Semantic-Aware Robotic Mapping in Unknown, Loosely Structured Environments

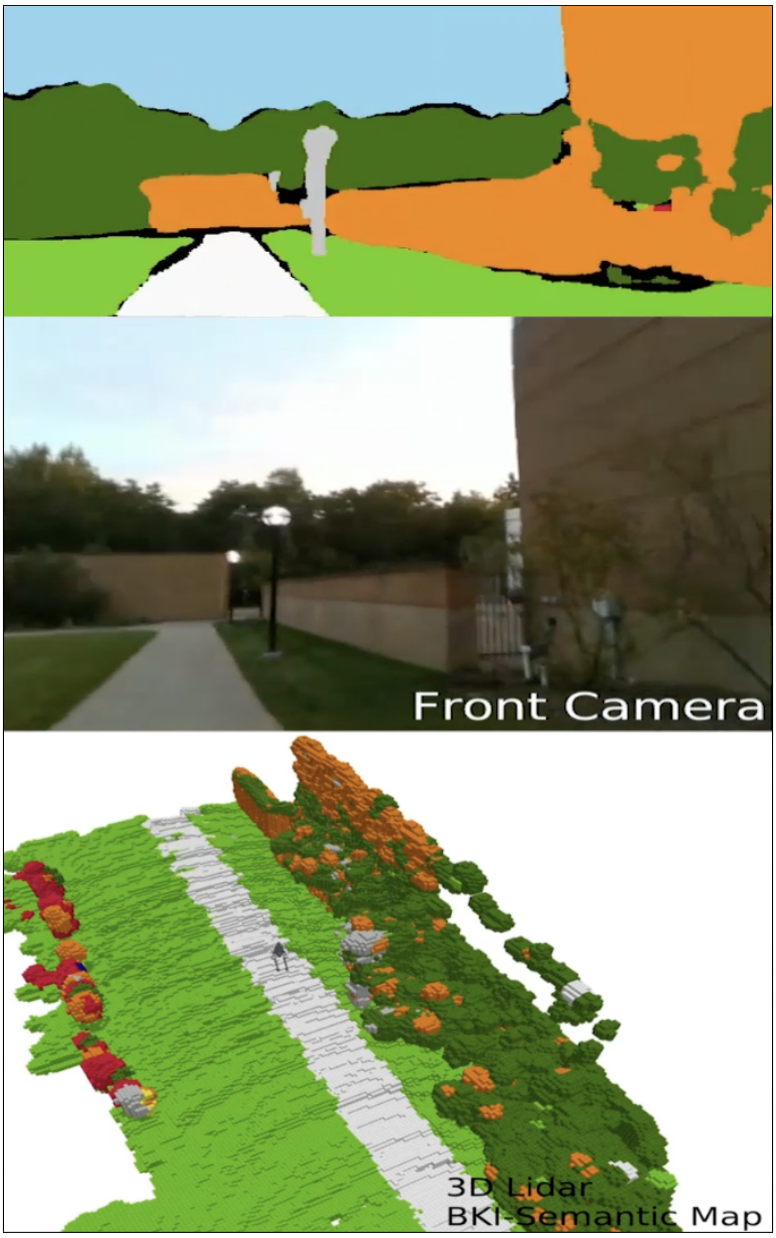
by

Lu Gan

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Robotics)
in the University of Michigan
2022

Doctoral Committee:

Professor Ryan M. Eustice, Co-Chair
Assistant Professor Maani Ghaffari, Co-Chair
Professor Jessy W. Grizzle, Co-Chair
Professor Odest Chadwicke Jenkins
Assistant Professor Katherine A. Skinner


Front Camera
3D Lidar
BKI-Semantic Map

Lu Gan

ganlu@umich.edu

ORCID iD: 0000-0003-0911-8032

© Lu Gan 2022

ACKNOWLEDGMENTS

"The doctoral tam is Aladdin's lamp which gives you anything you want." I still remember my mother saying this to me when I was little. At that time, I imagined the doctoral tam as a deep round bowl, and it would be filled with candies and toys whenever I asked. I can't help smiling when I think about this, as I realize how silly we were.

But still, I am getting the long-awaited "Aladdin's lamp" now. It has been quite a journey to stand here, full of loss and gain, joy and pain, hills and valleys. There have been too many things that I regret, but I am glad I have done this one thing right, and made it all worthwhile. I certainly would never have made it this far if not for so many people.

First of all, I am deeply grateful to my advisors, Prof. Ryan Eustice, Prof. Jessy Grizzle and Prof. Maani Ghaffari. Ryan, I would never forget the day when I received the Ph.D. offer with you being my advisor. I couldn't imagine how much confidence and expectation you were giving to me by giving chance to someone you had never met. My luck extended to my second year here when I had Jessy and Maani, and since then you have been giving me all of your guidance, support and encouragement. Jessy, I am surprised at and highly appreciate how much care you are giving to your students. You have more than 20000 citations but are as approachable as a friend. I still remember the assignment you gave to me when I had a low spirit is going out for a walk and sending you photos of the scene. Thank you for all of your support, your encouragement, and your unreasonably consistent compliments. I don't know the correct way to interpret them, but they worked for me. I would like to thank Maani for being my friend, advisor and role model. You might not even remember how much you have helped me over the past five years. I sincerely hope one day I could be the one like you for my students.

Next, I would like to thank my committee, Prof. Odest Chadwicke Jenkins and Prof. Katherine Skinner for valuable feedback and discussion on the current research and future directions. And Denise Edmund, thank you for always being so kind, helpful and patient. I am thankful to the early members of PeRL and Biped Robotics Lab, Jie Li, Arash Ushani, Josh Mangelson, Steven Parkison, Derrick Dominic, Ross Hartley, and Zhenyu Gan. Starting a new journey abroad was difficult, but you have made me feel welcome and helped me get through the transition. You are also so talented that I wish I were in a better situation to learn more from you. I would also like to acknowledge Cassie's perception team mem-

bers, Bruce Huang, Yukai Gong, and Ray Zhang. Together we spent countless hours on the Wave Field over 100°F getting Cassie to walk autonomously in a semantic map, providing one of the most important results presented in this thesis. The research will be outdated, the technology will be updated, but our memory of working together on a project like this will remain forever. Additionally, I would like to thank all the lab members, Omar Harib, Eva Mungai, Grant Gibson, Oluwami Dosunmu-Ogunbi, Minghan Zhu, Tribhi Kathuria, Cynthia Lin, Justin Lin, Sangli Teng, Bo Fu, Joseph Wilson, Aishwarya Unnikrishnan and Jingwei Song. More or less, we have spent time together working hard and having fun. I hope the friendship will live on. And Jinsun Liu, thanks for playing games together during a tough time.

Last but not least, I want to express my great appreciation to my parents. It is your love that motivates me to go this far and to keep going. Thank you for the education you have given to me. My mother always said, penny wise and pound foolish. On the other hand, if you are willing to take a loss, sometimes you might gain more than what you expected. I deeply believe this is the reason I have been helped by so many people in my life. Thank you for letting me know that there is something more important, even than the “Aladdin’s lamp”. That is being a good person.

TABLE OF CONTENTS

ACKNOWLEDGMENTS . . . . . **ii**

LIST OF FIGURES . . . . . **vii**

LIST OF TABLES . . . . . **xii**

ABSTRACT . . . . . **xiv**

CHAPTER

1 Introduction **1**

1.1 Motivation 1
1.2 Contributions 3
1.3 Outline 5

2 Gaussian Processes Semantic Map Representation **8**

2.1 Introduction 8
2.2 Related Work 11
2.3 Problem Statement and Preliminaries 12
2.3.1 Gaussian Processes Regression 13
2.3.2 Gaussian Processes Classification 13
2.3.3 Model Selection 14
2.3.4 Large-scale Inference 14
2.4 Gaussian Processes Semantic Map 15
2.5 Semantic OctoMap 17
2.6 Computational Complexity Analysis 18
2.7 Results and Discussion 19
2.7.1 Experimental Setup and Evaluation Criterion 19
2.7.2 Sparse and Missing Measurements Effects 20
2.7.3 Mapping under Noisy and Misclassified Labels 21
2.7.4 Discussion and Limitations 23
2.8 Conclusion and Possible Extensions 24

3 Compressed 3D Semantic Map Inference via Relevance Vector Machines . . . **25**

3.1 Introduction 25
3.2 Related Work 27
3.2.1 2D Segmentation Methods 27

3.2.2 3D Raw Point Cloud Segmentation Methods . . . 28
3.2.3 3D Voxel-based Semantic Map Fusion . . . 28
3.2.4 Semantic Mapping Refinement . . . 29
3.3 Problem Statement . . . 29
3.4 Compressed 3D Semantic Mapping Pipeline . . . 31
3.5 Results and Discussion . . . 32
3.5.1 Evaluation Metrics . . . 32
3.5.2 KITTI Dataset . . . 32
3.5.3 NCLT Dataset . . . 38
3.6 Conclusion and Future Work . . . 39

4 Bayesian Spatial Kernel Smoothing for Scalable Dense Semantic Mapping . . . 41

4.1 Introduction . . . 41
4.2 Related Work . . . 43
4.3 Preliminaries and Semantic Counting Sensor Model . . . 45
4.4 Continuous Semantic Mapping via Bayesian Kernel Inference . . . 46
4.5 Experimental Results . . . 49
4.5.1 KITTI Dataset . . . 49
4.5.2 SemanticKITTI Dataset . . . 52
4.5.3 Occupancy Evaluation . . . 54
4.5.4 Impact of Parameters . . . 55
4.5.5 Experimental Results on a Cassie Bipedal Robot . . . 55
4.6 Discussions and Limitations . . . 57
4.7 Conclusion . . . 57

5 Multi-Task Learning for Scalable and Dense Multi-Layer Bayesian Map Inference . . . 59

5.1 Introduction . . . 59
5.2 Related Work . . . 63
5.2.1 Multi-Layer Robotic Mapping . . . 63
5.2.2 Self-Supervised Learning for Traversability Estimation . . . 64
5.2.3 Deep Multi-Task Learning . . . 66
5.3 A Unified Segmentation Model for Multi-Layer Mapping via Multi-Task Learning . . . 67
5.3.1 Multi-Task Network with Attention Mechanisms . . . 68
5.3.2 MTL for Semantic and Traversability Segmentation . . . 70
5.3.3 Self-Supervised Traversability Labeling . . . 71
5.4 Multi-Layer Bayesian Map Inference . . . 73
5.4.1 Continuous Semantic Mapping . . . 74
5.4.2 Semantic-Traversability Mapping . . . 74
5.5 Experimental Evaluation . . . 78
5.5.1 Implementation Details . . . 78
5.5.2 Ablation Study . . . 79
5.5.3 KITTI Dataset . . . 81
5.5.4 TartanAir Dataset . . . 83

5.5.5 Cassie Robot Data . . . 86
5.6 Discussion on Multi-Task Multi-Layer Mapping Framework . . . 87
5.6.1 Extensions to More Layers . . . 88
5.6.2 Limitations . . . 89
5.6.3 Reproducibility . . . 89
5.7 Conclusion . . . 90

6 Energy-based Legged Robots Terrain Traversability Modeling Using Deep Inverse Reinforcement Learning . . . 91

6.1 Introduction . . . 91
6.2 Related Work . . . 94
6.2.1 Terrain Traversability Modeling for Legged Robots . . . 94
6.2.2 IRL for Traversability Cost Learning . . . 94
6.3 Methodology . . . 95
6.3.1 Problem Statement . . . 95
6.3.2 Maximum Entropy Deep IRL . . . 96
6.3.3 Learning Inertial Features: Network Architecture . . . 96
6.3.4 Trajectory-ranked MEDIRL . . . 98
6.3.5 Locomotion Energy Ranked Reward Extrapolation . . . 100
6.4 Experiments . . . 100
6.4.1 Legged Robot Walking Dataset . . . 100
6.4.2 Inertial Feature Learning Evaluation . . . 102
6.4.3 Energy-based Reward Extrapolating Evaluation . . . 105
6.5 Conclusion . . . 107

7 Future Research Directions . . . 108

BIBLIOGRAPHY . . . 110

LIST OF FIGURES

2.1 The figure shows examples of the GPSM (bottom right) and SOM (bottom left) using NYU Depth V2 dataset. The point cloud with ground truth semantic labels is shown in the top right. The maps are built by uniformly down-sampling the original point cloud to one-third. The GP map can infer the missing labels and deal with sparse measurements. The query points are the same as original points in the top right. 9

2.2 A two-dimensional toy example of GP multi-class classification using the synthetic dataset shown in the right plot. The prediction is continuous, and the class probabilities can be queries at any point in the plane as shown in the left plot. Furthermore, the contour curves can be seen as decision boundaries. The plots show the log probabilities. 16

2.3 Semantic OctoMap built using the Stanford 2D-3D-Semantics Dataset [1], area 1, conference room 1. The semantic label observations are from the ground truth data. From the figure, the effect of discretization of the map and independent voxels inference are evident in the structural shape of the environment. 18

2.4 The figure shows GPSM and SOM results using sparse labeled point clouds. From left, each column respectively shows the RGB image, the point cloud with ground truth semantic labels before downsampling, the SOM, and the GPSM. The maps are built by uniformly down-sampling the original point cloud to one-third. The GP map can infer the missing labels and deal with sparse measurements through leaning the correlation between observations. For GPSM, the query points are the same as original points in the second column. 20

2.5 The results of mapping under noisy and misclassified labels. From left, each column respectively shows the RGB image, the image segmentation using SegNet, the SOM, and the GPSM results. The false positives in the image segmentation are directly transferred into the SOM results, while the GPSM uses the spatial distance between points to infer the class labels based on correlations between map points. 22

3.1 The semantic mapping and fusion pipeline of RVSM, and a comparison with the baseline [2]. Although the proposed system shares similar properties of the baseline such as final voxel map representation and Bayesian updates per semantic class, it consists of extra sparse model building step which allows for map compression and continuous queries at any desired locations. 26
3.2 Example of KITTI Sequence 15 after projecting the occupied voxels from the semantically fused grid map to the camera image plane: From the top down are RGB image, prior result, CRF [2] result and RVSM result, respectively. RVSM is able to capture the traffic sign in the left image, and the pole in the right image. 34
3.3 Query for different resolution. From (a) to (d): map resolution changes from $0.1\ \mathrm{m}$ to $0.4\ \mathrm{m}$. 35
3.4 mIoU at different resolutions for Prior, CRF, and RVSM. RVSM's only does one query on the prior points' spatial coordinates and generates the map for all the resolutions. Here the resolution means the size of the voxel. When the resolution decreases, more points (possibly with different labels) fall into one voxel, resulting in a lower mIoU value. The larger the voxels are, the more loss of information comes from CRF's discretization. On the contrary, RVSM performs compression before voxel map fusion, which preserves local semantic consistency better as opposed to post-processing. 36
3.5 RVSM semantic map fusion on NCLT subsequence 2012-04-29. Left: The satellite view of the path from Google Earth. Right: Semantic OctoMap fusion result after querying from RVSM. 39

4.1 Qualitative results on KITTI odometry sequence 05 dataset [3]. From top to bottom the figures show the 2D ground truth image, 3D semantic map, and variance map. 42
4.2 3D toy example on a simulated dataset. (a) Environment model in Gazebo. (d) Annotated point cloud raw data. (b), (e) Semantic and variance map of S-CSM. (c), (f) Semantic and variance map of S-BKI. Variance maps of two models (shown using the jet colormap) provide useful information for robotic navigation and exploration [4]. We found that Bayesian kernel inference decreases the variance of the wall by considering neighboring measurements. There are some artifacts, however, on the periphery of the wall where the variance is relatively high. 48
4.3 Qualitative results on KITTI odometry sequence 05 dataset [3]. From (a) to (d) the figures show 2D projected images from Yang et al. [2], BGKOctoMap-CRF, S-CSM and S-BKI, respectively. The projected image from Yang's semantic map contains more gaps than other maps, compared with the ground truth image where the road, buildings, and vegetation are continuous and dense, while the projected image of S-BKI has the least holes in those regions, which resembles the ground truth better. BGKOctoMap-CRF outperforms Yang's method, in spite of the misclassification of the sidewalk to road. . 51
4.4 S-BKI map of a conference room in Area 3 of Stanford 2D-3D-Semantics Dataset [1]. 54

4.5 Impact of parameters on mapping performance for KITTI dataset sequence 15 [5] (stereo camera) and SemanticKITTI dataset sequence 04 [6] (LiDAR). Only one parameter at a time is varied while the others are kept at the values in Table 4.1. Both figures show reasonable robustness to the parameter variations. 55
4.6 Top Left: Cassie Blue has a custom designed torso on which is mounted an Intel RealSense depth camera capable of providing both RGB images and corresponding organized point clouds in outdoor environments. Top right: Google satellite map of the Wave Field of the University of Michigan - North Campus. Bottom: From left to right are the 3D and 2D views of S-BKI map. While the robot is navigating along the sidewalk, S-CSM produces discontinuous semantic maps from sparse sensor measurements, which may cause the robot's planner to regard the gaps in the map as unwalkable areas, a practical problem when we conduct autonomous walking experiments with Cassie Blue (a video of the experiment is available at https://youtu.be/uFyT8zCg1Kk). S-BKI model produces a continuous and smooth map, where gaps are assigned with labels inferred from local correlations in the map. 56

5.1 Overview of our proposed multi-task multi-layer Bayesian mapping framework. We first employ exteroceptive sensor data and robot 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 multi-task learning (MTL) using available data. Predictions of the MTL network are processed in our multi-layer Bayesian inference sequentially and efficiently. Inter-layer correlations are also leveraged to improve the map posterior. Our framework is designed to be easily extended to include additional layers. 61
5.2 Space decomposition of self-supervised traversability learning methodologies according to the type of supervisory signals employed, namely, proprioceptive, geometry-based and semantic-based approaches, the latter two comprising the domain of exteroceptive approaches. 65
5.3 Network architecture of each task using an illustrative backbone. Task-specific parameters consist of Residual Adapter (RA) and Squeeze-and-Excitation (SE) modules in residual blocks of the shared encoder and the entire decoder. This architecture can be built upon any feed-forward neural network with residual blocks and trained end-to-end. 69
5.4 Detailed visualization of our MTL network based on DeepLabv3+ architecture with WideResNet38 backbone. Semantic segmentation task (purple) and traversability segmentation task (blue) have their own set of attention modules attached to the shared encoder (orange), and their own decoders. All task-specific modules have the same design except the last convolutional layer (the number of output channels depends on the number of classes in that task), although their weights are individually learned. 71

5.5 Our self-supervised traversability labeling pipeline in which exteroceptive sensor 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 trajectories. A 2D traversability map is then estimated from the elevation map and projected into images. The lower part considers semantics; We use the predicted semantic labels to filter out the noises in projected traversability images. Combining both geometry-based and semantic-based self-supervised approaches, the final traversability ground truth images can be generated automatically and accurately. 72
5.6 3D qualitative results of our multi-layer Bayesian mapping algorithm on KITTI dataset. (a)-(c) are the semantic layer, traversability layer and semantic-traversability layer, respectively. (d)-(f) are the corresponding uncertainty maps showing variance value of each voxel using Jet colormap. We can see the improved estimation in (c) compared with (b), and reduced variance in (f) compared with (e) by leveraging semantic layer in traversability inference. Relatively high variance values are also observed for the voxels with estimation error. 82
5.7 2D quantitative results of our multi-layer Bayesian mapping algorithm on KITTI dataset. (a) Input color images. (b) Projected images from traversability mapping. (c) Projected images from semantic-traversability mapping. (d) Traversability ground truth images generated automatically by the method described in Section 5.3.3. We can see the improved estimation in (c) compared with (b) based on the ground truth (d). 82
5.8 Qualitative results of our multi-task learning (MTL) model compared with single-task learning (STL) network on TartanAir dataset. (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 by the method described in 5.3.3. We can see that (c) and (f) have better segmentation performance compared with (b) and (e). The improvement on both tasks obtained for the same input image also shows the correlation between these tasks. 84
5.9 Qualitative results of our multi-layer Bayesian mapping algorithm on TartanAir dataset. (a)-(c) are the semantic layer, traversability layer and semantic-traversability layer, respectively. (d)-(f) are the corresponding uncertainty maps showing the variance value of each voxel using Jet colormap. 85
5.10 The Cassie Blue robot platform equipped with a depth camera and LiDAR sensor. We use the LiDAR point clouds to build a 2D local traversability map based on geometric information. We then project traversability scores into the image plane to generate traversability ground truth labels automatically. . . . 86
5.11 Qualitative results of our multi-layer mapping algorithm on data collected by Cassie Blue. The left shows the color image collected by the on-board camera, which is used for semantic and traversability segmentation. The middle shows the semantic layer built up to the timestamp of the left image. The right shows the traversability layer. 87

5.12 Qualitative results of our multi-layer mapping algorithm on data collected by Cassie Blue. The left column shows the semantic layer and the corresponding uncertainty map. The right column shows the traversability layer and its uncertainty map. 87

6.1 The proposed deep reward network architecture and training process for terrain traversability cost learning. The network has two stages where the first stage is for feature learning and the second stage is for feature fusion (yellow). To extract useful features from both proprioceptive and exteroceptive sensory data, the first stage consists of an inertial branch (blue) taking inputs of raw IMU signals and a UNet-like environmental branch (pink) taking inputs of the elevation map, elevation variance map and color map. The outputs of both branches are concatenated and fused in the second stage to approximate the reward map. The network is trained in an end-to-end manner by backpropagating both MEDIRL gradient (green flow) and trajectory ranking loss (red flow). 97
6.2 MIT Mini-Cheetah robot with a customized sensor suite and satellite map of the data collection environment for this work. 101
6.3 Visualization of the generated data/network inputs in different environments, for instance, sidewalk, trail and forest. (a) Elevation map. (b) Elevation variance map. (c) Color map. The demonstrated past (red) and future (cyan) trajectories are overlaid for visualization only and not part of the feature map. (d) Third-person view of the current state (not an input). (e) The IMU signals collected within 0.5 sec. 102
6.4 Qualitative results on traversability cost learning and future trajectory prediction in Experiment 1. (a)-(c) are the inferred reward maps of the baseline method [7], our method without inertial feature learning and our method with inertial feature learning, respectively. (d)-(f) are the SVF visualizations of the three methods representing the future trajectory distributions. (g) are the corresponding input color maps overlaid with past (red) and future (cyan) trajectories. 103
6.5 Left block: Qualitative results on traversability cost learning and future trajectory prediction in Experiment 2. From left, the first and second columns are the inferred reward maps and future trajectory distributions obtained using the MEDIRL algorithm. The third and fourth columns are those obtained using our proposed T-MEDIRL algorithm with energy-based trajectory ranking. The fifth column is the corresponding input color maps overlaid with past (red) and future (cyan) trajectories. Right block: Mini-Cheetah Simulator. The bottom row shows the robot is following the optimal future trajectory. 105

LIST OF TABLES

2.1 The comparison of Gaussian processes semantic map and semantic OctoMap using the area under the receiver operating characteristic curve (AUC). The maps are built using downsampled point clouds from the NYU Depth V2 dataset [8] and the corresponding frame numbers are shown in the table. 21

2.2 The comparison of Gaussian processes semantic map and semantic OctoMap using the area under the receiver operating characteristic curve (AUC). The maps are built using noisy image segmentation labels produced by SegNet. The figures for both methods are lower than the first experiment due to the presence of false positive labels in the observations. 23

3.1 KITTI Experiment RVM Model Selection: We run sequential RVM inference for 500 iterations per class on 0.01 of the training data. The current kernel and hyperparameters are chosen by using a sample data and training a Gaussian process [9]. 33

3.2 Quantitative results on KITTI odometry sequence 15 test set [5] for 7 semantic classes, containing 25 ground truth images. The query points of RVSM are chosen to be at all the observed prior locations. 36

3.3 KITTI sequence 15 compression test for the frame 0-30. The mIoU is evaluated on the 15 ground truth images. In the compression experiment, query points are the centroids of the occupancy grid. Priors are discarded after the relevance vectors are obtained, thus RVSM does not have access to prior locations, but only to the positions of centroids in the occupancy grid. 37

3.4 NCLT Experiment RVM Model Selection: We run sequential RVM inference for 1000 iterations per class on 0.05 of the training data. The current kernel and hyperparameters are chosen by using a sample data and training a Gaussian process [9]. 38

4.1 Kernel and Dirichlet prior hyperparameters for all experiments. 49

4.2 Quantitative results on KITTI odometry sequence 05 test set [3] for 8 common semantic classes, containing 40 images. 52

4.3 Quantitative results on KITTI odometry sequence 15 test set [5] for 8 common semantic classes, containing 25 images. 52

4.4 Mean IoU on SemanticKITTI dataset sequence 00-21 [6] for 19 semantic classes. SqueezeSegV2-kNN (Sq.-kNN). Darknet53-kNN (Da.-kNN). Training (00-07, 09-10), Validation (8), Test (11-21). 53

4.5 Comparison of map quality using the Area Under ROC Curve (AUC) and runtime of the four methods on the example shown in Fig. 4.4. 54

5.1 Ablation study of deep multi-task learning network for semantic and traversability segmentation. Mean intersection over unions (mIoUs) of 19 semantic classes and 2 traversability classes are reported on a validation set from KITTI sequence 06-07. Multi-task performance $\Delta_{\texttt{MTL}}$ indicates the average relative improvement over the single-task baseline (see (5.17)). 80
5.2 Quantitative results of our system on KITTI data set using intersection over unions (IoUs) for traversability (trav.) classification. 81
5.3 Quantitative results of our multi-task learning model on TartanAir dataset. Mean intersection over unions (mIoUs) of 30 semantic classes and 2 traversability classes, and multi-task performance ($\Delta_{\texttt{MTL}}$) are reported. 84
5.4 Quantitative results of our multi-layer Bayesian mapping algorithm on TartanAir data set using intersection over unions (IoUs) for traversability (trav.) classification. 85

6.1 Traversability Cost Learning/Future Trajectory Prediction Performance Quantitative Comparison in Experiment 1. 104
6.2 Traversability Cost Learning/Future Trajectory Prediction Performance Quantitative Comparison in Experiment 2. Using a $24\text{V}, 6\text{Ah}$ battery, the difference shown below in AEC corresponds to about seven minutes extra operation. . . . 106

ABSTRACT

Robotic mapping is the problem of inferring a representation of a robot's surroundings using noisy measurements as it navigates through an environment. As robotic systems move toward more challenging behaviors in more complex scenarios, such systems require richer maps so that the robot understands the significance of the scene and objects within. This dissertation focuses on semantic-aware robotic mapping in unknown, loosely structured environments. The first main contribution is a Bayesian kernel inference semantic mapping framework that formulates a unified probabilistic model for occupancy and semantics, and provides a closed-form solution for scalable dense semantic mapping. This framework significantly reduces the computational complexity of learning-based continuous semantic mapping and achieves high accuracy in the meantime. Next, a novel and flexible multi-task multi-layer Bayesian mapping framework is proposed to provide even richer environmental information. A two-layer robotic map of semantics and traversability is built as a strong example. Moreover, it is readily extendable to include more layers according to needs. Both mapping algorithms were verified using publicly available datasets or through experimental results on a Cassie-series bipedal robot. Finally, instead of modeling the terrain traversability using metrics defined by domain knowledge, an energy-based deep inverse reinforcement learning method is developed to learn the traversability from demonstrations. The proposed method considers robot proprioception and can learn reward maps that lead to more energy-efficient future trajectories. Experiments are conducted using a dataset collected by a Mini-Cheetah robot in a campus environment with various scenes.

CHAPTER 1

Introduction

1.1 Motivation

The problem of simultaneously constructing a model of the environment (a *map*), and estimating the state of the robot moving within it, is known as simultaneous localization and mapping (SLAM). SLAM is generally regarded as one of the most important problems in the pursuit of full robotic autonomy. Over the last 30 years, astonishing progress has been made in this area. This not only allows the successful transition of SLAM from lab research to industrial applications (e.g., self-driving cars, surveillance drones and agricultural robots), but also benefits many branches of related research where SLAM is regarded as a must-have functionality of a robot.

With the significant progress in SLAM, debates about whether *SLAM is solved* have been arisen. In this regard, leaders in the area gave their answers in the seminal paper [10]. They argue that we are entering a third era for SLAM, the *robust-perception age*, where *high-level understanding* is one of the key requirements: the SLAM system goes beyond basic geometric reconstruction to obtain a high-level understanding of the environment (e.g., high-level geometry, semantics, physics, affordances) [10].

Mapping is the most fundamental and direct way for robots to understand the world. In SLAM, the map, estimated together with robot poses, is usually represented as sparse or semi-dense point clouds. These types of robotic maps allow limiting the error committed in estimating the robot states by adding more constraints in the optimization, for instance, sensor observations and loop closure factors in a factor graph [11]. However, they are limited in enriching the robot's understanding of the world, and supporting other tasks, such as path planning or environment exploration. This is where mapping with known poses plays a role.

Mapping with known poses, referred to as robotic mapping in this dissertation, deals with the problem of mapping an $a\ priori$ unknown environment using a robot equipped

with on-board sensors, given the robot states. Traditional robotic mapping algorithms can provide dense geometric information of the environment to facilitate robot navigation. For instance, occupancy grid mapping [12] discretizes the 2D or 3D space into 2D cells or 3D voxels where each cell/voxel contains the probability of it being occupied. This map allows the robot to detect the obstacles and determine a feasible path within it probabilistically. However, for robots to navigate and interact more richly with the world around them, they will likely require a deeper understanding of the world beyond geometry.

This research is thus motivated with a simple question: how do we design a mapping system that allows a robot to have a deeper understanding of the world? This question is not easy to answer at once. It is strongly tied to the question of *what* information of the environment the robot needs to know and *how* to represent that information in a map. Then we need to answer the questions of how to build the map accurately and efficiently to enable real-time robotic applications, how to scale the mapping system to a large environment, and further how to achieve lifelong learning, etc.

To answer the first and the easiest question, and also just to take one step further from the current geometric robotic maps, we think that a robot needs to know *what the object is* in addition to *where the object is* in its environment. A simple application is that a service robot needs to know which part in its map is a kitchen and what is a cup to perform the task of "grab a cup from the kitchen". In robotics and related research fields, the study of this level of understanding is often referred to as *semantics*, which dictates *what does the world "mean" to a robot conceptually* [13].

Integrating semantics in a map, also known as semantic mapping, will open new paths for robot navigation. In traditional robotic maps where the environment can only be interpreted as a series of accessible and inaccessible areas, robot navigation is usually based on "find a feasible/shortest path from point A to point B". In contrast, in a semantic map, the robot is able to plan a path to a place/object instead of fixed coordinates, avoid some places/objects, and perform some tasks at the same time. This is a whole new level of navigation, and closer to how humans navigate.

In this dissertation, we focus on semantic-aware robotic mapping in unknown, loosely-structured environments. We consider the term semantics in a robotics context to be about the *meaning* of things: the meaning of places, objects, other entities occupying the environment. In other words, everything that occupies an environment should have a meaning to the robot. To this end, we first propose a Gaussian processes framework for semantic mapping in Chapter 2 as a means to integrate semantic information into the map representation. Concerning about the large amount of memory usage for storing a semantic map, a semantic map compression method using Bayesian inference is proposed in Chapter 3.

In Chapter 4, a Bayesian spatial kernel smoothing framework for scalable dense semantic mapping is proposed and reduces the computational complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(\log n)$. To provide even richer information beyond semantics, a multi-task multi-layer Bayesian mapping framework is proposed in Chapter 5. Finally, an energy-based legged robots terrain traversability modeling framework using deep inverse reinforcement learning (IRL) is presented in Chapter 6.

1.2 Contributions

The main contributions of this dissertation are as follows:

- A high-dimensional Gaussian processes semantic map representation is proposed. The map representation is the natural extension of Gaussian processes occupancy maps from binary to multi-class form. Given a point cloud measurement that is (possibly partially) assigned with noisy semantic class labels, the framework can infer a semantic map representation of the point cloud as a Gaussian process. The map can infer missing labels and deal with sparse measurements through learning the correlation in data. It is also continuous, and can be inferred with any resolution.

- A compressed 3D semantic map representation and its inference pipeline is proposed. To reduce the memory consumption and enable large-scale mapping, we propose to use a sparse Bayesian model, in particular, the relevance vector machine, for semantic map compression. Specifically, we represent the semantic map as a set of parameters and basis functions in the parametric statistical model. The proposed semantic map representation has comparable prediction performance with the state-of-the-art semantic mapping techniques, while requiring a lower memory footprint.

- A Bayesian continuous 3D semantic occupancy mapping framework is developed that statistically models the occupancy and semantic probabilities in a unified framework. By exploiting Bayesian kernel inference which is an approximation of Gaussian processes, this framework reduces the computational complexity of $\mathcal{O}(n^3)$ in Gaussian processes semantic map to $\mathcal{O}(\log n)$, where n is the number of training points (sensor measurements). The mapping framework can be run at 2 Hz on a laptop CPU, and enables online robot planning using the semantic information of the map.

- A novel multi-task multi-layer Bayesian map framework which introduces flexible and readily expendable attribute map layers is proposed. We show a strong example

of the framework where a two-layer map for semantics and traversability is built. This framework uses deep multi-task learning for sensor measurements processing, and leverages inter-layer correlations in the closed-form Bayesian map inference. The framework can be easily extended to include more map layers to provide richer information of the environment. In addition, a self-supervised labelling pipeline for traversability is proposed to automatically generate training data.

- A deep inverse reinforcement learning method for legged robots terrain traversability modeling is proposed. Instead of only using exteroceptive sensory data to learn a robot-agnostic reward function, we incorporate robot proprioception as inputs and learn a reward function that depends on robot states during deployment. We design a deep reward network that can learn robot inertial features directly from raw sensor measurements and show the improvement over handcrafted kinematic features. We further propose to train the network using maximum entropy inverse reinforcement learning algorithm with an additional trajectory ranking loss to deal with the demonstration suboptimality for legged robots. We use the average locomotion energy consumption for trajectory ranking to learn a reward function that leads to more energy-efficient path planning.

The work in this thesis has appeared (or will appear) in the following publications:

- Maani Ghaffari Jadidi, **Lu Gan**, Steven A. Parkison, Jie Li, Ryan M. Eustice. *Gaussian processes semantic map representation*. In RSS 2017 Workshop on Spatial-Semantic Representations in Robotics.

- **Lu Gan**, Ray Zhang, Jessy W. Grizzle, Ryan M. Eustice, and Maani Ghaffari Jadidi. *Bayesian spatial kernel smoothing for scalable dense semantic mapping*. In IEEE Robotics and Automation Letters (RA-L), Vol. 5, No. 2, April 2020, pp. 790-797.

- **Lu Gan**$*$, Youngji Kim$*$, Jessy W. Grizzle, Jeffrey M. Walls, Ayoung Kim, Ryan M. Eustice, and Maani Ghaffari. *Multi-task learning for scalable dense multi-layer Bayesian map inference*. Second-round review. IEEE Transactions on Robotics (T-RO).

- **Lu Gan**, Jessy W. Grizzle, Ryan M. Eustice, and Maani Ghaffari. *Energy-based legged robots terrain traversability modeling via deep inverse reinforcement learning*. Under review. IEEE Robotics and Automation Letters (RA-L) and 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Other related publications that do not directly include the work presented in this thesis:

- Steven A. Parkison, **Lu Gan**, Maani Ghaffari Jadidi, and Ryan M. Eustice. *Semantic iterative closest point through Expectation-Maximization*. In 2018 British Machine Vision Conference (BMVC), p. 280. 2018

- Ross Hartley, Josh Mangelson, **Lu Gan**, Maani Ghaffari Jadidi, Jeffrey M. Walls, Ryan M. Eustice, and Jessy W. Grizzle. *Legged robot state-estimation through combined forward kinematic and preintegrated contact factors*. In 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–8. IEEE, 2018.

- Ross Hartley, Maani Ghaffari Jadidi, **Lu Gan**, Jiunn-Kai Huang, Jessy W. Grizzle, and Ryan M. Eustice. *Hybrid contact preintegration for visual-inertial-contact state estimation using factor graphs*. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3783–3790. IEEE, 2018.

- Steven A. Parkison, Maani Ghaffari, **Lu Gan**, Ray Zhang, Arash K. Ushani, and Ryan M. Eustice. *Boosting shape registration algorithms via reproducing kernel Hilbert space regularizers*. In IEEE Robotics and Automation Letters (RA-L), Vol. 4, No. 4, October 2019, pp. 4563-4570.

Open source implementation of the proposed algorithms is provided as follows:

- https://bitbucket.org/perl-sw/rvsm
- https://github.com/ganlumomo/BKISemanticMapping
- https://github.com/ganlumomo/semantic-segmentation
- https://github.com/ganlumomo/MultilayerMapping
- https://github.com/ganlumomo/traversability_projection_ws
- https://github.com/ganlumomo/minicheetah-traversability-irl

1.3 Outline

This dissertation proposal is organized into 7 chapters, and structured as follows:

Chapter 2 is our first attempt to incorporate semantic information in a robotic map. We naturally extend Gaussian processes occupancy maps from binary (occupied vs. unoccupied) to multi-class (semantic classes pre-defined for the environment) form. We first give a brief introduction of Gaussian processes and GP classification, then formulate the semantic

mapping problem as a multi-class GP classification problem using a *one-vs.-rest* approach. In experiments, we extend the OctoMap [14] to semantic OctoMap for comparison. Experimental results on single RGBD scans show the GP semantic map can handle sparse measurements, missing labels in the point cloud, as well as noise corrupted labels.

In Chapter 3, we propose a compressed 3D semantic map representation that exploits sparse Bayesian modeling and a pipeline to build the compressed semantic map. We compare the prediction performance with other 3D semantic mapping systems and show the continuity and compression properties of the proposed map representation. The experiments show our compressed semantic map has comparable prediction performance with the state-of-the-art techniques while requiring a lower memory footprint. The proposed semantic mapping system can be used in robotic applications where the semantic prediction needs to be made efficiently with limited onboard resources.

In Chapter 4, we focus on deriving a closed-form Bayesian semantic mapping framework that can take multi-scan sensor measurements sequentially and efficiently to build a full semantic map for robot navigation. We first extend a counting sensor model for occupancy mapping to a semantic counting sensor model where the Categorical and Dirichlet distribution (conjugate prior of the former) is used to model the semantic likelihood and its parameters, respectively. Then, we leverage Bayesian kernel inference with a sparse kernel to make the model continuous. It is an approximation to Gaussian processes with only $\mathcal{O}(\log n)$ computations. Experiments on publicly available datasets KITTI and SemanticKITTI show the proposed framework outperforms the state-of-the-art semantic mapping methods. An autonomy experiment on our bipedal robot Cassie Blue also shows the mapping efficiency and ability to facilitate robot planning.

In Chapter 5, we want to extend the Bayesian semantic mapping framework in Chapter 4 to provide richer information of the environment in addition to semantics. We propose a novel and flexible multi-task multi-layer Bayesian mapping framework in this chapter. And as an example, we show how to build a two-layer Bayesian map of semantics and traversability. We first design a deep multi-task learning neural network with attention mechanisms as a unified reasoning block for our mapping system. This network can output semantic and traversability prediction for a monocular image simultaneously. To train the network, a self-supervised approach to automatically generate the traversability ground truth labels is also proposed. Finally, we formulate a closed-form Bayesian inference model for a semantic-traversability layer which leverages inter-layer correlations. We present extensive experimental results on publicly available datasets and the data collected by our bipedal robot Cassie Blue. A discussions on how to extend the proposed framework to include more layers is given at the end of this chapter.

