Terrain-aware autonomous ground navigation in unstructured environments informed by human demonstrations: a dissertation in Electrical Engineering.

Ellis, Christian C.

University of Massachusetts Dartmouth
Department of Electrical and Computer Engineering

Terrain-aware Autonomous Ground Navigation in Unstructured Environments Informed by Human Demonstrations

A Dissertation in
Electrical Engineering
by
Christian C. Ellis

Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

January 2024

We approve the dissertation of Christian C. Ellis

Date of Signature

Lance Fiondella
Associate Professor, Department of Electrical and Computer Engineering
Dissertation Advisor

Hong Liu
Commonwealth Professor, Department of Electrical and Computer Engineering
Dissertation Committee

Jiawei Yuan
Associate Professor, Department of Computer and Information Science
Dissertation Committee

Maggie Wigness
Senior Computer Scientist, Army Research Laboratory, Adelphi MD
Dissertation Committee

Craig Lennon
Mathematical Statistician, Army Research Laboratory, Adelphi MD
Dissertation Committee

Liudong Xing
Graduate Program Director, Electrical Engineering

Dayalan Kasilingam
Chairperson, Department of Electrical and Computer Engineering

Jean VanderGheynst
Dean, College of Engineering

Tesfay Meressi
Associate Provost for Graduate Studies

Abstract

Terrain-aware Autonomous Ground Navigation in Unstructured Environments Informed by Human Demonstrations

by Christian C. Ellis

Mobile robots equipped with the capability to perform autonomous waypoint navigation can replace humans for applications such as humanitarian assistance, nuclear cleanup, reconnaissance, and transportation. In such tasks, the robot must be able to perform complex navigation behaviors, including the ability to navigate accurately and reliably over unstructured terrain while responding to unseen situations, similar to how a human would. To successfully complete missions in unstructured natural environments, agents must (i) respond to previously unseen environmental features, and (ii) be able to develop an accurate perception representation of the current environment. This dissertation provides a solution for each of the two aforementioned sub-problems using human teleoperated demonstrations. Learning from demonstration has been shown to be advantageous for navigation tasks as it allows for machine learning non-experts to quickly provide information needed for robots to learn complex traversal behaviors.

First, I present a Bayesian methodology which quantifies uncertainty over the weights of a linear reward function given a dataset of minimal human demonstrations to operate safely in dynamic environments. This uncertainty is quantified and incorporated into a risk averse set of weights used to generate cost maps for trajectory planning. This results in a robot which follows risk averse trajectories by expressing uncertainty in the designed reward function while considering all possible reward functions that satisfy the training environment and human demonstrations.

Second, I present a methodology to obtain a perception subsystem for an autonomous

ground vehicle by mapping a set of non-semantic classes from an unsupervised algorithm to a set of high-level semantic classes using only unlabeled images and human demonstrations. This enables the robot to obtain a unique fine-tuned set of abstract and semantic classes which represent the current operational environment while avoiding time consuming and expensive ground truth data labeling.

Lastly, I outline a framework which combines the two sub-problems, resulting in a methodology which takes a data stream of unlabeled images as input, and provides as output- (i) a refined perception representation, and (ii) a risk-averse reward function for unstructured terrain aware navigation. This formulation enables engineers to drop a robot in an environment it has never been in before, learn an appropriate perception representation, and learn the costs associated with the environmental terrains, using only unlabeled data and a minimal set of human demonstrations.

Acknowledgements

Completion of this dissertation would not have been possible if it were not for the mentorship provided to me by several colleagues.

First, to Professor Lance Fiondella- who was always willing to help guide me and provide rewarding opportunities throughout my professional career. He saw something in me starting in my freshman year of undergraduate study, and was resilient enough to continually push I go graduate school and obtain a doctoral degree.

Second, to Dr. Craig Lennon- who gave me the final motivation I needed to pursue graduate school. I was so excited to meet someone who was passionate about autonomous system safety as I was.

Third, to Dr. Maggie Wigness- whose unprecedented technical ability, novel research ideas, and top tier mentorship made completion of this dissertation possible. Whenever I was at a wall technically, she was able to help provide potential solutions. The way she encouraged me to think about and frame research problems is one of the best skills I have obtained in the past years.

Fourth, to Professor Hong Liu- whose personality and positive attitude about university study and research is inspirational. I hope to share the same attitude with my future mentees.

Fifth, to the many fellow graduate students I have gotten close with during the past few years both in and outside of the university- I always left our late night research conversations feeling like we were working on something meaningful.

And lastly, to potential future graduate students- if you have a natural desire and love for learning and are considering going to graduate school, let this sentence be the final motivation you need. It is a journey, but worth it.

This research was sponsored by the Army Research Laboratory accomplished under cooperative agreement number(s) ARL W911NF-19-2-0285 and W911NF23-2-0211. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies; either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposed notwithstanding any copyright notation herein.

Contents

List of Figures

List of Tables

Chapter 1 Problem Statement

1.1 Overview

Mobile robots equipped with the capability to perform autonomous waypoint navigation can replace or assist humans for applications such as humanitarian assistance, nuclear cleanup, reconnaissance, and transportation. In such tasks, the robot must be able to perform complex navigation behaviors, including the ability to navigate accurately and reliably over unstructured terrain while responding to unseen situations, similar to how a human would.

To communicate desired robot behavior in both positive and negative environmental scenarios such as staying on the road while avoiding the mud, roboticists assign rewards (or inversely, costs) which direct the robot toward states leading to the achievement of goals while avoiding negative, potentially dangerous states [71]. In the context of unstructured autonomous navigation, environmental features such as terrain and obstacle information are used to define the set of possible states. States encode information about the environmental features present for a particular area of physical space in time. The set of features which describe states form what is known as a reward basis. A reward basis scopes what is and is not possible for a robot to learn; if a robot's features do not capture information about the underlying terrain, it will not be able to reason over terrain. Therefore, to reach specified goals, states must encode information accurately using representative features which describe the current environment. For robots that need to quickly transition between varying unstructured environments, defining rewards prior to understanding all future features is often unachievable as either (i) the robot is unable to perceive the new feature, and/or (ii) the numerical reward for each feature is unknown.

Moreover, to implement complex behaviors beyond obstacle avoidance, many current approaches employ machine learning methods requiring large amounts of labeled data. While

simulations can quickly generate large amounts of labeled data, the same cannot be said for real world environments, limiting adoption of mobile robots to complete the aforementioned applications. Furthermore, solutions are often brittle, exhibiting poor performance when operating outside of environments where they were designed or trained. Therefore, there is a need for methods which can learn navigation behaviors from limited data while being able to adapt to novel scenarios.

In such scenarios, it is more effective for a human supervisor to provide examples of desired behavior than for a engineer to explicitly define it. Explicit encoding often leads to a misalignment of what the human intended the robot's goals to be, and the actual behavior exhibited [68]. This results in negative side effects [4] such as the robot taking a dangerous trajectory to reach a goal or ruthlessly following directions without considerations of how their actions affect other actors in the environment. Contrarily, inverse reinforcement learning (IRL) [7] seeks to learn a reward function given a limited dataset of demonstrations, which eliminates the need for an expert to hand code these rewards. However, the performance of such approaches is limited to the states only visited in the demonstrations, and their underlying feature set. Many IRL techniques make the unrealistic assumption that reward information obtained during training accurately characterizes future operational environments. Certain states and the features that describe states may have never been encountered, leading to uncertainty in their reward [33]. This presents a challenge for autonomous navigation since it is desirable for a robotic system to be robust to operation in unknown, possibly adversarial, environments which are likely to contain unforeseen conditions.

Therefore, this dissertation focuses on a ground robot's ability to incorporate human demonstrations as a supervisory signal to solve each respective preceding problem by (i) obtaining a semantic perception model capable of classifying terrains present in the current environment given a sequence of unlabeled images and (ii) using Bayesian inverse reinforce-

ment learning to learn rewards associated with the terrains identified to build cost-maps online for autonomous waypoint traversal.

1.2 Dissertation Scope

Recent approaches in image-based supervised perception algorithms have enabled autonomous robots to semantically understand the current operational environment. However, such approaches are limited to a set of known semantic features and require expensive and time-consuming labeled training data. Even in scenarios where the set of all semantic features to complete a mission is known, the relative rewards and preferences among them which result in the optimal navigation trajectory is unknown. If the predefined reward function fails to capture all aspects of the agent's operational environment, undesired behavior will occur when the agent fails to optimize the true reward function and therefore express indifference to potentially dangerous, unseen scenarios. More generally, current approaches to autonomous navigation are limited to environments representative of the robot's previous experience and brittle to anything outside it.

If the robot is unable to recognize new environmental features, it will be unable to respond to them. Consider a robot which has been optimized in a wooded, off-road environment described by a representative reward function. Once the robot is placed in a different environment such as an urban area, the original reward function will fail to accurately describe the desired behavior due to the presence of new environmental features, leading to potentially dangerous behavior. Therefore, in order to complete missions in unstructured natural environments, agents must succeed at the following tasks:

1. Responding to previously unseen environmental features similar to how a human supervisor would

2. Obtain an accurate perception representation of the current environment

This dissertation provides a solution for each of the two aforementioned tasks using human teleoperated demonstrations. Learning from demonstration has been shown to be advantageous for navigation tasks as it allows for machine learning non-experts to quickly provide information needed for robots to learn complex traversal behaviors.

With respect to task 1, I present a Bayesian technique which quantifies uncertainty over the weights of a linear reward function given a dataset of minimal human demonstrations to operate safely in dynamic environments. This uncertainty is quantified and incorporated into a risk averse set of weights used to generate cost maps for trajectory planning. This results in a robot which follows risk averse trajectories by expressing uncertainty in the designed reward function by considering all possible parameterizations of a reward function that satisfy the training environment and human demonstrations.

