsection discusses a friction coefficient layer and the way to use it in traversability layer inference.

We assume the friction coefficient observation of the j-th map cell follows a univariate Gaussian distribution with known variance σ^2; $f_i \sim \mathcal{N}(\mu_j, \sigma^2)$, where f_i is the i-th measurement acquired at 3D position x_i in map cell j [1]. Given observations $\mathcal{D}_f := \{(x_i, f_i)\}_{i=1}^N$, we seek the posterior $p(\mu_j|\mathcal{D}_f)$ in friction mapping. A conjugate prior for this Gaussian likelihood function is also a Gaussian distribution $\mu_j \sim \mathcal{N}(\mu_0, \sigma_0^2)$ [91]. The mean and variance of the posterior distribution are given by:

$$\mathbb{E}[\mu_j|\mathcal{D}_f] = \frac{\sigma^2}{M\sigma_0^2 + \sigma^2}\mu_0 + \frac{\sigma_0^2}{M\sigma_0^2 + \sigma^2}\sum_{i=1}^{M} f_i,$$

$$\mathbb{V}[\mu_j|\mathcal{D}_f] = \frac{\sigma^2\sigma_0^2}{\sigma^2 + M\sigma_0^2},$$

where M is the number of training points in the j-th map cell.

To apply Bayesian kernel inference, we can set $\sigma_0^2 = \sigma^2/\lambda$, where λ is a hyperparameter reflecting the confidence in the

[1]Of course, obtaining accurate friction coefficient measurements is a highly challenging problem and an interesting future research direction in robotic mapping. Here, we only discuss the mathematical derivation and generalizability of the developed mapping framework.

prior [83]. The kernel version of mean and variance are then derived as:

$$\mathbb{E}[\mu_*|\mathcal{D}_f] = \frac{\lambda\mu_0 + \sum_{i=1}^{N} k(x_*, x_i) f_i}{\lambda + \sum_{i=1}^{N} k(x_*, x_i)},$$
$$\mathbb{V}[\mu_*|\mathcal{D}_f] = \frac{\sigma^2}{\lambda + \sum_{i=1}^{N} k(x_*, x_i)}.$$

Similar to the derivation of our semantic traversability Bayesian inference, we can convert the friction coefficient (Gaussian) posterior distribution $p(\mu_j|\mathcal{D}_f)$ to a Bernoulli distribution Bernoulli(p) by setting thresholds for the Gaussian cumulative distribution function based on the correlation between friction coefficient and traversability, where $p = P(f_{low} \leq \mu_j \leq f_{high})$. We then sample from this Bernoulli distribution to get friction-traversability measurements $\mathcal{F}' := \{f'_1, ..., f'_N | f'_i \in \{0, 1\}\}$ as done in Algorithm 1.

To incorporate fiction coefficient layer inference into the traversability mapping, we assume the measurements $\mathcal{F}'$ are independent to $\mathcal{Z}$ and $\mathcal{Y}'$, and also have Bernoulli likelihood:

$$p(f'_i|\phi_j) = \phi_j^{f'_i}(1 - \phi_j)^{1-f'_i},$$

where $\phi_j, \mathcal{Z}$ and $\mathcal{Y}'$ are defined in Section IV-B. Following the rest derivations in Section IV-B, we are able to leverage the correlations to both semantic layer and fiction coefficient layer in traversability mapping. The traversability posterior $p(\phi_j|\mathcal{D}_{y'}, \mathcal{D}_{f'}, \mathcal{D}_z)$ can then be obtained as Beta(α_j, β_j) with:

$$\alpha_j := \alpha_0 + \sum_{i=1}^{N} k(x_j, x_i)(y'_i + f'_i + z_i),$$
$$\beta_j := \beta_0 + \sum_{i=1}^{N} k(x_j, x_i)(3 - y'_i - f'_i - z_i).$$

The mean and variance of ϕ_j remain the same as in (18).

Remark 2. *We note that the provided example does not correspond to the best approach for modeling the stated problem. It merely serves as a constructive example of adding more map layers. The general Bayesian inference [91], of course, can produce more accurate results; however, the motivation for the example is to maintain the closed-form Bayesian inference of each map layer.*

B. Limitations

There are several limitations in the current work. The current Bayesian map inference is based on a static-world assumption, i.e., $p(\mathcal{M}_t) = p(\mathcal{M}_{t-1})$, where t indicates the measurement timestamp. To deal with dynamic objects and environmental change, a prediction step $p(\mathcal{M}_t|\mathcal{M}_{t-1}, \mathcal{D})$ needs to be modeled given measurements from change detection or scene flow estimation. The work of [92] is an extension of [19] to dynamic environments using a simple autoregressive transition model for scene propagation in closed form. For a recent literature review on dynamic semantic mapping and scene flow estimation, we refer to [92] and references therein.