Built on the self-supervised traversability estimation in Chapter 5, Chapter 6 proposes an energy-based terrain traversability modeling method using deep inverse reinforcement learning. The motivation is to build robot-specific traversability cost map by learning from demonstration. Two issues have been mainly considered. First, unlike vehicles, legged robots with high mobility tend to have more sudden motions. Existing deep IRL works using purely environmental features or handcrafted vehicle kinematic features might fail to capture these sudden movements. We propose a deep reward network that can effectively learn robot inertial features from proprioceptive sensory data and integrate the learned features into the reward map. Secondly, we propose a Trajectory-ranked Maximum Entropy Deep Inverse Reinforcement Learning algorithm and locomotion trajectory ranking strategy using energy to deal with the suboptimality of legged robot demonstrations. By introducing a preference label for each demonstrated trajectory, the inferred reward map can extrapolate beyond suboptimal demonstrations and lead to a better-than-demonstration policy. We conduct the experiments using a dataset collected by Mini-Cheetah robot and in a robot simulator.

Finally, Chapter 7 outlines several main ideas for future research, based on the findings in this dissertation and my vision of future directions for robotic perception and learning.

CHAPTER 2

Gaussian Processes Semantic Map Representation

In this chapter, we develop a high-dimensional map building technique that incorporates raw pixelated semantic measurements into the map representation. The proposed technique uses Gaussian Processes (GPs) multi-class classification for map inference and is the natural extension of GP occupancy maps from binary to multi-class form. The technique exploits the continuous property of GPs and, as a result, the map can be inferred with any resolution. In addition, the proposed GP Semantic Map (GPSM) learns the structural and semantic correlation from measurements rather than resorting to assumptions, and can flexibly learn the spatial correlation as well as any additional non-spatial correlation between map points. We extend the OctoMap to Semantic OctoMap representation and compare with the GPSM mapping performance using NYU Depth V2 dataset. Evaluations of the proposed technique on multiple partially labeled RGBD scans and labels from noisy image segmentation show that the GP semantic map can handle sparse measurements, missing labels in the point cloud, as well as noise corrupted labels.

2.1 Introduction

Semantic knowledge in robotic perception systems can use representations such as hierarchical maps [15], objects as higher level landmarks [16], or voxelized reconstruction [17]. Dense robotic maps such as occupancy grids and the OctoMap [14, 18–21] traditionally contain geometric knowledge of the environment. Grid/voxel-based maps have been successful in many applications such as localization, robotic exploration, and navigation tasks [22–25]. However, these techniques ignore available correlations in data by simplifying the mapping problem into a set of marginalized random variables. Furthermore, the map resolution is often fixed or limited and once the map is inferred the resolution

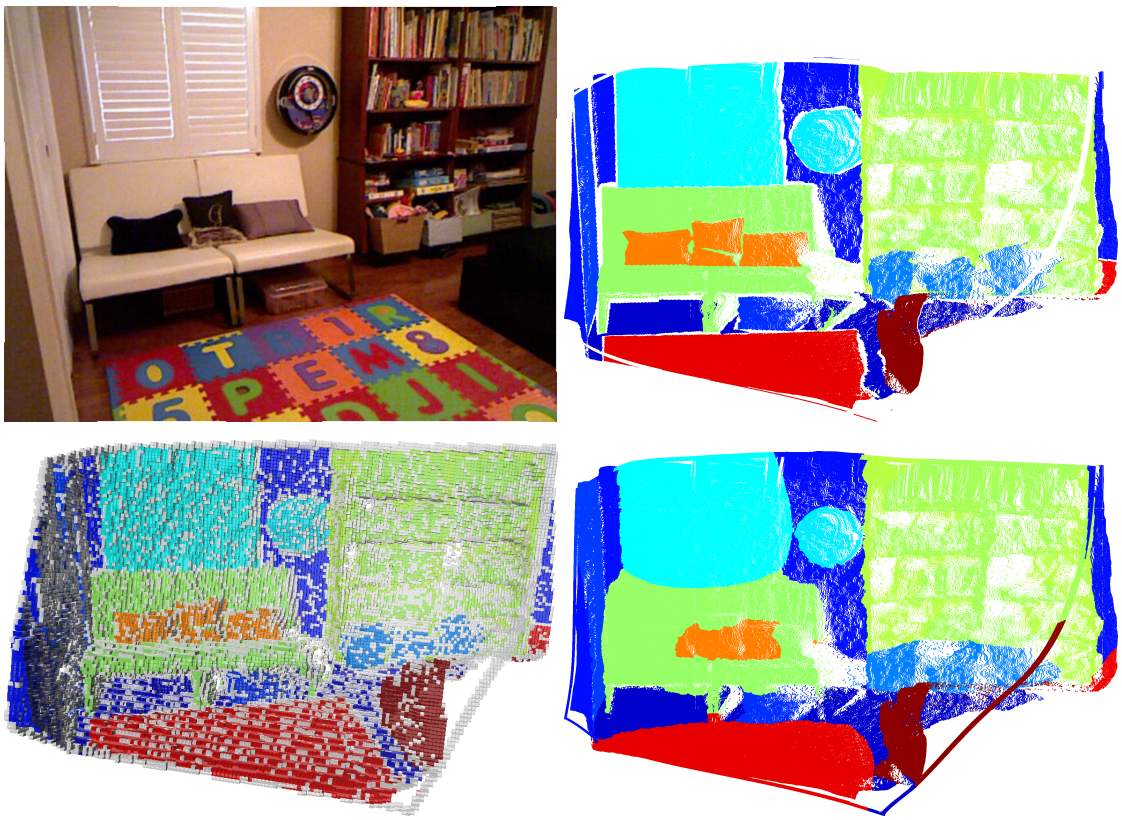


Figure 2.1: The figure shows examples of the GPSM (bottom right) and SOM (bottom left) using NYU Depth V2 dataset. The point cloud with ground truth semantic labels is shown in the top right. The maps are built by uniformly down-sampling the original point cloud to one-third. The GP map can infer the missing labels and deal with sparse measurements. The query points are the same as original points in the top right.

cannot be increased. In the context of high-dimensional occupancy mapping, at the cost of higher computational time, *Gaussian Processes* (GPs) have improved the map building performance by taking into account the correlation between map points and treating the map inference as a *binary classification* problem [26–29].

Semantic segmentation of the scene has long been an active topic in computer vision. The best performing algorithms used to rely on classifiers trained on a set of hand-crafted features [30,31]. However, the computational efficiency limits their application in real-time mobile robotics scenarios. More recent literature use the advances in the *deep convolutional neural network*. New architectures designed for semantic segmentation, including [32–35], have achieved superior performance in both indoor and outdoor benchmarks. Furthermore, the processing time for their pixel-wise estimation is also promising.

In this chapter, we formulate the semantic mapping as a *multi-class classification* prob-

lem and use raw pixelated semantic measurements (class labels) to generalize the traditional occupied and unoccupied class assignment. The proposed Gaussian Processes Semantic Map (GPSM) is inherently continuous, and prediction can be made at any desired location. We use *kernel methods* in the form of GPs to systematically accept inputs with any dimensionality as well as heterogeneous bases such as spatial coordinates and colors. Figure 2.1 shows examples of Semantic OctoMap (SOM) and GPSM built using a single frame of NYU Depth V2 dataset [8]. In particular, this chapter has the following contributions:

1. The proposed technique infers the structural and semantic correlation from measurements rather than resorting to assumptions. Through learning the correlation in data, GPSM can infer missing labels and deal with sparse measurements.

2. GPSM is continuous, and queries can be made at any desired locations; therefore, the map can be inferred with any resolution.

3. GPSM is the extension of GPs occupancy maps, i.e. binary maps, to the multi-class semantic representation which provides rich maps for robotic planning tasks.

4. GPSM is agnostic to the input dimensions and can handle an arbitrary number of non-spatial dimensions [1].

In the following section, a review of the related work is given. The problem statement and preliminaries are discussed in Section 2.3. The detailed formulation of the GP semantic mapping is presented in Section 2.4. The semantic OctoMap as an alternative map building technique is described in Section 2.5. Section 2.6 includes the time complexity analysis of both GP semantic map and semantic OctoMap. The Comparison of mapping results using a publicly available dataset is presented in Section 2.7; and finally, Section 2.8 concludes the chapter and discusses possible extensions of this work.

Notation

In the present article probabilities and probability densities are not distinguished in general. Matrices are capitalized in bold, such as in $\boldsymbol{X}$, and vectors are in lower case bold type, such as in $\boldsymbol{x}$. Vectors are column-wise and $1:n$ means integers from 1 to n. The Euclidean norm is shown by $\|\cdot\|$. $|\boldsymbol{X}|$ denotes the determinant of matrix $\boldsymbol{X}$. For the sake of compactness, random variables, such as X, and their realizations, x, are sometimes denoted interchangeably where it is evident from context. $x^{[i]}$ denotes a reference to the i-th element of the

[1] Without loss of generality, in this work, we only use spatial inputs to compare the results with semantic OctoMap.

variable. An alphabet such as $\mathcal{X}$ denotes a set, and the cardinality of the set is denoted by $|\mathcal{X}|$. A subscript asterisk, such as in $\boldsymbol{x}_*$, indicates a reference to a test set quantity. The n-by-n identity matrix is denoted by $\boldsymbol{I}_n$. We use $\mathrm{vec}(\boldsymbol{x}, \boldsymbol{y})$ to construct a vector by stacking $\boldsymbol{x}$ and $\boldsymbol{y}$. The function notation is overloaded based on the output type and denoted by $k(\cdot)$, $\boldsymbol{k}(\cdot)$, and $\boldsymbol{K}(\cdot)$ where the outputs are scalar, vector, and matrix, respectively. Finally, $\mathbb{E}[\cdot]$ and $\mathbb{V}[\cdot]$ denote the expected value and variance of a random variable, respectively.

2.2 Related Work

Early work in the context of robotic semantic mapping focused on room and building structure semantics, similar to topological mapping. Kuipers and Byun [15] create a network of distinctive places to map and explore large scale environments. Mozos et al. [36] assign labels to a 2D map of the indoor space corresponding to different parts of an indoor environment, such as room, corridor, and doorway through applying a classifier to the information collected by a range sensor. Then a *Hidden Markov Model* is used to encode transition probabilities to different labels, and thus using topological information. Pronobis et al. [37] extend that framework to include visual features, namely *Scale-Invariant Feature Transform* (SIFT) [38] and *Composite Receptive Field Histograms* (CRFH) [39].

Another interesting form of semantics for *Simultaneous Localization And Mapping* (SLAM) is object level classification. While room level semantics and topology are useful for navigation and exploration, object level semantics enable finer grained tasks such as *robotic manipulation*. Castle et al. [40] use a database of known planar objects to match SIFT features of a monocular video stream in an *extended Kalman filter* SLAM framework. Civera et al. [41] extend this approach to arbitrary three-dimensional (3D) geometries. Bao and Savarese [42] use an object detector in the structure from motion setup to jointly estimate camera parameters, 3D points, and object instances and poses. These approaches are developed to improve scene estimation by providing more geometric constraints. Conversely, Pillai and Leonard [43] use monocular SLAM to aggregate multiple views of a single object to provide more evidence to the object detector.

Dense 3D priors of objects have also been used for mapping and scene understanding. Kim et al. [44] learn models that are a collection of primitive 3D shapes; and show once a model based on primitives is learned, the object can quickly be recognized in an environment. Salas-Moreno et al. [16] align 3D mesh model priors of objects to the RGBD frame. The technique treats objects as landmarks and each alignment as a *factor* in the *graphical* SLAM framework. Choudhary et al. [45] also use objects as landmarks, but instead of having a dense 3D prior for every object, the objects are discovered via segmentation, and

then their models are produced during the mapping process.

Beyond just object models, there has been work to produce 3D maps with dense semantic labels. Kundu et al. [3] jointly reconstruct the 3D scene and perform semantic segmentation. The technique uses a *Conditional Random Field* (CRF) to infer the semantic category and occupancy for each voxel jointly. Sengupta and Sturgess [46] also look at semantic segmentation and reconstruction of the 3D scene; the technique uses stereo images, estimated camera pose, and a CRF defined over voxels and *supervoxels* of the *octree* to infer the semantic octree representation of the 3D scene. Vineet et al. [47] propose an incremental dense stereo reconstruction and semantic segmentation technique. To address the challenge of dealing with moving objects, Kochanov et al. [17] present a method to incorporate temporal updates into the map using scene flow measurements.

In this chapter, we propose the GP semantic map to infer the structural and semantic representation of the scene concurrently. Our approach fundamentally differs from the above-mentioned techniques as it does not discretize the map and is continuous. While in the present work, we focus on the problem formulation and 3D reconstruction of a single RGBD scan, the proposed framework can be used for the 3D scene reconstruction using map fusion algorithms [29]. Furthermore, the techniques mentioned earlier ignore pose estimation uncertainty in the map building process; however, our framework is systematically capable of accepting uncertain inputs [48].

2.3 Problem Statement and Preliminaries

The objective is formulate and solve the mapping process in a fully probabilistic framework. Since the environment representation we consider here is dense and measurements can be sparse due to the limited sensor field of view and range, a simple inference on individual voxels can lead to a poor mapping performance. Therefore, we devise a joint inference scheme based on GPs by assuming map points are normally distributed. In particular, the following assumption are made.

Assumption 1 (Static map representation) *The environment is static.*

Assumption 2 (Gaussian map points) *Any sampled point from the semantic map representation of the environment is a random variable whose distribution is Gaussian.*

From Assumption 2, and placing a joint distribution over map points, the mapping process by definition can be modeled as a GP.

Definition 1 (Gaussian process [9]) *A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.*

Problem 1 (Gaussian processes semantic map) *Given a point cloud measurement that is (possibly partially) assigned with noisy semantic class labels, infer a semantic map representation of the point cloud as a Gaussian process.*

2.3.1 Gaussian Processes Regression

GPs are nonparametric Bayesian regression techniques that employ statistical inference to learn dependencies between points in a data set [9]. The joint distribution of the observed *target* values, $\boldsymbol{y}$, and the function values (the latent variable), $\boldsymbol{f}_*$, at the query points can be written as

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n & \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_*) \\ \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}) & \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}), \tag{2.1}$$

where $\boldsymbol{X}$ is the $d \times n$ design matrix of aggregated input vectors $\boldsymbol{x}$, $\boldsymbol{X}_*$ is a $d \times n_*$ query points matrix, $\boldsymbol{K}(\cdot, \cdot)$ is the GP covariance matrix, and σ_n^2 is the variance of the observation noise which is assumed to have an independent and identically distributed (i.i.d.) Gaussian distribution. The predictive conditional distribution for a single query point $f_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}_* \sim \mathcal{N}(\mathbb{E}[f_*], \mathbb{V}[f_*])$ can be derived as

$$\mathbb{E}[f_*] = \boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x}_*)^T [\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n]^{-1} \boldsymbol{y}, \tag{2.2}$$

$$\mathbb{V}[f_*] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x}_*)^T [\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n]^{-1} \boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x}_*). \tag{2.3}$$

2.3.2 Gaussian Processes Classification

Supervised classification is the problem of learning input-output mappings from a training dataset for discrete outputs (class labels). In binary GP Classification (GPC), we define class labels as $y \in \{\pm 1\}$. In GPC, the inference is performed in two steps. First computing the predictive distribution of the latent variable corresponding to a query case, $f_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}_* \sim \mathcal{N}(\mathbb{E}[f_*], \mathbb{V}[f_*])$, and then a probabilistic prediction, $p(y_* = +1 | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}_*)$, using a *sigmoid function*, $\sigma(\cdot)$, that assigns class labels with a probability that increases monotonically with the latent. The non-Gaussian likelihood and the choice of the sigmoid function can make the inference analytically intractable. Hence, approximate inference techniques are required.

2.3.3 Model Selection

In GPs, one can learn free parameters of mean, covariance, and likelihood functions (*hyperparameters*) through optimizing a cost function. Practically, The hyperparameters, $\boldsymbol{\theta}$, can be computed by minimization of the negative log of the marginal likelihood (NLML) function:

$$\begin{aligned}\log p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta}) = &-\frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X})+\sigma_n^2\boldsymbol{I}_n)^{-1}\boldsymbol{y} \\ &-\frac{1}{2}\log|K(\boldsymbol{X},\boldsymbol{X})+\sigma_n^2\boldsymbol{I}_n| - \frac{n}{2}\log 2\pi. \end{aligned} \tag{2.4}$$

In (2.4), the first term $-\frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X})+\sigma_n^2\boldsymbol{I}_n)^{-1}\boldsymbol{y}$ corresponds to data-fit, the second term $\frac{1}{2}\log\ |K(\boldsymbol{X},\boldsymbol{X})+\sigma_n^2\boldsymbol{I}_n|$ penalizes the model complexity, and the last term is a constant.

Remark 1 *Note that the selection of NLML function for the optimization problem is entirely optional. However due to the marginalization property of the multivariate normal distribution, the latent variable $\boldsymbol{f}_*$ can be conveniently marginalized.*

Remark 2 *Generally speaking, building maps using GPs can handle sparse sensor observations and consequently sparse training data. However, in practice, the kernel function describes the correlation between training points. A smooth kernel such as the squared exponential can cover a larger area with fewer training points, and a rough kernel such as Matérn ($\nu = 1/2$) can only cover the vicinity of sparse training points, see [9, Figure 4.1]. Therefore, model selection is crucial to fully exploit capabilities of GPs. Furthermore, hyperparameters have a significant effect on the shape of the map.*

The GPC model implemented in this work uses a constant mean function, Matérn ($\nu = 5/2$) covariance function [49] with automatic relevance determination [50], the error function likelihood (*probit regression*), and *Laplace* technique for approximate inference and is done using the open source GP library in [9]. It is argued in [49] that Matérn covariance functions are more suitable for modeling physical processes.

2.3.4 Large-scale Inference

The main bottleneck in GP regression is computation of the term $[\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X})+\sigma_n^2\boldsymbol{I}_n]^{-1}$ where the covariance matrix of training data has to be inverted. This limitation reveals itself more when one tries to use a large number of training data. In general, the number of measurements in a point cloud exceeds a few hundred and can be up to several hundred

thousands. In such cases, approximate inference techniques such as Laplace, *expectation propagation* [51], or *variational Bayes* [52] can become time-consuming. The *Fully Independent Training Conditional* (FITC) [53, 54] method is based on a low-rank plus diagonal approximation to the exact covariance matrix and is computationally more attractive while it preserves the desirable properties of the full GP [53]. In particular, FITC uses a set of *inducing points* to shift the computational cost on the *cross-covariances* computation between training, test, and inducing points.

2.4 Gaussian Processes Semantic Map

In this section, we formulate the GP semantic map to solve Problem 1. While the formulation we present here is general, for the classification purpose, we use a binary GPC as the base classifier for each class and *one-vs.-rest* approach to building a multi-class classifier. However, we acknowledge that the *multi-class Laplace* approximation in [9, Section 3.5] is an interesting approach to building a true probabilistic multi-class classifier.

Let $\mathcal{M}$ be the set of possible semantic maps. We consider the map of the environment as an n_m-tuple random variable $(M^{[1]}, \dots, M^{[n_m]})$ whose elements are described by a normal distribution $m^{[i]} \sim \mathcal{N}(\mu^{[i]}, v^{[i]})$, $i \in \{1\colon n_m\}$. Let $\mathcal{X} \subset \mathbb{R}^3$ be the set of spatial coordinates to build a map on, and $\mathcal{C} = \{c^{[j]}\}_{j=1}^{n_c}$ be the set of semantic class labels. Let $\mathcal{Z} \subset \mathcal{X} \times \mathcal{C}$ be the set of possible measurements. The observation consists of an n_z-tuple random variable $(Z^{[1]}, \dots, Z^{[n_z]})$ whose elements can take values $\boldsymbol{z}^{[k]} \in \mathcal{Z}$, $k \in \{1\colon n_z\}$ where $\boldsymbol{z}^{[k]} = (\boldsymbol{x}^{[k]}, y^{[k]})$, $\boldsymbol{x}^{[k]} \in \mathcal{X}$, and $y^{[k]} \in \mathcal{C}$.

We define a training set $\mathcal{D} = \{(\boldsymbol{x}^{[i]}, y^{[i]})\}_{i=1}^{n_t}$ and the target vector $\boldsymbol{y} = \mathrm{vec}(y^{[1]}, \dots, y^{[n_t]})$ where $\mathcal{D} \subseteq \mathcal{Z}$, and $n_t \leq n_z$ is the number of training points. Given observations $Z = \boldsymbol{z}$, we wish to estimate $p(M = m \mid Z = \boldsymbol{z})$. The map can be inferred as a Gaussian process by defining the process as the function $y : \mathcal{Z} \to \mathcal{M}$, therefore

$$y(\boldsymbol{x}) \sim \mathcal{GP}(f_m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')). \tag{2.5}$$

It is often the case that we set the mean function $f_m(\boldsymbol{x})$ as zero, unless it is mentioned explicitly that $f_m(\boldsymbol{x}) \neq 0$. For a given query point in the map, $\boldsymbol{x}_*$, GP predicts a mean, μ, and an associated variance, v. Thus, for any map point, we have $m^{[i]} = y(\boldsymbol{x}_*^{[i]}) \sim \mathcal{N}(\mu^{[i]}, v^{[i]})$.

Once the mean and variance of the latent f_* are available, we predict the averaged

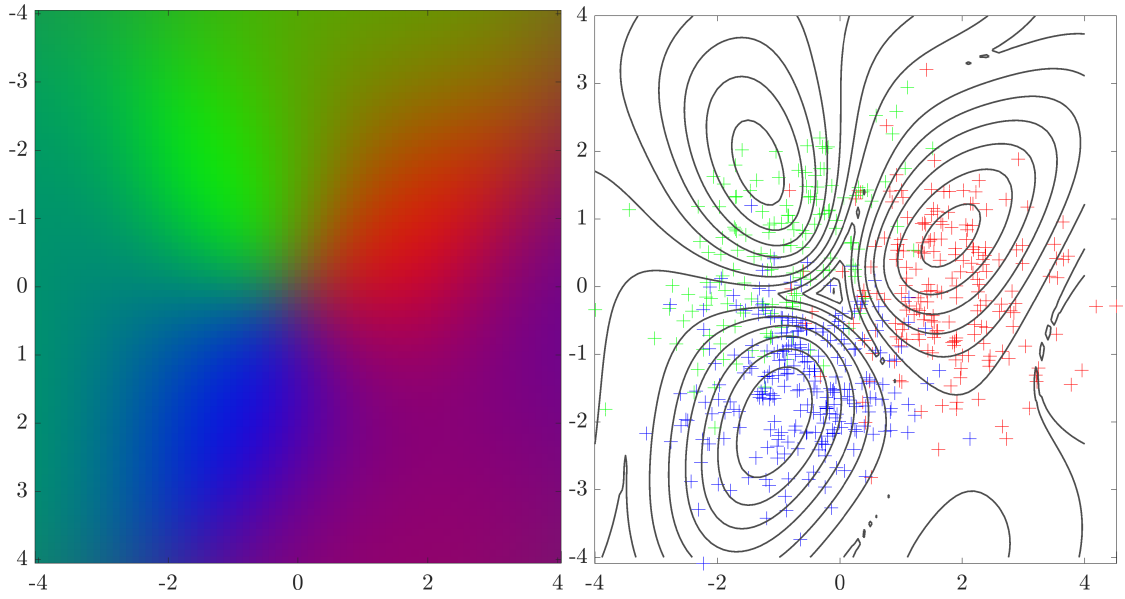


Figure 2.2: A two-dimensional toy example of GP multi-class classification using the synthetic dataset shown in the right plot. The prediction is continuous, and the class probabilities can be queries at any point in the plane as shown in the left plot. Furthermore, the contour curves can be seen as decision boundaries. The plots show the log probabilities.

predictive probability of the class $c^{[j]}$ as follow [9, Chapter 3]:

$$p(c^{[j]} = +1|\mathcal{D}, \boldsymbol{x}_*) = \int \sigma(u)\mathcal{N}(u \mid \mathbb{E}[f_*], \mathbb{V}[f_*])du. \tag{2.6}$$

Note that once the n_c binary GPC are trained, and the prediction at query points (map points) are performed, we normalize the class probabilities to get $p(M = c^{[j]}|\boldsymbol{z})$ for the j-th semantic class. One straightforward way to assign hard labels to map points is to find the class with the maximum probability.

Remark 3 *The actual representation of the map depends on the distribution of query points. It is often the case to use uniformly distributed points. In general, query points can have any desired distributions. However, in this work, we use the original dense point cloud as the query points to facilitate comparison with the ground truth.*

Example 1 (Two-dimensional Toy Example) *Figure 2.2 illustrates a two-dimensional toy example of GP multi-class classification that can be extended to the three-dimensional semantic mapping. The continuity and smoothness of the probabilistic inference are evident from the left plot, while the corresponding contour curves and the synthetic training points are shown in the right plot. The plots show the log probabilities.*

2.5 Semantic OctoMap

The OctoMap [14] is a popular robotic occupancy mapping tool which builds a discretized model of the 3D world using the octree data structure. The octree enables the multi-resolution map representation which has the advantage of being memory-efficient. In this section, we develop a simple extension of this technique to semantic OctoMap which will serve as the alternative approach in the presented evaluations in Section 2.7. In the basic OctoMap implementation, voxels only contain occupancy probability represented as log-odds. In the developed semantic OctoMap, in addition, voxels include semantic labeling and color information for visualization. The incorporation of the semantic knowledge into the OctoMap is also discussed in [46].

Following the problem formulation in the previous section, given observations $Z = \boldsymbol{z}$, we wish to estimate $p(M = m \mid Z = \boldsymbol{z})$. The fundamental assumptions of this approach are as follows. First, the map posterior is approximated as the product of its marginals [55]:

$$p(M = m \mid Z = \boldsymbol{z}) = \prod_{i=1}^{n_m} p(M = m^{[i]} \mid Z = \boldsymbol{z}), \tag{2.7}$$

which indicates the distribution of each voxel is independent of the others. Second, we assume that for each voxel, the occupancy and semantic probabilities are independent. Based on these tow assumptions, we can process occupancy and semantic beliefs separately. We first use OctoMap library to compute the occupancy probability of each voxel. The principle is that voxels correspond to endpoints are updated as hits, while voxels along the ray between the sensor origin and the endpoint are updated as misses.

We follow the idea in [17] to define the semantic likelihood. The idea is to simply average the discrete semantic class labels that fall within a voxel. Let $\mathcal{L}^{[i]}$ be a multiset that contains semantic label observation, $l^{[i,j]} \in \mathcal{C}$, that are inside the i-th voxel. The likelihood function to update the semantic belief of the j-th class for the i-th voxel can be defined as follows:

$$p(c^{[j]} = +1 \mid Z) = \frac{1}{|\mathcal{L}^{[i]}|} \sum_{s^{[k]} \in \mathcal{L}^{[i]}} p(s^{[k]} \mid Z). \tag{2.8}$$

For the visualization purpose, we only visualize the occupied voxels with their corresponding hard labels which are computed by finding the label with the maximum semantic probability. Figure 2.3 shows an example of the developed semantic OctoMap.

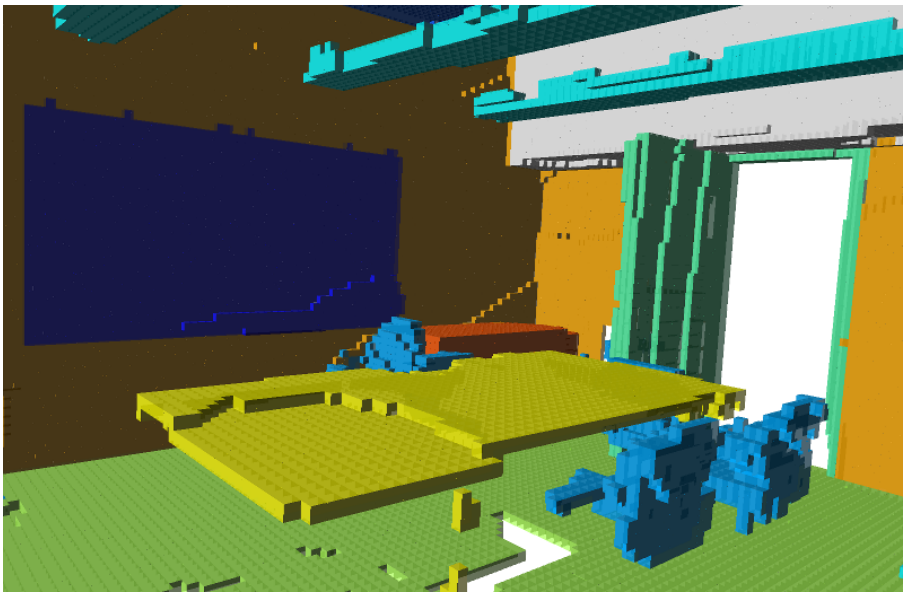


Figure 2.3: Semantic OctoMap built using the Stanford 2D-3D-Semantics Dataset [1], area 1, conference room 1. The semantic label observations are from the ground truth data. From the figure, the effect of discretization of the map and independent voxels inference are evident in the structural shape of the environment.

2.6 Computational Complexity Analysis

FITC uses inducing points (active set) to base the computations on cross-covariances between training, test, and inducing points; hence, the computational cost is dominated by the matrix multiplication and reduced from $\mathcal{O}(n_t^3)$ to $\mathcal{O}(n_t n_u^2)$ where n_u is the number of inducing points and $n_u \ll n_t$. The computational complexity of the GP semantic map is dominated by FITC approximation and is the same.

The computational complexity of semantic OctoMap is determined by octree data access complexity, which is $\mathcal{O}(n_d) = \mathcal{O}(\log n_n)$, where n_n is the total number of nodes and n_d is the depth of the octree data structure. As semantics fusion process queries the corresponding voxel on an octree for each measured point in a scan, the computational complexity of this part is $\mathcal{O}(n_p \log n_n)$, where n_p is the number of points in the point cloud. It is worth mentioning that the maximum depth of octree is fixed in implementation; therefore, the practical computational complexity of the semantic fusion process is $\mathcal{O}(n_p)$.

2.7 Results and Discussion

We now present mapping results using the proposed GP semantic map and the semantic OctoMap. We evaluate the proposed technique by comparing the mapping performance with that of the semantic OctoMap using NYU Depth V2 dataset [8]. The GPSM is implemented in MATLAB using the GPML library [9] and the SOM is implemented in C++ using by developing the original OctoMap implementation [14].

In the following, we explain the experimental setup and the performance criterion used for the comparison. We run two experiments; first, we study the effect of sparse measurements and missing labels by downsampling ground truth point clouds. Second, we use an image segmentation technique to label the entire point cloud to study the effect of misclassification (false positives) in observations.

2.7.1 Experimental Setup and Evaluation Criterion

The NYU Depth V2 dataset provides a large set of aligned RGB and depth images in indoor environments that are recorded by the Kinect sensor. The dataset also contains a subset of pixel-wise multi-class labeled images where there are unlabeled pixels in the annotated images for missing structures. We first convert RGBD scans to 3D point clouds with point-wise labels which serve as observation sets. Then we build the GPSM and the SOM. Note that, in this work, the map is referred to a local map built using a single scan (labeled point cloud). In the first experiment, point clouds are labeled using the ground truth semantic labels and downsampled by one-third to replicate sparse measurements and missing labels. In the second experiment, semantic labels are computed by the image segmentation technique SegNet [56]. The resolution of semantic OctoMap is set to $0.02\,\mathrm{m}$ for all results. The observation set contains about $300,000$ points. While the GP training points are a much smaller subset of the original observation set (typically less than 5000 points), the query points are all points in the original point cloud.

The evaluations include the comparison of mapping performance using the *Area Under the receiver operating characteristic Curve* (AUC). The probability that the classifier ranks a randomly chosen positive instance higher than a randomly chosen negative instance can be understood using the AUC of the classifier; furthermore, the AUC is useful for domains with skewed class distribution and unequal classification error costs [57]. The AUC is originally a measure of the discriminability of a pair of classes. The extension of this method to the multi-class case is discussed in [58]. In order to maintain the performance measure insensitive to class distribution and error costs, following our one-vs.-rest multi-class classification approach, we calculate an AUC for each classifier (GP) and then take the

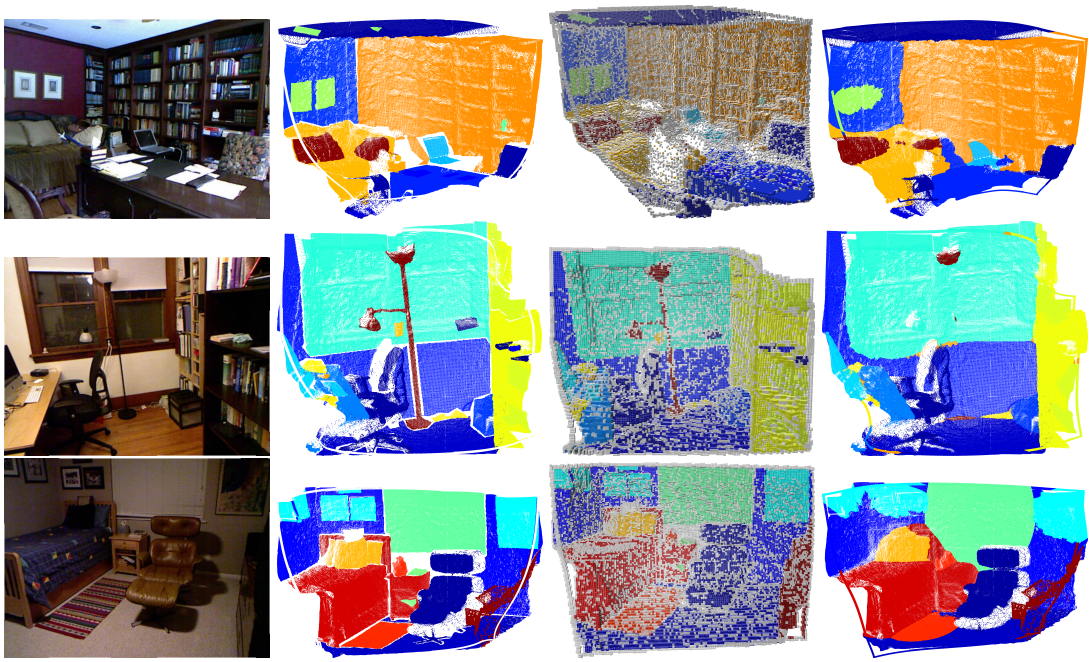


Figure 2.4: The figure shows GPSM and SOM results using sparse labeled point clouds. From left, each column respectively shows the RGB image, the point cloud with ground truth semantic labels before downsampling, the SOM, and the GPSM. The maps are built by uniformly down-sampling the original point cloud to one-third. The GP map can infer the missing labels and deal with sparse measurements through leaning the correlation between observations. For GPSM, the query points are the same as original points in the second column.

average AUC as the overall performance measure. Thus, the total AUC can be computed as follows:

$$\mathrm{AUC}_{\mathrm{total}} = \frac{1}{|\mathcal{C}|} \sum_{j \in \mathcal{C}} \mathrm{AUC}^{[j]}. \tag{2.9}$$

2.7.2 Sparse and Missing Measurements Effects

In the first experiment, we use four samples from the dataset for map building. The point clouds are labeled using the ground truth data. We uniformly downsample the labeled point clouds by one-third to replicate sparse measurements. As a result, since the number of labeled points is reduced, the point cloud contains many unlabeled points which makes the mapping process challenging. Figures 2.1 and 2.4 show the results of GPSM and SOM using the selected frames. Table 2.1 shows the quantitative comparison between GPSM and SOM.

From the result, the marginalization effect of the SOM on the map inference appears

Table 2.1: The comparison of Gaussian processes semantic map and semantic OctoMap using the area under the receiver operating characteristic curve (AUC). The maps are built using downsampled point clouds from the NYU Depth V2 dataset [8] and the corresponding frame numbers are shown in the table.

Frame Number	GP Semantic Map $\mathrm{AUC}_{\mathrm{total}}$	Semantic OctoMap $\mathrm{AUC}_{\mathrm{total}}$
NYU V2 - 282	0.9624	0.88525
NYU V2 - 374	0.9622	0.86251
NYU V2 - 555	0.9675	0.88648
NYU V2 - 965	0.9644	0.83569

as many unlabeled voxels. Essentially, any voxel that does not contain any labeled points remains as unlabeled as there is no information available at that particular location. Intuitively, it is likely that neighboring locations share the same semantic class label and if any two points are spatially close, this chance can be even higher due to the present structural correlations in the environment. The lower mapping performance of the SOM, while dealing with sparse measurements, approves this claim.

The GP semantic map places a joint distribution on the observations and query points. This high-dimensional approach enables the method to infer the semantic class labels continuously. Furthermore, the available information in observations are captured in the GP covariance function which effectively models the correlation between map points. The GPSM is not limited to a fixed resolution and provides a probabilistic prediction and any desired location. The GPSM performs on all samples equally well with minor misclassified regions in each map. The maps are visualized using the class label with the maximum probability.

2.7.3 Mapping under Noisy and Misclassified Labels

In the second experiment, we use SegNet [56] to generate semantic labels. We first perform the RGB image segmentation and then assign each pixel's label to its corresponding point in the point cloud. The mapping results are computed using the same four examples from the previous experiment, and the performance of each method is evaluated using the ground truth labeled point clouds in Figure 2.4. Figure 2.5 shows the results of the image segmentation (second column from left), the SOM (third column from left), and the GPSM (fourth column from left). Table 2.2 shows the quantitative comparison between GPSM and SOM. The challenge in this test is the presence of false positive labels in the observations and, as expected, both methods achieve lower mapping performance on the same set of data. Note

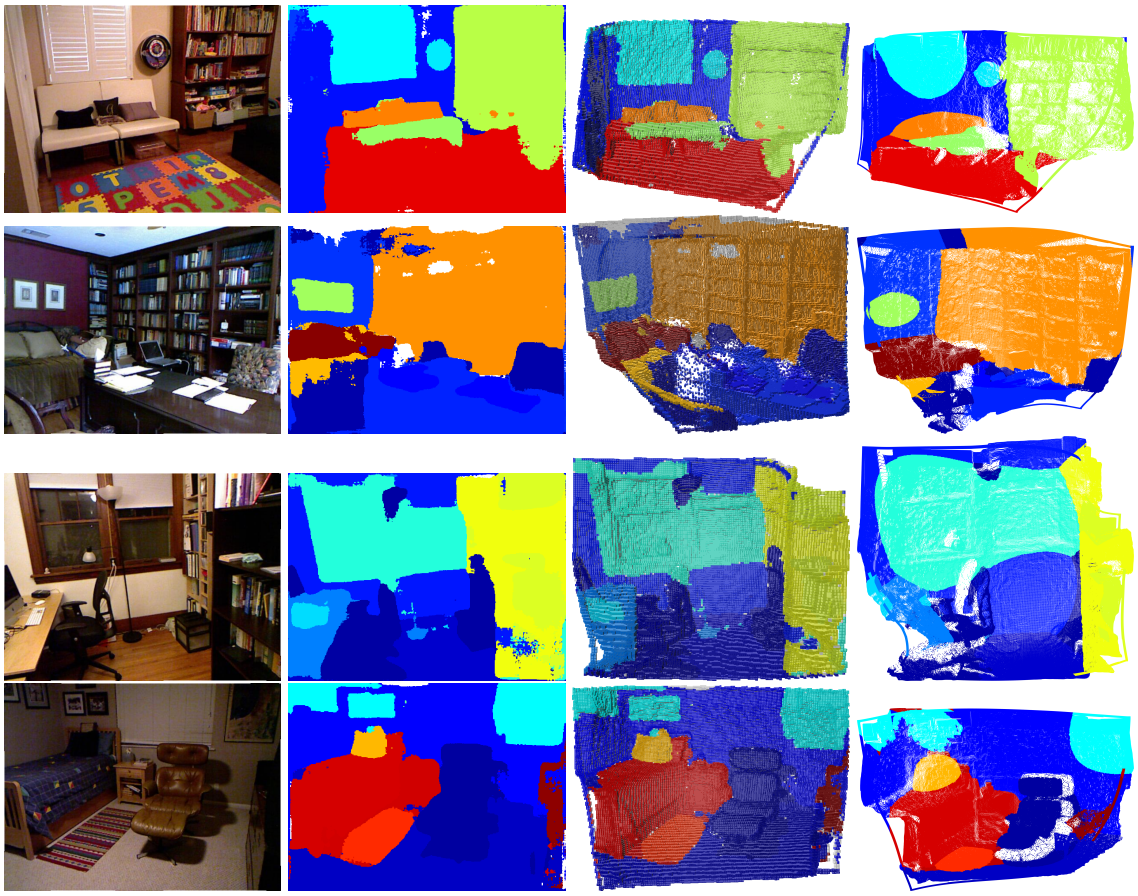


Figure 2.5: The results of mapping under noisy and misclassified labels. From left, each column respectively shows the RGB image, the image segmentation using SegNet, the SOM, and the GPSM results. The false positives in the image segmentation are directly transferred into the SOM results, while the GPSM uses the spatial distance between points to infer the class labels based on correlations between map points.

that in this experiment the point clouds are not downsampled; however, the GP training set is a subset of observations as explained in the previous subsection.