Regarding task 2, I first present a technique to obtain a perception subsystem for an autonomous ground vehicle by mapping a set of non-semantic feature labels from an unsupervised algorithm to a set of high-level semantic labels. This enables the robot to obtain a unique fine-tuned set of abstract labels which represent the current operational environment while avoiding time-consuming and expensive ground truth data labeling. Then, the resulting label set is used as the reward basis for inverse reinforcement learning.

Lastly, I outline an algorithm which combines the two tasks, resulting in a technique which takes a data stream of unlabeled images and a set of human demonstrations as input, and learns a reward function for unstructured terrain-aware navigation as output.

1.3 Research Timeline

Work completed during this dissertation has focused on three lines of related research, in order from most effort to least; (i) Terrain-aware autonomous ground navigation in unstructured environments informed by human demonstrations (this dissertation*), (ii) Formal

*Supported by ARL W911NF-19-2-0285.

verification of autonomous ground vehicles learning from human demonstration[†], and (iii) Review of assurance, test, and evaluation techniques for autonomous ground systems.

(i) Initial work by Ellis et al [25] showed that the uncertainty over terrain features traversed during demonstrations can be quantified through the normalized Shannon entropy of a feature's corresponding marginal distribution. This enabled an autonomous ground vehicle to avoid terrains which were never seen during training. Recently submitted work [24], addresses the limitations of quickly training robot perception systems by providing a methodology to learn environmental features using only unlabeled data and human demonstrations. This capability allows non-technical users to quickly train new perception models which are tailored to the current operational environment. Moreover, this methodology refines the over-segmentation found in an initial unsupervised segmentation model by discarding poor labels while simultaneously merging labels representing the same semantic concept. Lastly, the results from [24] are used to form a reward basis that utilizes [25].

(ii) Recent work [20] with UT Austin collaborators resolved information asymmetry between a demonstrator who has full knowledge of the underlying environment and a learner with limited information by modeling limited sensor range. Moreover, side information about the environment provided by the human and given to the agent are encoded as temporal logic constraints. Experiments performed in ARL's unity simulator show that a learner finds policies that achieve more cumulative reward when explicitly considering limited sensor observations and utilizing side information. In comparison to [25], this work seeks to obtain safe systems by explicitly modeling unsafe states rather than assuming certain environmental features are unsafe.

(iii) More broadly, [23] informs the autonomous systems community of the status, challenges, and remaining work regarding assurance of learning enabled autonomous systems.

[†]Supported by ARL W911NF-20-2-0132.

We reviewed state of the art technical approaches to assurance and identified where they fit in the systems engineering process. This work views autonomous system safety from a broader, more philosophical lens- outlining the key issues that arise when attempting to develop safe autonomous systems, and the current approaches to resolving such issues.

Chapter 2 Background Summary

2.1 Autonomy Stack

2.1.1 Overview

As previously mentioned the methods presented in this dissertation are intended for mobile wheeled ground robots operating in unstructured environments. Unstructured refers to environments which are off-road in nature, in contrast to clearly defined and marked roads often traversed by passenger vehicles. Unstructured environments often contain a combination of unmarked asphalt or concrete roads; loosely defined paths made of gravel, dirt, or grass; and obstacles throughout such as large rocks, bushes, trees, and fallen trees. Due to their unpredictable nature, resulting autonomous navigation systems often differ greatly than the ones found today on passenger vehicles. Specifically, the sensors used, localization and mapping techniques, obstacle detection subsystems, and planning techniques must accommodate the unpredictable, and complex nature of such environments. For example, an autonomous system operating only on highways in the USA can attempt to plan a set amount of time ahead with structured cues to reason and act, such as by following and staying within the dotted lines marking lanes on a highway. However, in unstructured environments the existence of dotted lines marking lanes cannot be assumed.

To accommodate such environments the methods presented are deployed on rugged robotic systems capable of autonomously traversing unstructured terrain. Specifically, the methods presented are integrated into an existing autonomous software stack explicitly designed for the same purpose. Resulting methods developed in subsequent chapters target additions to particular subsystems of this software stack to increase overall performance and capability to complete pre-defined missions. The autonomous software stack is capable of running the same exact code on both real world mobile field robotics, and in a 3D graphics

engine which simulates unstructured environments. The primary difference between simulated and real world results is in quality of perceived inputs from sensors. Simulated results often have perfect sensor readings, while in the real world they may be degraded due to intrinsic issues with the sensor such as calibration, or external factors such as mud blocking a sensor.

2.1.2 Example Levels of Autonomous Capability

As previously mentioned, the methods presented in this dissertation are integrated and deployed within an existing software stack developed by the U.S Army Research Laboratory, referred to as the ARL autonomy stack. The primary goal of this software stack is to provide autonomous navigation and subsequent intelligence subsystems to enable navigation tasks such as exploration, mapping, perception, and reasoning. As background, in this subsection I describe the different levels of capability one often likes to achieve when performing autonomous waypoint navigation. Please note that the levels of capability identified are brief descriptions of common use cases for mobile wheeled ground robotics aimed at informing the reader, and that there exists many other levels of capability as determined by the missions one would like to perform.

At the lowest capability, a robot finds the shortest path to a goal location while avoiding obstacles in the robots global frame, enabled by a simultaneous localization and mapping (SLAM) system. Moving up a level of capability, a robot finds the best obstacle free path to a goal location. This capability is possible by adding object detection subsystems enabled by the robot's camera and/or laser radar (LIDAR) sensors. In the final level of capability, the robot keeps track of a semantic map of the world, the terrain, obstacles, and other features of interest. The resulting semantic map(s) act as input to guide the robot to the specified waypoint, while following certain mission constraints, such as staying on the road when it is available, avoiding water, avoiding certain dangerous areas, certain elevations, etc. With

respect to the methods presented in this dissertation, they target this level. That is, the use of semantic information, alongside traditional SLAM to inform autonomous waypoint navigation.

2.1.3 Robot Operating System (ROS)

Aforementioned capabilities are implemented using Robot Operating System (ROS) [65]. Specifically, as of writing this dissertation, the system uses ROS 1.0 version Melodic. ROS is a common choice in the academic community due to its open-source nature containing several packages implementing common robotics subsystems and problems, widespread adoption, and active community. Breifly, ROS is an open-source software framework and middle-ware layer primarily written in C++ and Python3. Core functionality packaged within ROS includes- message passing between multiple processes (referred to as *ROS nodes*, or simply *nodes*) using TCP/IP and direct process to process communication, heterogeneous computer clustering (many nodes running on multiple connected computers), low-level hardware control and abstraction, package management, and build tools. Each major ROS release targets a particular Ubuntu (UNIX) distribution, and as a result, ROS 1.0 runs on top of the Ubuntu operating system.

2.2 Preliminaries

This section explains preliminary material and reviews related fields in which this dissertation's scope and problem formulation occupy. First, preliminary concepts required to understand the dissertation including machine learning (Sec. 2.2.1), and Markov decision processes (Sec. 2.2.1) are explained. Then, a literature review of the following fields is provided; learning from demonstration (LfD) (Sec. 2.3.1), Bayesian inverse reinforcement learning (B-IRL) (Sec. 2.3.2), safe autonomy (Sec. 2.3.3), semantic segmentation (Sec. 2.4.1), unsupervised learning system adaptation (Sec. 2.4.2), and unsupervised visual representation learning (Sec. 2.4.3). Fields are grouped together by their relation to each respective sub-problem identified in the last sentence of the abstract, namely, (i) learning a perception model to identify terrains present in the current environment alongside unlabeled images and (ii) using Bayesian inverse reinforcement learning (IRL) to learn the traversal rewards associated with the terrains identified to build cost-maps online for autonomous waypoint traversal.

2.2.1 Machine Learning

In machine learning (ML), tasks are completed by training a model from data to perform function approximation using a combination of mathematical optimization and statistical techniques [12]. This results in computer programs which are able to complete a task without constructing a set of exact solution instructions ahead of time. There are three main forms of learning, including supervised, unsupervised, and reinforcement learning. In supervised learning, each training sample from the dataset is associated with a set of features and a corresponding label to train a model. For example, a neural network can be trained on a dataset of images containing handwritten digits, where each sample's corresponding label is

0 through 9. In unsupervised learning, each training sample is only represented by a set of extracted features, which are subsequently used to identify the underlying feature patterns throughout the dataset. For example, clustering techniques divide a dataset into k distinct groups, where all data points in a group are similar with respect to some distance measure. Finally, in reinforcement learning, an autonomous agent learns the optimal way to act over time via interaction with the environment, such as an autonomous robot learning how to move its actuators and joints without hitting obstacles.

2.2.2 Markov Decision Process

An autonomous robot navigating in an environment is modeled using a Markov decision process (MDP) [64], M, represented by the following tuple

$$M = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle \tag{2.1}$$

where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, T is the state transition distribution over the next state given the present state and action represented by $T(s_{t+1}|s_t, a)$, R is the reward function representing the numerical reward received by taking action $a \in A$ in state $s \in S$ represented by $R(s, a) : \mathcal{S} \times \mathcal{A} \rightarrow R$, and $\gamma \in [0, 1)$ is the discount factor representing the weight on future unseen rewards. The solution to an MDP is a policy $\pi(s) : \mathcal{S} \rightarrow \mathcal{A}$, determining the action a a robot will take in state s. The optimal policy for an MDP (π^*) maximizes the expected cumulative reward.

2.3 Teaching Robots to Learn From Humans

This section describes how human demonstrations can be used to learn a reward model for a MDP.