In addition, the current traversability notion is robot-agnostic, as we only use exteroceptive sensor data as the supervisory signal. For navigation and exploration purposes, a *dynamic cost* map layer could be learned from the robot proprioceptive or multi-modal sensory information, where the cost reflects dynamic traversability based on the current operating point and robot behavior [93]. The framework developed in this work provides the foundation for implementing these ideas in the future.

C. Reproducibility

The current results and maps are reproducible using the same datasets for two reasons. First, the traversability ground truth labels are generated using the provided data without any manual labeling. Second, the multilayer Bayesian map inference provides exact solutions without approximation. This framework can also be used on customized data collected using other robotic platforms as long as the image and depth information are included.

VII. Conclusion

This paper developed a multitask multilayer Bayesian mapping framework that uses a deep MTL network as a unified reasoning block. The proposed framework

1) provides multiple high-level measurements simultaneously,
2) learns map attributes other than semantics in a self-supervised manner, and
3) infers a multilayer dense map in closed form where inter-layer correlations are leveraged.

As a constructive example and a useful case, we specifically build a robotic map with semantic and traversability layers. Experimental results on publicly available datasets and data collected by our robot platform show the advantage of using an MTL network for multilayer mapping, and the performance improvement of traversability inference when the correlation with semantic layer is incorporated.

The proposed framework is highly extendable to include additional map layers containing greater detail, such as friction coefficient, affordance, dynamic planning cost, etc., to assist more sophisticated robotic behavior planning. The map is also continuous and contains uncertainty information, which is desirable for decision-making problems. We hope that the proposed mapping framework is adopted and extended to fulfill many advanced real-world robotic applications.

References

[1] S. Thrun *et al.*, "Robotic mapping: A survey," *Exploring artificial intelligence in the new millennium*, vol. 1, no. 1-35, p. 1, 2002. 1

[2] R. Dubé, A. Cramariuc, D. Dugas, H. Sommer, M. Dymczyk, J. Nieto, R. Siegwart, and C. Cadena, "Segmap: Segment-based mapping and localization using data-driven descriptors," *Int. J. Robot. Res.*, vol. 39, no. 2-3, pp. 339–355, 2020. 1

[3] A. Kim and R. M. Eustice, "Perception-driven navigation: Active visual SLAM for robotic area coverage," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2013, pp. 3196–3203. 1

[4] M. Ghaffari Jadidi, J. Valls Miro, and G. Dissanayake, "Gaussian processes autonomous mapping and exploration for range-sensing mobile robots," *Auton. Robot.*, vol. 42, no. 2, pp. 273–290, 2018. 1

[5] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989. 1

[6] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robot.*, vol. 34, no. 3, pp. 189–206, 2013. 1

[7] 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. 1, 7, 9

[8] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *IEEE International Symposium on Mixed and Augmented Reality*. IEEE, 2011, pp. 127–136. 1

[9] 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.* IEEE, 2017, pp. 1366–1373. 1

[10] E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers, "Real-time camera tracking and 3D reconstruction using signed distance functions." in *Proc. Robot.: Sci. Syst. Conf.*, vol. 2, 2013, p. 2. 1

[11] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments," in *Experimental robotics*. Springer, 2014, pp. 477–491. 1

[12] D. F. Wolf and G. S. Sukhatme, "Semantic mapping using mobile robots," *IEEE Trans. Robot.*, vol. 24, no. 2, pp. 245–258, 2008. 1

[13] M. Ghaffari Jadidi, L. Gan, S. A. Parkison, J. Li, and R. M. Eustice, "Gaussian processes semantic map representation," *arXiv preprint arXiv:1707.01532*, 2017. 1

[14] L. Gan, M. Ghaffari Jadidi, S. A. Parkison, and R. M. Eustice, "Sparse Bayesian inference for dense semantic mapping," *arXiv preprint arXiv:1709.07973*, 2017. 1

[15] 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*. IEEE, 2017, pp. 4628–4635. 1

[16] 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*. IEEE, 2020, pp. 1689–1696. 1

[17] M. Hiller, C. Qiu, F. Particke, C. Hofmann, and J. Thielecke, "Learning topometric semantic maps from occupancy grids," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2019, pp. 4190–4197. 1

[18] E. Zobeidi, A. Koppel, and N. Atanasov, "Dense incremental metric-semantic mapping via sparse Gaussian process regression," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2020, pp. 6180–6187. 1

[19] L. Gan, R. Zhang, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Bayesian spatial kernel smoothing for scalable dense semantic mapping," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 790–797, 2020. 1, 2, 7, 14, 15, 16

[20] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "SLAM++: Simultaneous localisation and mapping at the level of objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 1352–1359. 1

[21] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid, "Meaningful maps with object-oriented semantic mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2017, pp. 5079–5085. 1

[22] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, "Volumetric instance-aware semantic mapping and 3D object discovery," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3037–3044, 2019. 1