The SOM directly uses the available points within a voxel to infer the voxel labels. Therefore, any misclassified label is transferred to the map. However, the GPSM infers the class labels based on the available correlations in the observation sets. Specifically, in the presented examples, the spatial correlations reduce the number of the misclassified points in the final map. For example, two adjacent pixels in the image space can have a similar label while their spatial distance can be large. Such cases are trivially handled in the GPSM framework since a large spatial distance implies insignificant correlation.

Table 2.2: The comparison of Gaussian processes semantic map and semantic OctoMap using the area under the receiver operating characteristic curve (AUC). The maps are built using noisy image segmentation labels produced by SegNet. The figures for both methods are lower than the first experiment due to the presence of false positive labels in the observations.

Frame Number	GP Semantic Map $\mathrm{AUC}_{\mathrm{total}}$	Semantic OctoMap $\mathrm{AUC}_{\mathrm{total}}$
NYU V2 - 282	0.7192	0.68402
NYU V2 - 374	0.7625	0.73552
NYU V2 - 555	0.7032	0.66881
NYU V2 - 965	0.8612	0.77456

2.7.4 Discussion and Limitations

Since we seldom have sufficient information about a process, minimization of the NLML is a useful approach as it provides flexibility for model selection. However, the problem of minimizing NLML is an ill-posed problem and using maximum likelihood estimate we can often find a local minimum. Therefore, it is always possible that the solution suffers from overfitting, especially if the number of hyperparameters is large [9].

The common way for map representation in robotics is using a dense set of points with a particular distribution that is possibly suitable for navigation tasks. However, there is no restriction for any other desired representation such as approximate belief representations in [59], that can be useful for other applications such as predictions. In fact, it is shown in [60] that by modeling the underlying process as GPs, one can perform information-theoretic planning using nonparametric information gain with possibility to handle the state estimate uncertainty [48].

In heteroscedastic processes, noise is state dependent. Heteroscedastic Gaussian process regression [61,62] can be an alternative to model the structural correlation more accurately. However, to model the prediction uncertainty, they require a second GP in addition to the GP governing the noise-free output value. The computational cost is roughly twice that of the standard GP [63,64]. To avoid increasing the computational time in the proposed mapping technique, we do not consider the heteroscedasticity in training data; however, applying these techniques to the problem at hand is an interesting direction to follow.

2.8 Conclusion and Possible Extensions

In this chapter, we developed the Gaussian processes semantic map for 3D semantic map representations. We formulated the problem as a multi-class classification by defining the map spatial coordinates and semantic class labels as the input and output of the process, respectively. The proposed GP semantic map learns the structural and semantic correlation from measurements rather than resorting to assumptions, and exploits the spatial correlation (and possibly any additional non-spatial correlation) between map points for inference. In particular, the proposed map inference can infer missing labels and deal with sparse measurements, is continuous and queries can be made at any desired locations, and can deal with false positives better.

While, in this work, we only considered the spatial coordinates as the input, GPSM is agnostic to the input dimensions and can handle an arbitrary number of non-spatial dimensions. It would be interesting to extend the proposed method to an incremental form as well as add other available dimensions such as color and intensity.

CHAPTER 3

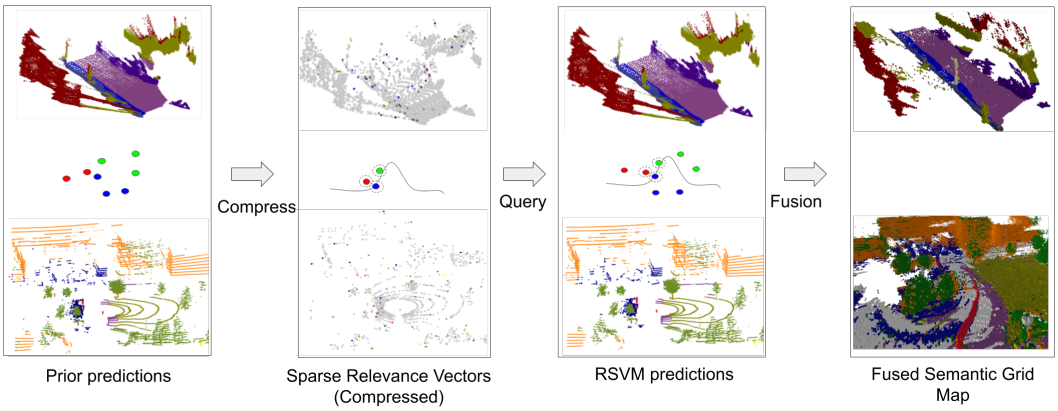
Compressed 3D Semantic Map Inference via Relevance Vector Machines

Dense robotic mapping is a challenging, high-dimensional, sequential inference problem. This chapter reports on development of a compressed 3D semantic map inference pipeline through sparse Bayesian models, in particular, the relevance vector machine. The technique is based on the principle of automatic relevance determination and produces sparse local models that use a small subset of the original dense training set as the dominant basis. The resulting map posterior is continuous, and queries can be made efficiently at any resolution. We evaluate the proposed relevance vector semantic map using two publicly available datasets. The experimental results show that the proposed mapping system has comparable performance with the state-of-the-art techniques while requiring a lower memory footprint. The open source implementation of the proposed system is available at: https://bitbucket.org/perl-sw/rvsm.

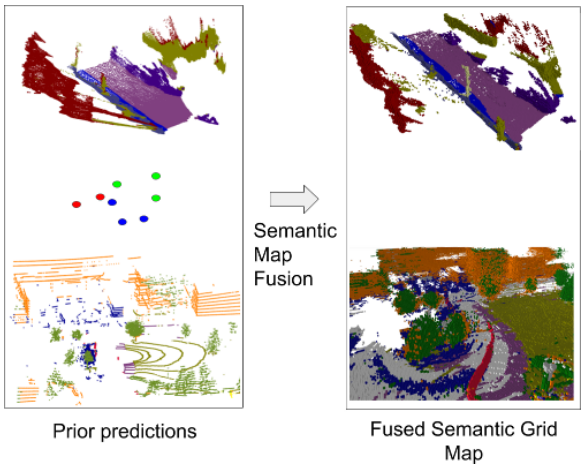
3.1 Introduction

Enriching geometric maps with semantic knowledge improves scene understanding, navigation, and interaction abilities of robots. Current mobile robots rely on three-dimensional (3D) maps for these ends. A 3D semantic mapping system requires to be not only accurate, but also memory and computationally efficient so that it can be used in real robotic applications. In addition, the system needs to be scalable for long-term mapping experiments; that is to say, the mapping system has the ability to reconstruct large maps on demand. In particular, the latter feature is a necessity as the drift caused by odometry and tracking algorithms is often rectified by *loop-closures* within Simultaneous Localization And Mapping (SLAM) systems, resulting in discrete changes in the trajectory of the robot.

The most common approach to build a semantic map is using a voxelized representation of 3D space and *undirected graphical models* (such as *Markov random fields*) or



(a) Proposed compressed 3D semantic mapping pipeline.



(b) Pipeline proposed by [2].

Figure 3.1: The semantic mapping and fusion pipeline of RVSM, and a comparison with the baseline [2]. Although the proposed system shares similar properties of the baseline such as final voxel map representation and Bayesian updates per semantic class, it consists of extra sparse model building step which allows for map compression and continuous queries at any desired locations.

Conditional Random Fields (CRFs) to model the conditional independence between voxels [65]. CRFs are *discriminative* models that are suitable for spatial inference. However, a primary drawback of these approaches is that the resulting representation is discrete from the beginning. The map's resolution is often fixed and once the map has been inferred from raw sensory data, its resolution cannot be increased. Furthermore, although this approach produces satisfactory local maps, to store a semantic map, all voxels have to store the label distribution which requires substantial amount of memory and is not scalable.

In this chapter, we propose a semantic robotic mapping technique, shown in Fig. 3.1, that can *compress* dense semantic information of the scene via *sparse Bayesian mod-*

els [66], in particular, the *Relevance Vector Machine* (RVM), without or with minor compromise on prediction performance. While CRFs perform inference on fixed voxels after the 3D space is discretized, RVMs are able to perform high-dimensional sequential inference directly on point clouds with prior distributions. After a Relevance Vector Semantic Map (RVSM) is trained, we can uncompress the semantic information by performing efficient *queries* any desired location. For application, RVSMs can be efficiently fused into a semantic voxel map using Bayesian updates. In summary, the main contributions of the present work are as follows:

1. We propose a compressed and continuous semantic map representation and integrate it within a pipeline to construct semantic occupancy grid maps.

2. The open source implementation of the proposed system is available at: https://bitbucket.org/perl-sw/rvsm.

3. We provide experimental results for evaluating the performance of the proposed mapping technique using two publicly available datasets, namely, KITTI [67] and NCLT [68].

The remainder of this chapter, after reviewing the related work in the following section, is organized as follows. Section 3.3 explains the problem setup via sparse Bayesian modeling. Section 3.4 shows the proposed pipeline for compressed 3D semantic map inference. The experimental results using two publicly available datasets are presented in Section 3.5. Finally, Section 3.6 concludes this chapter and provides future work directions.

3.2 Related Work

We refer to prior distribution as the class distribution of each 3D point in the local frame from 2D/3D semantic segmentation methods. Existing methods include performing pixel-wise segmentation on 2D images, and then transforming them to the 3D space, or directly performing point-wise segmentation for the raw 3D point cloud. In RVSM, the prior distribution over classes can be generated from both 2D segmentation and 3D raw point cloud segmentation methods. In this work, we use the 2D semantic segmentation neural networks from DeeplabV3+ [69] and Dilated Convolutional Neural Networks (CNN) [70].

3.2.1 2D Segmentation Methods

Advances in deep CNN for 2D semantic segmentation provide sufficient accuracy [32, 69–72] in both indoor and outdoor datasets [8, 73]. In [74], the original point cloud is projected

onto multiple camera views, each segmented with CNN, and then back-projected to the 3D space. The label of a single point is decided by the class of the nearest back-projected 2D pixel.

3.2.2 3D Raw Point Cloud Segmentation Methods

Qi et al. propose PointNet [75] that combines spatial feature blocks of points into a point-wise representation. They further extend PointNet into large-scale scenes in [76], by aggregating blocks of multiple scales and sharing features of neighboring points. Similarly, local structures of different scales are exploited by performing hierarchical sampling and grouping nearby points. SqueezeSeg in [77] also processes raw point clouds, but instead of taking them as unstructured sets, SqueezeSeg spherically projects all points onto a dense 2D image, which will be used as the network input. Furthermore, Landrieu and Simonovsky [78] introduce attributed directed graphs as a form of compact point-group representations, which can capture the contextual relations between neighboring objects. Qi et al. [79] leverage 2D object detectors to find regions of interest in the projected 3D point cloud, and then assigns labels to the frustum of each region using PointNet [75].

3.2.3 3D Voxel-based Semantic Map Fusion

After obtaining priors in local frames, fusing the predictions of nearby points from multiple frames produces a global semantic map. Early work in dense semantic mapping back-projected labels from a segmented (2D) image to the reconstructed 3D points and then assigned each voxel or mesh face to the most frequent label according to a label histogram [5, 80]. Bayesian frameworks have been utilized to fuse labels from multiple views into a voxel-based 3D map. In [81], probabilistic segmentation of multiple (2D) images obtained by *Random decision Forests* (RFs) were transferred into 3D and updated using a Bayes filter. Xiang and Fox proposed DA-RNN [82], which integrates a deep network for RGB-D video labeling into a dense 3D reconstruction framework built by KinectFusion [83]. DA-RNN yields consistent semantic labeling of indoor 3D scenes; however, it is assumed that semantic labels and geometric information are independent, and therefore, the consistency largely depends on the performance of data association computed by KinectFusion. The 3D label fusion is done by updating a probability vector of semantic classes and choosing the label with the maximum probability. Sun et al. [84] leverage recurrent neural cells to fuse the label updates.

RVSM also does not directly fuse the prior distributions into the voxel map. Instead, it compresses the priors into a sparse set of relevance vectors through training of local

RVMs. Then, query results from an RVM is fused into the voxel map using Bayesian updates similar to [2].

3.2.4 Semantic Mapping Refinement

In semantic mapping, structure prediction methods such as CRFs are used to improve the map consistency in the 3D space. In [85], a voxel-CRF model is proposed to capture the geometric and semantic correlation by constructing a CRF over the 3D volume. In [86], a Kalman filter is used to transfer 2D class probabilities obtained by RFs to the 3D model; then 3D labels are further refined through a dense pairwise CRF over the point cloud. In [87], a higher-order CRF model is used to enforce temporal label consistency by generating higher-order *cliques* from *superpixels* correspondences in an RGB-D video, which improves the precision of the semantic maps.

A common feature of the works mentioned earlier is the discretization of the space prior to map inference. That is, once the map is inferred, the prediction cannot be computed at an arbitrary point. In this chapter, we propose an alternative solution for the problem of dense 3D map building that is continuous and at the same time sparse. RVSM incrementally learns relevance vectors which are dominant basis vectors in the data, and builds a sparse Bayesian model [66] of the 3D map. As a result, the semantic information is compressed into compact relevance vectors, and query can be made efficiently at any desired location.

3.3 Problem Statement

In this section, we formulate the semantic mapping problem, and introduce a sequential sparse Bayesian learning algorithm to incrementally build a semantic map. Let $\mathcal{X} \subset \mathbb{R}^3$ be the set of spatial coordinates for map inference and let $\mathcal{C} = \{1, 2, 3, ..., n_c\}$ be the set of semantic class labels. A subset $\mathcal{Z} \subset \mathcal{X} \times \mathcal{C}$ is the set of possible observations, where an observation is a spatial coordinate associated with a semantic label. A set of existing observations consists of an n_t-tuple random variable $(Z_1, \ldots, Z_{n_t})$ whose elements can take values $z_i \in \mathcal{Z}$, $i \in \{1 \colon n_z\}$ where $z_i = [x_i, t_i]^\top$, $x_i \in \mathcal{X}$, and $t_i \in \mathcal{C}$. The training set is a subset of the observations, denoted by $\mathcal{D} \subseteq \mathcal{Z}$, where $n_t \leq n_z$ is the number of training points. Let $\mathcal{M}$ be the set of possible assignments of semantic maps, that is, all random variables defined on $\mathcal{X}$ with values in $\mathcal{C}$. A semantic map $m \in \mathcal{M}$ of the environment associates to each point x_i in $\mathcal{X}$ a probability distribution over $\mathcal{C}$. Given observations $z = (z_1, .., z_{n_z})$, we wish to estimate $p(M = m \mid Z = z)$.

To simplify the problem, we use one binary RVM classifier for each class and adopt a

one-vs.-rest approach for building a n_c-class classifier. Let $x \in \mathcal{X}$ be a 3D input vector as before taking the spatial coordinates of the raw point cloud. Given the existing observations $\mathcal{D} = \{(x_i, t_i) : t_i \in \mathcal{C}\}_{i=1}^{n_t}$, for any new input x, the estimate of x's posterior probability of class membership for each class $c_k \in \mathcal{C}$, can be computed from the logistic sigmoid function as follows:

$$p(c_k = +1|\mathcal{D}, x) = \sigma(y(x; w)) = \sigma(\sum_{j=1}^{n_b} w_j \phi_j(x)) = \sigma(\phi(x)w), \tag{3.1}$$

where $y(x; w)$ is a linear model (linear in weights w) to approximate the *latent* function of interest, and $\phi(x) \triangleq [\phi_1(x), \ldots, \phi_{n_b}(x)]$ is the kernel basis feature vector. The basis vector $\phi_j(x)$ is replaced with a vector of kernel functions $k(x_1, x_j), \ldots, k(x_{n_b}, x_j)$, where $k(\cdot, \cdot)$ evaluates the similarity between two inputs by implicitly mapping them to a *feature space* [88].

The objective is to learn an optimized weight vector w as well as the number of basis vectors n_b, such that it generalizes well to new inputs x_* (test data). When a significant portion of weights becomes small or zero, the linear model becomes sparse [89].

For better generalization, regularization in the form of zero mean Guassian Distribution on weights is added to encourage the target function to be smooth and infer which input vectors are relevant [50]. Instead of computing a *point estimate* for the weights, a Bayesian predictive framework infers the *posterior distribution* over w via Bayes' rule:

$$p(w|t, \alpha) = \frac{p(t|w)p(w|\alpha)}{p(t|\alpha)}. \tag{3.2}$$

Here, $p(t|w)$ is the *likelihood* of the observations in the form of a Bernoulli likelihood

$$p(t|w) = \prod_{i=1}^{n_t} \sigma(y(x_i; w))^{t_i} \big[1 - \sigma(y(x_i; w))\big]^{1-t_i}, \tag{3.3}$$

$p(t|\alpha) = \int p(t|w)p(w|\alpha)\mathrm{d}w$ is the *marginal likelihood*, and $p(w|\alpha)$ is the zero mean Gaussian prior defined over w using the principle of ARD [50],

$$p(w|\alpha) = (2\pi)^{-n_b/2} \prod_{j=1}^{n_b} \alpha_j^{1/2} \exp(-\frac{\alpha_j w_j^2}{2}) \tag{3.4}$$

with the inverse variance *hyperparameter* $\alpha \triangleq [\alpha_1, \ldots, \alpha_{n_b}]^\top$. Each α_j is individually controlling the effect of the prior over its associated weight w_j. During inference, many of these hyperparameters approach infinity, that is the posterior distributions of their as-

sociated weights are peaked around zero. Consequently, only a few basis functions with non-zero weights survive in the final classification model, resulting in a sparse model. The basis functions that survive are called *relevance vectors* [90], forming a sparse representation of the semantic map. The most-probable weights $w^\star$ can be estimated iteratively with an approximate inference strategy [66]:

$$w^\star = \arg\max_w \sum_{j=1}^{n_t} [t_j \log y_j + (1 - t_j) \log(1 - y_j)] - \frac{1}{2} w^\top A w, \tag{3.5}$$

where $A \triangleq (\alpha_1, \ldots, \alpha_{n_b})$ and $y_j \triangleq \sigma(y(x_j; w))$ is the prediction on the j-th observation point cloud. Then the Laplace approximation method provides a locally Gaussian approximation of the weight posterior around $w^\star \sim \mathcal{N}(\mu, \Sigma)$:

$$\mu = A^{-1} \Phi^\top (t - y), \quad \Sigma = (\Phi^\top B \Phi + A)^{-1}, \tag{3.6}$$

where $\Phi_{ni} \triangleq \phi_i(x_n)$, $B \triangleq (\beta_1, \ldots, \beta_{n_t})$, and $\beta_i \triangleq \sigma(y_i)[1 - \sigma(y_i)]$.

3.4 Compressed 3D Semantic Mapping Pipeline

As shown in Fig. 3.1, the compressed semantic map inference system comprises three steps: `compress()`, `query()` and `fusion()`. In the following, we describe each step briefly.

- `compress()`: As described in (3.5) and (3.6) in Section 3.3, `compress()` takes prior semantic predictions for the point cloud as the input, and outputs the sparse relevance vectors for each class as the compressed semantic map representation. The prior semantic predictions as existing observations either come from back-projecting 2D image segmentation to 3D or from raw 3D point cloud segmentation. These observations are used to train the internal relevance vector machine for each frame, and infer the weight parameters of the corresponding classification model. The resulting sparse relevance vectors store the semantic information of the entire map.

- `query()`: In order to generate a semantic voxel map with a certain resolution, we query at occupied locations in an occupancy grid map using the classification prediction model in (3.1) obtained from `compress()`. Thus, the input of this stage can be the centroids of the occupied voxels (if we discard all the priors to reduce memory consumption), or simply the points in the previous observations (for maximum possible accuracy). Given the spatial coordinates of those query points, we obtain the

probability of each class. At the end of this stage, we can already build a posterior semantic point cloud map.

- `fusion()`: With the resolution chosen as above, we can fuse the prediction results from `query()` for sequential frames using grid mapping systems such as OctoMap [14] and scrolling occupancy grids in [91]. To integrate semantic information into the grid, we store the distribution for each class in all the voxels and adopt a recursive counted averaging label update. A voxel's probability distribution of semantic labels is obtained as follows: Given n points that already fall into the voxel, with its current label distribution being $\mathbf{p}^{(n)}$, when a new point with label distribution of $\mathbf{p}_{\mathsf{new}}$ falls into this voxel, the updated distribution for the voxel can be computed as $\mathbf{p}^{(n+1)} \propto n\mathbf{p}^{(n)} + \mathbf{p}_{\mathsf{new}}$. Finally, the color of each voxel is decided by the label with maximum probability.

3.5 Results and Discussion

We evaluate the semantic prediction performance and multi-resolution query ability of the proposed semantic mapping method with two publicly available datasets, dense pointcloud from the stereo camera on KITTI [67] and sparse pointcloud from the laser scans on NCLT [68].

3.5.1 Evaluation Metrics

To evaluate the semantic prediction accuracy of RVSM and compare with the state-of-the-art semantic mapping systems, three metrics are used to quantify the results, i.e., *Recall (Sensitivity)* $= \frac{\mathrm{TP}}{\mathrm{TP+FN}}$, *Precision* $= \frac{\mathrm{TP}}{\mathrm{TP+FP}}$, and *Mean Intersection over Union (mIoU)* $= \frac{\mathrm{TP}}{\mathrm{TP+FN+FP}}$. We calculate these metrics for each semantic class and report the class average results.

3.5.2 KITTI Dataset

3.5.2.1 Experimental Setup

The KITTI odometry dataset consists of 22 outdoor-scene stereo sequences recorded by a sensor mounted on a driving vehicle. We evaluate our method on 25 test images from sequence 15 with labelled data [5].

We choose a CRF optimized 3D semantic occupancy mapping system [2] as our baseline, and quantitatively compare our results with it. The baseline back-projects a 2D con-

Table 3.1: KITTI Experiment RVM Model Selection: We run sequential RVM inference for 500 iterations per class on 0.01 of the training data. The current kernel and hyperparameters are chosen by using a sample data and training a Gaussian process [9].

Kernel	Length-Scales (x, y, z)	Signal Variance	Iterations	Downsample Rate
Matern ($\nu = 5$) [9]	$(1.3452, 1.2746, 1.9771)$ m	2.2133	500	0.01

volutional neural network segmentation to 3D as the prior prediction, and directly fuses the prior from multiple views into a scrolling occupancy grid map [91] by a Bayesian update procedure. After the grid map is built, higher-order CRFs are utilized to optimize each grid's label with the assumption that the labels of voxels corresponding to a superpixel are consistent.

In our comparison, we adopt the same data pre-processing as the baseline. We first use LIBELAS [92] to generate dense and accurate depth maps from rectified graylevel stereo pairs. By projecting depth maps to 3D space, discarding points further than $40m$ away, we obtain dense point clouds as the training data for RVSM. To obtain the prior, we use the same network as adopted by the baseline, Dilated CNN [70], which is trained on KITTI data from another sequence, and back-project the 2D segmentation to 3D. After the relevance vector machine is trained for each frame, we attach each relevance vector machine to a pose obtained using ORBSLAM2 Stereo [93]. At this point, we have the compressed and continuous semantic map representation RVSM in the global coordinate frame. For evaluation, we uncompress the RVSM by fusing the query points into the scrolling occupancy grid map. As described in Section 3.4, queries can be done at any occupied location with any resolution.

We compare the prediction results for 7 common semantic classes, i.e., *building, vegetation, car, road, fence, sidewalk, pole*. As our baseline, we project all labelled grids onto the camera image plane, ignoring grids that are further than $40m$ away from the camera, and evaluate them using 2D ground truth labels. Sky labels are discarded when evaluating due to the inaccurate depth values. RVM model selection in this experiment is described in Table 3.1.

3.5.2.2 Qualitative Results

We demonstrate the advantage of RVSM over the semantic prior map (directly transferring labels from 2D to 3D), and a semantic map optimized by CRF [2]. Fig. 3.2 is generated by projecting the grid map into the camera image plane. Due to the small size of the sign and the color similarity of the pole and building in the raw RGB images, the prior map is

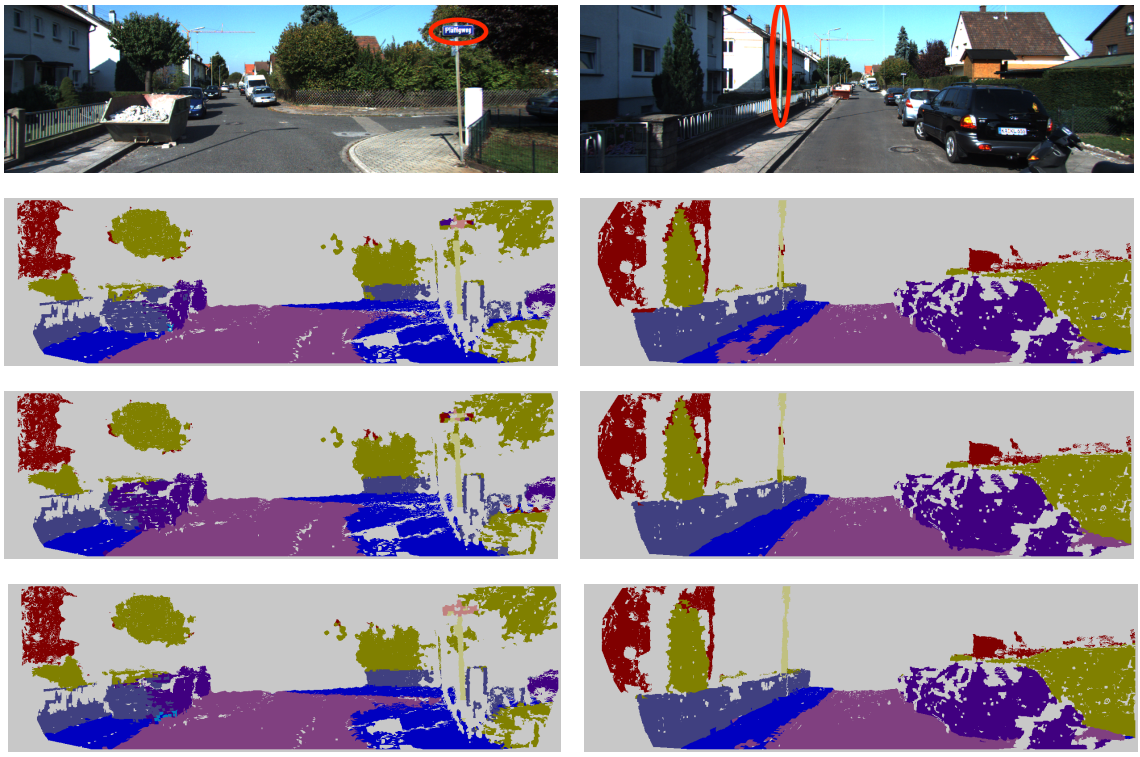


Figure 3.2: Example of KITTI Sequence 15 after projecting the occupied voxels from the semantically fused grid map to the camera image plane: From the top down are RGB image, prior result, CRF [2] result and RVSM result, respectively. RVSM is able to capture the traffic sign in the left image, and the pole in the right image.

unable to discriminate them from the background. The CRF optimized map is also unable to recover the correct labels completely due to inaccurate superpixel segmentation when the objects become small in the input images. RVSM, on the other hand, which builds the inference model before fusing the semantics and discretizing the space, successfully recovers the correct labels for even detailed objects.

We also show the continuity of RVSM in Fig. 3.3 by querying voxel centroids of grid maps with multiple resolutions. We build the RVSM from prior prediction for one frame. After the RVSM is built, the semantic information of the scene is compressed in the format of several relevance vectors. Due to RVSM's continuity property, we can build grid maps afterwards with any desired resolution depending on the applications.

3.5.2.3 Quantitative Results

In this section, we quantitatively evaluate the proposed technique in terms of overall performance, multi-resolution performance, and compression ability, as well as comparing it

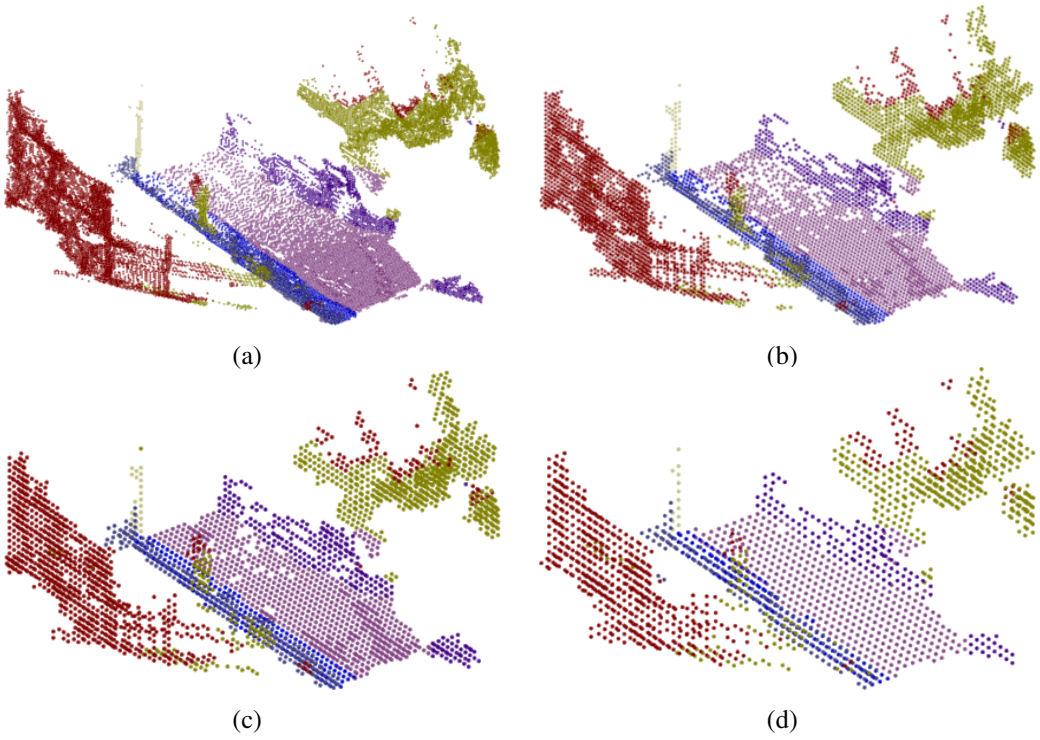


Figure 3.3: Query for different resolution. From (a) to (d): map resolution changes from 0.1 m to 0.4 m.

with other approaches.

Comparison on overall performance: We compare the overall semantic prediction performance with the prior map and the CRF map [2] for each class using three metrics: recall, precision and IoU. We build three grid maps for the first 100 frames in sequence 15, and project the grids into 25 images with ground truth for evaluation, discarding points that are further than $40m$ away from the camera for all approaches. For CRF map, we use the code provided by the authors. The quantitative results are given in Table 3.2. *Average* represents the mean over all the classes. Although the RVSM is compressed, it still outperforms other maps on average recall and mean IoU while has comparable precision.

Comparison on multi-resolution performance: Using the same setup as above, we change different grid sizes, ranging from $0.1m$ to $1m$ when fusing semantic query results for the occupancy map for the prior, the baseline and RVSM, as shown in Fig. 3.4. The RVSM compression is done once and the relevance vectors are held fixed for different resolutions, while in the baseline's result one CRF post-processing is applied for each different resolution. Except for a grid size of $0.9m$, RVSM demonstrates better overall IoU on all other grid resolutions.

Compression evaluation: Table 3.3 shows the compression and query results for the

Table 3.2: Quantitative results on KITTI odometry sequence 15 test set [5] for 7 semantic classes, containing 25 ground truth images. The query points of RVSM are chosen to be at all the observed prior locations.

Metric	Method	Building	Vege.	Car	Road	Fence	Sidewalk	Pole	Average
Recall	Prior Map	98.2	**99.2**	98.6	91.4	**96.3**	82.8	58.2	89.2
	CRF Map	98.2	98.5	**98.7**	91.6	96.0	**86.0**	58.0	89.6
	RVSM	97.7	98.9	98.2	**94.5**	96.0	85.0	**70.0**	**91.4**
Precision	Prior Map	86.5	88.9	**81.0**	74.5	**76.5**	76.7	**58.0**	77.4
	CRF Map	88.3	**91.4**	80.5	76.1	73.0	76.8	56.7	**77.6**
	RVSM	**88.4**	90.6	80.6	**77.1**	75.1	**77.2**	51.6	77.2
IoU	Prior Map	85.2	88.3	**80.0**	69.6	**74.3**	66.2	40.9	72.1
	CRF Map	**86.9**	**90.2**	79.7	71.2	70.9	68.2	40.2	72.5
	RVSM	86.6	89.7	79.4	**73.9**	72.8	**70.0**	**42.3**	**73.3**

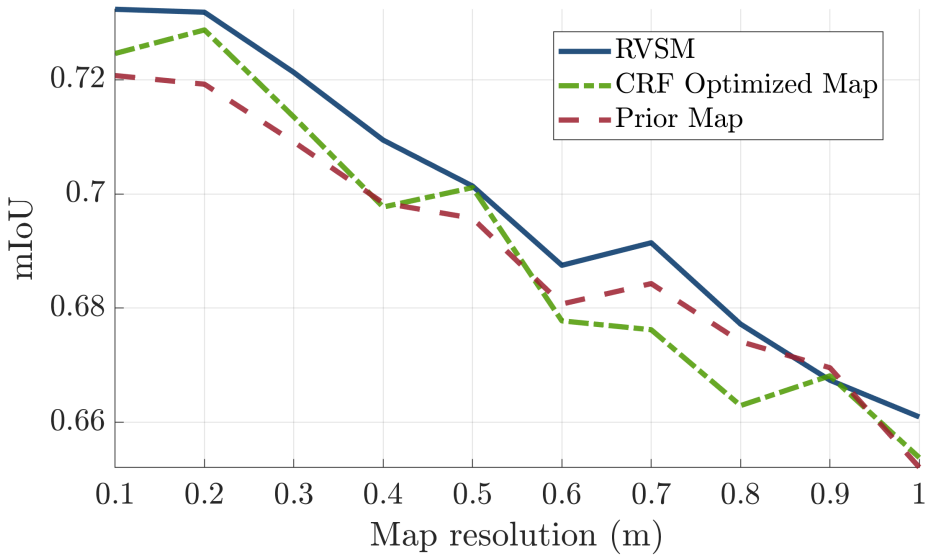


Figure 3.4: mIoU at different resolutions for Prior, CRF, and RVSM. RVSM's only does one query on the prior points' spatial coordinates and generates the map for all the resolutions. Here the resolution means the size of the voxel. When the resolution decreases, more points (possibly with different labels) fall into one voxel, resulting in a lower mIoU value. The larger the voxels are, the more loss of information comes from CRF's discretization. On the contrary, RVSM performs compression before voxel map fusion, which preserves local semantic consistency better as opposed to post-processing.

Table 3.3: KITTI sequence 15 compression test for the frame 0-30. The mIoU is evaluated on the 15 ground truth images. In the compression experiment, query points are the centroids of the occupancy grid. Priors are discarded after the relevance vectors are obtained, thus RVSM does not have access to prior locations, but only to the positions of centroids in the occupancy grid.

Method	mIoU	No. of occupied voxels	No. of points/ voxels with labels	(Estimated) Size of semantic voxel maps
Prior Map	73.6	2499662	2499662	109.985MB
RVSM	72.2	2499662	473	40.002MB

first 31 frames in KITTI Sequence 15. Every third frame is used as a prior; we transform them into the global frame for RVSM compression. After obtaining 473 relevance vectors for all classes, we query on the voxel centroids of the corresponding occupancy map of resolution $0.1m$ for every frame, and store the query results into the scrolling semantic occupancy map. In the table, the mIoU is evaluated on the 15 images that have ground truth of the total 31 prior images. The number of occupied voxels is computed from the scrolling occupancy map, counting how many of the voxels have adequate occupancy log likelihood. If the map fusion works directly on the priors such as the baseline, we would need to use the same number of voxels to store the distribution of all the classes in the map, one for each voxel. In contrast, RVSM only stores the relevance vectors, resulting in a more compact way of storing semantic information.

The estimated size of the semantic map is computed as follows:

- N_o: Number of occupied grid cells. Each grid cell has 4 floating point numbers, $(x, y, z, \text{occupancy})$
- n_f: Size of a floating point (in Bytes)
- N_s: Number of grid cells that have semantic label distribution
- n_c: Number of semantic classes
- N_{rv}: Number of relevance vectors.

With these definitions, the size of the semantic maps fused directly from the prior can be computed by $N_o \times (n_f \times 4) + N_s \times (n_f \times n_c)$. Meanwhile, the size of the semantic maps generated by fusing RVSM queries is $N_o \times (n_f \times 4) + N_{rv} \times (n_f \times 4)$. The second term is changed because RVSM computes the class of a voxel by (3.1), while each relevance vector v has 4 floating base numbers: (x_v, y_v, z_v) and weight w_v.

Table 3.4: NCLT Experiment RVM Model Selection: We run sequential RVM inference for 1000 iterations per class on 0.05 of the training data. The current kernel and hyperparameters are chosen by using a sample data and training a Gaussian process [9].

Kernel	Length-Scales (x, y, z)	Signal Variance	Iterations	Downsample Rate
Matern ($\nu = 5$) [9]	$(3.8096, 3.8096, 1.9090)$ m	0.8	1000	0.05

On this test sequence, RVSM has over 60% less memory consumption than directly fusing the map, at the cost of 1.4% mIoU drop. This is an interesting tradeoff when building autonomous systems that are much smaller than a passenger car, such as a bipedal robot.

3.5.3 NCLT Dataset

To evaluate the RVSM's performance on a sparse LiDAR point cloud, we use the NCLT dataset [68]. It contains 27 sequences collected on the University of Michigan's North Campus, with synchronized Velodyne HDL-32E 3D LiDAR, an omnidirectional Ladybug3 camera, and proprioceptive odometry sensors [68]. To obtain ground truth semantic labels, we deployed LabelMe [94] annotation tools on Amazon Mechanical Turk and labeled 1194 training and 457 validation segmented images from NCLT's Ladybug3 camera. Each image frame is synchronized with one laser scan, covering the full 360° field of view. Projection of LiDAR points onto the segmented images yields their labels. The labeled classes are: *background, water, road, sidewalk, terrain, building, vegetation, car, person, bike, pole, stair, sign, and sky*. We perform a qualitative experiment on a subset of the 2012-04-29 sequence, as shown in Fig. 3.5.