2.3.1 Learning from Demonstration

In reward learning, the goal is to obtain a unique reward function from human demonstration that maps human provided trajectories through a state space to scalar rewards [37]. More generally, autonomous agents learning solely from demonstrations to replicate behavior is called imitation learning [61]. There are two distinct sub-fields within imitation learning, behavioral cloning [10, 76] and IRL [57]. A robot learns a policy directly in behavioral cloning, whereas a robot implementing IRL learns a reward function, which may then be used to obtain a policy. Because of this extra step, it is typically more complex to obtain a solution with IRL methods. The chosen approach is dependent upon the following - *what is the most parsimonious model description of the desired behavior, reward or policy?* [61]. Behavioral cloning is sufficient for problems that can be solved by modeling a distribution over demonstrated trajectories, where feature extraction is difficult and environments are essentially static [58]. Problems which involve inferring expert intent or operating in a large space of potential environments requiring generalization, such as navigation, may be solved with IRL methods [86, 81].

IRL is an ill-posed problem [57] because many possible reward functions can characterize robot behavior. Attempts to identify effective solutions have led to several competing methodologies. For a comprehensive survey, we refer the reader to the following references [61, 8, 92, 6]. Nevertheless, previous work shows that learning from demonstration scales to real robotic systems for both linear [67, 81], and non-linear problems [87]. Similarly, deep learning approaches to IRL have been successful in the Atari [34] and MuJoCo envi-

ronments [13]. Since this dissertation seeks to learn a representative reward function from minimal demonstrations, a linear model is chosen instead of a non-linear deep learning approach.

2.3.2 Bayesian Inverse Reinforcement Learning

By taking the Bayesian viewpoint, a robot can quantifiably establish a belief over multiple reward functions and evaluate their uncertainty. In contrast to maximum likelihood approaches, Bayesian IRL methods provide an approach to reason about many different reward functions. Each reward function is assigned a point probability from a posterior distribution. Pre-computing uncertainty also potentially allows a systems engineer to intervene and directly specify or correct reward values relative to those learned from demonstrations (e.g. to specify that grass is as good as dirt). Ramachandran and Amir [66] proposed Bayesian IRL as a methodology to build a posterior density over reward functions given a dataset of demonstrations. Choi and Kim [17] developed a framework subsuming previous IRL methods and showed that the maximum a posteriori estimator is a better estimator than the posterior mean proposed in [66]. This dissertation also takes the Bayesian viewpoint to enable the robot to quantifiably establish a belief over multiple reward functions and evaluate their uncertainty. In contrast to Refs. [66, 17], this dissertation's methodology differs by explicitly considering safety when learning from demonstration.

2.3.3 Safe Autonomy

The reward function obtained from demonstrations in a training environment may not be well suited for guiding robot behavior in a new operational environment, leading to negative side effects[4, 45]. Safe imitation learning seeks to obtain behavior that avoids negative side effects [91, 52, 14]. Although these approaches explicitly consider safety, they do not directly address changes in the environment (distributional shift). Lutjens et al. [49] obtained model

uncertainty estimates to avoid novel obstacles from perception systems in the reinforcement learning framework, but did not consider the imitation learning scenario. Janson et al. [36] sought safe motion planning in unknown environments, but only considered obstacle avoidance and not a preference over different terrains. Hadfield-Menell et al. [33] considered safety to distributional shift, but do not explicitly consider learning from demonstration and their weight selection technique changes at runtime. Menda et al. [52], use an ensemble of neural networks to quantify the uncertainty of a non-expert policy obtained from an imitation learning method. This dissertation differs in two ways, (i) we take a linear approach which takes a set of semantic features as input and outputs a single reward function rather than a policy (ii) we explicitly consider the agent to navigate in operational environments separate than the training environment. Our intended use case is one in which training occurs from a dataset of demonstrations in an environment that is likely to be different than the operational environment. Consequently, this dissertation presents a method to obtain a single reward model before operating in a new environment, both for traceability and to save time and computational resources, when operating online in the deployment environment.

2.4 Robotic Perception

In order to assign rewards to environmental features, robots must first be able to develop feature representations which describe what they are currently perceiving from their sensors. The initial applications of scene understanding came predominantly from advances in image-based semantic segmentation. Recent achievements have been aided by sub-fields including unsupervised learning system adaptation, and unsupervised visual representation learning. A concise review of each follows.

2.4.1 Semantic Segmentation

Semantic segmentation [59, 90] is the task of identifying boundaries (segments) while simultaneously classifying every pixel in an image to a discrete label. Formally, a function $f : X \rightarrow Y$ maps the space of images, X, to a label space, Y, which consists of n semantic classes. This function directly maps every pixel in an image to a single label in a set of labels. State-of-the-art semantic segmentation approaches [9, 15, 29] are supervised in nature and make use of deep neural network architectures. In practice, these advances have enabled vision-based autonomous navigation, giving rise to fine-grained trajectory generation that can reason about specific terrain or object classes [54, 83] and the uncertainty associated with them [25, 74].

Although supervised semantic segmentation has produced high quality visual perception output, approaches are limited by (i) the need for large datasets of images and their associated ground truth labels, (ii) their inability to generalize well across domains (e.g., from structured to unstructured environments [39, 82]), and (iii) their discrete class output that does not allow for novel class discovery or open-world operation. Resolving such limitations is a critical need for off-road autonomous navigation as limited a priori knowledge of the environment is available. To address the shortcomings, unsupervised semantic segmentation [38, 16] has demonstrated the ability to learn without labels.

Stream-based unsupervised segmentation [89, 80] has focused on using temporal information to segment video streams through time, making them ideal for online operation. This enables novel concept discovery as encountered over time. Because these algorithms operate in an online fashion without batch data for learning, they tend to be more conservative to ensure segmented regions respect true class boundaries in the data stream. Often this results in unsupervised segmentation output that is over-segmented, such that many segments map to the same high-level ground truth label. The pipeline proposed in this dissertation (Sec. 4)

utilizes a stream-based unsupervised segmentation algorithm and addresses the shortcoming of over-segmentation, while automatically associating the learned segments with high-level semantics that can be used for terrain-aware navigation.

2.4.2 Learning System Adaptation

Partly based on the realization that static learning systems limit real world applicability, much research effort has been devoted to developing learning systems which can adapt to new data and their associated output classes. Specifically, such systems must adapt without the need for retraining the entire system, or using human effort to label new data. In transfer learning [94], weights from a pre-trained model are used as the initial weights of a model performing a different but similar task, whereas in incremental learning [48], an initial trained model is tasked to adapt online to new incoming data without forgetting the existing knowledge from previous data points in the stream. Similar work in domain adaptation [75] extends a model trained in a source domain to execute tasks in a similar although different target domain with a separate underlying data generation distribution. While the proposed algorithm (Sec. 4.4) loosely fits in the category of methods discussed, the proposed approach is distinct as we semantically classify (over-segment) the different terrain features seen in an environment for a single task, without explicitly considering the target distribution.

2.4.3 Unsupervised Visual Representation Learning

Unsupervised representation learning [11] searches for an appropriate hierarchy of concepts, or features, which explain a dataset using unlabeled data. Initial work [21] incorporates self-supervision in the form of a pretext signal which relates parts of the unlabeled data spatially. This relation is used as an intermediate supervisory signal aimed at increasing overall performance [26]. Similar work incorporates linear transformations [30] or temporal [44] relations. Although self-supervised visual representation learning achieves superior performance than

their unsupervised counterparts, neural network architecture designs which minorly affect supervised models may majorly affect performance of self-supervised models [42].

Most similar to this dissertation, weakly supervised representation learning approaches [93] utilize sparse, incomplete labeling to learn a model. Distinctly, this dissertation utilizes human feedback as a weak supervisory signal to aid unsupervised visual representation learning. Specifically, inexact supervision in the form of a human teleoperated robot navigation trajectories with a single label at each time step representing which terrain the robot traversed is used.

Chapter 3 Task 1 - Responding to Previously Unseen Environmental Features

3.1 Abstract

Traditional imitation learning provides a set of methods and algorithms to learn a reward function or policy from expert demonstrations. Learning from demonstration has been shown to be advantageous for navigation tasks as it allows for machine learning non-experts to quickly provide information needed to learn complex traversal behaviors. However, a minimal set of demonstrations is unlikely to capture all relevant information needed to achieve the desired behavior in every possible future operational environment. Due to distributional shift among environments, a robot may encounter features that were rarely or never observed during training for which the appropriate reward value is uncertain, leading to undesired outcomes. This chapter proposes a Bayesian technique which quantifies uncertainty over the weights of a linear reward function given a dataset of minimal human demonstrations to operate safely in novel environments. This uncertainty is quantified and incorporated into a risk averse set of weights used to generate cost maps for planning. Experiments in a 3-D environment with a simulated robot show that our proposed algorithm enables a robot to avoid dangerous terrain completely in two out of three test scenarios and accumulates a lower amount of risk than related approaches in all scenarios without requiring any additional demonstrations.

3.2 Introduction

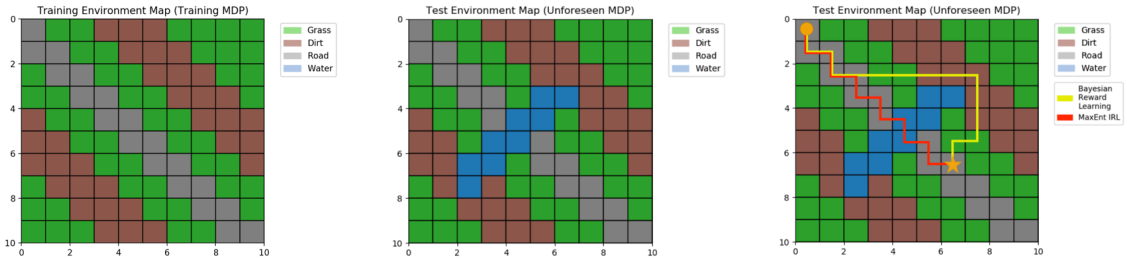

Figure 3.1: Simplified illustration to demonstrate the impact of distributional shift. Left: Training environment. Middle: Test environment. Right: Planned trajectories in the testing environment for various learning methods.