[23] J. J. Gibson, *The ecological approach to visual perception: classic edition*. Psychology Press, 2014. 2

[24] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016. 2

[25] F. Verdoja and V. Kyrki, "On the potential of smarter multi-layer maps," *arXiv preprint arXiv:2005.11094*, 2020. 2

[26] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: A survey," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, pp. 1373–1385, 2013. 2, 3

[27] B. Kuipers and Y.-T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," *Robot. and Auton. Syst.*, vol. 8, no. 1-2, pp. 47–63, 1991. 2

[28] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, vol. 99, no. 1, pp. 21–71, 1998. 2

[29] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller, "An atlas framework for scalable mapping," in *Proc. IEEE Int. Conf. Robot. and Automation*, vol. 2. IEEE, 2003, pp. 1899–1906. 2

[30] N. Tomatis, I. Nourbakhsh, and R. Siegwart, "Hybrid simultaneous localization and map building: a natural integration of topological and metric," *Robot. and Auton. Syst.*, vol. 44, no. 1, pp. 3–14, 2003. 2

[31] B. Kuipers, "The spatial semantic hierarchy," *Artificial intelligence*, vol. 119, no. 1-2, pp. 191–233, 2000. 2

[32] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J.-A. Fernandez-Madrigal, and J. González, "Multi-hierarchical semantic maps for mobile robotics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2005, pp. 2278–2283. 2

[33] H. Zender, O. M. Mozos, P. Jensfelt, G.-J. Kruijff, and W. Burgard, "Conceptual spatial representations for indoor mobile robots," *Robot. and Auton. Syst.*, vol. 56, no. 6, pp. 493–502, 2008. 2

[34] A. Pronobis and P. Jensfelt, "Large-scale semantic mapping and reasoning with heterogeneous modalities," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2012, pp. 3515–3522. 2

[35] K. Jiang, D. Yang, C. Liu, T. Zhang, and Z. Xiao, "A flexible multi-layer map model designed for lane-level route planning in autonomous vehicles," *Engineering*, vol. 5, no. 2, pp. 305–318, 2019. 2

[36] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese, "3D scene graph: A structure for unified semantics, 3D space, and camera," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 5664–5673. 3

[37] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone, "3D Dynamic Scene Graphs: Actionable Spatial Perception with Places, Objects, and Humans," in *Proc. Robot.: Sci. Syst. Conf.*, 2020. 3

[38] K. Zheng and A. Pronobis, "From pixels to buildings: End-to-end probabilistic deep networks for large-scale semantic mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2019, pp. 3511–3518. 3

[39] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone, "Kimera: From slam to spatial perception with 3D dynamic scene graphs," *Int. J. Robot. Res.*, vol. 40, no. 12-14, pp. 1510–1546, 2021. 3

[40] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception engine for 3D scene graph construction and optimization," *arXiv preprint arXiv:2201.13360*, 2022. 3

[41] J. Nieto, J. Guivant, and E. Nebot, "Denseslam: Simultaneous localization and dense mapping," *Int. J. Robot. Res.*, vol. 25, no. 8, pp. 711–744, 2006. 3

[42] P. Nordin and P. Degerman, "Multi layered maps for enhanced environmental perception," SAE Technical Paper, Tech. Rep., 2011. 3

[43] T. Zaenker, F. Verdoja, and V. Kyrki, "Hypermap mapping framework and its application to autonomous semantic exploration," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2020, pp. 133–139. 3

[44] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2016, pp. 1184–1189. 3, 7, 9

[45] P. Filitchkin and K. Byl, "Feature-based terrain classification for littledog," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2012, pp. 1387–1392. 3

[46] D. Stavens and S. Thrun, "A self-supervised terrain roughness estimator for off-road autonomous driving," *arXiv preprint arXiv:1206.6872*, 2012. 3

[47] M. A. Bekhti and Y. Kobayashi, "Regressed terrain traversability cost for autonomous navigation based on image textures," *Applied Sciences*, vol. 10, no. 4, p. 1195, 2020. 3

[48] C. A. Brooks and K. Iagnemma, "Self-supervised terrain classification for planetary surface exploration rovers," *J. Field Robot.*, vol. 29, no. 3, pp. 445–468, 2012. 3

[49] L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, and M. Hutter, "Where should I walk? predicting terrain properties from images via self-supervised learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1509–1516, 2019. 3

[50] J. Zürn, W. Burgard, and A. Valada, "Self-supervised visual terrain¨ classification from unsupervised acoustic feature learning," *IEEE Trans. Robot.*, 2020. 3

[51] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. R. Bradski, "Self-supervised monocular road detection in desert terrain." in *Proc. Robot.: Sci. Syst. Conf.*, vol. 38. Philadelphia, 2006. 4