The data processing pipeline is similar to the KITTI experiment except for the way of obtaining priors and the tool used for map fusion. The prior distributions of the LiDAR pointcloud are obtained from a 2D deep neural network generating predictions from the omnidirectional camera. Starting from the pretrained weight from Cityscapes, we fine-tune the 2D segmentation network DeeplabV3+ [69] on the training set and reached 0.553 mIoU on the 2D validation set. Given NCLT's camera-LiDAR transformation, for each LiDAR scan, we project it onto the five image planes of the Ladybug camera, so as to obtain the LiDAR point's prior distribution. Transformed by NCLT's ground truth poses, these priors are compressed into RVSM and then fused into a semantic OctoMap, which is the regular occupancy OctoMap [14] equipped with Bayesian label fusion. RVM model selection for NCLT experiment is given in Table 3.4.

Figure 3.5 shows a satellite view of the route from Google Earth and the corresponding RVSM predictions. Here we use a grid size of $0.2m$ for the semantic OctoMap. The result

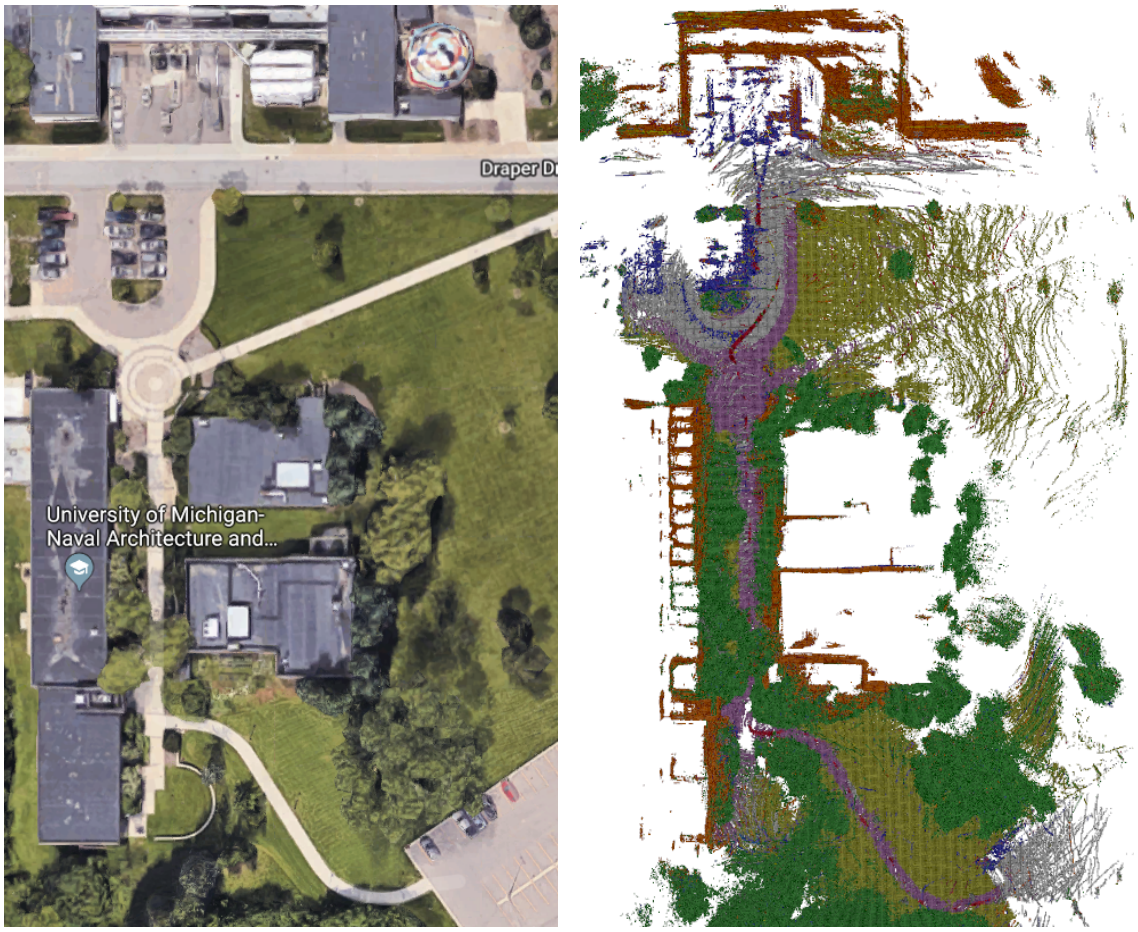


Figure 3.5: RVSM semantic map fusion on NCLT subsequence 2012-04-29. Left: The satellite view of the path from Google Earth. Right: Semantic OctoMap fusion result after querying from RVSM.

shows the ability of RVSM to recognize and reconstruct most of the categories in a large-scale mapping application. It also demonstrates the feasibility of RVSM on sparse LiDAR data.

3.6 Conclusion and Future Work

We proposed a compressed 3D semantic map representation that exploits sparse Bayesian modeling and a pipeline to build 3D semantic occupancy grid maps. We compared the prediction performance with other 3D semantic mapping systems and showed the continuity and compression properties of the proposed map representation. The experiments showed the proposed RVSM has comparable performance with the state-of-the-art techniques while requiring a lower memory footprint. The proposed semantic mapping system can be used

in robotic applications where the semantic prediction needs to be made efficiently with limited onboard resources.

In the future, we are interested in generalizing the likelihood to the multinomial distribution to integrate an inherently multi-class classifier into the RVSM framework. The optimization of the hyperparameters and model selection (selection of the best kernel) are also interesting future directions. In the current implementation, the hyperparameters (parameters of the kernel) are optimized using a sample dataset and training a Gaussian process. While this method produces the presented results, the joint optimization of the hyperparameters during RVM training is more convenient.

CHAPTER 4

Bayesian Spatial Kernel Smoothing for Scalable Dense Semantic Mapping

This chapter develops a Bayesian continuous 3D semantic occupancy map from noisy point clouds by generalizing the Bayesian kernel inference model for building occupancy maps, a binary problem, to semantic maps, a multi-class problem. The proposed method provides a unified probabilistic model for both occupancy and semantic probabilities and nicely reverts to the original occupancy mapping framework when only one occupied class exists in obtained measurements. The Bayesian spatial kernel inference relaxes the independent grid assumption and brings smoothness and continuity to the map inference, enabling to exploit local correlations present in the environment and increasing the performance. The accompanying software uses multi-threading and vectorization, and runs at about 2 Hz on a laptop CPU. Evaluations using multiple sequences of stereo camera and LiDAR datasets show that the proposed method consistently outperforms current baselines. We also present a qualitative evaluation using data collected with a bipedal robot platform on the University of Michigan - North Campus.

4.1 Introduction

Robotic mapping is the problem of inferring a representation of the robot's surroundings using noisy measurements as it navigates through an environment. This problem is traditionally solved using occupancy grid mapping techniques [14, 18, 19]. As robotic systems move toward more challenging behaviors in more complex scenarios, such systems require richer maps so that the robot understands the significance of the scene and objects within. Hence, the integration of semantic knowledge into the map has been the focus of robotic research in recent years [2, 46, 47, 95–97].

A semantic occupancy map as shown in Fig. 4.1, besides possessing properties similar to an occupancy grid map, maintains for each cell a set of probabilities of semantic classes.

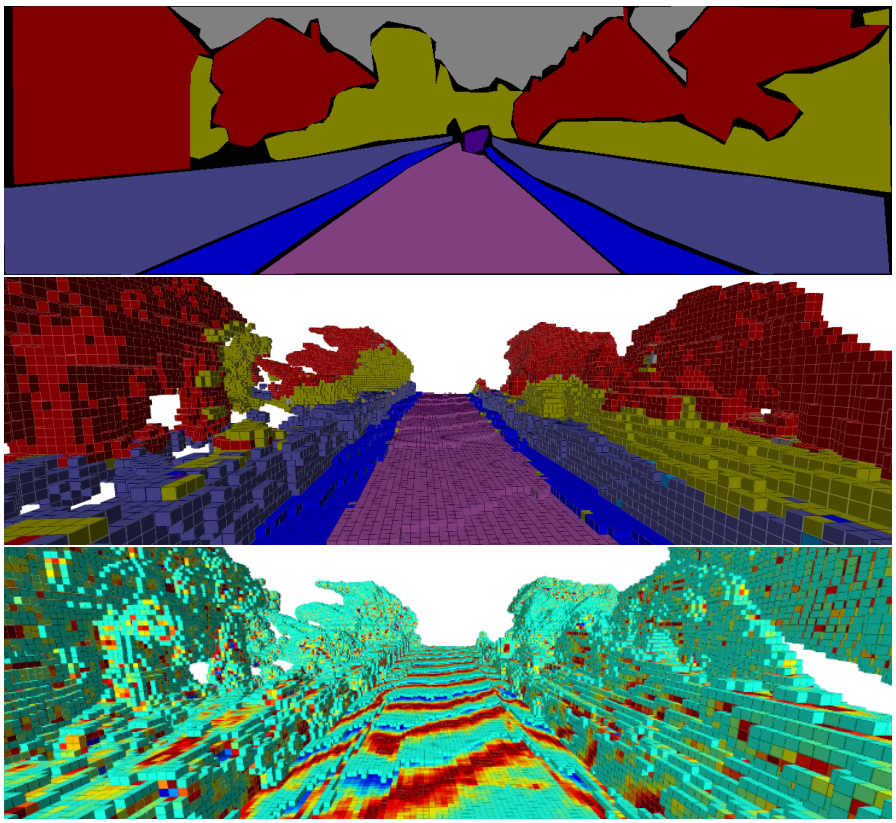


Figure 4.1: Qualitative results on KITTI odometry sequence 05 dataset [3]. From top to bottom the figures show the 2D ground truth image, 3D semantic map, and variance map.

These probabilities are often updated using a Bayes filter [2,81], and then Conditional Random Fields (CRF) or Markov Random Fields (MRF) are subsequently applied to mitigate discontinuities and inconsistencies in the semantic map [2,46,47,85,87]. In principle, CRF models encourage label consistency among neighboring grids in super-voxels [46] or 2D superpixels [2, 87]. However, CRF optimization is only applied as a post-processing step, and therefore, it is unable to predict semantics of partially observed regions in the map.

Occupancy grid maps assume the grids are statistically independent. However, a series of investigations on continuous occupancy mapping shows that taking local spatial correlations into account increases mapping performance [98–105]. Building on a similar idea, continuous semantic maps [106, 107] can deal with sparse sensor measurements by inferring semantics of partially observed regions from neighboring measurements. Recent work on Bayesian generalized kernel inference for occupancy map prediction (BGKOctoMap) proposed in [105] uses a kernel inference approach to generalize the counting sensor model [108] to continuous maps while maintaining the scalability of the method.

In this chapter, we extend BGKOctoMap [105] to continuous semantic mapping where

the inference reverts to the original framework when only one occupied class exists. In particular, the contributions of this chapter are:

1. We develop a continuous statistical model for semantic occupancy mapping which models occupancy and semantic probabilities in a unified framework and queries can be made at any resolution;

2. We provide an open-source implementation of the proposed method. The current implementation exploits multi-threading and vectorization and can be run at about 2 Hz using a laptop CPU;

3. We present extensive experiments using both stereo camera and LiDAR data. The evaluations show that the proposed method consistently outperforms state-of-the-art systems.

Related work is given in Section 4.2. Section 4.3 presents preliminaries and semantic counting sensor model. Section 4.4 describes an extension to continuous mapping. Experimental results are presented in Section 4.5. Limitations of this work are discussed in Section 4.6 and Section 4.7 concludes the chapter.

4.2 Related Work

Discrete 3D Semantic Mapping. Early semantic mapping work uses traditional pixel-wise image segmentation methods and directly transfers image labels from 2D to 3D. Labels from multiple images are fused in 3D without any further 3D optimization [5, 80, 81]. He et al. [80] build a semantic octomap by using an MRF for image segmentation and selecting the most frequent label of the 3D points inside each grid. Sengupta et al. [5] build a semantic volumetric map by adopting a CRF for 2D semantic segmentation and assigning labels by a voting scheme. Stückler et al. [81] use random decision forests to segment object classes in images and fuse soft labels in a voxel-based 3D map using a Bayesian update. While these methods are similar to our semantic counting sensor model in a way that the maximum of semantic labels in a 3D element is picked in label fusion, the latter is a closed-form Bayesian inference which outputs the mean and variance of the posterior.

To deal with noisy 2D predictions, 3D CRF optimization has been introduced as a refinement technique and it is widely applied in 3D semantic mapping [47, 85, 109]. In [3, 46, 87], a higher-order dense CRF model is used to further optimize the semantic predictions for 3D elements. Basic CRF models encourage label consistency for adjacent 3D

elements, while higher-order dense CRFs can model long-range relationships within a region, such as grids in super-voxels [46] or grids corresponding to 2D superpixels [87], and further improve the mapping performance. More recent work uses deep Convolutional Neural Networks (CNNs) for 2D image segmentation, and follows the same framework for building 3D semantic maps [2, 110]. However, CRF optimization post-processes the inferred occupied grids, which does not change the principle of discrete semantic map inference. In [111], a semantic Simultaneous Localization and Mapping (SLAM) system, SuMa++, builds a surfel-based semantic map using SemanticKITTI dataset [6] as its byproduct. However, surfel-based maps do not model occupied or free space, thus are not used for robot navigation.

Continuous Mapping. Gaussian Process Occupancy Map (GPOM) [98] takes into account the correlation between map points and treats the map inference as a binary classification at an arbitrary resolution. Hilbert maps [102] are more scalable and can be updated in linear time where a logistic regression classifier is trained online through stochastic gradient descent. GPOM has been extended from binary to multi-class case in [106]. However, the complexity of the model grows with the number of data points, resulting in $\mathcal{O}(n^3)$ cost without approximation. The cost also grows with the number of semantic classes as a one-vs.-rest approach is used to build the multi-class classifier. Similarly, Hilbert maps can also be extended to the multi-class maps using a multinomial model. However, as discussed in [105], the logistic regression classifier used by Hilbert map-based approaches does not provide associated uncertainties in probability estimates.

Bayesian Kernel Inference. Bayesian Kernel Inference (BKI) was introduced in [112] as an approximation to Gaussian processes that requires only $\mathcal{O}(\log N)$ computations instead of $\mathcal{O}(N^3)$, where N is the number of training points. It generalizes local kernel estimation to the context of Bayesian inference for the exponential family of distributions. Instead of approximating inference on the model, the approximation is made at the stage of model selection. Assuming latent training parameters are conditionally independent given the target parameters, exact inference on this model is possible for any likelihood function from the exponential family. In [113], BKI is successfully applied to a visual odometry problem for modeling sensor uncertainty. In [114], BKI has been used on a Bernoulli-distributed random event with Beta-distributed prior to model collision in safe high-speed navigation problems and could achieve safe behavior in a novel environment with no relevant training data. BKI was first used in the context of mapping problems in [104, 105], to generalize the discrete counting sensor model [108] to continuous occupancy mapping. Later, the applications of BKI in elevation regression and traversability classification are explored in [115]. Inspired by these works, we apply BKI in our semantic counting sensor

model and generalize it to continuous semantic mapping. In particular, we use BKI on a Categorical likelihood with a Dirichlet distribution as its conjugate prior.

4.3 Preliminaries and Semantic Counting Sensor Model

The counting sensor model describes occupancy probability via a Bernoulli likelihood function. It counts for each grid how often a beam has ended in that grid and how often a beam has passed through it. This model has comparable performance to Bayesian updates in occupancy grid mapping [116]. The semantic counting sensor model is its natural generalization from occupancy (binary) mapping to semantic (multi-class) mapping.

Let $\mathcal{K} = \{1, 2, ..., K\}$ be the set of semantic class labels, i.e., K categories, and $\mathcal{X} \subset \mathbb{R}^3$ be the map spatial support. For any map point $x_i \in \mathcal{X}$, we have a one-hot-encoded measurement tuple $y_i = (y_i^1, ..., y_i^K)$, where $y_i^k \geq 0$ and $\sum_{k=1}^K y_i^k = 1$. In practice, y_i is the output of a max function computed using the output of a deep network for multi-class classification. The training set (data) can be defined as $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^N$.

Assuming map cells are indexed by $j \in \mathbb{Z}^+$, the jth map cell can take on one of K possible categories with the probability of each category separately specified as $\theta_j = (\theta_j^1, ..., \theta_j^K)$, where $\sum_{k=1}^K \theta_j^k = 1$. The jth map cell with semantic probability θ_j is described by a Categorical distribution as:

$$p(y_i|\theta_j) = \prod_{k=1}^{K} \left(\theta_j^k\right)^{y_i^k}. \tag{4.1}$$

In semantic mapping, we seek the posterior over θ_j; $p(\theta_j|\mathcal{D})$.

For incremental Bayesian inference, we adopt a Dirichlet prior distribution over θ_j, given by $Dir(K, \alpha_0)$, as the conjugate prior of the Categorical likelihood, where $\alpha_0 = (\alpha_0^1, ..., \alpha_0^K)$, $\alpha_0^k \in \mathbb{R}^+$ are concentration parameters (hyperparameters). Applying Bayes' rule, the posterior is given by $Dir(K, \alpha_j)$, $\alpha_j = (\alpha_j^1, ..., \alpha_j^K)$, where α_j^k is

$$\alpha_j^k := \alpha_0^k + \sum_{i,\ x_i \text{ in cell } j} y_i^k. \tag{4.2}$$

Because α_j^k counts the number of measurements which falls into the jth cell and indicate the kth category, we call this model the Semantic Counting Sensor Model (S-CSM). Given concentration parameters α_j, the mode of θ_j has the following closed form, which is

also the maximum-a-posteriori estimate of θ_j:

$$\hat{\theta}_j^k = \frac{\alpha_j^k - 1}{\sum_{k=1}^K \alpha_j^k - K} \text{ and } \alpha_j^k > 1. \tag{4.3}$$

We also have the closed-form expected value and variance of θ_j as follows:

$$\mathbb{E}[\theta_j^k] = \frac{\alpha_j^k}{\sum_{k=1}^K \alpha_j^k} \text{ and } \mathbb{V}[\theta_j^k] = \frac{\frac{\alpha_j^k}{\sum_{k=1}^K \alpha_j^k}(1 - \frac{\alpha_j^k}{\sum_{k=1}^K \alpha_j^k})}{\sum_{k=1}^K \alpha_j^k + 1}. \tag{4.4}$$

We use (4.2) to calculate the parameters of the posterior Dirichlet distribution for cell j and given the posterior parameter α_j, the statistics of cell j can be computed by (4.3) and (4.4).

For free-class measurements, we use free-space points linearly interpolated along each sensor beam. We note that in the particular case when $K = 1$ represents the free-space class and $K = 2$ represents the occupied class, the semantic counting sensor model nicely reverts to the original counting sensor model.

However, the semantic counting sensor model inherits the traditional occupancy grid mapping limitations because the posterior parameters for each cell are only correlated with measurements that directly fall into or pass through that cell. To mitigate this shortcoming, we use BKI to convert the discrete semantic counting sensor model to a continuous model by taking into account local correlations in the map.

4.4 Continuous Semantic Mapping via Bayesian Kernel Inference

Bayesian kernel inference, as introduced by Vega-Brown et al. [112], relates the extended likelihood $p(y_i|\theta_*, x_i, x_*)$ and the likelihood $p(y_i|\theta_i)$ by a smoothness constraint, where θ_* is the value of the latent variable for the query point x_*. In this framework, the maximum entropy distribution g, satisfying $D_{\text{KL}}(g\|f)$, has the form $g(y) \propto f(y)^{k(x_*,x)}$, where $D_{\text{KL}}(\cdot\|\cdot)$ is the Kullback-Leibler Divergence (KLD), and $k(\cdot,\cdot)$ is a kernel function. Let g be the extended likelihood and f the likelihood, we define a smooth distribution over semantics as having bounded KLD between the two distributions. Given a kernel function operating on 3D spatial inputs $k : \mathcal{X} \times \mathcal{X} \to [0, 1]$, we have

$$\prod_{i=1}^N p(y_i|\theta_*, x_i, x_*) \propto \prod_{i=1}^N p(y_i|\theta_*)^{k(x_*,x_i)}. \tag{4.5}$$

Using Bayes' rule, we can write

$$p(\theta_*|x_*, \mathcal{D}) \propto p(\mathcal{D}|\theta_*, x_*)p(\theta_*|x_*), \tag{4.6}$$

and by substituting (4.5) into (4.6), we have:

$$p(\theta_*|x_*, \mathcal{D}) \propto \left[\prod_{i=1}^{N} p(y_i|\theta_*)^{k(x_*,x_i)}\right] p(\theta_*|x_*). \tag{4.7}$$

We adopt the Categorical likelihood and place a prior distribution $Dir(K, \alpha_0)$ over θ_*. Subsequently, (4.6) becomes:

$$\begin{aligned} p(\theta_*|x_*, \mathcal{D}) &\propto \left[\prod_{i=1}^{N}\left[\prod_{k=1}^{K}\left(\theta_*^k\right)^{y_i^k}\right]^{k(x_*,x_i)}\right]\prod_{k=1}^{K}\left(\theta_*^k\right)^{\alpha_0^k-1} \\ &= \prod_{k=1}^{K}\left(\theta_*^k\right)^{\alpha_0^k+\sum_{i=1}^{N} y_i^k k(x_*,x_i)-1}, \end{aligned} \tag{4.8}$$

which is proportional to the posterior $Dir(K, \alpha_*)$ where $\alpha_* = (\alpha_*^1, ..., \alpha_*^K)$ is defined as

$$\alpha_*^k := \alpha_0^k + \sum_{i=1}^{N} k(x_*, x_i) y_i^k. \tag{4.9}$$

The mode, mean, and variance for the continuous model can be computed exactly as given in (4.3) and (4.4).

Compared with (4.2), (4.9) not only considers measurements which fall into a cell but also adjacent measurements with a weighting coefficient defined by the kernel function, i.e., the distance to the query point. We note that the kernel neither needs to be positive-definite nor symmetric. To reduce the computational complexity, we choose the sparse kernel [117] as

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

where $d = \|x - x'\|$, $l > 0$ is the length-scale, and σ_0 is kernel scale parameter (signal variance).

The derived continuous semantic model can deal with sparse and noisy sensor measurements better and allows for queries at an arbitrary resolution. In the context of semantic occupancy mapping, the query points are chosen to be the grid centroids. Thus, (4.9) can be

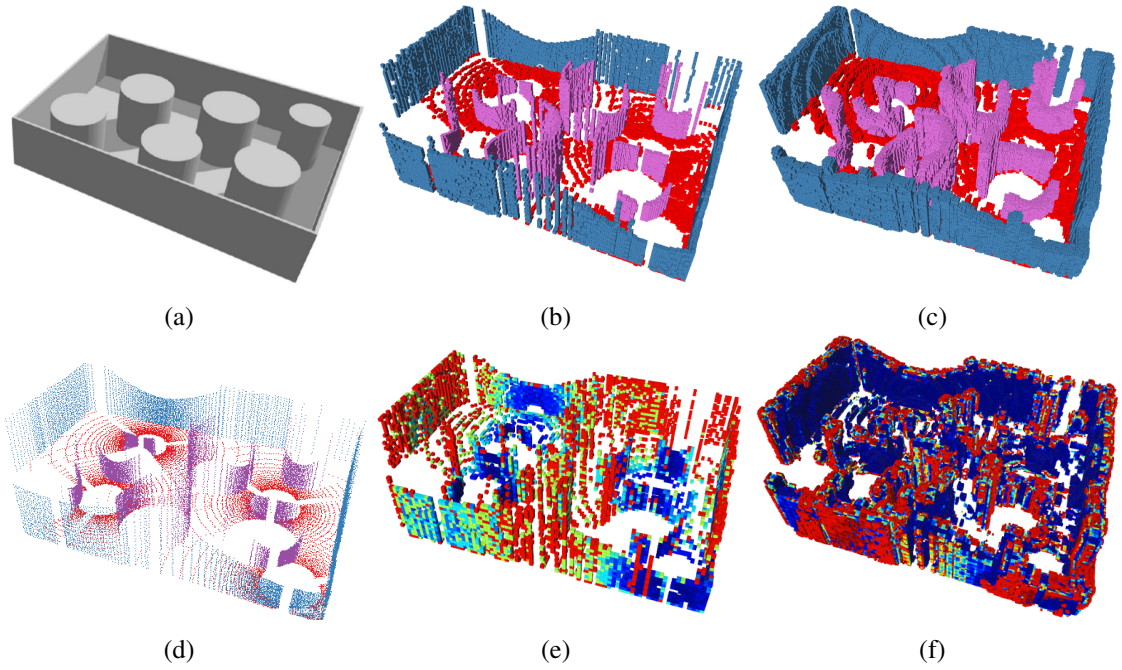


Figure 4.2: 3D toy example on a simulated dataset. (a) Environment model in Gazebo. (d) Annotated point cloud raw data. (b), (e) Semantic and variance map of S-CSM. (c), (f) Semantic and variance map of S-BKI. Variance maps of two models (shown using the jet colormap) provide useful information for robotic navigation and exploration [4]. We found that Bayesian kernel inference decreases the variance of the wall by considering neighboring measurements. There are some artifacts, however, on the periphery of the wall where the variance is relatively high.

used to recursively update the posterior parameters for each grid. We use a block to contain a number of grids according to the block depth, where each block is an octree of grids. For every block of test data, the corresponding training data is comprised of all portions of the new measurements that pass through the block's extended block [101], which is defined as the set of neighboring blocks with faces adjacent to the block containing the test data of interest.

Example 2 (Three-dimensional Toy Example) *Figure 4.2 illustrates a three-dimensional toy example of the continuous semantic mapping via Bayesian kernel inference using a simulated dataset made in Gazebo, with annotated semantic labels. The simulated dataset has dimensions* $10.0 \times 7.0 \times 2.0\mathrm{m}$. *We manually annotate the raw data into three semantic classes: ground, wall, and cylindrical obstacles. Semantic occupancy maps with resolution 0.05* m *for both S-CSM and Semantic Bayesian Kernel Inference (S-BKI) models are built using the annotated point clouds as sensor measurements. The figure shows that S-CSM can reconstruct the 3D environment with correct semantic information but has a limited*

Table 4.1: Kernel and Dirichlet prior hyperparameters for all experiments.

Hyperparameter	Description	Value
l	Kernel length-scale	0.3 m
σ_0	Kernel scale	0.1
α_0^k	Dirichlet prior	0.001

predictive capability where sensor coverage is sparse. The S-BKI map can interpolate the gaps in the walls due to the continuity and smoothness of Bayesian kernel inference.

4.5 Experimental Results

We now present experiments for evaluating semantic segmentation accuracy, occupancy prediction accuracy, and the impact of parameters using multiple real datasets. We also compare the proposed methods with state-of-the-art systems and present a qualitative evaluation using data collected with a bipedal robot. The C++ implementations of the proposed methods are available open source [1], and make use of the Learning-Aided 3D Mapping Library [105], the Robot Operating System (ROS) [118], and Point Cloud Library (PCL) [119]. The parameters in Table 4.1 were manually tuned but remained fixed throughout all experiments. For baselines, we used the available open-source implementations without any modification. All experiments are conducted on an Intel i7 processor with 8 cores and 32 GB RAM.

4.5.1 KITTI Dataset

KITTI dataset with semantically labeled images contains 40 test images from sequence 05 [3], and 25 test images from sequence 15 [5] in KITTI odometry dataset. We qualitatively and quantitatively compare the mapping performance of our methods with the state-of-the-art CRF-based semantic mapping system proposed by Yang et al. [2]. However, Yang's method only predicts semantic labels on occupied voxels using a discrete occupancy grid mapping algorithm. For a fair comparison with respect to the occupancy model, we implement another baseline, BGKOctoMap-CRF [2], by replacing Yang's discrete occupancy grid map with the continuous BGKOctoMap, and then applying the same hierarchical CRF model to refine the voxel labels.

[1] https://github.com/ganlumomo/BKISemanticMapping
[2] https://github.com/zeroAska/BGKOctoMap-CRF

We adopt the same data pre-processing methods as used by Yang et al. [2]. We use ELAS [120] to generate depth maps from stereo image pairs, ORB-SLAM [121] to estimate 6DoF camera poses, and the deep network dilated CNN [122] for prior semantic label predictions. The superpixels used in Yang's CRF module and BGKOctoMap-CRF are generated by the SLIC algorithm [123]. The common parameters for occupancy mapping in all methods are set according to Yang's work: the resolution of 0.1 m, free and occupied thresholds as 0.47 and 0.6, respectively.

4.5.1.1 Qualitative Results

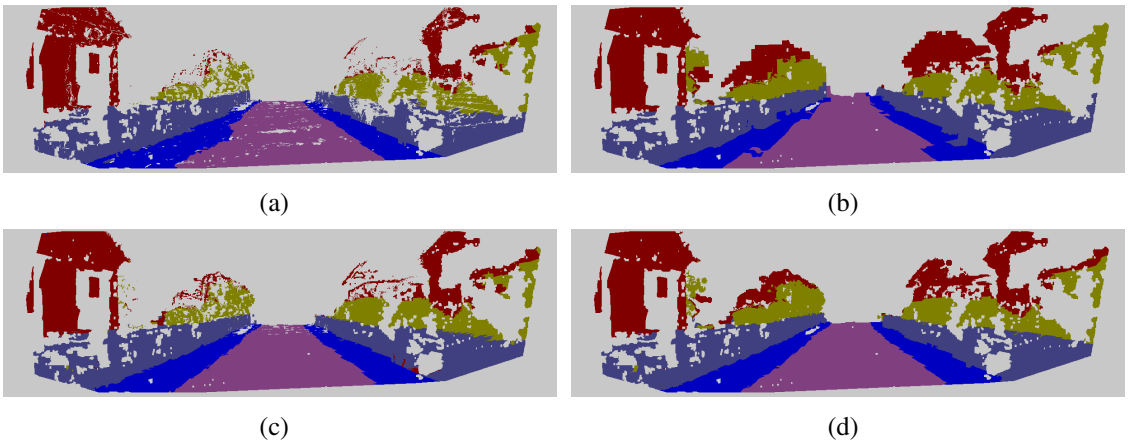
The 3D view of the semantic map built by S-BKI model is given in Fig. 4.1. Our approach is able to recognize and reconstruct general objects such as road, sidewalk, building, fence and vegetation. We also show the same view of the corresponding variance map of S-BKI in Fig. 4.1. Most of the grids on the surface have relatively low variance (cyan); the middle grids have the lowest variance (blue) where the sensor measurements are dense, while the grids on the margins of sensor scans show relatively high variance (red) where the sensor measurements are sparse. It can also be noticed that the uneven parts of the road in the semantic map have high variance, which might be caused by the discontinuity of the estimated camera poses.

We also found that a small portion of grids of the fence on the left side are misclassified as vegetation, where the corresponding variance is high. This nice property enables us to reject misclassified grids by setting a variance threshold. If the variance is too high, we can regard the state of the grid as unknown and thus build safer semantic maps for robot navigation. To compare the mapping performance, we project semantic maps onto 2D left camera views and compare with 2D ground truth images as shown in Fig. 4.3.

4.5.1.2 Quantitative Results

We follow the evaluation method given in [2] by projecting 3D semantic map onto the 2D left image plane, ignoring voxels that are too far from the camera (40 meters for all the methods), and calculating the standard metric of Intersection over Union (IoU) based on labeled ground truth left images. IoU is defined as TP/(TP+FN+FP), where T/F P/N stands for true/false positive/negative.

Yang et al. [2] *exclude* the data that has not been projected onto images (gray color in the projected images), *even when there exists corresponding ground truth data of it* (as shown in the ground truth images in Fig. 4.1). For a fair comparison, we follow this approach for all methods and call it *IoU Exclusive*. However, this evaluation ignores the classification



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

Figure 4.3: Qualitative results on KITTI odometry sequence 05 dataset [3]. From (a) to (d) the figures show 2D projected images from Yang et al. [2], BGKOctoMap-CRF, S-CSM and S-BKI, respectively. The projected image from Yang's semantic map contains more gaps than other maps, compared with the ground truth image where the road, buildings, and vegetation are continuous and dense, while the projected image of S-BKI has the least holes in those regions, which resembles the ground truth better. BGKOctoMap-CRF outperforms Yang's method, in spite of the misclassification of the sidewalk to road.

error of gaps in the map, and cannot show the advantage of continuous mapping. Therefore, we compute a more rigorous *IoU* by taking all projected data except the sky class into account.

The quantitative results are given in Table 4.2 and 4.3, where the two metrics are computed. For this experiment, the average runtime of Yang's method is 4.41 sec/scan, BGKOctoMap-CRF is 1.10 sec/scan, S-CSM is 0.75 sec/scan, and S-BKI is 0.36 sec/scan. S-BKI has the highest IoU among almost all semantic classes compared with other maps, and S-CSM is the second-best method. We reiterate that the IoU Exclusive is not a reasonable metric for mapping performance evaluations; nevertheless, S-CSM and S-BKI still outperform the compared baselines using this metric. In the latter case, as expected, S-CSM and S-BKI perform similarly.

BGKOctoMap-CRF has a higher IoU than Yang's method because of the continuous occupancy model of BGKOctoMap. The gaps in the measurements are interpolated and CRF fills the labels from adjacent voxels. S-CSM outperforms both CRF-based methods, because even if the 3D CRF model further optimizes the grid labels, it is only post-processing pre-calculated occupied grids and, therefore, it cannot recover the correct semantic labels for misclassified occupancy or unknown grids. Specifically, even if BGKOctoMap-CRF is a continuous model for occupancy, it is not continuous for semantics and color. Thus, the predicted occupied voxels might not contain observation of semantics and color, leading to

Table 4.2: Quantitative results on KITTI odometry sequence 05 test set [3] for 8 common semantic classes, containing 40 images.

Metric	Method	Building	Road	Vege.	Sidewalk	Car	Signate	Fence	Pole	Average
IoU Exclusive	Yang et al. [2]	86.2	91.5	**85.3**	74.1	**77.1**	16.8	78.5	28.0	67.2
	BGKOctoMap-CRF	86.1	88.0	82.3	73.6	71.9	15.5	73.8	27.7	64.9
	S-CSM	86.3	93.2	84.3	**80.0**	76.8	**25.5**	77.5	**30.1**	**69.2**
	S-BKI	**87.4**	**93.3**	84.7	79.9	76.9	18.6	**78.7**	29.2	68.6
IoU	Yang et al. [2]	32.5	70.1	45.2	55.7	39.5	13.0	46.6	18.9	40.2
	BGKOctoMap-CRF	43.5	70.9	49.4	55.5	40.2	12.7	46.4	13.9	41.6
	S-CSM	40.2	74.1	49.5	62.1	42.1	**20.3**	47.7	22.8	44.9
	S-BKI	**45.6**	**75.5**	**52.8**	**62.9**	**43.3**	14.9	**49.3**	**22.9**	**46.0**

Table 4.3: Quantitative results on KITTI odometry sequence 15 test set [5] for 8 common semantic classes, containing 25 images.

Metric	Method	Building	Road	Vege.	Sidewalk	Car	Signate	Fence	Pole	Average
IoU Exclusive	Yang et al. [2]	**95.6**	90.4	**92.8**	70.0	94.4	0.1	**84.5**	49.5	72.2
	BGKOctoMap-CRF	94.7	93.8	90.2	81.1	92.9	0.0	78.0	49.7	72.5
	S-CSM	94.4	**95.4**	90.7	**84.5**	95.0	22.2	79.3	**51.6**	76.6
	S-BKI	94.6	**95.4**	90.4	84.2	**95.1**	**27.1**	79.3	51.3	**77.2**
IoU	Yang et al. [2]	32.9	85.8	59.0	**79.3**	61.0	0.9	46.8	33.9	50.0
	BGKOctoMap-CRF	**50.0**	86.6	64.1	74.9	61.0	0.0	47.5	**36.7**	52.6
	S-CSM	42.6	87.3	62.9	77.9	62.6	17.1	47.7	34.8	54.1
	S-BKI	49.3	**88.8**	**69.1**	78.2	**63.6**	**22.0**	**49.3**	**36.7**	**57.1**

improper initialization of them for CRF potentials. In contrast, the counting sensor model uses a statistical model to infer the grid statistics. By adding the Bayesian kernel inference, S-BKI outperforms S-CSM as it can fill the gaps in the map using nearby measurements. Even for fully observed regions, by considering local correlations the map becomes more robust to noisy measurement.

4.5.2 SemanticKITTI Dataset

SemanticKITTI [6] is a large-scale dataset based on the KITTI odometry dataset. It provides dense annotations for each LiDAR scan of 22 sequences including camera poses estimated from a surfel-based SLAM approach (SuMa) [124]. The input data of this dataset is collected by a Velodyne HDL-64E laser scanner. The semantic measurements are gen-

Table 4.4: Mean IoU on SemanticKITTI dataset sequence 00-21 [6] for 19 semantic classes. SqueezeSegV2-kNN (Sq.-kNN). Darknet53-kNN (Da.-kNN). Training (00-07, 09-10), Validation (8), Test (11-21).

Seq.	Method	Car	Bicycle	Motorcycle	Truck	Other Vehicle	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other Ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic Sign	Average
Training	Sq.-kNN	88.2	14.4	45.7	67.3	60.9	33.3	58.7	63.1	92.6	62.0	81.3	49.2	77.3	63.6	76.7	34.5	71.5	32.8	49.5	59.1
	S-CSM (w/ Sq.-kNN)	92.6	21.6	62.2	73.1	70.6	44.1	80.3	67.4	94.3	70.9	85.1	52.3	82.0	69.1	81.4	47.8	75.4	50.8	65.0	67.7
	S-BKI (w/ Sq.-kNN)	93.5	29.1	73.9	82.0	77.0	54.6	87.2	73.7	93.8	73.6	84.2	55.7	83.8	70.1	82.8	53.9	75.9	54.6	70.4	72.1
	Da.-kNN	94.7	42.3	81.8	83.4	69.4	69.4	72.5	53.7	96.7	88.6	92.6	82.1	95.4	85.0	92.0	71.0	88.3	70.6	82.4	79.6
	S-CSM (w/ Da.-kNN)	96.0	48.6	88.3	84.5	71.4	77.3	83.6	54.3	**96.8**	89.7	**93.3**	84.2	96.6	86.7	93.5	79.2	90.0	80.0	88.9	83.3
	S-BKI (w/ Da.-kNN)	**96.9**	**53.2**	**90.9**	**85.9**	**73.3**	**83.5**	**88.4**	**59.8**	**96.8**	**89.9**	93.1	**85.4**	**97.3**	**87.4**	**94.2**	**81.3**	**90.9**	**82.0**	**90.6**	**85.3**
Validation	Sq.-kNN	86.7	14.4	24.6	21.0	23.3	23.5	40.9	0.0	90.1	32.4	74.8	**1.2**	79.6	42.7	79.2	36.5	71.1	28.3	24.8	41.8
	S-CSM (w/ Sq.-kNN)	90.5	23.0	34.9	26.8	29.1	32.4	49.4	0.0	92.6	38.7	79.0	1.1	84.6	51.6	83.3	48.3	72.9	44.1	31.6	48.1
	S-BKI (w/ Sq.-kNN)	92.3	30.0	39.7	29.3	**32.1**	38.8	54.7	0.0	92.9	40.9	79.9	1.1	86.6	54.6	84.9	52.3	74.2	47.9	34.7	50.9
	Da.-kNN	91.0	25.0	47.1	40.7	25.5	45.2	62.9	0.0	93.8	46.5	81.9	0.2	85.8	54.2	84.2	52.9	72.7	53.2	40.0	52.8
	S-CSM (w/ Da.-kNN)	92.6	32.5	54.9	43.4	26.2	51.3	69.2	0.0	**94.6**	49.2	**84.0**	0.1	87.9	58.4	85.8	59.9	73.3	61.7	43.0	56.2
	S-BKI (w/ Da.-kNN)	**93.5**	**33.5**	**57.3**	**44.5**	27.2	**52.9**	**72.1**	0.0	94.4	**49.6**	**84.0**	0.0	**88.7**	**59.6**	**86.9**	**62.5**	**75.3**	**63.6**	**45.1**	**57.4**
Test	Da.-kNN	82.4	26.0	34.6	21.6	18.3	6.7	2.7	**0.5**	91.8	65.0	75.1	27.7	87.4	58.6	80.5	55.1	64.8	47.9	55.9	47.5
	S-BKI (w/ Da.-kNN)	**83.8**	**30.6**	**43.0**	**26.0**	**19.6**	**8.5**	**3.4**	0.0	**92.6**	**65.3**	**77.4**	**30.1**	**89.7**	**63.7**	**83.4**	**64.3**	**67.4**	**58.6**	**67.1**	**51.3**

erated by RangeNet++ [125], which is a state-of-the-art LiDAR-only semantic segmentation deep neural network. To investigate mapping performance on noisy data, we choose two backends provided in RangeNet++: the best-performing one, Darknet53-kNN, and SqueezeSegV2-kNN which has lower performance. All maps are built at a resolution of 0.1 m and without any pre-processing of the input data.