Robot behavior can be described through a reward function which directs the robot toward states leading to the achievement of goals and developer specifications [41]. This encoding often focuses on goals defined during system design, implicitly expressing indifference to all others, resulting in negative side effects [4] such as the robot traversing a harmful terrain or crashing into an obstacle. Inverse reinforcement learning (IRL) [57, 1, 96] seeks to learn a reward function given a dataset of demonstrations, which eliminates the need for an expert to hand code these rewards. However, the performance of these approaches are sensitive to the features seen in the demonstrations. Certain states and the features that describe them may have never been encountered, leading to uncertainty in their reward [45, 33]. This presents a challenge for autonomous navigation since it is desirable for a robotic system to be robust to operation in unknown, possibly adversarial, environments which are likely to contain unforeseen conditions.

As an example, consider an autonomous ground robot which learns to navigate in an environment consisting of four types of terrain, including grass, dirt, road, and water (Fig. 3.1 middle). In the training environment (Fig. 3.1 left), none of the states contain the water

feature, and subsequently neither do the demonstrations. Maximum likelihood approaches such as [96] implicitly set the reward weight for water to their initialization value because its feature count is zero. Since the demonstrations do not provide full reward information, the robot fails to avoid water in the test environment (Fig. 3.1 right). Water may have been avoided if different initial rewards were used, but this would not resolve the more general problem of how to handle states possessing uncertain reward values. As a step toward solving this problem, we propose a systematic approach which utilizes Bayesian analysis to quantify uncertainty for each terrain's reward weight. This approach allows the robot to achieve risk averse behavior by avoiding terrain possessing high uncertainty (Fig. 3.1 right).

Previous imitation learning methods have provided a Bayesian framework to incorporate prior information and obtain a unique reward function [66, 17]. Most similar to our work, Hadfield-Menell et al. [33] provide a Bayesian technique, which explicitly considers safety to distributional shift in environments. The difference between their methodology and ours is the assumption of a Bayesian posterior. Specifically, they obtain a posterior over reward functions given a proxy reward function and a world model, while we do not assume a proxy reward function is given, but rather a dataset of demonstrations. This change in assumptions allows developers to obtain a reward function which explicitly considers safety solely from observed behavior. However, the main difference between Ref. [33] and our methodology is in the reward function selection technique. The reward function obtained in Ref. [33] changes at runtime, requiring planning with multiple reward functions online, whereas our technique outputs a single reward function at the end of training. Online planning with a single reward function reduces computational overhead, thereby increasing the capability to scale to larger environments.

We build upon previous IRL work [81], which learned traversal behavior reward models for autonomous navigation. This method assigns high rewards to state features visited more

frequently during demonstration, implicitly assuming that high visitation frequency means the state feature is better than those visited less frequently. Our inclusion of uncertainty modifies this assumption such that if a state feature is visited less it is associated with more uncertainty. We propose *risk averse Bayesian reward learning (RABRL)*, as a method to obtain a unique, risk averse linear reward function solely from a dataset of demonstrations for autonomous waypoint navigation. The contributions are twofold, we provide a methodology to quantify uncertainty over reward functions from a set of human demonstrations, and provide a weight selection technique, which chooses a single weight set during training.

The remainder of the chapter is organized as follows. Section 3.3 presents our methodology showing how a posterior over reward functions is used to obtain a unique set of risk averse weights. Section 3.4 describes experimental setup and results obtained from an autonomous ground robot navigating in a 3-D simulation. Section 3.5 concludes with areas this research can impact.

3.3 Methodology

An outline of the methodology follows. Section 3.3.1 formally models the problem as an MDP without rewards. Section 3.3.2 formulates the problem. Section 3.3.3 explains how the likelihood of a demonstrator's reward intent over terrains is modeled. Section 3.3.4 describes two distributions to encode prior reward information. Finally, section 3.3.5 describes our methodology to select the reward weights for each feature.

3.3.1 Environment and Robot Modeling

Recall from Sec. 2.2.2 that an autonomous robot navigating in an environment is modeled using a Markov Decision Process (MDP), M, represented by the following tuple

$$M = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle \tag{3.1}$$

where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, T is the state transition distribution over the next state given the present state and action represented by $T(s_{t+1}|s_t, a)$, R is the reward function representing the numerical reward received by taking action $a \in A$ in state $s \in S$ represented by $R(s, a) : \mathcal{S} \times \mathcal{A} \rightarrow R$, and $\gamma \in [0, 1)$ is the discount factor representing the weight on future unseen rewards. The solution to an MDP is a policy $\pi(s) : \mathcal{S} \rightarrow \mathcal{A}$, determining the action a a robot will take in state s. The optimal policy for an MDP (π^*) maximizes the expected cumulative reward.

Scenarios where the agent finds itself in an unknown environment are modeled by omitting R in Eq. (3.1).

$$M \backslash R = \langle \mathcal{S}, \mathcal{A}, T, -, \gamma \rangle \tag{3.2}$$

In the reinforcement learning framework, a reward function is considered to be the most succinct, robust, and transferable definition of a task [57]. To learn a reward function, an agent is supplied demonstrations in the form of trajectories that depict the desired behavior. For a navigation task, the cumulative reward associated with a trajectory demonstration can be found through its state sequence $\xi = [s_1, s_2, ..., s_T]$, where T is the number of time steps. Although we only consider reward functions that are a function of state $R(s_t)$, one may also wish to consider reward functions that are a function of the state and action $R(s_t, a_t)$ or reward functions that are a function of the state, action, and next state $R(s_t, a_t, s'_t)$. This modeling choice depends on the goals and preferences the system designer desires the agent to learn [72]. Formally, this is an example of a reward design problem [71], where the true reward function is unobservable, but possible reward functions are assessed by a fitness function given a distribution of environments the agent may find itself in. The behavior of an agent operating in $M \backslash R$ is summarized as a probability distribution over trajectories given a vector of reward weights $P(\xi|\hat{w})$ and is referred to as a robot model. For a list of robot models incorporating human feedback, we refer the reader to [37].

3.3.2 Problem Formulation

We seek a posterior over the weights describing a reward function,

$$P(w = \hat{w}|\mathcal{D}) = \frac{P(\mathcal{D}|\hat{w})P(\hat{w})}{P(\mathcal{D})} \tag{3.3}$$

where w is a random vector describing the weights of a reward function, $\hat{w}$ is an estimator of w, $\mathcal{D}$ is a dataset of demonstrations (navigation trajectories) such that $\mathcal{D} = \{(\xi_i), (\xi_{i+1}), \ldots, (\xi_n)\}_{i=1}^{n}$. We consider reward functions as a function of trajectory states expressed as a linear combination between estimated weights $\hat{w}$ and features $\phi(s)$ representing a state such that $\phi : \mathcal{S} \rightarrow \mathbb{R}^{\mathbb{D}}$, where $\mathbb{D}$ indicates the dimension of the feature space.

$$R(s) = \hat{w}^T \phi(s) \tag{3.4}$$

The total reward for a trajectory is the sum of its state rewards.

$$R(\xi) = \sum_{s_i \in \xi} \hat{w}^T \phi(s_i) \tag{3.5}$$

We define the reward space $\mathcal{R}$ as a discrete set of fixed weight vectors, which parameterize a reward function. The discrete set $\mathcal{W}$ contains all possible values for a single element in the weight vector. The number of possible weight vectors is therefore represented by the cardinality of $\mathcal{W}$ raised to the dimension of the feature space, $|\mathcal{R}| = |\mathcal{W}|^{\mathbb{D}}$. The reward space $\mathcal{R}$ should contain values representative of the number of features and their scaled differences. At a minimum, $|\mathcal{W}|$ should be equal to $\mathbb{D}$, so that each feature may be assigned a distinct value, representing the preference over features. However, to enable a reward model to consider the scaled reward difference, such as "water is 10 times worse than grass,"

a larger or non uniformly spaced set of values can be specified to capture such preference. The training time of the model increases as $|R|$ and $|\mathcal{S}|$ increase, so it is important to choose a value of $|R|$ relative to the complexity of the domain. The posterior is computed at $|\mathcal{R}|$ discrete points to obtain a nominal probability for each point. These nominal points are subsequently divided by their marginal probability producing a valid discrete posterior distribution.

Alternatively, with the use of Markov Chain Monte Carlo [5], a continuous posterior over reward functions can be obtained, allowing one to model an infinite number of reward functions. However, this requires numerically approximating integrals, adding computational complexity during training. Furthermore, in the context of our weight selection technique (Section 3.3.5), a continuous posterior would require using differential entropy, which is difficult to interpret [53]. Due to these limitations and the performance achieved with a small number of reward functions (Section 3.4.3), we chose the discretization approach.

3.3.3 Likelihood Modeling

The likelihood of a demonstrator assuming independent and identically distributed trajectories is defined as the product of the individual trajectories.

$$\begin{aligned} P(\mathcal{D}|\hat{w}) &= P(\xi_1|\hat{w}) \times P(\xi_2|\hat{w}) \times ... \times P(\xi_n|\hat{w}) \\ &= \prod_{\xi_i \in \mathcal{D}} P(\xi_i|\hat{w}) \end{aligned} \tag{3.6}$$

More specifically, we model the demonstrator using the maximum entropy IRL distribution [96].

$$P(\mathcal{D}|\hat{w}) \propto \prod_{\xi_i \in \mathcal{D}} \exp\left(\beta \hat{w}^T \mathbb{E}[\phi(\xi_i)|\xi_i \sim P(\xi_i|\hat{w})]\right) \tag{3.7}$$

where $\beta \in [0, 1]$ represents the level of confidence in a demonstrator. $\beta = 0$ indicates low confidence, such as random behavior from a demonstrator, while $\beta = 1$ indicates optimal behavior with respect to the reward preference over features. The expected feature count is high when a trajectory ξ_i obtains high rewards for a given robot model $P(\xi|\hat{w})$ relative to all other trajectories and vice versa, as represented by the dot product between weights and the expected feature count. Therefore, an increase in the reward increases a weight vector's desirability, quantifying a preference over features with respect to $\mathcal{D}$.