[52] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun, "Learning long-range vision for autonomous off-road driving," *J. Field Robot.*, vol. 26, no. 2, pp. 120–144, 2009. 4

[53] D. Barnes, W. Maddern, and I. Posner, "Find your own way: Weakly-supervised segmentation of path proposals for urban autonomy," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2017, pp. 203–210. 4

[54] M. Broome, M. Gadd, D. De Martini, and P. Newman, "On the road: Route proposal from radar self-supervised by fuzzy lidar traversability," *AI*, vol. 1, no. 4, pp. 558–585, 2020. 4

[55] S. Ruder, "An overview of multi-task learning in deep neural networks," *arXiv preprint arXiv:1706.05098*, 2017. 4

[56] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997. 4

[57] S. Vandenhende, S. Georgoulis, W. Van Gansbeke, M. Proesmans, D. Dai, and L. Van Gool, "Multi-task learning for dense prediction tasks: A survey," *arXiv preprint arXiv:2004.13379*, 2020. 4, 9, 10, 13

[58] M. Crawshaw, "Multi-task learning with deep neural networks: A survey," *arXiv preprint arXiv:2009.09796*, 2020. 4

[59] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, "Multinet: Real-time joint semantic reasoning for autonomous driving," in *IEEE Intelligent Vehicles Symposium*. IEEE, 2018, pp. 1013–1020. 4

[60] Y. Yang and T. M. Hospedales, "Trace norm regularised deep multi-task learning," *arXiv preprint arXiv:1606.04038*, 2016. 4

[61] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 3994–4003. 4

[62] Y. Gao, J. Ma, M. Zhao, W. Liu, and A. L. Yuille, "Nddr-cnn: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 3205–3214. 4

[63] S. Liu, E. Johns, and A. J. Davison, "End-to-end multi-task learning with attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 1871–1880. 4, 5

[64] K.-K. Maninis, I. Radosavovic, and I. Kokkinos, "Attentive single-tasking of multiple tasks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 1851–1860. 4, 5

[65] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. Feris, "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 5334–5343. 4

[66] C. Rosenbaum, T. Klinger, and M. Riemer, "Routing networks: Adaptive selection of non-linear functions for multi-task learning," *arXiv preprint arXiv:1711.01239*, 2017. 4

[67] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 7482–7491. 4

[68] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *International Conference on Machine Learning*. PMLR, 2018, pp. 794–803. 4

[69] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," *arXiv preprint arXiv:1810.04650*, 2018. 4

[70] T. Standley, A. R. Zamir, D. Chen, L. Guibas, J. Malik, and S. Savarese, "Which tasks should be learned together in multi-task learning?" *arXiv preprint arXiv:1905.07553*, 2019. 4

[71] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 7132–7141. 5

[72] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," *arXiv preprint arXiv:1705.08045*, 2017. 5

[73] ——, "Efficient parametrization of multi-domain deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8119–8127. 5

[74] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2017. 5

[75] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3431–3440. 6

[76] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. European Conf. Comput. Vis.*, 2018, pp. 801–818. 6

[77] Z. Wu, C. Shen, and A. Van Den Hengel, "Wider or deeper: Revisiting the resnet model for visual recognition," *Pattern Recognition*, vol. 90, pp. 119–133, 2019. 6

[78] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. Newsam, A. Tao, and B. Catanzaro, "Improving semantic segmentation via video propagation and label relaxation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 8856–8865. 6, 9

[79] 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.* IEEE, 2012, pp. 3354–3361. 6, 9, 10

[80] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, 2018. 7, 9

[81] 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. Q. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. 7, 8

[82] A. Melkumyan and F. Ramos, "A sparse covariance function for exact gaussian process inference in large datasets." in *IJCAI*, vol. 9, 2009, pp. 1936–1942. 7

[83] T. Shan, J. Wang, B. Englot, and K. Doherty, "Bayesian generalized kernel inference for terrain traversability mapping,"

in *Conference on Robot Learning*, 2018, pp. 829–838. 7, 16

[84] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual SLAM," *arXiv preprint arXiv:2003.14338*, 2020. 9, 11

[85] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 3213–3223. 9, 10

[86] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, 2017. 10

[87] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian Conference on Computer Vision*, 2010. 10

[88] J.-K. Huang and J. W. Grizzle, "Improvements to target-based 3d lidar to camera calibration," *IEEE Access*, vol. 8, pp. 134 101–134 110, 2020. 14

[89] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *Int. J. Robot. Res.*, vol. 39, no. 4, pp. 402–430, 2020. 14

[90] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of michigan north campus long-term vision and lidar dataset," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2016. 15

[91] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006. 15, 16

[92] 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," *arXiv preprint arXiv:2108.03180*, 2021. 16

[93] L. Gan, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Energy-based legged robots terrain traversability modeling via deep inverse reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8807–8814, 2022. 16