For evaluation, we use all sequences in SemanticKITTI. For training (00-07, 09-10) and validation (08) sequences, we compare the 3D predictions with ground truth labels. To obtain the IoU metrics for test (11-21) sequences, we submitted our results to the official evaluation server which are shown on the multi-scan leaderboard [3]. As our method is for static environments, we cannot differentiate between static and dynamic objects. For static semantic classes, we outperforms Darknet53-kNN for 18 out of 19 classes on test sequences.

Quantitative results on all sequences are given in Table 4.4. For this experiment, the average runtime of S-CSM is 9.48 sec/scan, S-BKI is 1.67 sec/scan. For all sequences, our semantic mapping methods can improve the prior segmentation IoU by fusing multiple scans. We note that S-BKI consistently outperforms S-CSM in almost all semantic classes, which shows the advantage of Bayesian kernel inference and continuous semantic maps. When S-CSM does outperform S-BKI, the IoUs are close to each other. Moreover, the mapping improvement over SqueezeSegV2-kNN is much higher than Darknet53-kNN, which shows our methods can deal with noisy input data.

[3] https://competitions.codalab.org/competitions/20331#results (ganlumm)

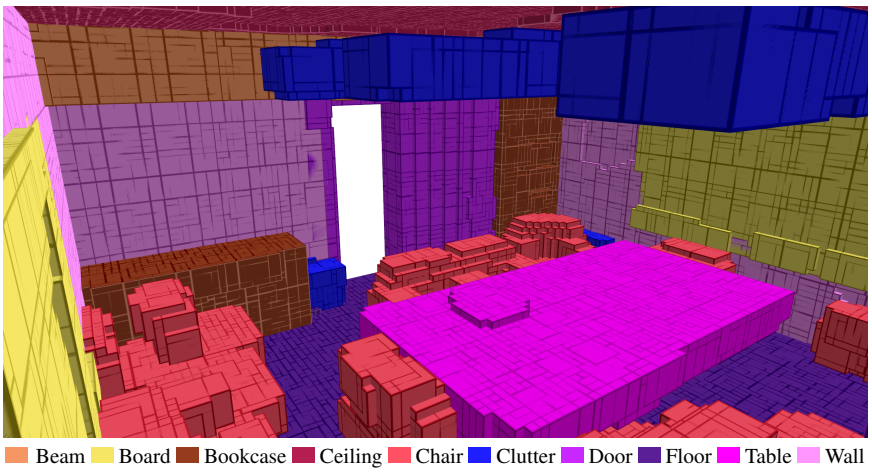


Figure 4.4: S-BKI map of a conference room in Area 3 of Stanford 2D-3D-Semantics Dataset [1].

Table 4.5: Comparison of map quality using the Area Under ROC Curve (AUC) and runtime of the four methods on the example shown in Fig. 4.4.

Method	OctoMap	BKIOctoMap	S-CSM	S-BKI
AUC	0.7226	**0.7801**	0.7274	**0.7801**
Runtime (s)	252.32	73.30	480.44	**68.17**

4.5.3 Occupancy Evaluation

To support the claim that S-BKI is a semantic occupancy mapping method, we evaluate the accuracy of occupancy prediction of S-CSM, S-BKI, OctoMap and BGKOctoMap. The experiment is performed using a conference room in Area 3 of Stanford 2D-3D-Semantics Dataset [1], as ground-truth occupancy values are provided. For S-CSM and S-BKI, we use the annotated point clouds to build the semantic occupancy maps, and the same point clouds without semantics for OctoMap and BGKOctoMap. The semantic map built by S-BKI is shown in Fig. 4.4. Comparisons of map quality and runtime of the four methods are given in Table 4.5. For S-CSM and S-BKI, the probability of occupancy is computed as the sum of all probabilities of valid semantic classes. Among all methods, S-BKI and BGKOctoMap have the highest (identical) performance, which shows that S-BKI reduces to BGKOctoMap when only occupancy is of interest, not only theoretically, but also experimentally. S-CSM is slower than S-BKI because the block depth is set to one, thus S-CSM has more blocks to be computed.

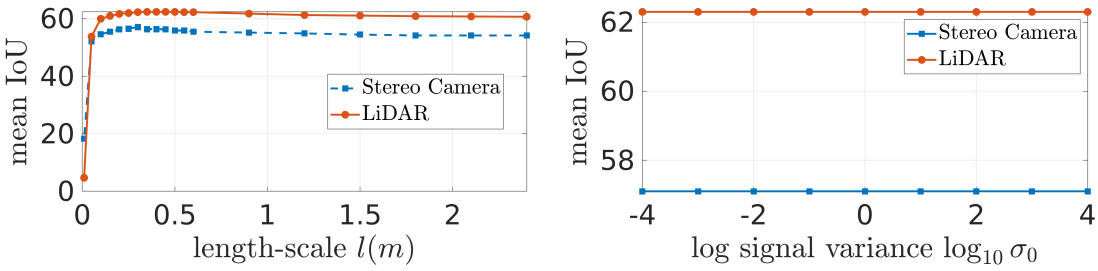


Figure 4.5: Impact of parameters on mapping performance for KITTI dataset sequence 15 [5] (stereo camera) and SemanticKITTI dataset sequence 04 [6] (LiDAR). Only one parameter at a time is varied while the others are kept at the values in Table 4.1. Both figures show reasonable robustness to the parameter variations.

4.5.4 Impact of Parameters

We empirically study the sensitivity of S-BKI mapping to the kernel length-scale and signal variance. The experiments are conducted using KITTI dataset sequence 15 [5] for stereo camera and SemanticKITTI datase sequence 04 [6] for LiDAR data. All other parameters are fixed to the values indicated in Table 4.1. In Fig. 4.5, we plot the kernel length-scale l and signal variance σ_0 against the mean IoU metrics. The influence of the kernel length-scale on mapping performance for both stereo camera and LiDAR data is similar: the mean IoU increases rapidly as the length-scale varies from 0.01 to 0.1, gradually increases to a peak value, and then drops gradually as the length-scale increases. S-BKI achieves the best performance when $l = 0.3$ for stereo camera data and $l = 0.4$ for LiDAR data. This is reasonable because LiDAR data is sparser than stereo camera data and longer distance should be considered. The mapping performance is insensitive to signal variance over a large scale; this is because we use the same signal variance for all semantic classes. To see an effect, one would need to allow signal variance to vary from class-to-class.

4.5.5 Experimental Results on a Cassie Bipedal Robot

We test our mapping methods on data collected using the bipedal robot Cassie Blue shown in Fig 4.6. To obtain semantic measurements, we manually annotated 1194 training images and 457 validation images from the NCLT dataset [126]. The NCLT dataset was selected because it shares a similar environmental domain as the Wave Field data, which includes *background, water, road, sidewalk, terrain, building, vegetation, car, person, bike, pole, stair, traffic sign and sky* for a total of 14 classes. We used these images to fine-tune a modified 2D segmentation network MobileNet [127] with a pre-trained model on the ImageNet dataset [128] for efficiency. The fine-tuned network segments the RGB images,

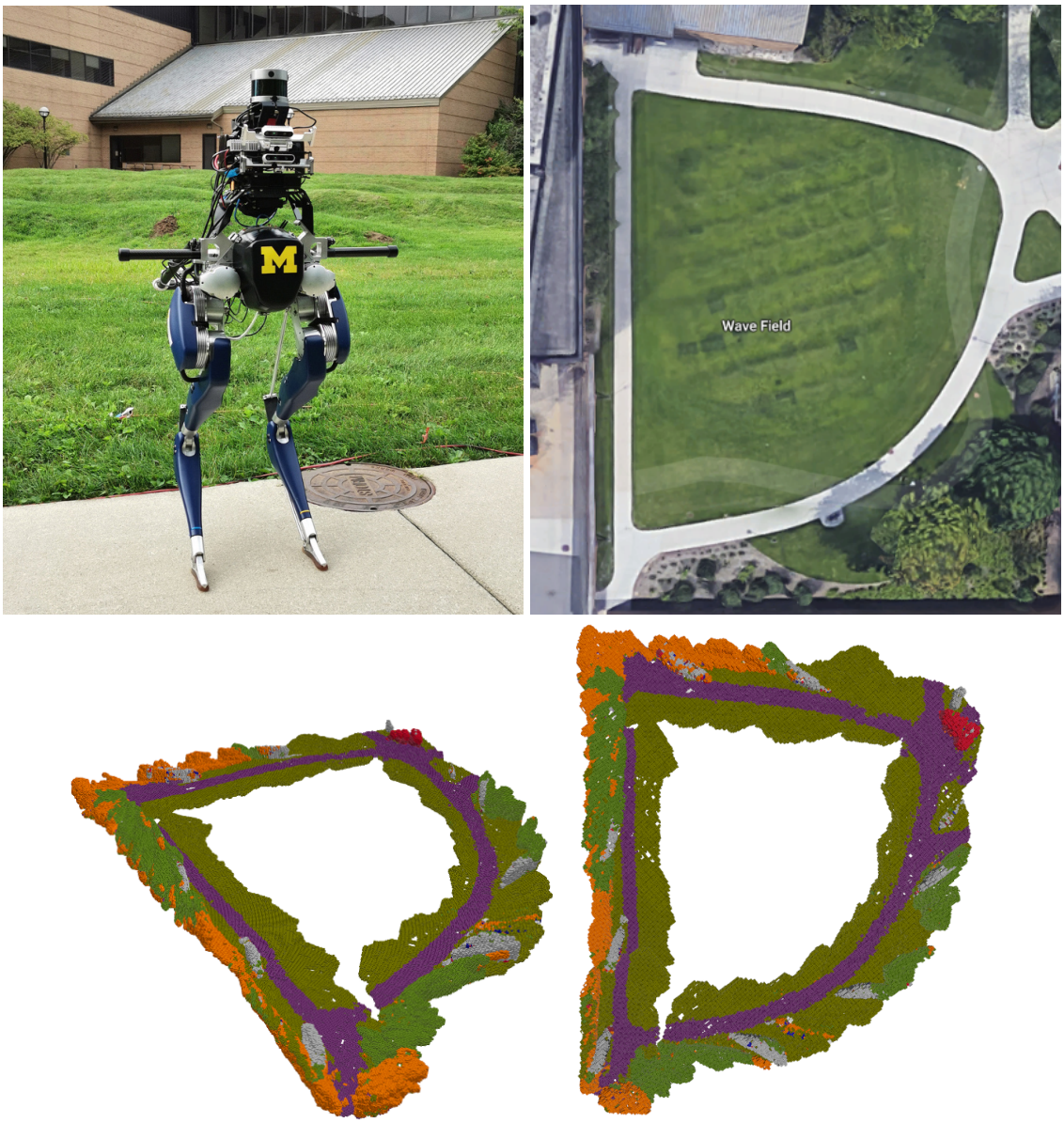


Figure 4.6: Top Left: Cassie Blue has a custom designed torso on which is mounted an Intel RealSense depth camera capable of providing both RGB images and corresponding organized point clouds in outdoor environments. Top right: Google satellite map of the Wave Field of the University of Michigan - North Campus. Bottom: From left to right are the 3D and 2D views of S-BKI map. While the robot is navigating along the sidewalk, S-CSM produces discontinuous semantic maps from sparse sensor measurements, which may cause the robot's planner to regard the gaps in the map as unwalkable areas, a practical problem when we conduct autonomous walking experiments with Cassie Blue (a video of the experiment is available at https://youtu.be/uFyT8zCg1Kk). S-BKI model produces a continuous and smooth map, where gaps are assigned with labels inferred from local correlations in the map.

and then we can directly label the organized point clouds.

The qualitative results are given in Fig. 4.6. To show the mapping performance of our methods on sparse data, we downsample the point clouds per scan to a resolution of $0.2\ \mathrm{m}$, and build a semantic occupancy map with a resolution of $0.1\ \mathrm{m}$. Our S-BKI runs at about $2\ \mathrm{Hz}$. The mapping drift after one full round of the Wave Field is because of the odometry system [129] instead of SLAM used in the experiment.

4.6 Discussions and Limitations

In practice, semantic measurements do not necessarily come in the form of a one-hot vector, but rather a pseudo-probability vector obtained from the softmax output of a classifier. Taking the max rather than the softmax results in the current formulation. Taking the softmax, on the other hand, results in other models corresponding to a set of model-averaging techniques (i.e., the linear opinion pooling and Nadaraya-Watson kernel-regression) that are similar, but not identical, to the Bayesian model presented in Sec. 4.3.

There are still several limitations in this work. First, the length-scale of the kernel function trades off predictive ability and classification accuracy. When the length-scale is large, the model can extrapolate large-scale trends in data, and thus be more predictive; however, the classification accuracy may drop for small objects in the environment. In the current approach, we manually tune the length-scale and use the same scale everywhere, independent of the class. Optimizing the hyperparameters in a Bayesian framework can be helpful. In addition, varying the length-scale and signal variance based on geometric features and semantic properties may further improve semantic mapping performance. Secondly, the memory and space storage for large-scale mapping is another limitation. We currently store the entire semantic map in computer memory without any pruning. However, with the current test-data octrees data structure, even when storing the map after pruning, the save in memory consumption is not substantial. How to compress the continuous semantic maps is an interesting future research direction. Finally, the current semantic map is for static environments, differentiating between static and dynamic semantic labels is also an interesting future work.

4.7 Conclusion

In this chapter, we extended the counting sensor model for occupancy grid mapping to a semantic counting sensor model for semantic occupancy mapping. To relax the independent-grid assumption in occupancy grid mapping, we used a Bayesian spatial kernel inference to

generalize the semantic counting sensor model to continuous semantic mapping. Extensive experimental results show the proposed methods work with both dense stereo camera and LiDAR data. We improved the mapping performance over the state-of-the-art semantic mapping system using the KITTI dataset, and increased the segmentation accuracy over a 3D deep neural network with kNN processing using the SemanticKITTI dataset. We labeled the NCLT dataset and collected data using Cassie Blue biped robot to further evaluate the mapping performance in real world experiments. The S-BKI model consistently outperforms S-CSM, which shows the advantage of using Bayesian kernel inference in continuous mapping.

CHAPTER 5

Multi-Task Learning for Scalable and Dense Multi-Layer Bayesian Map Inference

This chapter presents a novel and flexible multi-task multi-layer 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 existing inter-layer correlations. It removes the need for a robot to access and process information from many separate maps when performing a complex task and benefits from the correlation between map layers, advancing the way robots interact with their environments. To this end, we design a multi-task deep neural network with attention mechanisms as our front-end to provide multiple 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 data sets and data collected by a 3D bipedal robot platform on the University of Michigan North Campus 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.

5.1 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 [130]. 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 purposes such as localization [131], as a reference for navigation [132], and autonomous exploration [133]. Traditionally, maps built for robot localization contain geometric information about 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 [12, 14, 105]. 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 [134, 135]. For better visualization [136] and camera tracking [137], 3D maps are often 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 [96,106,107,110,138–141]. Object instances, as another form of semantic knowledge, are also incorporated into robotic maps [16,142,143].

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 legged robot can walk on firm pavement instead of unstable vegetation. However, even with semantics, higher level planning may still be limited. Planning systems may desire other attributes, for example, the traversability of the ground surface, the affordance [144, 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 one of the key requirements of the *robust-perception age* in robotics [10].

At present, each of these attributes is often estimated independently and represented in a separate map, even though the map can have multiple layers. To perform a complex task involving multiple attributes, a robot needs to access and process information across different maps. This not only complicates planning, but ignores the correlation among different sources of information.

In this work, we argue that instead of using an independent map for a specific task, information can be stored in one shared map containing several correlated layers where each layer contains one attribute. We propose a multi-task multi-layer Bayesian mapping framework which

1. provides rich information such as semantics and traversability and is extendable to more layers;

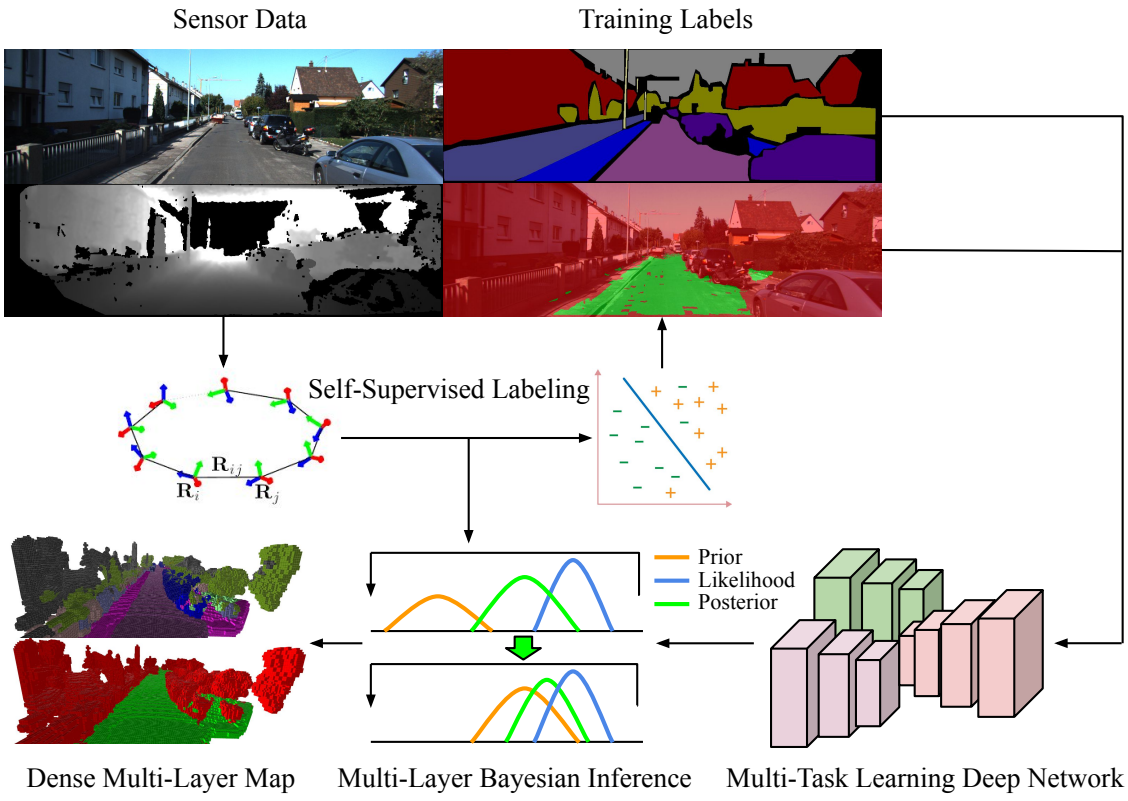


Figure 5.1: Overview of our proposed multi-task multi-layer Bayesian mapping framework. We first employ exteroceptive sensor data and robot trajectory to generate training data for map attributes other than semantics (traversability specifically) in a self-supervised manner. We then train a unified segmentation deep neural network for monocular image via multi-task learning (MTL) using available data. Predictions of the MTL network are processed in our multi-layer Bayesian inference sequentially and efficiently. Inter-layer correlations are also leveraged to improve the map posterior. Our framework is designed to be easily extended to include additional 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 chapter focuses on applying this framework to build a second *traversability* layer on top of the first *semantic* layer developed by [141]. 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" [145]. 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. 5.1. We first employ exteroceptive sensor data and robot trajectory to generate ground truth labels for traversability segmentation in a self-supervised manner. We then design and train a multi-task deep neural network as our front-end to provide multiple observations simultaneously. Our back-end runs a closed-form Bayesian inference for dense multi-layer mapping where semantic information is leveraged in traversability estimation. The Multi-Task Learning (MTL) front-end not only improves prediction performance over Single-Task Learning (STL), but also achieves systematic efficiency. The multi-layer Bayesian inference back-end is scalable and provide map uncertainty.

This chapter has the following contributions.

1. We propose a novel and extendable multi-task multi-layer Bayesian mapping framework to provide rich environmental information in a single mapping formalism.

2. We develop a multi-task deep neural network for scene semantic and traversability segmentation.

3. We present a self-supervised approach to automatically generate training data for traversability segmentation.

4. We formulate a scalable and dense multi-layer mapping algorithm via a 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 chapter is organized as follows. A review of the related work on multi-layer robotic mapping, self-supervised learning for traversability estimation and deep multi-task learning is given in Section 5.2. Section 5.3 describes the design of our multi-task deep network, and how we generate labeled traversability data without requiring manual annotation to train the network. Section 5.4 presents our multi-layer Bayesian mapping formulation in the semantic-traversability case. Experimental results and discussions are presented in Section 5.5 and 5.6, and finally, Section 5.7 concludes the chapter.

5.2 Related Work

In this section, we review the existing literature on multi-layer robotic mapping, self-supervised traversability estimation, and deep multi-task learning.

5.2.1 Multi-Layer Robotic Mapping

Multi-layer robotic mapping, also known as hybrid mapping, stems from the work of [15] 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 on multi-layer mapping [146–148]. 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* [149–151] where semantic knowledge extracted from vision or dialogue is integrated. Therefrom, multi-layer robotic maps include attributes more than geometry (i.e., heterogeneous) and are built using multi-sensor modalities.

More recently, computer vision and machine learning advancements significantly increase the potential of multi-layer robotic mapping. Pronobis and Jensfelt [152] develop a probabilistic framework based on chain graphs to build a four-layer map on three hierarchy levels using multi-modal sensory data: the low-level sensory layer, the middle-level place, and categorical layer, and the high-level conceptual layer. Jiang et al. [153] 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 multi-layer High-Definition (HD) map consisting of a road graph, lane geometry, and semantic features is also conventional 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 multi-layer mapping. In a scene graph, each layer has a set of *nodes* representing entities, each node has some *attributes*, and *edges* between nodes indicate their relations. Armeni et al. [154] construct a four-layer graph that spans an entire building and includes semantics on rooms/objects and camera locations, as well as the relations among them. Rosinol et al. [155] further propose 3D dynamic scene graphs to handle dynamic entities and capture actionable information in the scene. Although scene graphs provide photo-realistic visualization and plenty of environmental information, we argue that they are not ready for general robotic navigation/exploration yet. First, the actionability in scene graphs is still at the topological level of connectivity between places. For trajectory planning and control, information such

as occupancy and traversability of the whole 3D space is needed. Secondly, scene graphs are currently often constructed offline and only for indoor environments.

DenseSLAM by Nieto et al. [156] is similar to our work as it also builds a dense grid-based multi-layer map of the environment using a probabilistic framework. The fundamental difference is that it contains only geometric information acquired using range-finder sensors. The work of Nordin and Degerman [157] is also related to ours in which a multi-layer 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 nor explore inter-layer correlations. More recently, Zaenker et al. [158] 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 which 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.

5.2.2 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 survey done by Papadakis [145], the traditional approaches for estimating traversability are categorized into *geometry-based* analysis of digital elevation maps [159] and *appearance-based* classification of the terrain into a set of pre-defined classes using supervised learning [160]. 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, automates not only the labeling process but also gains flexibility in changing environments and platform variation (as the supervisory signals also vary 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.

Based on the type of supervisory signals employed in self-supervised traversability learning methods, we may project these methodologies onto a space characterized in Fig 5.2. We note that for traversability estimation problems, where we usually need to predict the traversability of the ground areas to be traversed, no matter what type of supervisory signals

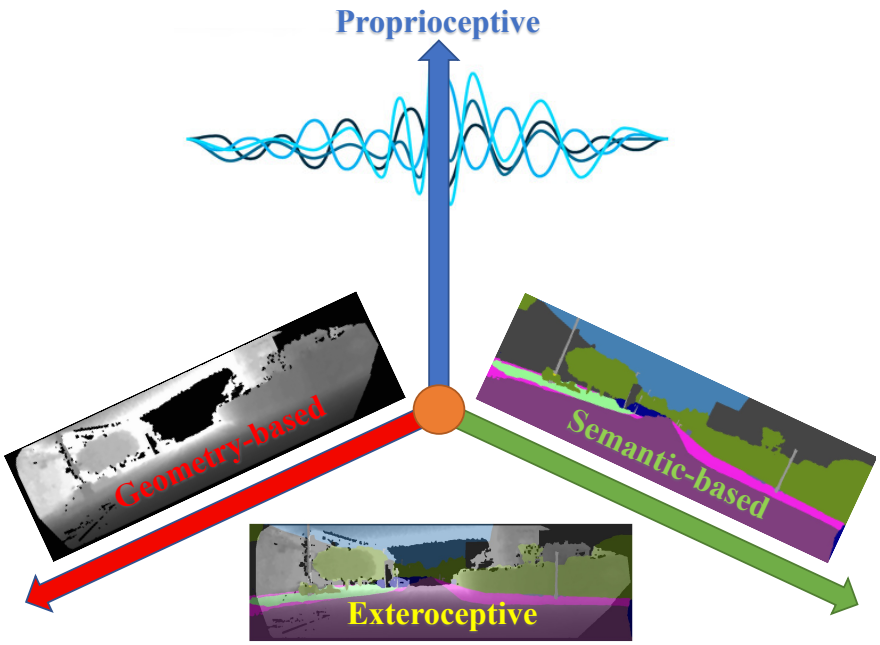


Figure 5.2: Space decomposition of self-supervised traversability learning methodologies according to the type of supervisory signals employed, namely, proprioceptive, geometry-based and semantic-based approaches, the latter two comprising the domain of exteroceptive approaches.

is employed, deployed sensory data tends to be exteroceptive.

Proprioceptive sensors that measure the internal state of the robot, e.g., acceleration [161, 162], force [163, 164], torque [164], vibration [163], are commonly used to generate supervisory signals for traversability as they directly reflect the physical robot-terrain interactions. Stavens and Thrun [161] generate the label of terrain roughness automatically from a vehicle's inertial measurements while driving. Acceleration signals are also used as labels for traversability cost regression based on image textures [162]. Brooks and Iagnemm [163] train two proprioceptive classifiers based on a traction force model and wheel vibration. Wellhausen et al. [164] 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 images to train a convolutional network for terrain property predictions. Recently, robot-terrain interaction sounds captured by microphones are also used as a proprioceptive modality to self-supervise a visual terrain classification network [165].

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. Heuristic methods may be necessary for generating negative samples. Thus, exteroceptive data (e.g., cam-

era, LiDAR) is more commonly 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 drivable surfaces and label the corresponding images [166]. Stereo images are used to self-supervise a vision-based traversability classifier in a near-to-far setting on a DARPA program LAGR platform [167]. To train a deep segmentation network for path proposals in urban environments, Barnes et al. [168] project vehicle future paths onto images and superimpose obstacle labels as detected by a LiDAR scanner onto them for labeling.

Similarly, Broome et al. [169] use geometric features extracted from accumulated LiDAR point clouds as inputs 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 multi-task learning and multi-layer mapping to improve the traversability estimation accuracy.

5.2.3 Deep Multi-Task Learning

Multi-task learning shares the same philosophy as our multi-layer map inference, aiming to improve the performance of multiple related learning tasks by leveraging useful information among them [170]. Theoretically, MTL models lead to a preference for hypotheses that explain multiple tasks, thus leading to reduced overfitting and improved generalization capabilities. While MTL emerged long before the advent of deep learning [171], 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 networks capable of learning shared representations from multi-task signals [172]. Particularly, three main problems have been researched in this field: network architectures, optimization strategies, and task relationships [172, 173]. For architectures, the key questions to answer are *what to share* and *how to share* parameters between tasks. The existing MTL methods are often categorized into *hard* and *soft parameter sharing* [170]. Hard parameter sharing networks share exactly the same model weights in shared layers [174], while in soft parameter sharing methods, tasks have separate weights but interact with each other through regularization [175], cross-talk [176, 177] or attention mechanism in more recent works [178, 179]. A new taxonomy of MTL architectures is also proposed in [172], which groups them into *encoder-focused* and *decoder-focused* 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 [180, 181].

However, with an increase in the number and type 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 between 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 [182] and Gradient Normalization (GradNorm) [183] 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 [184]. 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 [185], 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 [170].

Attention mechanism has often been used in deep learning to visualize/interpret the inner states of convolutional neural networks, and successfully applied to natural language processing tasks. More recently, the usage of attention in multi-task learning achieves promising performances [178, 179]. 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 the negative transfer issue in hard parameter sharing methods where the shared features are constrained to be the same. As attention modules are small compared to the backbone network, these methods also do not suffer from the scalability issue in common soft parameter sharing methods where each task owns a completely separate set of parameters. Furthermore, attention modules can be incorporated into any feed-forward backbone architectures and can be learned in an end-to-end manner.

5.3 A Unified Segmentation Model for Multi-Layer Mapping via Multi-Task Learning

Traditional 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 image or point cloud data for a higher-level understanding of the scene has become essential to

the pipeline. This reasoning block is usually chosen as a Deep Neural Network (DNN) for place recognition, object detection, or semantic segmentation, depending on the specific mapping framework.

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

In this section, a multi-task DNN for image segmentation is presented. The MTL network takes monocular RGB images as inputs and outputs several dense segmentation results to provide measurements for the multi-layer 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, while traversability labels are rare and usually robot-dependent. An automated traversability labeling approach using exteroceptive sensor data is also proposed at the end of this section to tackle this issue. Given 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.

5.3.1 Multi-Task Network with Attention Mechanisms

5.3.1.1 Architecture Design

Inspired by recent works on MTL with attention [178, 179], 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 decoder and an attention-modulated shared encoder. It reduces the risk of overfitting by encouraging most encoder parameters to learn to extract features that 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 task-specific attention mechanisms following the work of Maninis et al. [179]: Squeeze-and-Excitation (SE) blocks and Residual Adapters (RA). Squeeze-and-excitation block is proposed by Hu et al. [186] to adaptively recalibrate channel-wise feature responses by explicitly modeling interdependencies between channels. In other words,

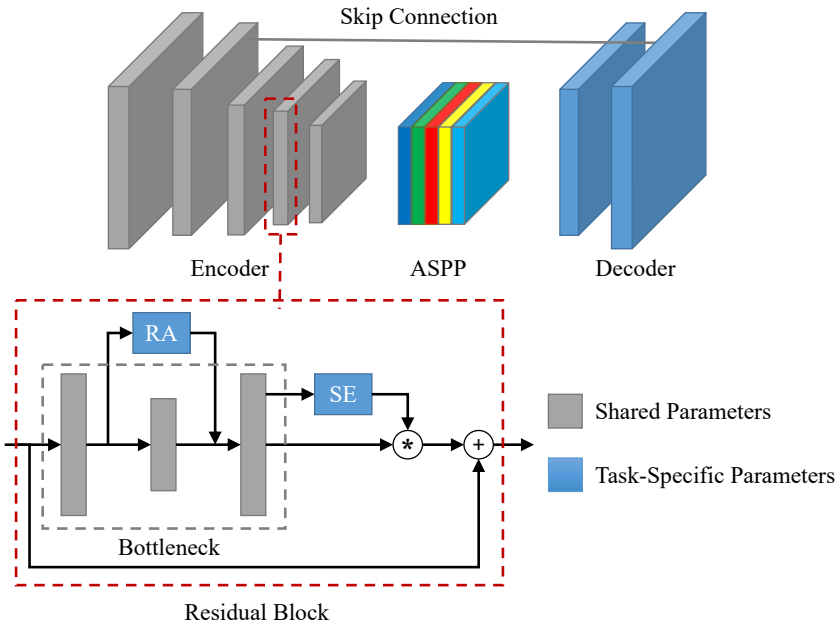


Figure 5.3: Network architecture of each task using an illustrative backbone. Task-specific parameters consist of Residual Adapter (RA) and Squeeze-and-Excitation (SE) modules in residual blocks of the shared encoder and the entire decoder. This architecture can be built upon any feed-forward neural network with residual blocks and trained end-to-end.

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), which maps the descriptor to a set of channel weights. Finally, the original feature maps are reweighted with the channel weights. We set the SE block parameters to be task-dependent so that each task can learn its own channel-wise relationships and utilize the shared encoder differently.

The idea of residual adapters is proposed by Rebuffi et al. [187] 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 degree of parameter sharing among domains is still maintained. As multi-domain learning can be considered as a specific multi-task learning problem, RAs have also been applied to MTL for adapting and refining the shared features of each task [179]. RA module is essentially a 1×1 filter bank in parallel with a skip connection, introducing a small amount of computational overhead to the shared backbone network. However, it can substantially improve accuracy [179] and control overfitting [187].

Figure 5.3 illustrates the general architecture of each task in our network. Each task

has its own RA and SE modules to modulate the shared encoder in a task-specific manner. The RA refines the shared features to be favorable to a specific task, while the SE works as a soft membership function to accept more relevant features and avoid less relevant ones. We note that this MTL architecture can be built on any feed-forward neural network with residual blocks. Thus, all task-shared and task-specific parameters can be learned in an end-to-end manner.

5.3.1.2 Model Objective

In a MTL setting, the optimization objective for K tasks is usually formulated as a weighted sum of all task-specific losses:

$$\mathcal{L}_{\texttt{MTL}}(\mathbf{X}, \mathbf{Y}_{1:K}) = \sum_{i=1}^{K} w_i \mathcal{L}_i(\mathbf{X}, \mathbf{Y}_i), \tag{5.1}$$

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

$$\mathcal{L}_i(\mathbf{X}, \mathbf{Y}_i) = -\sum_{c=1}^{M} \lambda_{i,c} \mathbf{Y}_{i,c} \log \hat{\mathbf{Y}}_{i,c}, \tag{5.2}$$

where $\mathbf{Y}_{i,c}$ is a binary variable indicating whether the ground truth label of the input $\mathbf{X}$ is class c, and $\hat{\mathbf{Y}}_{i,c}$ is the network predicted likelihood of $\mathbf{X}$ belonging to class c, $c = 1, 2, ..., M$. In order to compensate for unbalanced training data, we also weight the loss according to the number of samples in each class [32], i.e., class weight $\lambda_{i,c}$ will be higher if class c has fewer samples. Precisely, $\lambda_{i,c}$ is set to the reciprocal of the frequency of class c in training data and is not part of the network parameters.

5.3.2 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 multi-layer 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 mea-

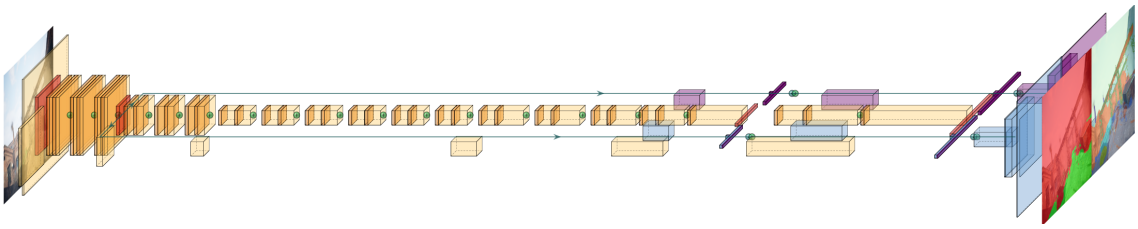


Figure 5.4: Detailed visualization of our MTL network based on DeepLabv3+ architecture with WideResNet38 backbone. Semantic segmentation task (purple) and traversability segmentation task (blue) have their own set of attention modules attached to the shared encoder (orange), and their own decoders. All task-specific modules have the same design except the last convolutional layer (the number of output channels depends on the number of classes in that task), although their weights are individually learned.

surements.

For segmentation tasks, we choose DeepLabv3+ architecture [188] with WideResNet38 backbone [189]. The combination is used in NVIDIA semantic segmentation work [190] and yields state-of-the-art performance on KITTI dataset [191]. Figure 5.4 gives a detailed visualization of our network built upon DeepLabv3+, where the shared parameters and task-specific parameters are shown in different colors. As we can see, the semantic/traversability segmentation task has its own decoder and its own set of attention modules attached to the shared encoder. 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 (traversable vs. untraversable); thus, the number of classes, in this case, is two.

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

5.3.3 Self-Supervised Traversability Labeling

This subsection presents a self-supervised approach to generate 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 traversabil-

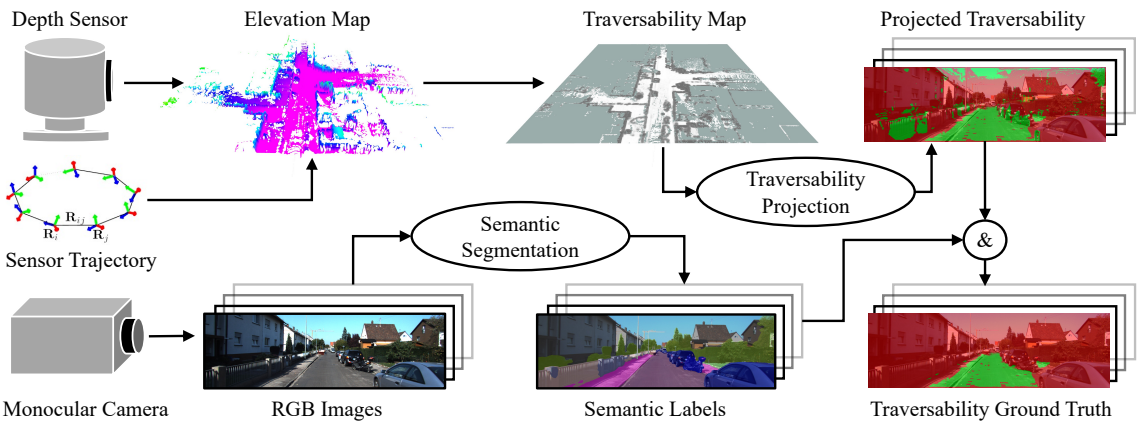


Figure 5.5: Our self-supervised traversability labeling pipeline in which exteroceptive sensor data is used as supervisory signals. The upper part concerns geometry; An elevation map is built using point cloud data captured by a depth sensor (LiDAR, depth camera or stereo camera) and the sensor trajectories. A 2D traversability map is then estimated from the elevation map and projected into images. The lower part considers semantics; We use the predicted semantic labels to filter out the noises in projected traversability images. Combining both geometry-based and semantic-based self-supervised approaches, the final traversability ground truth images can be generated automatically and accurately.

ity labeling process is illustrated in Fig. 5.5. 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.

5.3.3.1 3D Geometry-Based Traversability Computation

We employ a geometry-based traversability estimation method with low complexity proposed by Wermelinger et al. [159]. The traversability is estimated based on terrain characteristics and the capabilities of the robot. A robot-centric *elevation map* of the terrain is built from an exteroceptive sensor (such as a laser scanner, a depth camera, or a stereo camera) data and poses. The elevation map is a 2.5D grid map in which each cell stores a height value and its variance [192].