The feature expectation of all demonstrated trajectories is:

$$\mathbb{E}[\phi(\xi)] = \sum_{\xi_i \in \boldsymbol{\xi}} P(\xi_i|\hat{w})\phi(\xi_i) \tag{3.8}$$

Where $\boldsymbol{\xi}$ represents the set of all possible trajectories that can be taken in the MDP.

Although several candidate robot models may be suitable [37], we use maximum entropy IRL [96], which also contains an algorithm to compute Eq. (3.8).

$$P(\xi|\hat{w}) = \frac{\exp(\hat{w}^T\phi(\xi))}{Z(\xi)} \tag{3.9}$$

Calculating the feature expectation directly is infeasible because it requires an agent to consider all possible trajectories in the MDP, as captured in the normalization constant $Z(\xi)$. However, this can be approximated by using either the forward backward algorithm or value iteration [95].

3.3.4 Prior Modeling

There are multiple ways to incorporate prior information about reward weights, $P(\hat{w})$. For our work, we chose a modified uniform prior as an uninformative prior, and a Dirichlet prior as an informative prior.

If a demonstrator does not prefer any one weight paramaterization, a uniform prior may be used. However, if a reward function has all the same weights for each feature, any set of demonstrations appear Boltzmann optimal [57]. Therefore, we use a modified version of the discrete uniform prior, where all weight sets have equal probability *unless* all its weights are the same, in which case its probability is zero.

$$P(\hat{w}) = \begin{cases} 0 & \text{if, } \hat{w}_1 = \hat{w}_2 = ... = \hat{w}_n \\ \frac{1}{|\mathcal{R}|-|\mathcal{W}|} & \text{otherwise} \end{cases} \tag{3.10}$$

Each element in the weight vector $\hat{w}$ above takes a value from the discrete set $\mathcal{W}$.

In some cases, a preference over terrains is known a priori, and therefore can be captured by a Dirichlet prior, a continuous multivariate generalization of the beta distribution

$$P(\hat{w}) = \frac{1}{Beta(\alpha)} \prod_{i=1}^{\mathbb{D}} \hat{w}_i^{\alpha_i - 1} \quad \forall i,\ 1 < \alpha_i \tag{3.11}$$

such that $\alpha \in \mathbb{R}_+^{\mathbb{D}}$ where each α_i represents our preference over a corresponding feature weight w_i. The higher the α_i, the more density the component possesses. That is, a large value of α_i, corresponds to preference over all other components j for which $\alpha_i > \alpha_j$ is true. The Dirichlet distribution is subject to the constraint $\sum_{i=1}^{|\mathcal{W}|} w_i = 1$. To satisfy this constraint, a softmax function is applied to the current weight vector, producing normalized weights.

$$\hat{w}_i \leftarrow \frac{\exp(\hat{w}_i)}{\sum_{j=1}^{|\mathcal{W}|} \exp(\hat{w}_j)} \tag{3.12}$$

In the Dirichlet prior, $\hat{w}$ is assumed to be continuous, while in Sec. (3.3.1) we have defined it to be discrete. However, a proper prior is obtained as a result of applying Eq. (3.12) to a given weight vector before obtaining the result in Eq. (3.11).

3.3.5 Planning Risk Averse Behavior

A Bayesian posterior is obtained by evaluating Eq. (3.3) for $|\mathcal{R}|$ different reward vectors. A point evaluation is obtained by multiplying the likelihood (Eq. (3.7)) and a prior (Eq. (3.11)) or (Eq. (3.10)) for each reward function in $\mathcal{R}$. Then, a normalization constant is computed by taking the sum of the point products. The system designer can then quantify uncertainty over $\hat{w}$. Uncertainty is expressed as the normalized Shannon entropy of the marginal probability distribution of each feature weight $\hat{w}_i \in \hat{w}$. The marginal probability is calculated by holding the weight being marginalized constant and summing over all possible values of the other $n-1$ variates.

$$p_{w_i}(k) = \sum_{\forall \hat{w}_j \in \hat{w} | \hat{w}_j \neq \hat{w}_i,\ \hat{w}_j \in |\mathcal{W}|} P(w_1, \ldots, w_i = k, \ldots, w_n) \tag{3.13}$$

Uncertainty is defined according to each individual feature, since each corresponds to a distinct semantic meaning. The overall uncertainty of each marginal distribution is quantified by the normalized Shannon entropy

$$H(w_i) = -\frac{\sum_{k=1}^{|\mathcal{R}|} p_{w_i}(w_k) \log_2 p_{w_i}(w_k)}{\log_2 |\mathcal{R}|} \tag{3.14}$$

yielding a value in the interval $[0, 1]$. An entropy of 0 indicates certainty in the weight's value, and a value of 1 represents maximum uncertainty, the uniform distribution. Weights are then chosen by their respective uncertainty using a specified level of acceptable risk, ϵ, where smaller values of ϵ indicate greater risk acceptance. When the entropy is greater than or equal to the threshold, $H(w_i) \geq 1 - \epsilon$, the lowest reward weight is chosen. Otherwise, if $H(w_i) < 1 - \epsilon$, the expected value $E[w_i]$ of the marginal distribution is chosen as the reward weight.

The weights of the linear reward function are then used to produce a costmap for a

navigation planning algorithm. Costmap generation is a simple dot product between the reward weight vector and the environment feature maps (discussed in Section 3.4.2). To assess model performance, we express risk as the percentage of time the robot traversed a potentially dangerous terrain, λ, throughout its trajectory,

$$Risk(\xi) = \sum_{s_i \in \xi} \frac{\lambda_{s_i}}{|\xi|} \tag{3.15}$$

where $|\xi|$ represents the length of the trajectory and λ_{s_i} is an indicator function returning 1 when $\phi(s_i)$ contains the dangerous feature λ and 0 otherwise. Assuming the unseen terrain's true reward is low or even negative, lower risk values should be correlated with the robot's safety.

Since risk alone is not a sufficient metric, we assess the overall performance as the tradeoff between risk and path length because longer paths tend to correspond with increased energy use and time to complete a mission.

3.4 Evaluation

This section evaluates RABRL in a 3-D simulated environment for an autonomous navigation task using a ground robot equipped with virtual sensors.

3.4.1 Experiment Overview

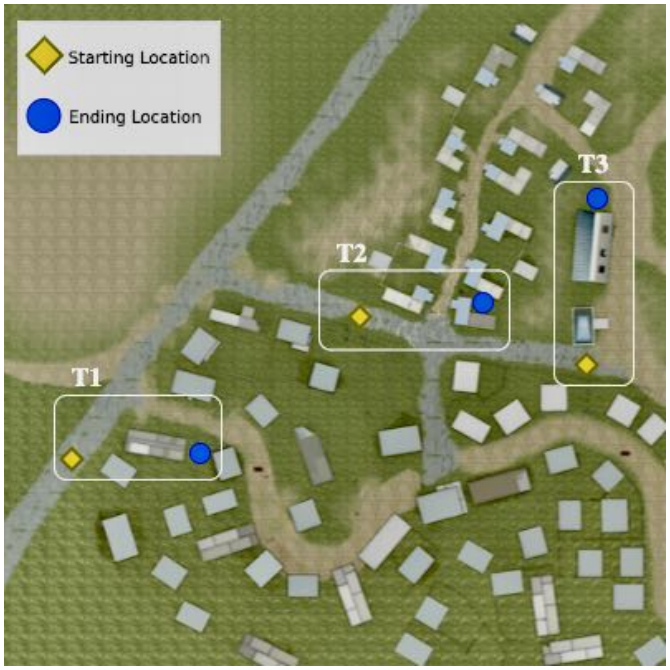

Figure 3.2: Birds-eye view of test scenarios in the simulated environment. Terrain colors are as follows: green corresponds to grass, brown to mud, and gray to asphalt.

All experiments were performed in a simulated 3-D Unity environment that represents a semi-structured village containing three terrain types describing our features, ϕ, namely *grass*, *asphalt* and *mud*, as well as several obstacles, including buildings, trees, and vehicles. Fig. 3.2 provides a birds-eye view of the three terrain types and buildings in the environment. This simulated environment possesses greater complexity than the toy illustration shown in

Fig. 3.1, including: (i) noisy demonstrations due to imperfect perception and mapping, and (ii) the requirement to plan kinematically feasible trajectories.

To facilitate objective comparison, each reward model learning approach was trained with the same set of demonstrations, which were collected by having a human teleoperate the robot in an area of the simulated environment. When collecting these trajectories, the demonstrator stayed almost entirely on asphalt, in an attempt to show the robot their preference for driving on roads instead of grass. To showcase how the approaches handle learning from training data that lacks a full representation of the operating environment, demonstrations were collected in an area where no mud was present, thereby setting λ to correspond to the mud feature. This methodology encompasses a number of real-world situations, including applications where training data cannot be collected in the precise operational environment or where there is significant time lapse or adverse weather conditions that cause an environment to change relative to the time at which training data was collected.

Three different reward models were trained, including (i) RABRL with a uninformed uniform prior, (ii) RABRL with an informed Dirichlet prior, and (iii) Maximum Entropy IRL [81], which serves as a baseline towards learning semantic terrain rewards from human demonstrations, since both approaches use Maximum Entropy as a robot model. Although the risk aversion idea from Hadfield-Menell et al. [33] motivated this work, the methodology described there seeks to resolve a given misspecified, partially defined reward function. As mentioned previously, RABRL seeks to learn a reward function solely from a dataset of demonstrations without ever being given a partially defined reward function, and is therefore difficult to compare directly.