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

$$t_{\texttt{score}} = 1 - w_1 \frac{s}{s_{\texttt{crit}}} - w_2 \frac{r}{r_{\texttt{crit}}} - w_3 \frac{h}{h_{\texttt{crit}}}, \tag{5.3}$$

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

To generate dense labels for the image at timestamp k, we first build a local elevation map using successive frames of point cloud data from timestamp k to $k+l$ using 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 k. We then compute the traversability score at each cell of the map using (5.3). As a result, a 2D local traversability map with the same resolution and size as the input elevation map is obtained for timestamp k.

5.3.3.2 Map-to-Image Projection

To project the traversability map onto the image domain, 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. 5.5.

5.3.3.3 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. 5.5, the projected traversability image mislabels some sky pixels as traversable. To correct these false-positive labels, we use semantic segmentation of the 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. 5.5 do not contain false positives.

5.4 Multi-Layer Bayesian Map Inference

Multi-layer mapping aims to use the semantic occupancy map as *prior* or complementary information in modeling other useful maps of the environment, such as traversability, friction, and concentration of a physical quantity, by considering their correlations with occupancy and semantic labels. We extend an incremental Bayesian inference method proposed in our previous work [141] due to the closed-form formulation and efficiency. This section first briefly introduces the Bayesian inference for constructing a semantic occupancy layer

and then describes how to leverage it in traversability layer inference. Extension of the method to more map layers is discussed in Section 5.6.1.

5.4.1 Continuous Semantic Mapping

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

$$p(y_i|\theta_j) = \prod_{k=1}^{K} \left(\theta_j^k\right)^{y_i^k}, \tag{5.4}$$

where $\theta_j = (\theta_j^1, ..., \theta_j^K)$ separately denotes the probability of map cell j taking the kth category from a set of semantic class labels $\mathcal{K} = \{1, 2, ..., K\}$, i.e., K categories, and $\sum_{k=1}^{K} \theta_j^k = 1$. The semantic measurement tuple $y_i = (y_i^1, ..., y_i^K)$ is a one-hot-encoded vector computed by taking the max of the semantic segmentation results of a deep network. Let $\mathcal{X} \subset \mathbb{R}^3$ be the map spatial support, i.e., 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 ith 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 closed-form solution, we place a Dirichlet distribution over θ_j as the conjugate *prior* of the Categorical likelihood, given by $\text{Dir}(K, \alpha_0)$, where $\alpha_0 = (\alpha_0^1, ..., \alpha_0^K)$, $\alpha_0^k \in \mathbb{R}^+$ are concentration parameters. By applying Bayes' rule and Bayesian kernel inference [112], the posterior $\text{Dir}(K, \alpha_*)$, $\alpha_* = (\alpha_*^1, ..., \alpha_*^K)$ can be computed as follows:

$$\alpha_*^k := \alpha_0^k + \sum_{i=1}^{N} k(x_*, x_i) y_i^k, \tag{5.5}$$

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 ith training point x_i. To exactly and efficiently evaluate the kernel over relevant measurements (i.e., measurements in the neighborhood of a query point), a sparse kernel [117] is employed with 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 [141, and references therein].

5.4.2 Semantic-Traversability Mapping

Bernoulli distribution is a common choice for modeling binary variables such as occupancy [105], and terrain traversability [115]. For the jth map cell, we treat its traversability

as a Bernoulli distributed (binary) random variable which takes the value 1 with probability ϕ_j. The measurement likelihood is then described as:

$$p(z_i|\phi_j) = \phi_j^{z_i}(1-\phi_j)^{1-z_i}, \tag{5.6}$$

where the measurements $\mathcal{Z} = \{z_1, ..., z_N | z_i \in \{0, 1\}\}$ indicating 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 a neural network. Given training set $\mathcal{D}_z := \{(x_i, z_i)\}_{i=1}^N$, Bayesian traversability mapping problem is thus formulated as the computation of posterior distribution $p(\phi_j|\mathcal{D}_z)$ for each map cell.

Semantic information of the environment can give insight into its traversability. For example, intuitively, the sky and buildings are untraversable to ground robots, while the road is traversable to an autonomous vehicle. Therefore, semantics can be used as another source of observations available for traversability mapping. Moreover, it is expected to achieve better traversability inference performance 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 the multivariate generalization of 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 with parameter ϕ'_j, where $\phi'_j = (\alpha'_j - 1)/(\alpha'_j + \beta'_j - 2)$ is the maximum a posterior (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 find the map cell j containing the corresponding position x_i, and generate a binary measurement y'_i according to $\text{Bernoulli}(\phi'_j)$ deduced from the semantic posterior of that cell. To incorporate these measurements into traversability inference, we assume that measurements $\mathcal{Y}'$ also have the same likelihood as in (5.6):

$$p(y'_j|\phi_j) = \phi_j^{y'_j}(1-\phi_j)^{1-y'_j}. \tag{5.7}$$

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:

$$p(\phi_j|\mathcal{D}_{y'}, \mathcal{D}_z) \propto p(\phi_j|\mathcal{D}_{y'})p(\phi_j|\mathcal{D}_z). \tag{5.8}$$

We note that $\mathcal{D}_{y'}$ 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 α_0 and β_0 are hyperparameters. We have $\phi_j|\mathcal{D}_{y'} \sim \text{Beta}(\alpha_{y'}, \beta_{y'})$, where:

$$\alpha_{y'} := \alpha_0 + \sum_{i,\, x_i \text{ in cell } j} y'_i \tag{5.9}$$

$$\beta_{y'} := \beta_0 + \sum_{i,\, x_i \text{ in cell } j} (1 - y'_i). \tag{5.10}$$

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

$$\alpha_j := \alpha_{y'} + \sum_{i,\, x_i \text{ in cell } j} z_i \tag{5.11}$$

$$\beta_j := \beta_{y'} + \sum_{i,\, x_i \text{ in cell } j} (1 - z_i). \tag{5.12}$$

Applying Bayesian kernel inference [112], the update equations for hyperparameters become:

$$\alpha_* := \alpha_0 + \sum_{i=1}^{N} k(x_*, x_i)(y'_i + z_i) \tag{5.13}$$

$$\beta_* := \beta_0 + \sum_{i=1}^{N} k(x_*, x_i)(2 - y'_i - z_i). \tag{5.14}$$

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

$$\hat{\phi}_j = \frac{\alpha_j - 1}{\alpha_j + \beta_j - 2} \text{ and } \alpha_j, \beta_j > 1. \tag{5.15}$$

We also have the closed-form expected value and variance of ϕ_j as follows:

$$\mathbb{E}[\phi_j] = \frac{\alpha_j}{\alpha_j + \beta_j} \text{ and } \mathbb{V}[\phi_j] = \frac{\alpha_j \beta_j}{(\alpha_j + \beta_j)^2(\alpha_j + \beta_j + 1)}. \tag{5.16}$$

An algorithmic implementation of the semantic traversability Bayesian inference procedure is provided in Algorithm 1. Lines 3-8 are procedures for semantic mapping and lines 9-30 for semantic-traversability mapping. We use a simple strategy (Lines 20-25) to sample semantic-traversability measurements from the deduced Bernoulli distributions. In this way, we can leverage more accurate, up-to-date semantic posteriors in traversability

Algorithm 1 Semantic Traversability Bayesian Inference

Input: Training data: $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$; Query point: x_*
Initialize: semantic hyperparameters $\alpha_*^k \leftarrow 0$
traversability hyperparameters $\alpha_*, \beta_* 0$
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 kth category is traversable **then**
        $\alpha'_j \leftarrow \alpha'_j + \alpha_j^k$
      else if kth category is untraversable **then**
        $\beta'_j \leftarrow \beta'_j + \alpha_j^k$
      end if
    end for ▷ Convert Dirichlet distribution to Beta
    $\phi'_j \leftarrow (\alpha'_j - 1)/(\alpha'_j + \beta'_j - 2)$ ▷ 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
  $k_i \leftarrow k(x_*, x_i)$
  $\alpha_* \leftarrow \alpha_* + k_i(y'_i + z_i)$ ▷ Traversable posterior
  $\beta_* \leftarrow \beta_* + k_i(2 - y'_i + z_i)$ ▷ Untraversable posterior
end for ▷ Semantic-Traversability Mapping
return $\alpha_*^k, \alpha_*, \beta_*$

inference instead of noisy single-frame semantic measurements.

The computational complexity for building the semantic layer is $\mathcal{O}(M \log N)$, where M and N are the number of test points and training points, respectively. For semantic-traversability mapping, the worst-case time complexity is $\mathcal{O}(M \log NK)$, where K is the number of semantic classes. However, by remembering the deduced Bernoulli distribution for cells that are already computed, it can still reach $\mathcal{O}(M \log N)$.

Remark 4 *The traversability layer inference is similar to the continuous occupancy mapping approach of [105]. While the mathematical derivations are the same, the traversability layer in this work performs two updates per iteration; one from deduced semantic distributions and one directly from a neural network.*

5.5 Experimental Evaluation

In this section, we evaluate our proposed method using two publicly available datasets for robotics: the KITTI dataset [191] and the TartanAir dataset [193]. For both datasets, qualitative and quantitative results are provided with discussion. We further validate the proposed method using a data set collected by our own bipedal robot platform, Cassie Blue.

5.5.1 Implementation Details

For the MTL network, we use the architecture shown in Fig. 5.4 based on DeepLabv3+ and WideResNet38 backbone. During training, we use an SGD optimizer with the initial learning rate of $1e \times 3$ and a polynomial learning rate decay with the power of 1.0. Momentum and weight decay are set to 0.9 and $1e \times 4$, respectively. In addition, we adopt synchronized batch normalization (SBN) as in [190] with a batch size of 8 distributed over two Nvidia TITAN RTX GPUs and further make it task-specific. For both datasets, we use the pre-trained semantic segmentation model on Cityscapes. The traversability-specific decoder is also initialized with the pre-trained model with the intuition that the two tasks are highly related. All task-specific attention modules are randomly initialized. Our network implementation is based on [190] using PyTorch. We provide our code at https://github.com/ganlumomo/semantic-segmentation.

The multi-layer Bayesian mapping algorithm is implemented in C++ with the Robot Operating System (ROS) and uses the Learning-Aided 3D Mapping Library (la3dm) [105]. 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 and Beta hyperparameters

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 mapping algorithm is available at https://github.com/ganlumomo/MultilayerMapping.

For self-supervised traversability labeling, we use the Robot-Centric Elevation Mapping library [192] and the Traversability Estimation package [159]. Data set loaders for KITTI and TartanAir are provided, along with the code for projecting traversability labels onto images. Elevation maps are built at a resolution of 0.1 m. The $s_{\texttt{crit}}$, $r_{\texttt{crit}}$ and $h_{\texttt{crit}}$ in (5.3) are set to 1.0, 0.05 and 0.12, respectively. 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_projection_ws.

5.5.2 Ablation Study

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

1. **STL**: The vanilla DeepLabv3+ with WideResNet38 backbone for single-task learning. This model is proposed by Zhu et al. [190] for semantic segmentation, and we also train it for traversability segmentation.

2. **MTL w/o Attentions, Fixed**: The standard multi-task learning baseline with shared encoder and task-specific decoders [172], trained with shared parameters fixed.

3. **MTL w/o Attentions**: The same architecture as above [172] with all parameters trained.

4. **MTL w/ Attentions**: The same architecture as above with task-specific attention modules and task-specific SBN (synchronized batch normalization) operators attached to the shared encoder. All parameters are trained.

As for evaluation criterion, in addition to reporting the performance on each individual task using the standard segmentation metric, mean intersection over union (mIoU), we also use a multi-task performance metric $\Delta_{\texttt{MTL}}$ defined by Vandenhende et al. [172]:

$$\Delta_{\texttt{MTL}} = \frac{1}{K}\sum_{i=1}^{K}(-1)^{l_i}(M_{m,i} - M_{s,i})/M_{s,i}, \tag{5.17}$$

Table 5.1: Ablation study of deep multi-task learning network for semantic and traversability segmentation. Mean intersection over unions (mIoUs) of 19 semantic classes and 2 traversability classes are reported on a validation set from KITTI sequence 06-07. Multi-task performance $\Delta_{\texttt{MTL}}$ indicates the average relative improvement over the single-task baseline (see (5.17)).

Setting	Model	Training	Segmentation mIoU (%)		$\Delta_{\texttt{MTL}}$ (%)
			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**

where $M_{m,i}$ and $M_{s,i}$ are the performance of the MTL and STL model on task i, respectively. $l_i = 0$ if a higher value means better performance for metric M_i, and 1 otherwise.

We train the four models using the same hyperparameters on KITTI dataset [191]. For traversability, we used automatically annotated traversability labels generated from KITTI odometry data set sequence 06 and 07 and evaluate them on a validation set generated from sequence 10. Because of the automated labeling process, we can obtain training and validation data amount to 200 images each. For semantic segmentation, we use KITTI semantic dataset. From the 200 semantically labeled training images, we randomly select 21 images as a validation set and exclude them from the training process. Therefore, the ratio of the validation set to the training set is about one to nine. The ablation study results are given in Table 5.1.

From the results in Table 5.1, the standard multi-task learning baselines (setting 2 and 3) fail to outperform their single-task equivalent (setting 1) for traversability segmentation on the KITTI data set while attain or slightly surpass the single-task semantic segmentation performance. Overall, these MTL models degrade the single-task performance (indicated by the negative $\Delta_{\texttt{MTL}}$ values). Similar observations are also reported by Vandenhende et al. [172] 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.

We conjecture that this is because the relationship between these tasks is unclear. The gradient of the traversability-specific loss to the shared encoder can sometimes conflict with that of semantic segmentation during training. Without any task-specific capabilities, the shared encoder tends to be undertrained for an individual task. This negative transfer issue can be alleviated by adding attention modules to modulate the shared encoder in a task-specific manner. Our results in setting 4 also align with this hypothesis, where a

Table 5.2: Quantitative results of our system on KITTI data set 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**

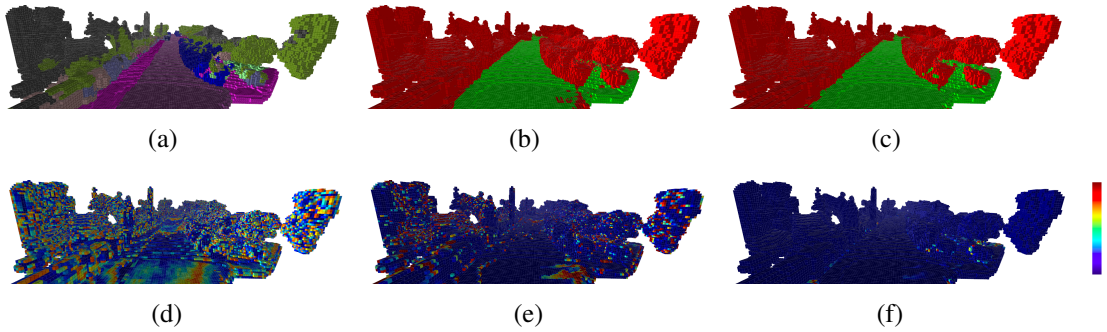
performance gain is obtained for both semantic segmentation ($+3.43\%$) and traversability segmentation ($+4.11\%$). Therefore, we use this model in the following experiments.

5.5.3 KITTI Dataset

KITTI dataset [191] 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 releases its semantic segmentation data set which can be used in our multi-task learning multi-layer mapping framework. However, none of these benchmarks exactly matches our work as they are designed for other purposes. 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 data set which consists of 200 training images. It defines the same 19 semantic classes as in Cityscapes, including *road*, *sidewalk*, *building*, *car*, etc. For traversability segmentation, we generate ground truth labels for KITTI odometry data set sequence 06 and 07, and randomly select 200 training images from them for task balancing. We first use ORB-SLAM2 [93] to estimate the 6-DoF camera poses from stereo measurements, and transform them to get the LiDAR poses using camera-LiDAR calibration. We then use the point clouds from LiDAR scanner and the corresponding LiDAR pose to generate traversability labels, as described in Section 5.3.3. To project the labels into images, we use the depth values estimated by stereo matching [92].

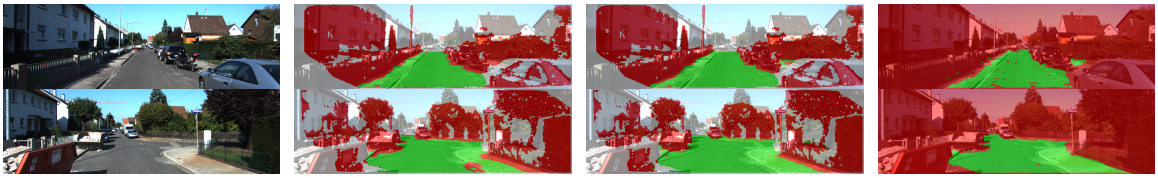
We test our framework on KITTI odometry sequence 15, and provide the quantitative results in Table 5.2. As we do not have ground truth semantic labels for the sequence (KITTI odometry data set does not provide corresponding semantic labels; however, we need sequential information for mapping), we only compare the traversability segmentation performance between the STL baseline and our MTL network. As shown in Table 5.2,



(a) (b) (c)

(d) (e) (f)

Figure 5.6: 3D qualitative results of our multi-layer Bayesian mapping algorithm on KITTI dataset. (a)-(c) are the semantic layer, traversability layer and semantic-traversability layer, respectively. (d)-(f) are the corresponding uncertainty maps showing variance value of each voxel using Jet colormap. We can see the improved estimation in (c) compared with (b), and reduced variance in (f) compared with (e) by leveraging semantic layer in traversability inference. Relatively high variance values are also observed for the voxels with estimation error.



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

Figure 5.7: 2D quantitative results of our multi-layer Bayesian mapping algorithm on KITTI dataset. (a) Input color images. (b) Projected images from traversability mapping. (c) Projected images from semantic-traversability mapping. (d) Traversability ground truth images generated automatically by the method described in Section 5.3.3. We can see the improved estimation in (c) compared with (b) based on the ground truth (d).

our MTL network improves about 10% mIoU for traversability segmentation, and especially improves 16.08% for the traversable class. Two reasons might justify this: First, 200 training images are insufficient to well train an STL network that is pretrained on another task; Second, traversability is highly correlated to semantics in an on-road environment, and the MTL network can learn this correlation quickly.

The performance of our multi-layer Bayesian mapping is also evaluated in Table 5.2, where traversability mapping is the inference without semantic information, and semantic-traversability mapping is the algorithm given in Section 5.4.2 which leverages inter-layer correlations. For a fair comparison, we project the 3D maps into 2D images and compare them with the ground truth images pixel-wisely. To do this, for each pixel in the test image, we use its depth value to query the traversability probability from the corresponding map

cell and convert it to a binary label by thresholding. We exclude all pixels with invalid depth from all result and ground truth images before computing the metrics. The quantitative results of 3D maps and 2D projected images are shown in Fig. 5.6 and Fig. 5.7, respectively.

The results show that through sequential Bayesian inference, traversability mapping alone can improve the classification accuracy (line 3 compared with line 2 in Table 5.2), which is an advantage of using multi-frame measurements. However, leveraging semantic layer in traversability inference achieves the best performance (line 4 in Table 5.2). This is also shown in Fig. 5.6, where the semantic-traversability mapping not only corrects some wrong predictions, but also reduces the map uncertainty.

5.5.4 TartanAir Dataset

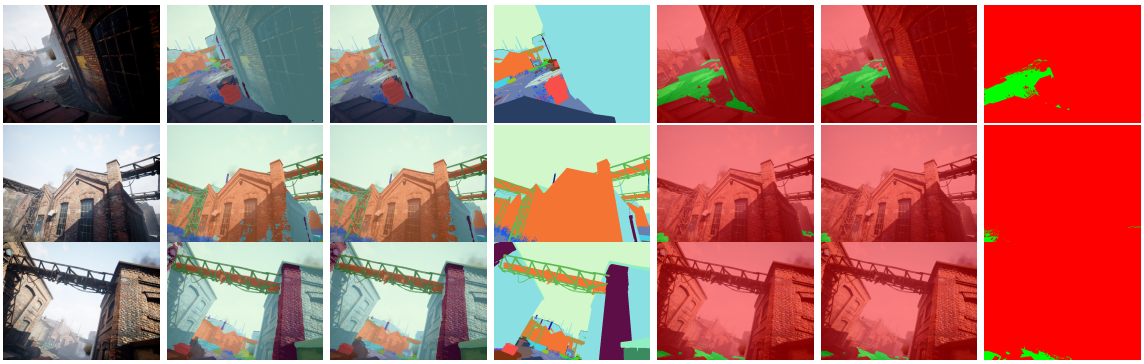
As KITTI data set aims for on-road autonomous vehicle applications, the environments are semblable and relatively easy. To further test our framework in other environments, we employ a new challenging dataset, the TartanAir [193]. TartanAir is mainly introduced as a benchmark for visual SLAM algorithms, 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, which are collected from similar camera viewpoints (pointing to the front from the same height), TartanAir has various camera viewpoints and diverse sensor motion patterns, posing a challenge for image feature extraction.

We choose the *abandoned factory* environment for this experiment. Specifically, we use sequence P000 of 2176 frames for training and P001 of 434 frames for testing. For semantic segmentation, we use the provided pixel-wise annotation including 30 semantic classes for training and evaluation. For traversability segmentation, we convert depth images to point clouds using camera intrinsics, and use the corresponding camera poses for ground truth generation. We use these data to train our MTL model, and compare with the STL network. The quantitative results are listed in Table 5.3.

We obtain a lower semantic segmentation mIoU from the TartanAir data set than that from KITTI. This is reasonable as KITTI has a similar environment and the exact class definition as Cityscapes, on which we pretrained the network. By contrast, TartanAir is more challenging due to the differences in environment, class definition and camera viewpoint. In this case, our MTL model still outperforms the STL network in both tasks (line 3 compared with line 1 in Table 5.3). However, we notice that the mIoU improvement for traversability segmentation is only 1%. We conjecture that the correlation between the

Table 5.3: Quantitative results of our multi-task learning model on TartanAir dataset. Mean intersection over unions (mIoUs) of 30 semantic classes and 2 traversability classes, and multi-task performance ($\Delta_{\texttt{MTL}}$) are reported.

Method	Semantic Seg. (mIoU)	Traversability Seg. (mIoU)	$\Delta_{\texttt{MTL}}$ (%)
STL	47.77	85.78	+ 0.00
MTL, GradNorm	49.62	**87.38**	+ 2.87
MTL	**52.68**	86.82	**+ 5.75**



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

Figure 5.8: Qualitative results of our multi-task learning (MTL) model compared with single-task learning (STL) network on TartanAir dataset. (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 by the method described in 5.3.3. We can see that (c) and (f) have better segmentation performance compared with (b) and (e). The improvement on both tasks obtained for the same input image also shows the correlation between these tasks.

two tasks in TartanAir might not be as high as in KITTI, so we might need a better task balancing strategy for training.

We then implement the GradNorm strategy [183] for training. GradNorm varies the task-specific weights in (5.1) according to the norm of the gradients of task-specific loss to shared layers during back-propagation. In our experiments, this strategy indeed improves the traversability segmentation performance, however, at the cost of a lower semantic segmentation mIoU (line 2 in Table 5.3). The multi-task performance metric $\Delta_{\texttt{MTL}}$ shows that our MTL network trained with fixed and uniform task-specific weights actually has a better overall performance than the GradNorm-trained model. Similar phenomenon is also observed in experiments conducted in [172], where most task-balanced MTL approaches do not perform as well as those using fixed weights in practice. In the end, we use our MTL model trained with fixed weights for multi-layer mapping. The qualitative results of this

Table 5.4: Quantitative results of our multi-layer Bayesian mapping algorithm on TartanAir data set using intersection over unions (IoUs) for traversability (trav.) classification.

Method	Untraversable (%)	Traversable (%)	Mean (%)
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**

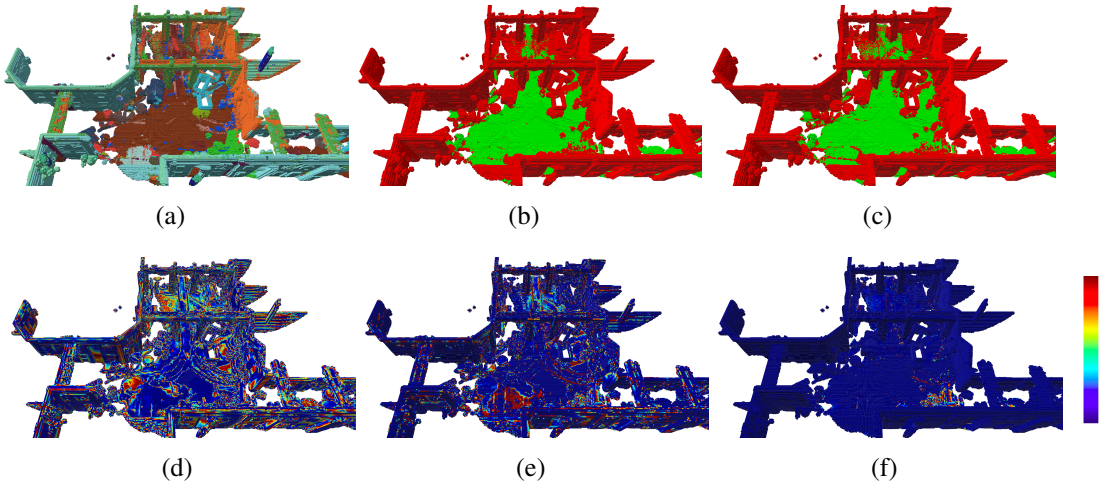


Figure 5.9: Qualitative results of our multi-layer Bayesian mapping algorithm on TartanAir dataset. (a)-(c) are the semantic layer, traversability layer and semantic-traversability layer, respectively. (d)-(f) are the corresponding uncertainty maps showing the variance value of each voxel using Jet colormap.

model are also given in Fig. 5.8, where we can see a performance improvement is obtained for both semantic and traversability segmentation compared with single-task learning.

The quantitative results of our multi-layer Bayesian mapping algorithm are given in Table 5.4. Following the same trend in KITTI experiments, our traversability mapping improves the classification accuracy of single-frame segmentation by training on multi-frame measurements (line 2 compared with line 1 in Table 5.4). Leveraging semantic layer 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 semantic posterior instead of noisy semantic segmentation in the formulation. Fig. 5.9 gives the qualitative results of those map layers with the corresponding uncertainty maps.

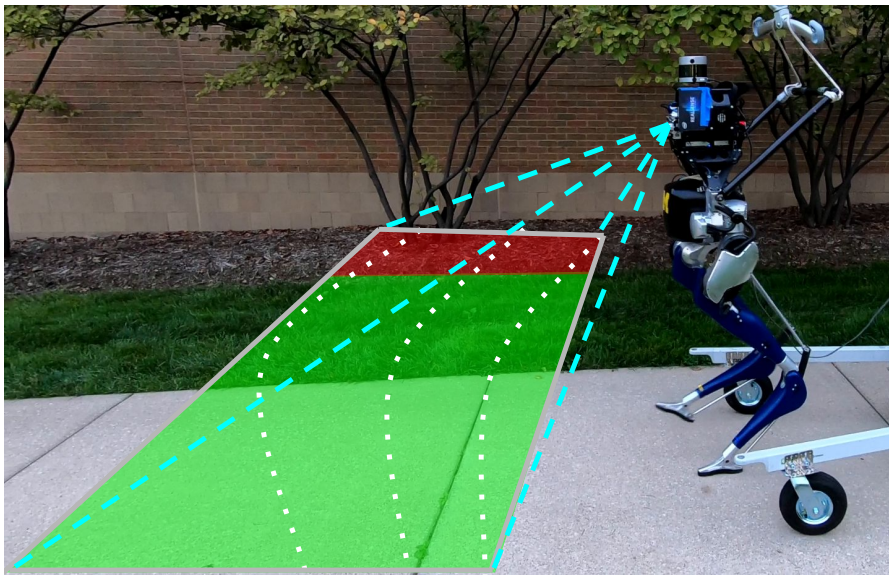


Figure 5.10: The Cassie Blue robot platform equipped with a depth camera and LiDAR sensor. We use the LiDAR point clouds to build a 2D local traversability map based on geometric information. We then project traversability scores into the image plane to generate traversability ground truth labels automatically.

5.5.5 Cassie Robot Data

Finally, we test our multi-layer mapping system on real-world data that was collected using our Cassie Blue robot platform in 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, shown in Fig. 5.10. A contact-aided invariant EKF [194] was running on-board for high-frequent robot pose estimation during data collection. The experiment setup is the same as in [141], except that we build multi-layer map offline using recorded data at the recording speed.

As we already have an optimized semantic segmentation network running on-board during data collection, we skip training the multi-task learning network for this experiment. Instead, we generate the traversability ground truth images from the data and use them to train a traversability segmentation network. The self-supervised labeling process is illustrated in Fig. 5.10. We use LiDAR point clouds and the corresponding robot poses to build a 2D traversability map, and then project the traversability score onto the color image plane using depth values.

Multi-layer mapping algorithm takes the on-board semantic segmentation and offline traversability segmentation for map building. The quantitative results in Fig. 5.11 and 5.12 are visually correct and pleasing. Most ground areas are mapped as traversable (green)

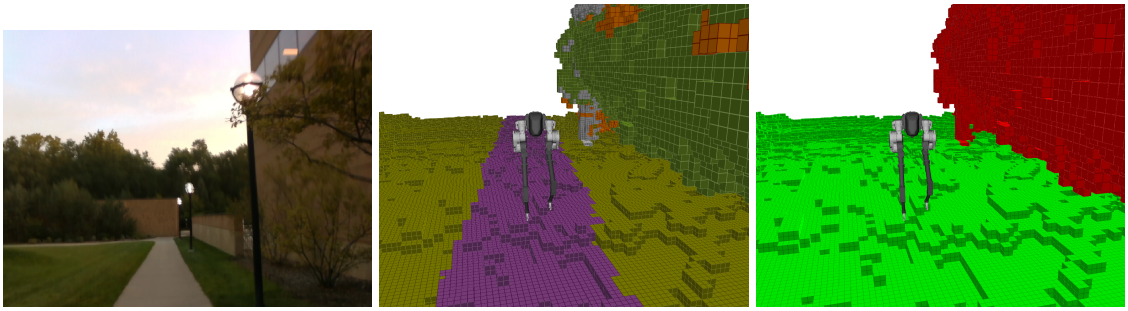


Figure 5.11: Qualitative results of our multi-layer mapping algorithm on data collected by Cassie Blue. The left shows the color image collected by the on-board camera, which is used for semantic and traversability segmentation. The middle shows the semantic layer built up to the timestamp of the left image. The right shows the traversability layer.

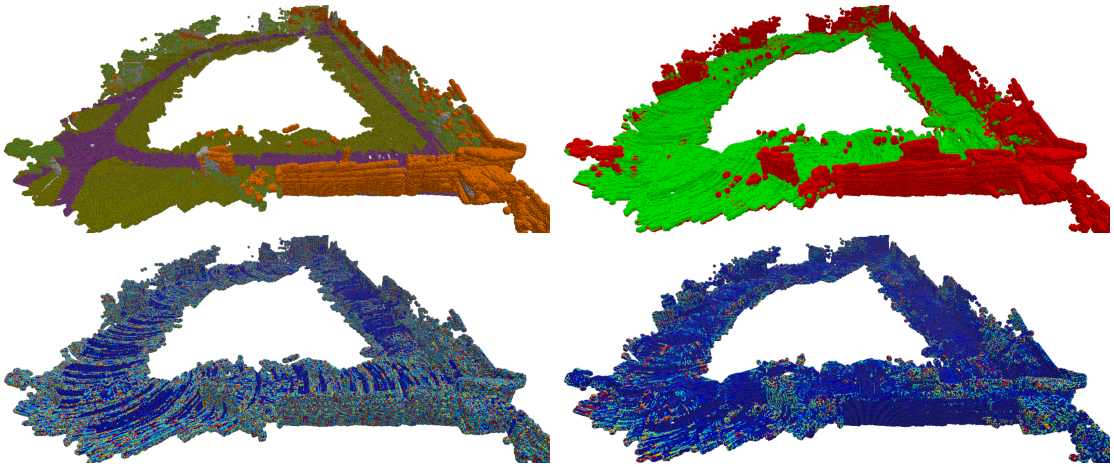


Figure 5.12: Qualitative results of our multi-layer mapping algorithm on data collected by Cassie Blue. The left column shows the semantic layer and the corresponding uncertainty map. The right column shows the traversability layer and its uncertainty map.

with some untraversable voxels (red) caused by moving humans in the environment. It is also noteworthy that the mapping algorithm processes the recorded data at the original collecting speed, which shows that our multi-layer Bayesian mapping is efficient, and ready for online navigation and exploration experiments.

5.6 Discussion on Multi-Task Multi-Layer Mapping Framework

In this section, we discuss the generality of the proposed multi-task multi-layer 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.

5.6.1 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 jth 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 ith 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 $\mu_j \sim \mathcal{N}(\mu_0, \sigma_0^2)$ [195]. 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 jth map cell.

To apply Bayesian kernel inference, we can set $\sigma_0^2 = \sigma^2/\lambda$, where λ is a hyperparameter reflecting confidence in the prior [115]. The kernel version of mean and variance are then derived as:

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

After Bayesian inference for the friction coefficient layer, we can convert the Gaussian posterior distribution $p(\mu_j|\mathcal{D}_f)$ to a Bernoulli distribution $p(\phi_j|\mathcal{D}_f) = \text{Bernoulli}(p)$ by setting thresholds for the Gaussian cumulative distribution function, where $p = Pr(d_{low} \leq$

[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.

$\mu_j \leq d_{high}$). 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 inference into the traversability mapping, we assume that f'_i also has a Bernoulli likelihood:

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

where ϕ_j is defined in (5.6), and $\mathcal{F}'$ are also independent to $\mathcal{Z}$ and $\mathcal{Y}'$. Following the same derivations in Section 5.4.2, we can leverage the correlations to both semantic layer and fiction coefficient layer in traversability mapping. The hyperparameter update equations then become:

$$\alpha_* := \alpha_0 + \sum_{i=1}^{N} k(x_*, x_i)(y'_i + f'_i + z_i),$$
$$\beta_* := \beta_0 + \sum_{i=1}^{N} k(x_*, x_i)(3 - y'_i - f'_i - z_i).$$

The mean and variance of ϕ_j remain the same as (5.16).

5.6.2 Limitations

There are several limitations of 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 timestamp. To deal with dynamic objects and environmental changing, 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, etc. Also, 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 multimodal sensory information, where the cost reflects dynamic traversability based on the current operating point and robot behavior. The framework developed in this chapter provides the foundation for implementing these ideas in the future.

5.6.3 Reproducibility

The current results and maps are reproducible using the same data sets for two reasons. First, the traversability ground truth labels are generated using the given data without any manual labeling. Second, the multi-layer Bayesian map inference provides exact solutions

without approximation. This framework can also be used on self-collected data by other robotic platforms as long as the image and depth information are included.

5.7 Conclusion

This chapter developed a multi-task multi-layer Bayesian mapping framework that uses a deep multi-task learning 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 multi-layer 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. Extensive experimental results on publicly available data sets and data collected by our robots show the advantage of using a multi-task learning network for multi-layer mapping and improving traversability mapping performance when the correlation with the semantic layer is leveraged.

The proposed framework is highly extendable to include additional layers containing greater detail, such as friction coefficient, affordance, dynamic planning cost, to assist more sophisticated robotic behavior planning. The map is also continuous and contains uncertainty information attributed to Bayesian kernel inference, 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.

CHAPTER 6

Energy-based Legged Robots Terrain Traversability Modeling Using Deep Inverse Reinforcement Learning

This chapter reports on developing a deep inverse reinforcement learning method for legged robots terrain traversability modeling that incorporates both exteroceptive and proprioceptive sensory data. Existing works use robot-agnostic exteroceptive environmental features or handcrafted kinematic features; instead, we propose to also learn robot-specific inertial features from proprioceptive sensory data for reward approximation in a single deep neural network. Incorporating the inertial features can improve the model fidelity and provide a reward that depends on the robot's state during deployment. We train the reward network using the Maximum Entropy Deep Inverse Reinforcement Learning (MEDIRL) algorithm and propose simultaneously minimizing a trajectory ranking loss to deal with the suboptimality of legged robot demonstrations. The demonstrated trajectories are ranked by locomotion energy consumption, learning an energy-aware reward map, and a more energy-efficient policy than demonstration. We evaluate our method using a dataset collected by an MIT Mini-Cheetah robot and a Mini-Cheetah simulator. The code and dataset are publicly available at https://github.com/ganlumomo/minicheetah-traversability-irl.

6.1 Introduction

For a robot to autonomously navigate an unknown and unstructured environment, end-to-end learning [196–198], and planning with terrain information [199–201] are two commonly applied approaches. Compared with end-to-end methods that directly map environmental sensing and robot states to actions, planning with terrain information decomposes the complex problem, thus being more data-efficient and interpretable [202, 203]. The

terrain information obtained in these methods can also be more generalizable to other applications or robots. The present work approaches legged robot autonomous navigation from the latter direction.

Motion and task planning for legged robots in unstructured environments involve explicitly modeling the terrain traversability and solving a sequential decision-making problem to find the "optimal" behavior. For unknown environments, terrain traversability modeling using noisy measurements is itself a challenging problem due to the lack of *a priori* knowledge of the surrounding terrain, the variability of terrain conditions, and the *robot-specific* characteristic of traversability.

A taxonomy of terrain traversability modeling methodologies includes exteroceptive approaches that further consist of appearance-based and geometry-based approaches and proprioceptive approaches [145]. Appearance-based approaches formulate terrain traversability modeling as a terrain type classification problem via image classification [160] or semantic segmentation [204]. However, such methods assume that each terrain class corresponds to the same level of traversability. Geometry-based approaches usually generate a geometric representation of the terrain from depth measurements and compute the traversability from local geometric features [115, 205]. Recent works [168, 206, 207] generate self-supervised traversability labels and train a neural network on appearance and geometry data to model robot-specific terrain traversability.

Traversability labels in exteroceptive-based methods ignore the robot's internal states while in motion across the terrain; hence, the degree of traversability is unknown, and the labels are usually binary (i.e., untraversable and traversable). In contrast, proprioceptive approaches can model terrain traversability as a continuous variable by defining a traversability cost using a ground reaction score [164], vibration [162] or stability [208] from a proprioceptive sensor modality. These metrics are handcrafted based on a user's domain knowledge and thus may not fully capture the terrain's traversability as experienced by the robot itself.

Recently, Inverse Reinforcement Learning (IRL) for terrain traversability modeling has received increased interest, particularly for autonomous vehicles [7, 209–211]. Instead of using a pre-determined metric to define traversability cost and associating it with terrain features, IRL-based methods aim to learn this cost directly from a robot's (traversing) behaviors. Doing so bypasses the effort and incompleteness of manually designing the cost function and parts of the learning pipeline. The resulting cost representations of IRL-based methods have also shown better robustness and scalability than the handcrafted cost and supervised learning methods [209, 212].

Maximum Entropy Deep IRL (MEDIRL) exploits the representational capacity of neu-

ral networks for reward structure approximation and has been successfully applied to traversability cost learning and trajectory prediction for autonomous vehicles [7, 209–211, 213, 214]. This work adopts the MEDIRL framework to tackle the problem of terrain traversability modeling for legged robots. A big challenge in IRL-based methods is the choice of representation of the agent's state. A higher dimensional state generally leads to higher model fidelity but at the expense of an exponential increase in computational complexity. Therefore, most works incorporate higher-order robot states into the reward approximation network as extra features instead of expanding the dimension of the agent's state. However, encoding handcrafted robot kinematic features, such as estimated trajectory curvature [7], average velocities [213] and overlaid past trajectory [211], is limited in improving the model fidelity.

A challenge of applying the MEDIRL framework to legged robot applications is the suboptimality of expert demonstration. While maximum entropy methods are robust to some level of suboptimal demonstrations, they fundamentally seek a reward function that justifies the demonstrated behavior, leading to a policy that explains but rarely outperforms the demonstrator [215, 216]. Unlike a human driver, the demonstrator for legged robots typically lacks adequate feedback of locomotion performance to make the demonstration optimal or even near-optimal. Thus, we propose to use robot proprioception as a source of feedback to rank each demonstrated trajectory and add a *trajectory ranking loss* to regulate the reward learning to learn a reward function that could extrapolate beyond suboptimal demonstrations and lead to a policy outperforming the demonstrator.

This chapter offers the following major contributions:

1. We propose a deep reward approximation network in the context of IRL that can learn robot inertial features from raw sensor measurements and incorporate the learned inertial features and environmental features into the traversability reward map.

2. We extend the MEDIRL framework into a Trajectory-ranked MEDIRL framework, and use locomotion energy as the trajectory preference label to learn an energy-aware reward map for legged robots.

3. We collect and generate an IRL dataset for a legged robot in various environments, and evaluate our method using the dataset as well as a simulation environment.

The remainder of this chapter is organized as follows. Section 6.2 introduces related. The proposed methodology is described in Section 6.3. Experimental results are shown in Section 6.4. Finally, we conclude this chapter in Section 6.5.

6.2 Related Work

This work is inspired by — and bridges the gap between — legged robots terrain traversability modeling and IRL-based traversability cost learning methods for autonomous vehicles.

6.2.1 Terrain Traversability Modeling for Legged Robots

Compared with ground vehicles, legged robots can traverse extreme, off-road, and unstructured terrains. However, terrain traversability modeling is crucial for path planning and foot placement and remains a challenging problem because of the complex terrain conditions and various legged robot models.

Wellhausen et al. [164] proposes a self-supervised learning framework to predict terrain properties for a quadruped robot. They compute a *ground reaction score* from recorded force-torque measurements and associate the score with image data by foothold projection. The annotated images are used to train a regression network. Similarly, Faigl et al. [208] also use robot proprioceptive features to characterize three traversal costs, i.e., mean instantaneous power consumption, mean forward velocity, and attitude stability (the variance of the robot's roll), and attach those costs to corresponding exteroceptive features for regression.

More recently, Buchanan et al. [217] propose a neural network for tactile terrain classification on an ANYmal robot. The network takes variable-length signals such as forces and torques and outputs the terrain class. Fan et al. [199] probabilistically model the traversability cost as a combination of multiple risk factors determined by the terrain geometry and robot states. A hierarchical risk-aware planner is also proposed to use the traversability costs and kinematic constraints to navigate wheeled and legged robots. Our method relates to these works as both exteroception and proprioception are exploited to model the terrain traversability for legged robots. But instead of manually defining the traversability cost function, we learn it from demonstrations.

6.2.2 IRL for Traversability Cost Learning

Early works include the applications of the Maximum Margin Planning (MMP) framework to traversal cost modeling using overhead terrain data [218, 219]. Later, the MEDIRL framework is proposed [209] to exploit the power of deep neural networks in expressing nonlinear reward functions and becomes a common paradigm for traversability cost learning.

In [209], the reward function is modeled using a fully convolutional network with inputs of environmental features for each state, i.e., mean height, height variance, and a binary visible indicator. Vehicle kinematics is gradually considered in the MEDIRL framework in addition to environmental context to improve the model fidelity. Zhang et al. [7] integrate the discretized vehicle's past velocity and trajectory curvature into the reward network. Similarly, the reward network in [213] also takes the encoded robot's motion features as inputs. Jung and Shim [211] propose to incorporate inertial context into the traversability map by overlaying the past trajectory on an occupied grid map. Zhu et al. [210] encode vehicle kinematic constraints into convolution kernels to accelerate the computation. We argue that these manually encoded kinematic features are limited in improving the model fidelity; instead, we let the reward network learn the inertial features from raw sensor measurements.

6.3 Methodology

We frame the problem of terrain traversability modeling under the IRL paradigm, in which the traversability cost map is obtained as the negative reward function learned from expert demonstrations. We first formulate our problem as a Markov Decision Process (MDP), and briefly introduce the MEDIRL framework. In order to better incorporate robot kinematics into the framework, we propose a reward network architecture that can simultaneously extract inertial features from raw sensor measurements. A Trajectory-ranked MEDIRL algorithm is also proposed to deal with suboptimal demonstrations for legged robots.

6.3.1 Problem Statement

We model the process of a legged robot walking over local terrain as an agent following an MDP. An MDP is defined as $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, r\}$ consisting of a set of states $\mathcal{S}$, actions $\mathcal{A}$, transition probabilities $\mathcal{T}$, a discount factor $\gamma \in [0, 1)$, and a reward function $r : \mathcal{S} \to \mathbb{R}$. A trajectory is defined as a sequence of state-action pairs $\tau := \{(s_0, a_0), (s_1, a_1), ..., s_{|\tau|}\}$ followed by the agent, and a demonstration is a set of trajectories under a policy $\mathcal{D} = \{\tau_0, \tau_1, ..., \tau_{|\mathcal{D}|}\}$. Given an MDP, a forward RL problem seeks the optimal policy $\pi^* : \mathcal{S} \to \mathcal{A}$, that maximizes the expected discounted reward (return) $J(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | \pi]$. We consider the IRL problem: Given an MDP$\backslash r$ and expert demonstration $\mathcal{D}$, to recover the underlying reward function that explains the demonstration, i.e., to seek the reward function under which the demonstrated behavior is optimal.

Specifically, the MDP formulation is as follows. **State space**: We discretize the local

terrain into a 2D grid map, and the agent's state is the current position of the robot in the grid map. **Action space**: The actions of the agent in the discretized state space are chosen to be $\mathcal{A} = \{up, down, left, right\}$, which allow the robot to transit from the current grid to an adjacent grid. It can also be easily extended to include 8 directions with evenly spaced 45 degree intervals. **Traversability cost**: The traversability cost of each state is defined as the negative reward $c = -r : \mathcal{S} \to \mathbb{R}$.

6.3.2 Maximum Entropy Deep IRL

Maximum Entropy IRL is formulated to address the challenge of reward ambiguity (i.e., multiple rewards can explain the same behavior) by treating trajectories with higher rewards as exponentially more likely [220]. Under the same assumption, MEDIRL [221] further approximates the reward function using a deep neural network with a set of parameters θ, i.e, $\hat{r}_\theta(s) = f(\phi(s); \theta)$, where $\phi(s)$ denote the features of state sec. It is then mathematically formulated as a maximum-a-posteriori (MAP) estimation problem that maximizes the logarithmic joint posterior distribution of the demonstration and network parameters under the predicted reward:

$$\begin{aligned} \mathcal{L}(\theta) &= \log \mathbf{P}(\mathcal{D}, \theta | r_\theta) = \log \mathbf{P}(\mathcal{D} | r_\theta) + \log \mathbf{P}(\theta) \\ &=: \mathcal{L}_\mathcal{D} + \mathcal{L}_\theta, \end{aligned} \tag{6.1}$$

which is divided into a demonstration term $\mathcal{L}_\mathcal{D}$ and a model regularizer $\mathcal{L}_\theta$. Applying the chain rule, the reward network can be trained by backpropagating the MEDIRL gradient $\frac{\partial \mathcal{L}}{\partial r_\theta}$.

6.3.3 Learning Inertial Features: Network Architecture

As the essence of maximum entropy IRL is feature matching, the features for each state (i.e., the inputs to the reward network) are important choices. Previous works integrated manually encoded robot kinematics and environmental context into the reward network [7, 211, 213]. In these works, the reward function can be written as $\hat{r}_\theta(s) = f(\phi_e(s), \phi_k(s); \theta)$, where $\phi_e(s)$ and $\phi_k(s)$ are the handcrafted environmental features and kinematic features of state sec. In this work, we propose a network architecture that can learn the robot inertial features directly from proprioceptive sensory data and integrate the learned features in reward approximation: $\hat{r}_\theta(s) = f(\phi_e(s), \phi_i(s; \theta); \theta)$.

Inspired by the work of [222] and [217] where 1D convolutional modules are employed to efficiently process temporal proprioceptive sensory data for legged robots, we design

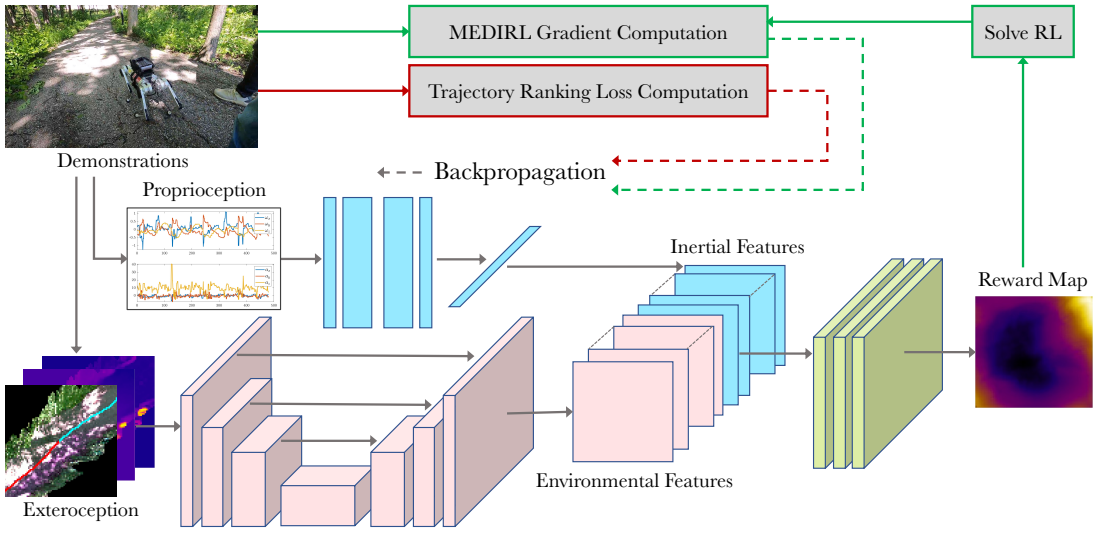


Figure 6.1: The proposed deep reward network architecture and training process for terrain traversability cost learning. The network has two stages where the first stage is for feature learning and the second stage is for feature fusion (yellow). To extract useful features from both proprioceptive and exteroceptive sensory data, the first stage consists of an inertial branch (blue) taking inputs of raw IMU signals and a UNet-like environmental branch (pink) taking inputs of the elevation map, elevation variance map and color map. The outputs of both branches are concatenated and fused in the second stage to approximate the reward map. The network is trained in an end-to-end manner by backpropagating both MEDIRL gradient (green flow) and trajectory ranking loss (red flow).

an inertial branch for the reward network using similar structures. The branch consists of 2 convolutional (Conv) blocks and 1 fully connected (FC) layer. Within the Conv block, two 1D convolutional layers are followed by a 1D max pooling layer. Dropout layers are also added to the Conv block for overfitting alleviation, serving as the model regularizer $\mathcal{L}_\theta$ in Eq. 6.1 [209]. For simplicity, we choose fixed-length IMU signals as the inputs to this branch. The 1D convolutional kernel is designed to move along the time domain of the signals to extract the temporal correlation. It is noteworthy that we have also tested on including other proprioceptive signals such as joint states, but have not observed improvement over using IMU signals alone. We conjecture that more sophisticated network design or pre-processing is needed to effectively use raw joint states.

To model traversability, the geometry and appearance of the local terrain also contain critical information. Thus, another branch of the network aims to incorporate environmental context into the reward learning. Similar to previous works [7,209,210,213], we choose the elevation map, elevation variance map and a bird's eye view color map as the environmental features. Due to the spatial nature of environmental inputs, we adopt a UNet [223]

architecture for image segmentation as the environmental branch. The main considerations are the simplicity and data efficiency. The UNet only receptively uses convolutions, ReLU and max pooling, consistent to our inertial branch. In addition, it exhibits good performance with a limited amount of training data.

The proposed reward network therefore has a two-stage architecture, as shown in Fig. 6.1. The first stage is comprised of the two aforementioned branches to process inertial (proprioceptive) data and environmental (exteroceptive) data individually. The inertial features learned/extracted from the first branch are then upsampled and concatenated with the outputs of the environmental branch. The second stage of the reward network is a block of 2D convolutions that fuses the two types of features into the final reward map.

6.3.4 Trajectory-ranked MEDIRL

In the existing MEDIRL framework, the reward network is trained by backpropagating the MEDIRL gradient $\frac{\partial \mathcal{L}}{\partial r_\theta}$, which can be obtained as the differences in the state visitation frequencies (SVF) from the demonstration and from the predicted reward [209, 220]. Therefore, the reward function obtained by only minimizing the differences will be the one best explaining the demonstrated behavior, and the corresponding policy will be suboptimal if the demonstration is suboptimal. In order to allow reward extrapolation when additional preference information is given, we propose to add a trajectory ranking loss to the MEDIRL framework and extend it to a Trajectory-ranked MEDIRL (T-MEDIRL) framework.

The trajectory ranking loss is inspired by the preference-based IRL methods [215, 224] which seek a reward function that explains the *ranking* over demonstrations (intention), in contrast to the *demonstrations* (behavior), thereby allowing for reward generalization and extrapolation. Specifically, given a sequence of ranked trajectories, the objective is to learn a reward function that assigns higher returns to higher-ranked trajectories. It can thus be formulated as a classification problem that predicts whether a trajectory has higher rank than another based on the predicted returns (originally based on the predicted reward function). The classifier can be trained by using the trajectory-rank pairs as data-label pairs and minimizing the pairwise trajectory ranking loss [224]:

$$\mathcal{L}_{i,j} = -\sum_{\tau_i \prec \tau_j} \log \frac{\exp \sum\limits_{s \in \tau_j} r_{\theta,j}(s)}{\exp \sum\limits_{s \in \tau_i} r_{\theta,i}(s) + \exp \sum\limits_{s \in \tau_j} r_{\theta,j}(s)}, \tag{6.2}$$

where $r_{\theta,i}$ and $r_{\theta,j}$ are the parameterized reward function for trajectory i and j, respectively, and $\tau_i \prec \tau_j$ means that trajectory j has a higher rank compared with trajectory i.

Algorithm 2 T-MEDIRL: Trajectory-ranked Maximum Entropy Deep Inverse Reinforcement Learning

Input: $\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, \mathcal{D}, \alpha$
Output: network parameters θ
function MEDIRL-GRAD($r_\theta, \tau, \mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma$)
    $\pi \leftarrow$ VALUE-ITERATION($r_\theta, \mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma$)
    $\mu_\tau \leftarrow$ STATE-VISITATIONS-TRAJ($\tau, \mathcal{S}, \mathcal{A}, \mathcal{T}$)
    $\mathbb{E}[\mu] \leftarrow$ STATE-VISITATIONS-POLICY($\pi, \mathcal{S}, \mathcal{A}, \mathcal{T}$)
    $\frac{\partial \mathcal{L}}{\partial r_\theta} \leftarrow \mu_\tau - \mathbb{E}[\mu]$
    return $\frac{\partial \mathcal{L}}{\partial r_\theta}$
end function

Randomly initialize θ^1
for n = 1 to N **do**
    Randomly choose τ_i and τ_j from $\mathcal{D}$
    $r_{\theta^n,i}(s) \leftarrow f(\phi_i(s); \theta^n)$ for $\forall s \in \mathcal{S}$
    $r_{\theta^n,j}(s) \leftarrow f(\phi_j(s); \theta^n)$ for $\forall s \in \mathcal{S}$
                                                ▷ Forward reward network
    $\frac{\partial \mathcal{L}_i}{\partial r_{\theta^n,i}} \leftarrow$ MEDIRL-GRAD($r_{\theta^n,i}, \tau_i, \mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma$)
    $\frac{\partial \mathcal{L}_j}{\partial r_{\theta^n,j}} \leftarrow$ MEDIRL-GRAD($r_{\theta^n,j}, \tau_j, \mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma$)
                                                ▷ Compute MEDIRL gradient
    $\mathcal{L}_{i,j} \leftarrow -\sum_{\tau_i \prec \tau_j} \log \frac{\exp \sum_{s \in \tau_j} r_{\theta^n,j}(s)}{\exp \sum_{s \in \tau_i} r_{\theta^n,i}(s) + \exp \sum_{s \in \tau_j} r_{\theta^n,j}(s)}$
                                                ▷ Compute trajectory ranking loss
    Compute $\nabla_{\theta^n} \mathcal{L}_i, \nabla_{\theta^n} \mathcal{L}_j, \nabla_{\theta^n} \mathcal{L}_{i,j}$ via chain rule
    $\theta^{n+1} \leftarrow \theta^n + \alpha \nabla_{\theta^n} \mathcal{L}_i + \alpha \nabla_{\theta^n} \mathcal{L}_j - \alpha \nabla_{\theta^n} \mathcal{L}_{i,j}$
                                                ▷ Update network parameters
end for

We use the aforementioned trajectory ranking loss to regularize the reward learning in the proposed T-MEDIRL algorithm (Algorithm 2). Given the MDP$\backslash r$ and demonstration with ranked trajectories, the T-MEDIRL framework learns to approximate the reward function r_θ by backpropagating the MEDIRL gradients and trajectory ranking loss together, as shown in Fig. 6.1. In each iteration, we randomly choose two (batches of) trajectories and obtain the corresponding predicted reward functions by forwarding the reward network. The per-trajectory MEDIRL gradient can be computed using the MEDIRL-Grad function, while the trajectory ranking loss is computed using (6.2) on a cross-trajectory basis. Finally, the network parameters are updated according to the learning rate α and gradients computed using the chain rule.

6.3.5 Locomotion Energy Ranked Reward Extrapolation

Energy cost is one of the major factors that can quantify the locomotion performance for legged robots, and an important consideration in path/motion planning. As a source of robot-terrain interaction, it also contains important information about the traversed terrain. Faigl et al. [208] use the mean power consumption as an indicator of terrain traversability cost, and Walas et al. [225] optimize the gait parameters for a humanoid robot based on the energy expenditure and locomotion stability. However, the energy cost is usually unknown or unpredictable to the human demonstrator when collecting demonstrations for legged robot walking, leading to a severe suboptimality issue, i.e., the demonstrated trajectory might be feasible but energy inefficient.

To extrapolate beyond the suboptimal demonstrations for legged robots and learn a reward function that leads to more energy-efficient path planning, we propose to use the Average Energy Consumption (AEC) of each trajectory for trajectory ranking in the T-MEDIRL framework. The AEC is computed by dividing the overall energy consumed for each trajectory by the length of the trajectory, and the overall energy consumed can be obtained using recorded robot joint states as [225]:

$$e_\tau = \sum_{i=1}^{n} \langle |\boldsymbol{u}_i|, |\Delta \boldsymbol{q}_i| \rangle, \tag{6.3}$$

where $|\cdot|$ takes the element-wise absolute values, $\langle \cdot, \cdot \rangle$ denotes the dot product, $\boldsymbol{u} \in \mathbb{R}^m$ and $\Delta \boldsymbol{q} \in \mathbb{R}^m$ are the joint torque and joint displacement for m joints between two timestamps, respectively, and n is the total length of the signals collected along the entire trajectory τ. In this work, if τ_j has a lower AEC value than τ_i, then $\tau_i \prec \tau_j$.

6.4 Experiments

We first introduce the process of data collection and generation. Then, two experiments are designed to evaluate the two contributions of this work.

6.4.1 Legged Robot Walking Dataset

We collect data using a quadruped robot with a customized sensor suite operating in a campus environment including diverse scenes. The robot platform, an MIT Mini-Cheetah, and the data collection environment are shown in Fig. 6.2. The Mini-Cheetah robot is equipped with an Intel RealSense depth camera, an IMU, and an Nvidia Jetson Xavier computing de-

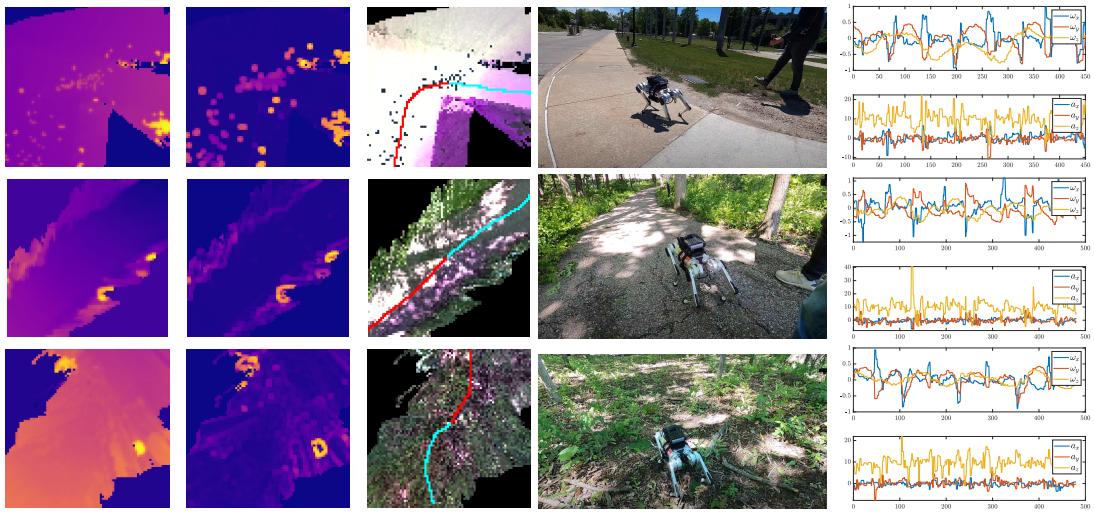


Figure 6.2: MIT Mini-Cheetah robot with a customized sensor suite and satellite map of the data collection environment for this work.

vice for data recording. We manually teleoperate the robot to walk around by sending the commands of longitudinal velocity, lateral velocity and yaw rate. The color/depth images, IMU signals, and joint state measurements are synchronized and recorded using the Robot Operating System.

We generate a dataset of over 1000 trajectories and their corresponding environmental features, IMU signals and AEC values using both exteroceptive and proprioceptive sensory data from different sequences. For the environmental features, we choose to build the elevation map, elevation variance map and color map of a region of $8m \times 8m$ with $0.1m$ resolution. Specifically, we use ORB-SLAM2 [226] to estimate the robot poses, and build a local robot-centric probabilistic elevation map using point clouds captured by the depth camera. Elevation mapping [192] models the height measurements by a Gaussian distribution and fuses multi-frame measurements using a Kalman filter. We generate the elevation map and elevation variance map using the mean and variance of each grid's Gaussian distribution. The color map is generated by averaging the RGB values of all points falling into each grid.

The demonstrated trajectory is generated by projecting the robot poses onto the local grid map, including a past trajectory and a future trajectory. The past trajectory is a sequence of grid indices traversed by the robot, whereas the future trajectory is for the same period of time after map building. The IMU signals used for the inertial feature learning are the signals collected within the past 0.5 sec, and the AEC value is computed using the joint state measurements of all 12 joints (3 joints per leg) along the past trajectory. Examples of the generated data are visualized in Fig. 6.3.



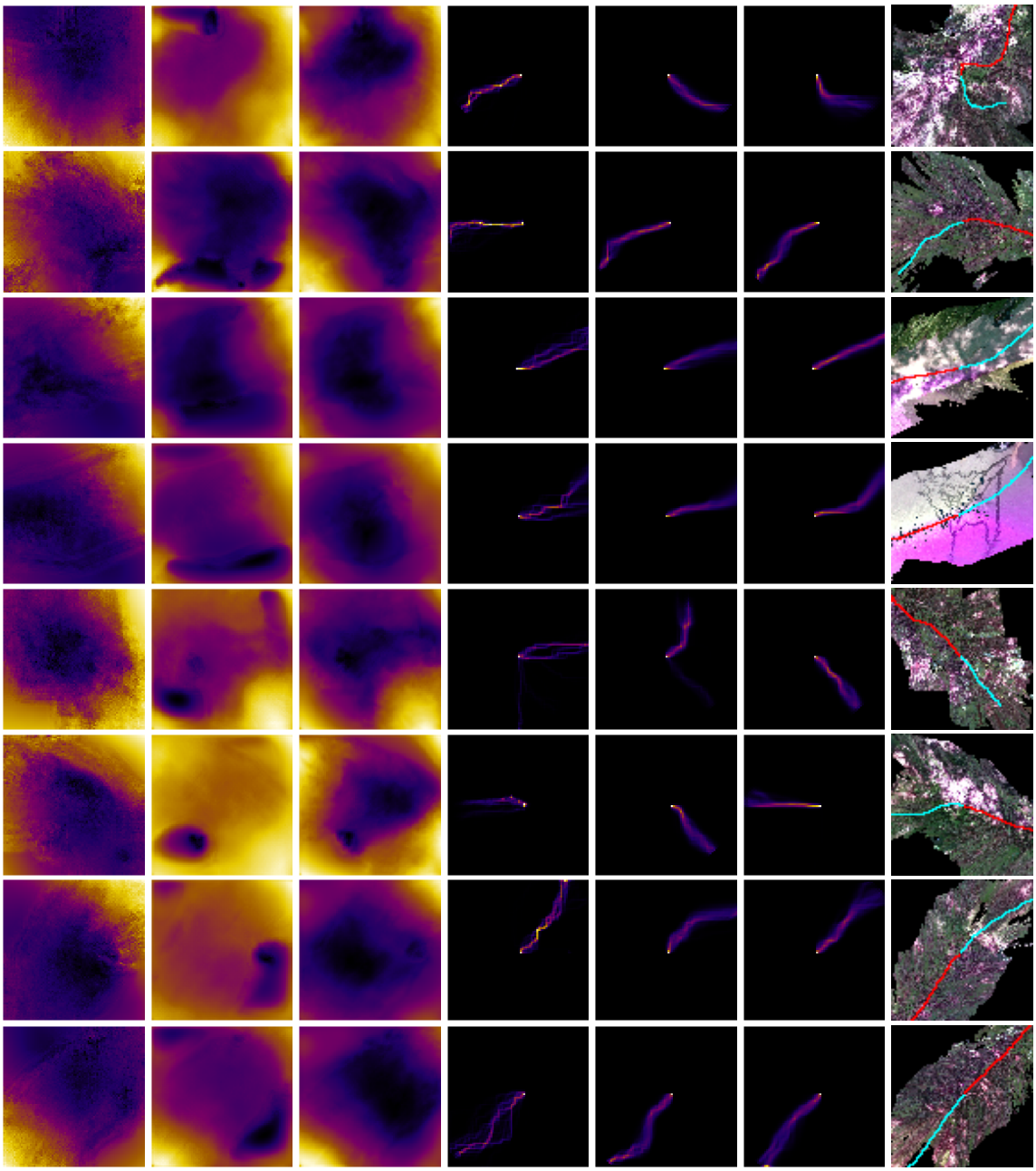
(a) (b) (c) (d) (e)

Figure 6.3: Visualization of the generated data/network inputs in different environments, for instance, sidewalk, trail and forest. (a) Elevation map. (b) Elevation variance map. (c) Color map. The demonstrated past (red) and future (cyan) trajectories are overlaid for visualization only and not part of the feature map. (d) Third-person view of the current state (not an input). (e) The IMU signals collected within 0.5 sec.

6.4.2 Inertial Feature Learning Evaluation

This experiment aims to evaluate the performance of the proposed reward network that can learn robot inertial features and incorporate the learned features for reward approximation. To this end, we compare our method with a baseline approach using handcrafted kinematic features [7], and our method without inertial feature learning. For a fair comparison on the network performance and model fidelity, three networks are trained using the MEDIRL framework and evaluated on how well the approximated reward function explains the expert demonstration. Therefore, we employ two metrics commonly used in IRL for quantitative evaluation: the Negative Log-Likelihood (NLL) of the expert demonstration under the learned policy, and the Hausdorff Distance (HD) [227] between demonstrations and trajectories sampled with the learned policy. Here, the learned policy is obtained by value iteration given the learned reward function (line 4 in Algorithm 2).

In this experiment, we apply the baseline's original implementation and kinematic feature encoding [7] to our dataset. For a fair comparison, we keep the same MEDIRL training process and only replace the reward network by our proposed architectures for each of our methods. For our method without inertial feature learning, we exclude the inertial branch and only use the environmental branch. We use a 70/30 split to divide our data into training and test sets.



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

Figure 6.4: Qualitative results on traversability cost learning and future trajectory prediction in Experiment 1. (a)-(c) are the inferred reward maps of the baseline method [7], our method without inertial feature learning and our method with inertial feature learning, respectively. (d)-(f) are the SVF visualizations of the three methods representing the future trajectory distributions. (g) are the corresponding input color maps overlaid with past (red) and future (cyan) trajectories.

Table 6.1: Traversability Cost Learning/Future Trajectory Prediction Performance Quantitative Comparison in Experiment 1.

Method	NLL ↓	HD ↓
Baseline [7] w/ kinematics	0.9269	12.1843
Ours w/o inertial features	0.7448	7.1942
Ours	**0.6995**	**6.3165**

The qualitative results are given in Fig. 6.4. The first three columns are the inferred reward maps on selected test data, and the next three columns are the corresponding SVF visualizations representing the future trajectory distributions under the inferred reward and policy. The input color maps are also shown in the last column for each scene, and overlaid with past trajectories (red) and ground truth future trajectories (cyan).

Fig. 6.4 clearly shows the effectiveness of inertial feature learning and integrating for reward approximation. The trajectories predicted by our method best align with the ground truth future trajectories in demonstration. Our network without the inertial features predicts correct future trend in most cases, but failed in the last two scenes. Even in the cases where it successfully predicts the future trend, our method with the learned inertial features predicts the trajectory curve more accurately, for instance in the top two forest scenes. Both methods perform similarly in the trail and sidewalk environments, shown in the middle two rows.

The baseline method using handcrafted kinematic features does not perform well in our experiment. We conjecture that the kinematic features used, i.e., the average velocity and curvature of the past trajectory, cannot capture the instantaneous inertial context and thus tend to fail when the robot suddenly changes its motion. This can also be observed from the top two rows in Fig. 6.4, where the trajectory predicted by the baseline seems to extrapolate the past trajectory. In contrast, by integrating the learned inertial features, our method can better capture the instantaneous motion changes.

The superiority of our method for legged robots can also be seen in the quantitative results in Table 6.1, where our method has the best prediction results on the test set using both metrics. By learning and incorporating robot inertial features in the proposed reward network, the inferred reward function (traversability cost) becomes highly robot-state-dependent, thus achieving better performance than purely environment-dependent reward.

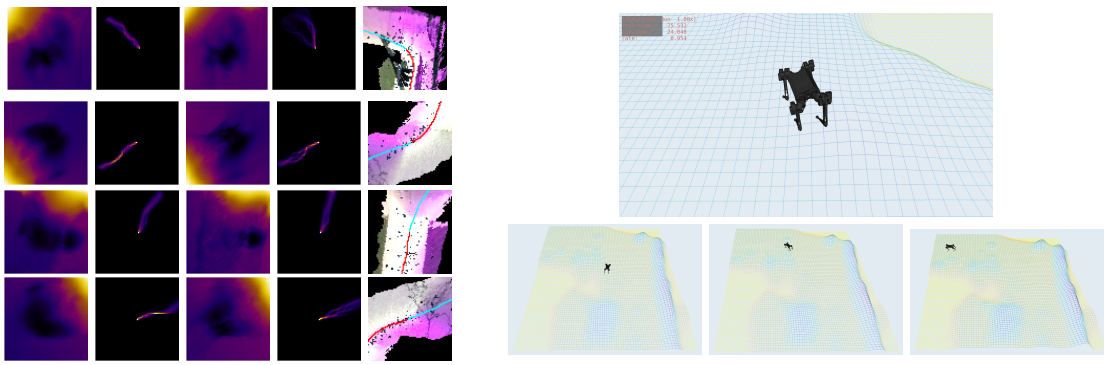


Figure 6.5: Left block: Qualitative results on traversability cost learning and future trajectory prediction in Experiment 2. From left, the first and second columns are the inferred reward maps and future trajectory distributions obtained using the MEDIRL algorithm. The third and fourth columns are those obtained using our proposed T-MEDIRL algorithm with energy-based trajectory ranking. The fifth column is the corresponding input color maps overlaid with past (red) and future (cyan) trajectories. Right block: Mini-Cheetah Simulator. The bottom row shows the robot is following the optimal future trajectory.

6.4.3 Energy-based Reward Extrapolating Evaluation

In the second experiment, we evaluate the effect of locomotion energy ranked reward extrapolation. We use the AEC value of each trajectory to rank the demonstrated trajectories in the training set, and train the proposed reward network using the proposed T-MEDIRL algorithm. For comparison, we choose to train the same reward network using the original MEDIRL algorithm as our baseline.

The qualitative results are given in Fig. 6.5. The first two columns are the inferred reward maps and the predicted trajectory distributions obtained using the MEDIRL algorithm on the test set, and the following two columns are obtained using our T-MEDIRL algorithm. The corresponding color maps and demonstrated trajectories are shown in the last column. We observe that the reward maps obtained by T-MEDIRL contain more details whereas those obtained by MEDIRL are more smooth. We believe that this is due to a higher state distinguishability brought by trajectory ranking. Without trajectory ranking, all states traversed by the agent have the same level of preference during training, whereas trajectory ranking introduces different preference labels to these states. Another observation is that the trajectory distributions predicted by T-MEDIRL tend to be multi-modal distributions, showing a better generalizing ability of T-MEDIRL.

As the purpose of introducing energy-based trajectory ranking loss is to learn a reward/policy better than demonstration instead of best explaining the demonstration, quantitative metrics other than NLL and HD are needed in this experiment. In classical IRL problems, the quality of the predicted reward can be evaluated by comparing with the

Table 6.2: Traversability Cost Learning/Future Trajectory Prediction Performance Quantitative Comparison in Experiment 2. Using a $24\mathrm{V}, 6\mathrm{Ah}$ battery, the difference shown below in AEC corresponds to about seven minutes extra operation.

Method	ours w/ MEDIRL [209]	ours w/ T-MEDIRL
NLL ↓	**0.6705**	0.6810
HD ↓	**5.0129**	5.5718
Accuracy ↑	39.13%	**60.86%**
AEC ↓	$4.596 \times 10^{-2}\mathrm{J}$	$4.088 \times 10^{-2}\mathrm{J}$

ground truth reward. Due to the lack of ground truth reward in real-world problems, we introduce the following two metrics for evaluation.

The first one is classification accuracy. The reward learning in T-MEDIRL framework can be formulated as a classification problem, where the trajectory with higher predicted return should be classified as higher ranked given the predicted reward. Thus, we compare the classification accuracy on the test set for both methods. Specifically, for any pair of trajectories in the test set, we compute their returns from the predicted reward maps, if the one having higher return corresponds to a smaller AEC value, it is considered to be correct. The accuracy is computed by dividing the total number of correct classifications by the total number of comparisons.

Secondly, to evaluate whether the reward map learned by our T-MEDIRL algorithm can lead to a more energy-efficient policy, we generate optimal future trajectories from the reward maps learned by both methods, and compare the AEC values of these planned trajectories. To this end, we utilize a simulation environment for Mini-Cheetah that provides the robot model, a state estimator and a controller.

We implement a simple PID-based waypoint follower to replace the remote controller in the original simulator for sending motion commands to the controller. For each planned trajectory of the test data, we load the corresponding elevation map to the simulator to simulate the terrain conditions, and make the robot follow the given trajectory under the trot mode, as shown in Fig. 6.5. We log the simulated robot joint states during the trajectory following for all planned trajectories in the test set, and compute the AEC for all trajectories. For each method, using a sampling time of $2\ \mathrm{ms}$, we compute the overall energy consumed for all planned trajectories using (6.3), and divide it by the length of signals.

The quantitative results are given in Table 6.2, where we report the classification accuracy and AEC besides NLL and HD for both methods. As expected, MEDIRL has better prediction results compared with T-MEDIRL (shown by the lower NLL and HD), which

indicates that MEDIRL learns a reward function that better explains the demonstration, and a policy closer to the demonstrated behavior. In contrast, the reward maps learned using T-MEDIRL lead to a much higher classification accuracy, which shows the effect of the energy-based trajectory ranking loss on reward regulation. Last but not least, the lower AEC of the optimal future trajectories planned by T-MEDIRL directly shows that the reward maps learned by T-MEDIRL are extrapolated beyond the suboptimal demonstrations and lead to a more energy-efficient policy than demonstration.

Another interesting observation from Table 6.2 is the close NLL values for both methods. Although the trajectories planned by T-MEDIRL are spatially further away from the expert demonstration (i.e., the relatively larger HD), the likelihood of the demonstrated trajectories under the policy learned by T-MEDIRL is still relatively high. It indicates that T-MEDIRL tends to plan a different trajectory to the demonstrated one, however, the demonstrated trajectory remains a good policy under the learned reward.

6.5 Conclusion

This chapter addresses the problem of terrain traversability modeling for legged robots using deep inverse reinforcement learning. Two issues have been mainly considered in this work. First, unlike vehicles, legged robots with high mobility tend to have more sudden motions. Existing DIRL works using purely environmental features or handcrafted vehicle kinematic features might fail to capture these sudden movements. The proposed deep reward network can effectively learn robot inertial features from proprioceptive sensory data and integrate the learned features into the reward map. We have shown that the proposed reward network outperforms the baselines for reward learning and trajectory prediction. Secondly, we have proposed a T-MEDIRL algorithm and locomotion trajectory ranking strategy using energy to deal with the suboptimality of legged robot demonstrations. By introducing a preference label for each demonstrated trajectory, the inferred reward map can extrapolate beyond suboptimal demonstrations and lead to a better-than-demonstration policy.

CHAPTER 7

Future Research Directions

This chapter outlines several main ideas for future research. Based on the findings in this dissertation and most recent related works, I have summarized them into three directions: end-to-end learning for robotic map representation, legged robots terrain traversability modeling for agile space exploration, and semantic SLAM in a multi-robot setting. The details are given as follows:

- **End-to-End Learning for Robotic Map Representation.** In this dissertation, the underlying map representation consists of different probability distributions, for instance, Categorical distribution for semantic maps in Chapter 4, Bernoulli distribution for occupancy and traversability maps in Chapter 5, etc. The most natural distribution is chosen to represent different environmental information, and to infer its corresponding posterior distribution by incorporating measurements obtained from a multi-task learning (MTL) deep network. In the future, I will explore the expressive power of recurrent neural networks (RNNs) in map representation learning. The expected result is a unified deep learning framework that takes temporal sequential multi-modal exteroceptive sensory data and outputs a novel dynamic robot-centric map representation. Specifically, as I envision that the ultimate map representation for robots will be heterogeneous and graph-based, I will focus on devising a multi-task recurrent graph neural network (RGNN) [228] for map representation learning.

- **Legged Robots Terrain Traversability Modeling for Agile Space Exploration.** Compared with ground vehicles, legged robots have the potential for traversing extreme, off-road, and unstructured terrains, largely expanding the scope of robot/human activities and scientific discoveries. For instance, in space exploration, many areas of scientific interest, such as lunar pits, icy-moon crevasses, and Martian craters, are inaccessible to current wheeled rovers [229]. Legged robots have the ability to traverse these tricky terrains, however, enabling fully autonomous traversing remains

challenging. In Chapter 6, an energy-based terrain traversability modeling method is proposed for legged robots path planning. Instead of fitting a manually-defined model-based traversability cost function, we learn the traversability cost as negative reward from expert demonstrations in a deep inverse reinforcement learning (IRL) framework. This work improves the reward/traversability cost prediction performance for legged robots and deals with the demonstration suboptimality. In order to increase the data efficiency and generalizing ability of the current approach, I will further explore the application of meta learning theory for task adaptive reward learning. Preliminary research related to this topic has also been presented as multi-task IRL [230] and meta-IRL [231].