We compare performance of each method on the three test scenarios shown in Fig. 3.2. To enable statistical analysis, including hypothesis testing, the robot started at the same location for each combination of test scenario and reward model, navigating to the same

goal waypoint. Moreover, each combination of test scenario and reward model was run five times to account for stochasticity from imperfect mapping. The percentage of the time the robot went into the mud was calculated for each trajectory with Eq. (3.15). Evaluation metrics are averaged across the five trial results.

3.4.2 Robotic Autonomy Stack

Figure 3.3: Simulated environment with Clearpath warthog running ROS equipped with an array of virtual sensors.

A Clearpath Warthog equipped with an array of sensors including a 3D LiDAR, IMU, and two monocular cameras (Fig. 3.3) was deployed for the simulations. The robot possesses a full autonomy stack (Section 2.1) consisting of three main subsystems, namely mapping, perception, and planning. We briefly describe these subsystems, discussing how our contribution to risk-averse costmap generation interfaces with each of these subsystems.

The mapping subsystem is based on OmniMapper [77] and provides the necessary localization for autonomous navigation. The map from this subsystem represents a costmap layer for obstacle avoidance used by the planning subsystem. The risk-averse costmap acts as a

second layer that provides terrain-awareness to the planning system, enhancing the overall navigation of the robot.

The perception subsystem includes semantic segmentation* of camera images for an ontology of terrain and object classes such as grass, asphalt, and building. As images are segmented, the semantic label of each pixel is used to accumulate evidence for binary occupancy terrain grids, which represent the semantic features, ϕ, used for reward model learning and risk-averse costmap generation.

As previously mentioned, the planning subsystem uses costmap layers generated from mapping and our IRL algorithm to plan paths during navigation. Specifically, costmaps serve as input to a global planner to search for the lowest-cost trajectory between the robot's location and a specified goal waypoint. In our system, global planning is computed with the Search-Based Planning Library [46] to find a kinematically achievable plan by searching combinations of motion primitives.

3.4.3 Results

The dataset of demonstrations was used to train three different reward models. If the robot were to operate fully online, imperfections from all other subsystems (perception, SLAM, mapping, and planning) would propagate. Therefore, to best capture the performance of the proposed methodology, a robot collects a map of its environment a priori and then builds a costmap with respect to the learned reward weights. However, the reward function obtained from RABRL could be used to produce costmaps in an online setting.

For both of the RABRL reward models, Model (i) and (ii), a multivariate posterior was obtained from Eq. (3.3). The confidence parameter was set to $\beta = 0.3$ to indicate relatively

*For the experiments reported, we use the ground truth semantic segmentation produced by the simulation.

low confidence in the optimality of the demonstrator and $\epsilon = 0.01$ was used, resulting in a threshold of $1 - 0.01 = 0.99$, indicating a relatively high tolerance to uncertainty.

The reward weight associated with a terrain was allowed to take on a value from the set $\mathcal{W} = \{-2, ..., 1\}$. yielding a reward space of 64 possible reward functions, since $|\mathcal{W}|^{\mathbb{D}} = 4^3 = 64$. This domain allowed each feature to be assigned a distinct value and also enabled a modest amount of reward scaling, since $|\mathcal{W}|$ was one larger than $\mathbb{D}$. For the maximum entropy reward model, training was performed according to the reward model outlined in [81].

Table 3.1: Marginal Entropy

Reward Model	**Entropy -** $H(w_i)$		
	$H(w_{grass})$	$H(w_{mud})$	$H(w_{asphalt})$
(i) RABRL Unifrom	0.974	0.981	$4.366 * 10^{-14}$
(ii) RABRL Dirichlet	0.759	0.797	$2.781 * 10^{-15}$

Table 3.1 shows the normalized Shannon entropy, as evaluated by Eq. (3.14), of each marginal distribution obtained from Eq. (3.13) for each terrain. Table 3.1 indicates that the normalized marginal entropy for mud was highest in both models, while the grass was second highest because the human demonstrations attempted to avoid grass. However, both models are virtually certain of the preference for asphalt, as indicated by an entropy level close to zero. Moreover, the entropy of Model (i) was higher because it employed an uninformed prior, whereas an informed prior, such as the Dirichlet prior in Model (ii), exhibited less uncertainty over the reward weights for each terrain. If the risk acceptance parameter ϵ was larger, resulting in less tolerance for risk, the respective reward weights would change. Specifically $\epsilon = 0.05$ results in weights of -2 for both grass and mud in the uniform model, thereby exemplifying the importance of an informed prior.

Table 3.2: Feature Weight Vectors

Reward Model	**Feature Weight**		
	w_{grass}	w_{mud}	$w_{asphalt}$
(i) RABRL w/Uniform	-0.258	-0.687	1.000
(ii) RABRL w/Dirichlet	-0.687	-1.253	1.000
(iii) Maximum Entropy IRL	-0.304	0.000	0.567

Table 3.2 shows the estimated reward weight vector $\hat{w} = \langle grass, mud, asphalt \rangle$ corresponding to the terrain features, which was determined with the weight selection technique described in Section 3.3.5. Model (iii) implicitly assigns the mud a reward weight of zero, thereby indicating a preference for mud over grass. Since mud was never encountered during training, its gradient during maximum likelihood optimization is always zero. Conversely, Models (i) and (ii) prefer every other terrain more than mud due to its high uncertainty.

Table 3.3: Total Average Risk Taken

Reward Model	**Total Risk**		
	T1	**T2**	**T3**
(i) RABRL w/Uniform	0.2090	**0.0000**	**0.0000**
(ii) RABRL w/Dirichlet	**0.0617**	**0.0000**	**0.0000**
(iii) Maximum Entropy IRL	0.3120	0.1724	0.8992

Table 3.3 shows the average risk, which was computed by averaging the values computed with Eq. (3.15) from the five runs for each combination of test scenario and reward model. Table 3.3 indicates that the proposed method, RABRL, achieved lower risk in each test scenario and, in scenarios T2 and T3, the robot found a path to traverse which avoided

mud entirely. Moreover, Table 3.3 shows that, in T1, the informed Dirichlet prior (Model (ii)) accumulated approximately one third of the risk of Uniform prior (Model (i)) and one approximately one fifth of the risk of Maximum Entropy IRL (Model (iii)). Furthermore, while the informed prior took less risk than its uninformative counterpart, scenarios T2 and T3 show that it is also possible to avoid side effects with uninformative priors.

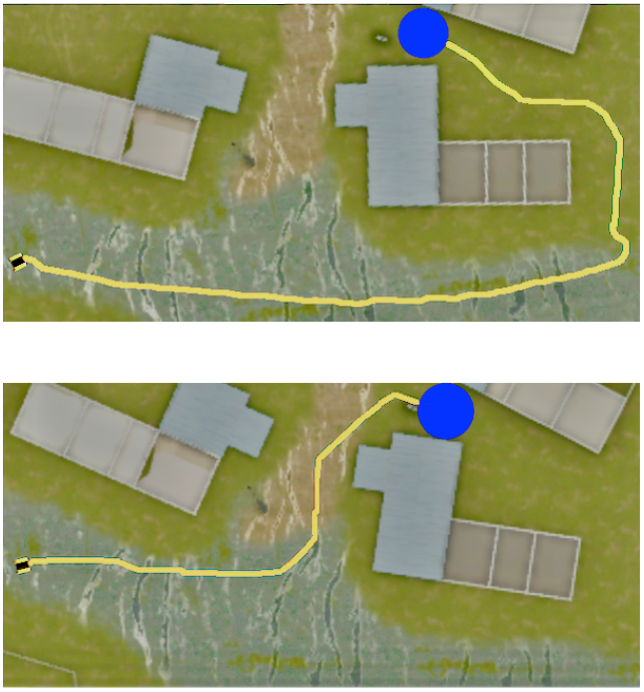

Figure 3.4: Trajectory for Test Scenario 2 (T2) generated using two reward models with waypoint region (blue circle).**Top:** (i) RABRL w/ Uniform **Bottom:** (iii) MaxEnt IRL

To further clarify the observations made in Table 3.3, Fig. 3.4 shows a birds eye view of the trajectories taken by Models (i) and (iii) in T2. Model (iii) took a shorter path traversing mud, while Model (i) took a longer path and never traversed mud. Therefore, in certain scenarios RABRL was able to avoid negative side effects, which occur due to distributional shift in environmental terrain.

Table 3.4: Average Path Length

Reward Model	**Test Scenario**		
	T1	**T2**	**T3**
(i) RABRL w/Uniform	335.8	436.0	**454.6**
(ii) RABRL w/Dirichlet	343.4	431.0	492.6
(iii) Maximum Entropy IRL	**327.0**	**267.2**	465.0

Table 3.4 shows the average path length computed with the five runs on each combination of test scenario and reward model. Table 3.4 indicates that Model (iii) took the shortest path in Test Scenarios T1 and T2. However, the path lengths of Models (i) and (ii) were competitive with Model (iii) in T1 and T2 took substantially more risk as noted in Fig. 3.4. Thus, in some scenarios, RABRL is able to substantially lower or eliminate risk while preserving a low path length. Moreover, in T3 Model (i) outperformed Model (iii) with respect to both total average risk and path length.