- **Multi-Agent Semantic SLAM.** Besides robotic mapping and environmental modeling, robot state estimation is also crucial for motion and task planning. Decades of research on simultaneously localization and mapping (SLAM) have made tremendous progress for single robot application. However, multi-agent SLAM such as collaborative SLAM [232] and distributed SLAM [233] remains an open problem. In the setting of multiple robots, SLAM is a hard optimization problem with ambiguous data association. I believe my works on semantic mapping can provide a new and efficient approach for probabilistic data association, thereby improving the pose estimation accuracy.

To conclude, my future research will continue on robot perception and learning for motion and task planning. Current end-to-end navigation deep networks can only complete specific navigation goals which are trained for [198, 234, 235]. However, a proper map representation and its versatile interfaces will provide a general and reliable solution for various robotic tasks, and eventually allow robots to assist humans in hazardous missions, and to aid with scientific discovery and exploration.

BIBLIOGRAPHY

[1] Armeni, I., Sax, A., Zamir, A. R., and Savarese, S., "Joint 2D-3D-Semantic Data for Indoor Scene Understanding," *ArXiv e-prints*, 2017.

[2] Yang, S., Huang, Y., and Scherer, S., "Semantic 3D Occupancy Mapping through Efficient High Order CRFs," 2017, pp. 590–597.

[3] Kundu, A., Li, Y., Dellaert, F., Li, F., and Rehg, J. M., "Joint semantic segmentation and 3D reconstruction from monocular video," *European Conf. on Computer Vision*, Springer, 2014, pp. 703–718.

[4] Ghaffari Jadidi, M., Valls Miro, J., and Dissanayake, G., "Sampling-based incremental information gathering with applications to robotic exploration and environmental monitoring," Vol. 38, No. 6, 2019, pp. 658–685.

[5] Sengupta, S., Greveson, E., Shahrokni, A., and Torr, P. H., "Urban 3D semantic modelling using stereo vision," 2013, pp. 580–585.

[6] Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., and Gall, J., "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," 2019, pp. 9297–9307.

[7] Zhang, Y., Wang, W., Bonatti, R., Maturana, D., and Scherer, S., "Integrating kinematics and environment context into deep inverse reinforcement learning for predicting off-road vehicle trajectories," *arXiv preprint arXiv:1810.07225*, 2018.

[8] Nathan Silberman, Derek Hoiem, P. K. and Fergus, R., "Indoor Segmentation and Support Inference from RGBD Images," *ECCV*, 2012.

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

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

[11] Kaess, M., Ranganathan, A., and Dellaert, F., "iSAM: Incremental smoothing and mapping," *IEEE Trans. Robot.*, Vol. 24, No. 6, 2008, pp. 1365–1378.

[12] Elfes, A., "Using occupancy grids for mobile robot perception and navigation," *Computer*, Vol. 22, No. 6, 1989, pp. 46–57.

[13] Garg, S., Sünderhauf, N., Dayoub, F., Morrison, D., Cosgun, A., Carneiro, G., Wu, Q., Chin, T.-J., Reid, I., Gould, S., et al., "Semantics for robotic mapping, perception and interaction: A survey," *arXiv preprint arXiv:2101.00443*, 2021.

[14] Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., and Burgard, W., "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous robots*, Vol. 34, No. 3, 2013, pp. 189–206.

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

[16] Salas-Moreno, R. F., Newcombe, R. A., Strasdat, H., Kelly, P. H., and Davison, A. J., "SLAM++: Simultaneous localisation and mapping at the level of objects," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2013, pp. 1352–1359.

[17] Kochanov, D., Ošep, A., Stückler, J., and Leibe, B., "Scene flow propagation for semantic mapping and object discovery in dynamic street scenes," *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 1785–1792.

[18] Moravec, H. P. and Elfes, A., "High resolution maps from wide angle sonar," *Proc. IEEE Int. Conf. Robot Automat.*, Vol. 2, IEEE, 1985, pp. 116–121.

[19] Elfes, A., "Sonar-based real-world mapping and navigation," *Robotics and Automation, IEEE Journal of*, Vol. 3, No. 3, 1987, pp. 249–265.

[20] Thrun, S., "Learning occupancy grid maps with forward sensor models," *Autonomous robots*, Vol. 15, No. 2, 2003, pp. 111–127.

[21] Merali, R. S. and Barfoot, T. D., "Optimizing online occupancy grid mapping to capture the residual uncertainty," *Proc. IEEE Int. Conf. Robot Automat.*, IEEE, 2014, pp. 6070–6076.

[22] Yamauchi, B., "A frontier-based approach for autonomous exploration," *Int. Sym. Comput. Intell. Robot. Automat.*, 1997, pp. 146–151.

[23] Stachniss, C., Grisetti, G., and Burgard, W., "Information Gain-based Exploration Using Rao-Blackwellized Particle Filters." *Robotics: Science and Systems*, Vol. 2, 2005.

[24] Charrow, B., Liu, S., Kumar, V., and Michael, N., "Information-theoretic mapping using Cauchy-Schwarz quadratic mutual information," *Proc. IEEE Int. Conf. Robot Automat.*, IEEE, 2015, pp. 4791–4798.

[25] Wolcott, R. W. and Eustice, R. M., "Robust LIDAR localization using multiresolution Gaussian mixture maps for autonomous driving," *The Int. J. Robot. Res.*, Vol. 36, No. 3, 2017, pp. 292–319.

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

[27] Kim, S. and Kim, J., "Occupancy Mapping and Surface Reconstruction Using Local Gaussian Processes With Kinect Sensors," *IEEE Transactions on Cybernetics*, Vol. 43, No. 5, 2013, pp. 1335–1346.

[28] Ghaffari Jadidi, M., Miro, J. V., Valencia, R., and Andrade-Cetto, J., "Exploration on Continuous Gaussian Process Frontier Maps," *Proc. IEEE Int. Conf. Robot Automat.*, 2014, pp. 6077–6082.

[29] Ghaffari Jadidi, M., Valls Miro, J., and Dissanayake, G., "Gaussian Process Autonomous Mapping and Exploration for Range Sensing Mobile Robots," *arXiv*, 2017.

[30] Brostow, G., Shotton, J., Fauqueur, J., and Cipolla, R., "Segmentation and recognition using structure from motion point clouds," *Computer Vision–ECCV 2008*, 2008, pp. 44–57.

[31] Shotton, J., Johnson, M., and Cipolla, R., "Semantic texton forests for image categorization and segmentation," *IEEE Conf. on Computer vision and pattern recognition*, IEEE, 2008, pp. 1–8.

[32] Long, J., Shelhamer, E., and Darrell, T., "Fully convolutional networks for semantic segmentation," *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[33] Badrinarayanan, V., Kendall, A., and Cipolla, R., "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv:1511.00561*, 2015.

[34] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L., "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *arXiv preprint arXiv:1606.00915*, 2016.

[35] Noh, H., Hong, S., and Han, B., "Learning deconvolution network for semantic segmentation," *Proceedings of the IEEE Int. Conf. on Computer Vision*, 2015, pp. 1520–1528.

[36] Mozos, O. M., Triebel, R., Jensfelt, P., Rottmann, A., and Burgard, W., "Supervised semantic labeling of places using information extracted from sensor data," *Robot. Auton. Syst.*, Vol. 55, No. 5, 2007, pp. 391–402.

[37] Pronobis, A., Martinez Mozos, O., Caputo, B., and Jensfelt, P., "Multi-modal semantic place classification," *The Int. J. Robot. Res.*, Vol. 29, No. 2-3, 2010, pp. 298–320.

[38] Lowe, D. G., "Distinctive image features from scale-invariant keypoints," *Int. journal of computer vision*, Vol. 60, No. 2, 2004, pp. 91–110.

[39] Linde, O. and Lindeberg, T., "Object recognition using composed receptive field histograms of higher dimensionality," *Proc. Int. Conf. Pattern Recognition*, Vol. 2, IEEE, 2004, pp. 1–6.

[40] Castle, R. O., Gawley, D. J., Klein, G., and Murray, D. W., "Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras," *Proc. IEEE Int. Conf. Robot Automat.*, 2007, pp. 4102–4107.

[41] Civera, J., Gálvez-López, D., Riazuelo, L., Tardós, J. D., and Montiel, J., "Towards semantic SLAM using a monocular camera," *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 1277–1284.

[42] Bao, S. Y. and Savarese, S., "Semantic structure from motion," *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, IEEE, 2011, pp. 2025–2032.

[43] Pillai, S. and Leonard, J., "Monocular SLAM Supported Object Recognition," *Robotics: Science and Systems*, Rome, Italy, July 2015.

[44] Kim, Y. M., Mitra, N. J., Yan, D.-M., and Guibas, L., "Acquiring 3D indoor environments with variability and repetition," *ACM Transactions on Graphics (TOG)*, Vol. 31, No. 6, 2012, pp. 138.

[45] Choudhary, S., Trevor, A. J., Christensen, H. I., and Dellaert, F., "SLAM with object discovery, modeling and mapping," *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 1018–1025.

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

[47] Vineet, V., Miksik, O., Lidegaard, M., Nießner, M., Golodetz, S., Prisacariu, V. A., Kähler, O., Murray, D. W., Izadi, S., Pérez, P., et al., "Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction," *Proc. IEEE Int. Conf. Robot Automat.*, 2015, pp. 75–82.

[48] Ghaffari Jadidi, M., Valls Miro, J., and Dissanayake, G., "Warped Gaussian Processes Occupancy Mapping with Uncertain Inputs," *IEEE Robotics and Automation Letters*, Vol. 2, No. 2, 2017, pp. 680 – 687.

[49] Stein, M., *Interpolation of spatial data: some theory for kriging*, Springer Verlag, 1999.

[50] Neal, R. M., *Bayesian learning for neural networks*, Vol. 118, Springer New York, 1996.

[51] Minka, T. P., *A family of algorithms for approximate Bayesian inference*, Ph.D. thesis, Massachusetts Institute of Technology, 2001.

[52] Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K., "An introduction to variational methods for graphical models," *Machine learning*, Vol. 37, No. 2, 1999, pp. 183–233.

[53] Snelson, E. and Ghahramani, Z., "Sparse Gaussian processes using pseudo-inputs," *NIPS*, 2006, pp. 1257–1264.

[54] Naish-Guzman, A. and Holden, S. B., "The Generalized FITC Approximation." *NIPS*, 2007, pp. 1057–1064.

[55] Thrun, S., Burgard, W., and Fox, D., *Probabilistic robotics*, Vol. 1, MIT press, 2005.

[56] Kendall, A., Badrinarayanan, V., , and Cipolla, R., "Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding," *arXiv preprint arXiv:1511.02680*, 2015.

[57] Fawcett, T., "An introduction to ROC analysis," *Pattern recognition letters*, Vol. 27, No. 8, 2006, pp. 861–874.

[58] Hand, D. J. and Till, R. J., "A simple generalisation of the area under the ROC curve for multiple class classification problems," *Machine learning*, Vol. 45, No. 2, 2001, pp. 171–186.

[59] Charrow, B., Kumar, V., and Michael, N., "Approximate representations for multi-robot control policies that maximize mutual information," *Auton. Robot*, Vol. 37, No. 4, 2014, pp. 383–400.

[60] Ghaffari Jadidi, M., Valls Miro, J., and Dissanayake, G., "Sampling-based Incremental Information Gathering with Applications to Robotic Exploration and Environmental Monitoring," *arXiv*, 2017.

[61] Goldberg, P. W., Williams, C. K., and Bishop, C. M., "Regression with input-dependent noise: A Gaussian process treatment," *NIPS*, 1998, pp. 493–499.

[62] Kersting, K., Plagemann, C., Pfaff, P., and Burgard, W., "Most likely heteroscedastic Gaussian process regression," *Proc. Int. Conf. Machine learning*, ACM, 2007, pp. 393–400.

[63] Ko, J. and Fox, D., "GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models," *Auton. Robot*, Vol. 27, No. 1, 2009, pp. 75–90.

[64] Titsias, M. K. and Lázaro-Gredilla, M., "Variational heteroscedastic Gaussian process regression," *Proc. Int. Conf. Machine learning*, ACM, 2011, pp. 841–848.

[65] Murphy, K. P., *Machine learning: a probabilistic perspective*, The MIT Press, 2012.

[66] Tipping, M. E., Faul, A. C., et al., "Fast marginal likelihood maximisation for sparse Bayesian models." *AISTATS*, 2003.

[67] Geiger, A., Lenz, P., and Urtasun, R., "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," 2012.

[68] Carlevaris-Bianco, N., Ushani, A. K., and Eustice, R. M., "University of Michigan North Campus long-term vision and lidar dataset," Vol. 35, No. 9, 2016, pp. 1023–1035.

[69] Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H., "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," 2018.

[70] Yu, F. and Koltun, V., "Multi-Scale Context Aggregation by Dilated Convolutions," *Proc. Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, 2016, pp. 1–13.

[71] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C., "MobileNetV2: Inverted Residuals and Linear Bottlenecks," 2018.

[72] Eigen, D. and Fergus, R., "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," 2015, pp. 2650–2658.

[73] Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K., and Pollefeys, M., "SEMANTIC3D.NET: A new large-scale point cloud classification benchmark," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. IV-1-W1, 2017, pp. 91–98.

[74] Boulch, A., Guerry, J., Saux, B. L., and Audebert, N., "SnapNet: 3D point cloud semantic labeling with 2D deep segmentation networks," *Computers & Graphics*, Vol. 71, 2018, pp. 189 – 198.

[75] Qi, C. R., Su, H., Mo, K., and Guibas, L. J., "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," 2017, pp. 77–85.

[76] Qi, C. R., Yi, L., Su, H., and Guibas, L. J., "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," 2017, pp. 5099–5108.

[77] Wu, B., Wan, A., Yue, X., and Keutzer, K., "SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud," IEEE, 2018, pp. 1887–1893.

[78] Landrieu, L. and Simonovsky, M., "Large-Scale Point Cloud Semantic Segmentation with Superpoint Graphs," 2018, pp. 4558–4567.

[79] Qi, C. R., Liu, W., Wu, C., Su, H., and Guibas, L. J., "Frustum pointnets for 3d object detection from rgb-d data," 2018, pp. 918–927.

[80] He, H. and Upcroft, B., "Nonparametric semantic segmentation for 3D street scenes," 2013, pp. 3697–3703.

[81] Stückler, J., Biresev, N., and Behnke, S., "Semantic mapping using object-class segmentation of RGB-D images," 2012, pp. 3005–3010.

[82] Xiang, Y. and Fox, D., "DA-RNN: Semantic Mapping with Data Associated Recurrent Neural Networks," 2017.

[83] Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., and Fitzgibbon, A., "KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera," *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, ACM, 2011, pp. 559–568.

[84] Sun, L., Yan, Z., Zaganidis, A., Zhao, C., and Duckett, T., "Recurrent-OctoMap: Learning State-Based Map Refinement for Long-Term Semantic Mapping With 3-D-Lidar Data," *IEEE Robotics and Automation Letters*, Vol. 3, No. 4, Oct 2018, pp. 3749–3756.

[85] Kim, B.-s., Kohli, P., and Savarese, S., "3D scene understanding by Voxel-CRF," 2013, pp. 1425–1432.

[86] Hermans, A., Floros, G., and Leibe, B., "Dense 3D semantic mapping of indoor scenes from RGB-D images," 2014, pp. 2631–2638.

[87] Zhao, Z. and Chen, X., "Building 3D semantic maps for mobile robots using RGB-D camera," *Intell. Serv. Robot.*, Vol. 9, No. 4, 2016, pp. 297–309.

[88] Schölkopf, B. and Smola, A. J., *Learning with kernels: support vector machines, regularization, optimization, and beyond*, MIT press, 2002.

[89] Candela, J. Q., *Learning with uncertainty-Gaussian processes and relevance vector machines*, Ph.D. thesis, PhD thesis, Technical University of Denmark, 2004.

[90] Tipping, M. E., "Sparse Bayesian learning and the relevance vector machine," *J. machine learning res.*, Vol. 1, No. Jun, 2001, pp. 211–244.

[91] Maturana, D., "scrollgrid," https://github.com/dimatura/scrollgrid, 2016.

[92] Geiger, A., Roser, M., and Urtasun, R., "Efficient Large-Scale Stereo Matching," *Asian Conf. Computer Vision*, 2010.

[93] Mur-Artal, R. and Tardós, J. D., "ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras," *IEEE Transactions on Robotics*, Vol. 33, No. 5, 2017, pp. 1255–1262.

[94] Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T., "LabelMe: a database and web-based tool for image annotation," *International journal of computer vision*, Vol. 77, No. 1-3, 2008, pp. 157–173.

[95] Nüchter, A. and Hertzberg, J., "Towards semantic maps for mobile robots," Vol. 56, No. 11, 2008, pp. 915–926.

[96] Wolf, D. F. and Sukhatme, G. S., "Semantic mapping using mobile robots," Vol. 24, No. 2, 2008, pp. 245–258.

[97] Kostavelis, I. and Gasteratos, A., "Semantic mapping for mobile robotics tasks: A survey," Vol. 66, 2015, pp. 86–103.

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

[99] Kim, S. and Kim, J., "GPmap: A Unified Framework for Robotic Mapping Based on Sparse Gaussian Processes," Springer, 2015, pp. 319–332.

[100] Ghaffari Jadidi, M., Valls Miró, J., Valencia, R., and Andrade-Cetto, J., "Exploration on continuous Gaussian process frontier maps," IEEE, 2014, pp. 6077–6082.

[101] Wang, J. and Englot, B., "Fast, accurate Gaussian process occupancy maps via test-data octrees and nested Bayesian fusion," IEEE, 2016, pp. 1003–1010.

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

[103] Ghaffari Jadidi, M., Valls Miro, J., and Dissanayake, G., "Gaussian processes autonomous mapping and exploration for range-sensing mobile robots," Vol. 42, No. 2, 2018, pp. 273–290.

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

[105] Doherty, K., Shan, T., Wang, J., and Englot, B., "Learning-aided 3-D occupancy mapping with Bayesian generalized kernel inference," *IEEE Transactions on Robotics*, Vol. 35, No. 4, 2019, pp. 953–966.

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

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

[108] Hahnel, D., Triebel, R., Burgard, W., and Thrun, S., "Map building with mobile robots in dynamic environments," Vol. 2, IEEE, 2003, pp. 1557–1563.

[109] Valentin, J. P., Sengupta, S., Warrell, J., Shahrokni, A., and Torr, P. H., "Mesh based semantic modelling for indoor and outdoor scenes," Portland, Oregon, 2013, pp. 2067–2074.

[110] McCormac, J., Handa, A., Davison, A., and Leutenegger, S., "Semanticfusion: Dense 3D semantic mapping with convolutional neural networks," 2017, pp. 4628–4635.

[111] Chen, X., Milioto, A., Palazzolo, E., Giguère, P., Behley, J., and Stachniss, C., "SuMa++: Efficient LiDAR-based Semantic SLAM," IEEE, 2019.

[112] Vega-Brown, W. R., Doniec, M., and Roy, N. G., "Nonparametric Bayesian inference on multivariate exponential families," *Advances in Neural Information Processing Systems*, 2014, pp. 2546–2554.

[113] Peretroukhin, V., Vega-Brown, W., Roy, N., and Kelly, J., "PROBE-GK: Predictive robust estimation using generalized kernels," IEEE, 2016, pp. 817–824.

[114] Richter, C., Vega-Brown, W., and Roy, N., "Bayesian learning for safe high-speed navigation in unknown environments," Springer, 2018, pp. 325–341.

[115] Shan, T., Wang, J., Englot, B., and Doherty, K., "Bayesian Generalized Kernel Inference for Terrain Traversability Mapping," 2018, pp. 829–838.

[116] Hähnel, D., *Mapping with mobile robots*, Ph.D. thesis, University of Freiburg, Freiburg im Breisgau, Germany, 2005.

[117] Melkumyan, A. and Ramos, F., "A Sparse Covariance Function for Exact Gaussian Process Inference in Large Datasets." *IJCAI*, Vol. 9, 2009, pp. 1936–1942.

[118] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y., "ROS: an open-source Robot Operating System," *ICRA workshop on open source software*, Vol. 3, 2009, p. 5.

[119] Rusu, R. B. and Cousins, S., "3D is here: Point cloud library (PCL)," IEEE, 2011.

[120] Geiger, A., Roser, M., and Urtasun, R., "Efficient large-scale stereo matching," Springer, 2010, pp. 25–38.

[121] Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D., "ORB-SLAM: A versatile and accurate monocular SLAM system," Vol. 31, No. 5, 2015, pp. 1147–1163.

[122] Yu, F. and Koltun, V., "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[123] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S., "SLIC superpixels compared to state-of-the-art superpixel methods," Vol. 34, No. 11, 2012, pp. 2274–2281.

[124] Behley, J. and Stachniss, C., "Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments," 2018.

[125] Milioto, A., Vizzo, I., Behley, J., and Stachniss, C., "RangeNet++: Fast and accurate LiDAR semantic segmentation," 2019.

[126] Carlevaris-Bianco, N., Ushani, A. K., and Eustice, R. M., "University of Michigan North Campus long-term vision and lidar dataset," Vol. 35, No. 9, 2016, pp. 1023–1035.

[127] Siam, M., Gamal, M., Abdel-Razek, M., Yogamani, S., and Jagersand, M., "RtSeg: Real-time semantic segmentation comparative study," IEEE, 2018, pp. 1603–1607.

[128] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L., "ImageNet: A large-scale hierarchical image database," IEEE, 2009, pp. 248–255.

[129] Hartley, R., Jadidi, M. G., Grizzle, J., and Eustice, R. M., "Contact-Aided Invariant Extended Kalman Filtering for Legged Robot State Estimation," June 2018.

[130] Thrun, S. et al., "Robotic mapping: A survey," 2002.

[131] Dubé, R., Cramariuc, A., Dugas, D., Sommer, H., Dymczyk, M., Nieto, J., Siegwart, R., and Cadena, C., "SegMap: Segment-based mapping and localization using data-driven descriptors," Vol. 39, No. 2-3, 2020, pp. 339–355.

[132] Kim, A. and Eustice, R. M., "Perception-driven navigation: Active visual SLAM for robotic area coverage," IEEE, 2013, pp. 3196–3203.

[133] Ghaffari Jadidi, M., Valls Miro, J., and Dissanayake, G., "Gaussian processes autonomous mapping and exploration for range-sensing mobile robots," Vol. 42, No. 2, 2018, pp. 273–290.

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

[135] Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., and Nieto, J., "Voxblox: Incremental 3D euclidean signed distance fields for on-board mav planning," IEEE, 2017, pp. 1366–1373.

[136] Bylow, E., Sturm, J., Kerl, C., Kahl, F., and Cremers, D., "Real-time camera tracking and 3D reconstruction using signed distance functions." Vol. 2, 2013, p. 2.

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

[138] Rosinol, A., Abate, M., Chang, Y., and Carlone, L., "Kimera: an open-source library for real-time metric-semantic localization and mapping," *arXiv preprint arXiv:1910.02490*, 2019.

[139] Hiller, M., Qiu, C., Particke, F., Hofmann, C., and Thielecke, J., "Learning topometric semantic maps from occupancy grids," IEEE, 2019, pp. 4190–4197.

[140] Zobeidi, E., Koppel, A., and Atanasov, N., "Dense Incremental Metric-Semantic Mapping via Sparse Gaussian Process Regression," IEEE, 2020, pp. 6180–6187.

[141] Gan, L., Zhang, R., Grizzle, J. W., Eustice, R. M., and Ghaffari, M., "Bayesian spatial kernel smoothing for scalable dense semantic mapping," *IEEE Robotics and Automation Letters*, Vol. 5, No. 2, 2020, pp. 790–797.

[142] Sünderhauf, N., Pham, T. T., Latif, Y., Milford, M., and Reid, I., "Meaningful maps with object-oriented semantic mapping," IEEE, 2017, pp. 5079–5085.

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

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

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

[146] Thrun, S., "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, Vol. 99, No. 1, 1998, pp. 21–71.

[147] Bosse, M., Newman, P., Leonard, J., Soika, M., Feiten, W., and Teller, S., "An atlas framework for scalable mapping," *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, Vol. 2, IEEE, 2003, pp. 1899–1906.

[148] Tomatis, N., Nourbakhsh, I., and Siegwart, R., "Hybrid simultaneous localization and map building: a natural integration of topological and metric," *Robotics and Autonomous systems*, Vol. 44, No. 1, 2003, pp. 3–14.

[149] Kuipers, B., "The spatial semantic hierarchy," *Artificial intelligence*, Vol. 119, No. 1-2, 2000, pp. 191–233.

[150] Galindo, C., Saffiotti, A., Coradeschi, S., Buschka, P., Fernandez-Madrigal, J.-A., and González, J., "Multi-hierarchical semantic maps for mobile robotics," *2005 IEEE/RSJ international conference on intelligent robots and systems*, IEEE, 2005, pp. 2278–2283.

[151] Zender, H., Mozos, O. M., Jensfelt, P., Kruijff, G.-J., and Burgard, W., "Conceptual spatial representations for indoor mobile robots," *Robotics and Autonomous Systems*, Vol. 56, No. 6, 2008, pp. 493–502.

[152] Pronobis, A. and Jensfelt, P., "Large-scale semantic mapping and reasoning with heterogeneous modalities," 2012, pp. 3515–3522.

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

[154] Armeni, I., He, Z.-Y., Gwak, J., Zamir, A. R., Fischer, M., Malik, J., and Savarese, S., "3d scene graph: A structure for unified semantics, 3d space, and camera," *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5664–5673.

[155] Rosinol, A., Gupta, A., Abate, M., Shi, J., and Carlone, L., "3D dynamic scene graphs: Actionable spatial perception with places, objects, and humans," *arXiv preprint arXiv:2002.06289*, 2020.

[156] Nieto, J., Guivant, J., and Nebot, E., "Denseslam: Simultaneous localization and dense mapping," Vol. 25, No. 8, 2006, pp. 711–744.

[157] Nordin, P. and Degerman, P., "Multi Layered Maps for Enhanced Environmental Perception," Tech. rep., SAE Technical Paper, 2011.

[158] Zaenker, T., Verdoja, F., and Kyrki, V., "Hypermap Mapping Framework and its Application to Autonomous Semantic Exploration," *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, IEEE, 2020, pp. 133–139.

[159] Wermelinger, M., Fankhauser, P., Diethelm, R., Krüsi, P., Siegwart, R., and Hutter, M., "Navigation planning for legged robots in challenging terrain," IEEE, 2016, pp. 1184–1189.

[160] Filitchkin, P. and Byl, K., "Feature-based terrain classification for littledog," IEEE, 2012, pp. 1387–1392.

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

[162] Bekhti, M. A. and Kobayashi, Y., "Regressed Terrain Traversability Cost for Autonomous Navigation Based on Image Textures," *Applied Sciences*, Vol. 10, No. 4, 2020, pp. 1195.

[163] Brooks, C. A. and Iagnemma, K., "Self-supervised terrain classification for planetary surface exploration rovers," *Journal of Field Robotics*, Vol. 29, No. 3, 2012, pp. 445–468.

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

[165] Zürn, J., Burgard, W., and Valada, A., "Self-supervised visual terrain¨ classification from unsupervised acoustic feature learning," 2020.

[166] Dahlkamp, H., Kaehler, A., Stavens, D., Thrun, S., and Bradski, G. R., "Self-supervised monocular road detection in desert terrain." *Robotics: science and systems*, Vol. 38, Philadelphia, 2006.

[167] Hadsell, R., Sermanet, P., Ben, J., Erkan, A., Scoffier, M., Kavukcuoglu, K., Muller, U., and LeCun, Y., "Learning long-range vision for autonomous off-road driving," *Journal of Field Robotics*, Vol. 26, No. 2, 2009, pp. 120–144.

[168] Barnes, D., Maddern, W., and Posner, I., "Find your own way: Weakly-supervised segmentation of path proposals for urban autonomy," IEEE, 2017, pp. 203–210.

[169] Broome, M., Gadd, M., De Martini, D., and Newman, P., "On the Road: Route Proposal from Radar Self-Supervised by Fuzzy LiDAR Traversability," *AI*, Vol. 1, No. 4, 2020, pp. 558–585.

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

[171] Caruana, R., "Multitask learning," *Machine learning*, Vol. 28, No. 1, 1997, pp. 41–75.

[172] Vandenhende, S., Georgoulis, S., Van Gansbeke, W., Proesmans, M., Dai, D., and Van Gool, L., "Multi-Task Learning for Dense Prediction Tasks: A Survey," *arXiv preprint arXiv:2004.13379*, 2020.

[173] Crawshaw, M., "Multi-Task Learning with Deep Neural Networks: A Survey," *arXiv preprint arXiv:2009.09796*, 2020.

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

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

[176] Misra, I., Shrivastava, A., Gupta, A., and Hebert, M., "Cross-stitch networks for multi-task learning," 2016, pp. 3994–4003.

[177] Gao, Y., Ma, J., Zhao, M., Liu, W., and Yuille, A. L., "Nddr-cnn: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction," 2019, pp. 3205–3214.

[178] Liu, S., Johns, E., and Davison, A. J., "End-to-end multi-task learning with attention," 2019, pp. 1871–1880.

[179] Maninis, K.-K., Radosavovic, I., and Kokkinos, I., "Attentive single-tasking of multiple tasks," 2019, pp. 1851–1860.

[180] Lu, Y., Kumar, A., Zhai, S., Cheng, Y., Javidi, T., and Feris, R., "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification," 2017, pp. 5334–5343.

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

[182] Kendall, A., Gal, Y., and Cipolla, R., "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," 2018, pp. 7482–7491.

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

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

[185] Standley, T., Zamir, A. R., Chen, D., Guibas, L., Malik, J., and Savarese, S., "Which Tasks Should Be Learned Together in Multi-task Learning?" *arXiv preprint arXiv:1905.07553*, 2019.

[186] Hu, J., Shen, L., and Sun, G., "Squeeze-and-excitation networks," 2018, pp. 7132–7141.

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

[188] Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H., "Encoder-decoder with atrous separable convolution for semantic image segmentation," 2018, pp. 801–818.

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

[190] Zhu, Y., Sapra, K., Reda, F. A., Shih, K. J., Newsam, S., Tao, A., and Catanzaro, B., "Improving semantic segmentation via video propagation and label relaxation," 2019, pp. 8856–8865.

[191] Geiger, A., Lenz, P., and Urtasun, R., "Are we ready for autonomous driving? the kitti vision benchmark suite," IEEE, 2012, pp. 3354–3361.

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

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

[194] Hartley, R., Ghaffari, M., Eustice, R. M., and Grizzle, J. W., "Contact-aided invariant extended Kalman filtering for robot state estimation," Vol. 39, No. 4, 2020, pp. 402–430.

[195] Bishop, C. M., *Pattern recognition and machine learning*, springer, 2006.

[196] Nguyen, A., Nguyen, N., Tran, K., Tjiputra, E., and Tran, Q. D., "Autonomous navigation in complex environments with deep multimodal fusion network," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 5824–5830.

[197] Pan, Y., Cheng, C.-A., Saigol, K., Lee, K., Yan, X., Theodorou, E. A., and Boots, B., "Imitation learning for agile autonomous driving," *The International Journal of Robotics Research*, Vol. 39, No. 2-3, 2020, pp. 286–302.

[198] Kahn, G., Abbeel, P., and Levine, S., "Badgr: An autonomous self-supervised learning-based navigation system," *IEEE Robotics and Automation Letters*, Vol. 6, No. 2, 2021, pp. 1312–1319.

[199] Fan, D. D., Otsu, K., Kubo, Y., Dixit, A., Burdick, J., and Agha-Mohammadi, A.-A., "STEP: Stochastic Traversability Evaluation and Planning for Risk-Aware Off-road Navigation," *Robotics: Science and Systems*, RSS Foundation, 2021, pp. 1–21.

[200] Dashora, N., Shin, D., Shah, D., Leopold, H., Fan, D., Agha-Mohammadi, A., Rhinehart, N., and Levine, S., "Hybrid Imitative Planning with Geometric and Predictive Costs in Off-road Environments," *arXiv preprint arXiv:2111.10948*, 2021.

[201] Guan, T., He, Z., Manocha, D., and Zhang, L., "TTM: Terrain traversability mapping for autonomous excavator navigation in unstructured environments," *arXiv preprint arXiv:2109.06250*, 2021.

[202] Sünderhauf, N., Brock, O., Scheirer, W., Hadsell, R., Fox, D., Leitner, J., Upcroft, B., Abbeel, P., Burgard, W., Milford, M., et al., "The limits and potentials of deep learning for robotics," *The International Journal of Robotics Research*, Vol. 37, No. 4-5, 2018, pp. 405–420.

[203] Guastella, D. C. and Muscato, G., "Learning-based methods of perception and navigation for ground vehicles in unstructured environments: a review," *Sensors*, Vol. 21, No. 1, 2021, pp. 73.

[204] Maturana, D., Chou, P.-W., Uenoyama, M., and Scherer, S., "Real-time semantic mapping for autonomous off-road navigation," *Field and Service Robotics*, Springer, 2018, pp. 335–350.

[205] Ahtiainen, J., Stoyanov, T., and Saarinen, J., "Normal Distributions Transform Traversability Maps: LIDAR-Only Approach for Traversability Mapping in Outdoor Environments," *Journal of Field Robotics*, Vol. 34, No. 3, 2017, pp. 600–621.

[206] Chavez-Garcia, R. O., Guzzi, J., Gambardella, L. M., and Giusti, A., "Learning ground traversability from simulations," *IEEE Robotics and Automation letters*, Vol. 3, No. 3, 2018, pp. 1695–1702.

[207] Gan, L., Kim, Y., Grizzle, J. W., Walls, J. M., Kim, A., Eustice, R. M., and Ghaffari, M., "Multi-Task Learning for Scalable and Dense Multi-Layer Bayesian Map Inference," *arXiv preprint arXiv:2106.14986*, 2021.

[208] Faigl, J. and Prágr, M., "On unsupervised learning of traversal cost and terrain types identification using self-organizing maps," *International Conference on Artificial Neural Networks*, Springer, 2019, pp. 654–668.

[209] Wulfmeier, M., Rao, D., Wang, D. Z., Ondruska, P., and Posner, I., "Large-scale cost function learning for path planning using deep inverse reinforcement learning," *The International Journal of Robotics Research*, Vol. 36, No. 10, 2017, pp. 1073–1087.

[210] Zhu, Z., Li, N., Sun, R., Xu, D., and Zhao, H., "Off-road autonomous vehicles traversability analysis and trajectory planning based on deep inverse reinforcement learning," *2020 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2020, pp. 971–977.

[211] Jung, C. and Shim, D. H., "Incorporating Multi-Context Into the Traversability Map for Urban Autonomous Driving Using Deep Inverse Reinforcement Learning," *IEEE Robotics and Automation Letters*, Vol. 6, No. 2, 2021, pp. 1662–1669.

[212] Osa, T., Pajarinen, J., Neumann, G., Bagnell, J. A., Abbeel, P., and Peters, J., "An algorithmic perspective on imitation learning," *arXiv preprint arXiv:1811.06711*, 2018.

[213] Deo, N. and Trivedi, M. M., "Trajectory forecasts in unknown environments conditioned on grid-based plans," *arXiv preprint arXiv:2001.00735*, 2020.

[214] Wang, T., Dhiman, V., and Atanasov, N., "Inverse reinforcement learning for autonomous navigation via differentiable semantic mapping and planning," *arXiv preprint arXiv:2101.00186*, 2021.

[215] Brown, D., Goo, W., Nagarajan, P., and Niekum, S., "Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations," *International conference on machine learning*, PMLR, 2019, pp. 783–792.

[216] Arora, S. and Doshi, P., "A survey of inverse reinforcement learning: Challenges, methods and progress," *Artificial Intelligence*, 2021, pp. 103500.

[217] Buchanan, R., Bednarek, J., Camurri, M., Nowicki, M. R., Walas, K., and Fallon, M., "Navigating by touch: haptic Monte Carlo localization via geometric sensing and terrain classification," *Autonomous Robots*, Vol. 45, No. 6, 2021, pp. 843–857.

[218] Silver, D., Bagnell, J., and Stentz, A., "High performance outdoor navigation from overhead data using imitation learning," *Robotics: Science and Systems IV, Zurich, Switzerland*, Vol. 1, 2008.

[219] Silver, D., Bagnell, J. A., and Stentz, A., "Learning from demonstration for autonomous navigation in complex unstructured terrain," *The International Journal of Robotics Research*, Vol. 29, No. 12, 2010, pp. 1565–1592.

[220] Ziebart, B. D., Maas, A. L., Bagnell, J. A., Dey, A. K., et al., "Maximum entropy inverse reinforcement learning." *Aaai*, Vol. 8, Chicago, IL, USA, 2008, pp. 1433–1438.

[221] Wulfmeier, M., Ondruska, P., and Posner, I., "Maximum entropy deep inverse reinforcement learning," *arXiv preprint arXiv:1507.04888*, 2015.

[222] Lin, T.-Y., Zhang, R., Yu, J., and Ghaffari, M., "Legged Robot State Estimation using Invariant Kalman Filtering and Learned Contact Events," *Conference on Robot Learning*, 2021.

[223] Ronneberger, O., Fischer, P., and Brox, T., "U-net: Convolutional networks for biomedical image segmentation," *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[224] Brown, D. S., Goo, W., and Niekum, S., "Better-than-demonstrator imitation learning via automatically-ranked demonstrations," *Conference on robot learning*, PMLR, 2020, pp. 330–359.

[225] Walas, K., Kanoulas, D., and Kryczka, P., "Terrain classification and locomotion parameters adaptation for humanoid robots using force/torque sensing," *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, IEEE, 2016, pp. 133–140.

[226] Mur-Artal, R. and Tardós, J. D., "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, Vol. 33, No. 5, 2017, pp. 1255–1262.

[227] Kitani, K. M., Ziebart, B. D., Bagnell, J. A., and Hebert, M., "Activity forecasting," *European conference on computer vision*, Springer, 2012, pp. 201–214.

[228] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y., "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, Vol. 32, No. 1, 2020, pp. 4–24.

[229] Paton, M., Strub, M. P., Brown, T., Greene, R. J., Lizewski, J., Patel, V., Gammell, J. D., and Nesnas, I. A., "Navigation on the Line: Traversability Analysis and Path Planning for Extreme-Terrain Rappelling Rovers," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 7034–7041.

[230] Gleave, A. and Habryka, O., "Multi-task maximum entropy inverse reinforcement learning," *arXiv preprint arXiv:1805.08882*, 2018.

[231] Xu, K., Ratner, E., Dragan, A., Levine, S., and Finn, C., "Learning a prior over intent via meta-inverse reinforcement learning," *International Conference on Machine Learning*, PMLR, 2019, pp. 6952–6962.

[232] Zou, D. and Tan, P., "Coslam: Collaborative visual slam in dynamic environments," *IEEE transactions on pattern analysis and machine intelligence*, Vol. 35, No. 2, 2012, pp. 354–366.

[233] Matsuka, K., Feldman, A. O., Lupu, E. S., Chung, S.-J., and Hadaegh, F. Y., "Decentralized formation pose estimation for spacecraft swarms," *Advances in Space Research*, Vol. 67, No. 11, 2021, pp. 3527–3545.

[234] Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al., "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[235] Tai, L., Paolo, G., and Liu, M., "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," IEEE, 2017, pp. 31–36.