Fig. 3.5 shows a scatter plot of the tradeoff between normalized risk (Eq. (3.15)) and path length, include all five runs for each combination of test scenario and reward model. To rigorously illustrate that RABRL reduced risk substantially, we performed a two means test on the risk taken for pairs of models. For example, a two-tailed test with a null hypothesis of equal risk in Models (ii) and (iii) was rejected in all three scenarios at the 99.95% confidence level with p-values of 3.046034×10^{-8}, 0.000449, and 1.122941×10^{-7} respectively, strongly favoring RABRL. Furthermore, we performed a two means test on the path lengths for pairs of models. In some cases, Model (iii) performed best, but in others the results were equivocal. Specifically, a two-tailed test with a null hypothesis of equal path lengths in Models (ii) and (iii) produced p-values of 0.247475, 1.334867×10^{-5}, and 0.227629 for the three scenarios, suggesting that the null hypothesis of equal path length could not be rejected at the 90% or

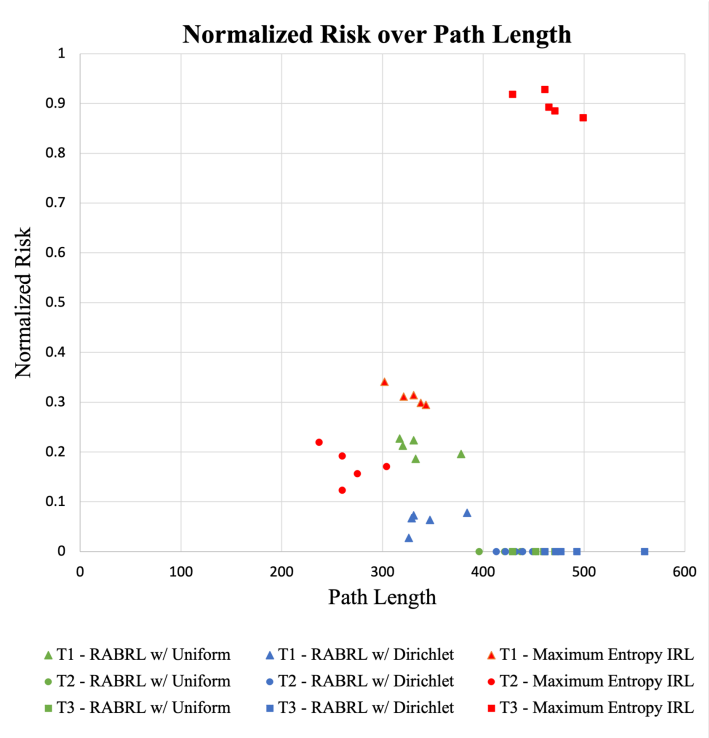

Figure 3.5: Tradeoff between risk and path length.

even 80% confidence level in Scenarios T1 and T3. Thus, while RABRL took a longer path to avoid mud in T2 as was shown in Fig. 3.4, the lower risk taken by RABRL demonstrated very strong statistical significance without a significant increase in path length in two of three scenarios.

3.5 Conclusion

This chapter proposed a Bayesian technique to express the uncertainty over the semantic terrain reward weights of a linear reward function obtained from a dataset of human demonstrations. With the use of normalized Shannon Entropy, the relative uncertainty over reward weights can be learned by considering a small space of reward functions. Experiments performed in a simulated 3D environment showed that a robot may leverage its uncertainty over

semantic terrains to choose a trajectory with less risk. However, this may require longer trajectories in some scenarios. Our proposed methodology, RABRL, enables an agent to avoid potentially dangerous terrain, while operating in an altered training environment.

The proposed technique is a member of the growing class of imitation learning techniques, which explicitly seek to avoid negative side effects that occur as a result of distributional shift of operational environments. Safe semi- or unstructured ground autonomy is likely to contain terrain scenarios never encountered during training. Rather than undertake the infeasible task of attempting to capture such training data, a proactive approach to quantify uncertainty will identify gaps in training data and adapt behavior appropriately. By resolving ambiguities, implicit biases, and misspecifications in reward models obtained from human demonstrations, robots will be able to make more informed decisions, leading to safer behavior relative to their predecessors.

Chapter 4 Task 2 - Learning an Accurate Perception Representation of the Current Environment

4.1 Definitions

Definitions of commonly occurring terms in the chapter that follows is provided below.

Perception System - A subsystem within an autonomous system which takes in raw sensor data as input, and outputs data products representing the overall understanding of the current state of the world. This system often performs scene understanding tasks such as object recognition and semantic segmentation.

Semantic - A noun that has a specific meaning. Such as the underlying terrains and objects present in the robot's operational environment.

Segment - A particular area/region of adjacent pixels within an image; normally all pixels in a segment capture the same characteristic; does not signify semantic meaning on its own.

Semantic Segment - A segment drawn around a specific semantic. Such as a shape drawn around all of the pixels in the image for a particular object or ground terrain.

Representation - A set of vectors capturing the feature basis for a sequence of images. Each vector contains the same number of dimensions and is referred to as a "feature vector".

Label (unsupervised learning) - A label gives clusters a name so they can be referenced. Although each label is a "semantic label" in the context of semantic segmentation, each label likely does not have a clear semantic meaning in unsupervised learning.

Terrain Projection System - A subsystem which consumes labeled segmented images as input, and produces a set of one-hot occupancy grid maps as output. This is done via a pinhole camera model which projects pixels of an image to the robot's ground plane with respect to the robot's current pose.

Traversal Evidence - Numerical feature counts, counting the number of times each label is

traversed (driven) over during a robot's trajectory. This is obtained by imposing the robot's trajectory over its global map and counting grid cells in the occupancy grid maps which intersect with the robot's trajectory.

4.2 Abstract

Rapid progress has been made in terrain-aware autonomous ground navigation, in part due to advances in visual perception, specifically supervised semantic segmentation. However, such approaches require expensive data collection and time-consuming ground truth labeling to train deep learning architectures. Moreover, some applications may require the autonomous vehicle to navigate in unseen, unstructured environments with competing ground terrains, of which no labeled dataset exists. To circumvent the need to perform dense pixel-wise labeling in a new environment for supervised learning, we consider how to leverage human demonstrations and unsupervised learning to quickly deploy a perception system for terrain-aware navigation. Specifically, this paper introduces a technique that uses minimal demonstration evidence as a weak supervisory signal to refine and align learned abstract concepts from unsupervised semantic segmentation to fine-grained terrain semantics. The presented methodology alleviates over-segmentation produced by the unsupervised approach and yields a refined feature representation which maps to the set of semantic classes found in the operating environment. We demonstrate the applicability of the methodology using both an over-segmentation simulator utilizing a 3D graphics engine, and the RUGD dataset, in which an initial over-segmented unsupervised semantic segmentation model is refined to learn a more accurate representation of the underlying environment, thereby enabling terrain-aware autonomous navigation.

4.3 Introduction

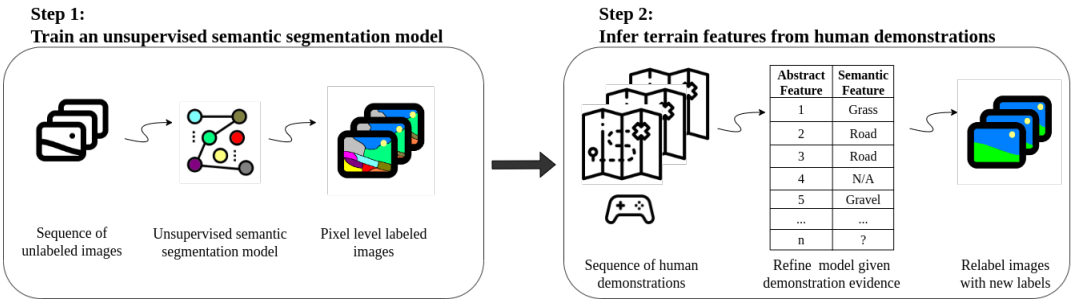

Figure 4.1: A visual representation of the proposed algorithm. Left: Sequence of steps to obtain an initial unsupervised semantic segmentation model which over-segments. Right: Sequence of steps which use human demonstrations as a weak supervisory signal to provide evidence for abstract label refinement and mapping to semantic labels.

Safety is critical if autonomous navigation is going to be adopted for mainstream use. In structured settings such as urban city navigation, the awareness of lane markings, pedestrians, and street signs are some examples of the context needed for an autonomous vehicle to make safe navigation decisions. Recent advances in autonomous vehicle perception [78], specifically supervised semantic segmentation [29, 43], provide fine-grained context of the environment by classifying each pixel in an image using a discrete class set. These supervised approaches achieve high performance when they are trained with large labeled datasets [27, 32].

However, application spaces such as humanitarian assistance and disaster relief [55, 56], require operation in semi-structured or unstructured environments. In these domains, autonomous vehicle safety is highly dependent on the capability to robustly traverse complex terrain. Unfortunately, existing datasets representing structured environments lack the terrain diversity seen in unstructured settings. Although some datasets have recently emerged

that represent off-road environments with more terrain diversity and natural environmental constraints [82, 39], supervised semantic segmentation approaches will always be limited by the availability of such datasets and the ground truth labels associated with them. For application spaces with extreme uncertainty in environment conditions where all possible future semantics are not known a pirori, supervised perception systems may not adequately provide the context needed for safe navigation.

In contrast, unsupervised semantic segmentation approaches learn semantic concept representations from unlabeled data. Without a supervisory signal, the unsupervised algorithm must determine the appropriate granularity of segmentation. To lessen this challenge, some approaches [38, 16] assume the number of classes within the data is known a priori to provide a weak signal during the learning stage. With this information, the approaches can take a large batch of unlabeled data and identify the underlying class representations. However, in scenarios where an autonomous vehicle is required to act in an unseen environment without sufficient time to collect (much less label) large domain-specific datasets, it is beneficial to have the capability of both (i) identifying an appropriate number of discrete semantic classes while also (ii) learning a representation of the current operational environment offline with a small amount of unlabeled data to avoid the risk of total mission failure.

Stream-based unsupervised semantic segmentation approaches [88, 80] have the ability to process data in an online fashion without making any underlying assumptions about the number of concepts that will be present in the environment. This makes them ideal for online operation under an open-world paradigm [62], i.e., where novel concepts not represented by the perception system can be encountered at any time during operation. However, such approaches have not been deployed to support visual perception for autonomous navigation because they often produce under or over-segmented representations relative to the goals of the perception system as no supervisory signal is present. As illustrated in fig-

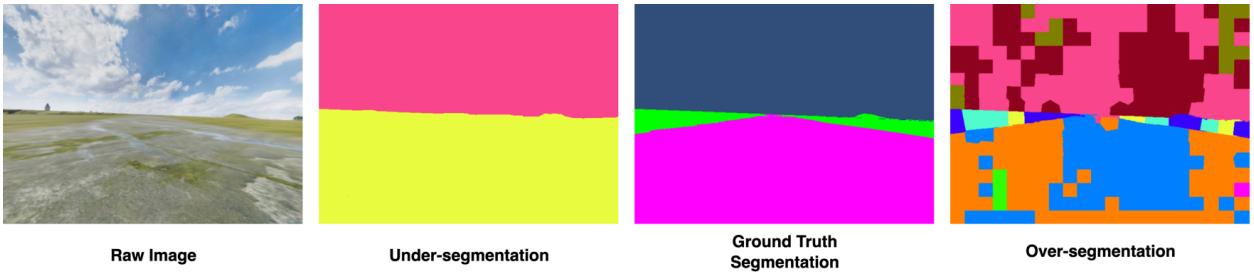

Figure 4.2: Segmentation at different levels of granularity. Column 1 - raw image; column 2 - example of under-segmentation; column 3 - example of perfect segmentation; column 4 - example of over-segmentation.

ure 4.2, under-segmented representations fail to identify all semantic concepts (column 2), while over-segmented representations obtain more labels then there are semantic concepts (column 4). Furthermore, the desired level of segmentation is dependent on the context of the mission's one would like to complete. Under-segmented representations do not contain the context needed, while over-segmented models offer too much context. We seek to address these shortcomings of stream-based unsupervised semantic segmentation using limited human demonstration effort as a weak supervisory signal to refine a set of labels that can be used for terrain-aware autonomous navigation. Specifically, as many stream-based segmentation algorithms often over-segment more then they under-segment [80], the proposed methodology seeks to alleviate over-segmentation by (i) identifying and discarding labels which classify multiple semantics, while (ii) over-segmented labels are merged together and mapped directly to semantic concepts.

Specifically, this chapter focuses on enabling an autonomous vehicle to perceive environmental terrains in an open-world setting using two forms of data (i) a sequence of unlabeled images, and (ii) human demonstrations with a single semantic label identifier. Unlabeled images are collected and segmented in an online fashion, i.e., frame by frame, as the vehicle

moves throughout the environment. Human demonstrations take the form of a teleoperated trajectory over a single terrain type in the environment. The (over)-segmented images are mapped to the trajectory demonstrations, where we reason about the demonstration evidence to assign semantic terrain labels to the unsupervised segmentation output. Fig. 4.1 provides a high level overview of the proposed algorithm resulting in two main steps. In step 1, an initial unsupervised segmentation model is trained online using a stream of unlabeled images, producing an initial label set. In step 2, the label set is refined using human demonstrations, reducing the number of overall labels while directly mapping labels with traversal evidence to semantic concepts. The final output is a refined unsupervised semantic segmentation representation and label set capable of classifying unstructured terrains which may act as a perception subsystem for robotic applications. This chapter's primary contribution is the evidence-based reasoning that supports the refinement of the unsupervised output, where a set of abstract labels are refined to reduce over-segmentation.

Similar navigation based approaches estimate terrain traversability directly[70, 40]. Our work differs in that rather than estimate traversability, we obtain a perception model which classifies semantic terrains present in the environment. This type of system decouples key functionality for autonomous traversal- rather than simultaneously classify and rank traversability, by only classifying terrains, subsequent systems can focus on the costs/rewards over terrain, as seen in Section 4.7. Moreover, this perception model is not coupled to any specific path planner, and therefore any path planner which utilizes semantic occupancy grids may be used.

The remainder of the chapter is organized as follows. Section 4.4 (i) introduces the stream-based unsupervised segmentation algorithm chosen for this study, Unsupervised Semantic Scene Labeling [80], and (ii) presents a methodology to refine a large set of over and under-segmented abstract labels to a smaller set of high-level semantic labels. Sec-

tion 4.5 (i) presents a framework to simulate over-segmentation in a 3D game engine, and (ii) utilizes the framework in a set of simulated robotic experiments to demonstrate the chapter's research claim that human demonstrations act as weak supervisory input capable of alleviating over-segmentation. Section 4.6 evaluates the presented methodology's performance on a real world robotics dataset, namely, the Robot Unstructured Ground Driving (RUGD) dataset [82]. Section 4.7 evaluates autonomous navigation capability using a refined perception model obtained from the presented methodology as the reward basis for inverse reinforcement learning. And lastly, section 4.8 concludes with a brief summary and identifies opportunities for future extensions.

4.4 Methodology

Our proposed solution to provide terrain-aware perception for autonomous navigation in a weakly supervised manner is composed of two stages (depicted in Fig. 4.1). First, we identify a representation of visual concepts in the environment using stream-based unsupervised segmentation. The concepts from this stage are abstract in that they have not been aligned to a high level semantic that the autonomous vehicle can use for decision making. Second, a limited number of human demonstrations are correlated with the unsupervised abstract concepts to refine (e.g., improve over-segmentation) and align these segments to high level semantics. Details of each stages are provided in the subsequent subsections.

4.4.1 Stage 1: Stream-based Unsupervised Segmentation

The first stage of our pipeline relies on an unsupervised stream-based segmentation algorithm to learn an initial set of segments that represent abstract classes. We refer to these as abstract classes because up to this point there is no supervisory signal to map these learned segment representations to a high-level semantic label. Although any unsupervised stream-based

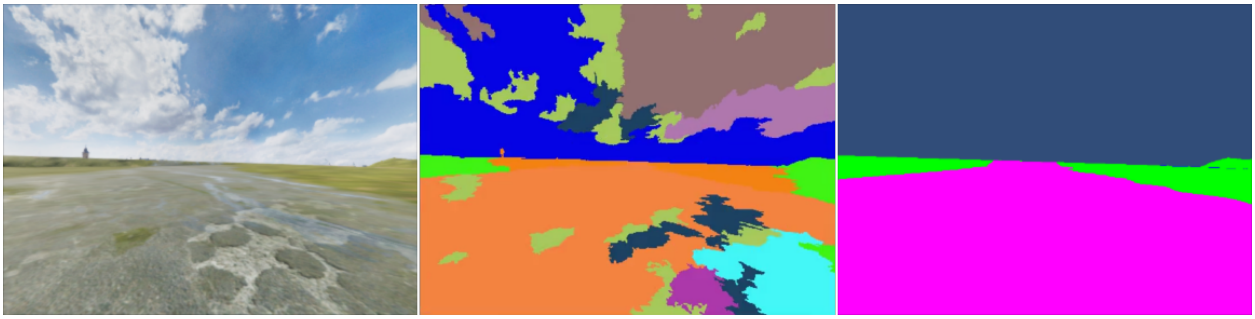

Figure 4.3: Example of oversegmentation output from USSL in a simulated unstructured outdoor environment with three classes, road, grass, and sky. Left: Raw camera image. Center: USSL model output. Right: Ground truth segmentation. Note that the segment colors between USSL and the ground truth do not correspond since the USSL output is generated from unlabeled data and is not tied to specific semantics.

segmentation algorithm could be leveraged in the first stage of our pipeline, we use the Unsupervised Semantic Scene Labeling (USSL) [80] algorithm.

USSL is an ensemble-based approach that agglomeratively clusters superpixels from images to automatically learn the number of visual concepts within a data stream. Creation of an ensemble is achieved by extracting visual features on overlapping sliding windows of the data stream, referred to as local modeling. We use a combination of visual features including, average color, color histograms, and local binary patterns. USSL encodes the segmentation evidence across the overlapping local models in a graph structure and extracts the connected components to generate a global set of abstract output labels for the entire stream, $M = m_1, m_2, ..., m_n$. Additional details on the USSL algorithm can be found in the accompanying appendix to this chapter (Section 4.9), which summarizes the original publication [80].

As previously mentioned, the output of stream-based unsupervised segmentation algorithms is often over-segmented. Fig. 4.3 shows example output from USSL. Over-segmentation

can be seen across nearly all visual classes in the image. For example, *sky* is represented by segments colored in blue, green and brown and *road* is represented by segments colored in orange, cyan, purple and navy. Now, although the over-segmentation of sky could be capturing details such as the different types of clouds, this is not relevant for the task at hand- the navigation of ground terrains. Stage two of our pipeline refines this output to produce a perception model that provides high-level semantic segmentation output similar to supervised models but without ground truth annotations.

4.4.2 Stage 2: Terrain Inference From Human Demonstration

Next, we describe how the unsupervised abstract labels, M, from stage one are refined to reduce over-segmentation and aligned to the terrains present using minimal human demonstration effort.

Algorithm Overview: We refer to the set of semantic terrains in an environment as $\mathbf{\Phi} = \{\Phi_1, \Phi_2, \dots \Phi_k\}$. These are not assumed to be known by the algorithm in stage one, but the human providing demonstrations is aware of them. Demonstrations are collected by teleoperating a robot across a single terrain, Φ_j. A demonstration is represented by the stream of images, $S_j = \{I_1, I_2, ..., I_n\}$, captured from the onboard camera during traversal and the trajectory, ξ_j, that captures the robot's poses sequentially throughout time. Via tele-operation, a set of demonstrations (minimum one per terrain required), $D = \{(S_1, \xi_1), (S_2, \xi_2) \dots (S_k, \xi_k)\}$ can be quickly collected.

Given the demonstration dataset, D, we can reason about the terrain semantics for abstract classes in M as follows: 1) Use the initial label set M to classify pixels from each image in S_j. 2) Project resulting perception information onto a map with the same reference frame as the robot's trajectory, ξ_j. 3) Accumulate the evidence of each terrain type in Φ being associated with each abstract label in M by evaluating how often each demonstration traversed through each abstract label. 4) Identify conflicting evidence caused

by noise from the unsupervised stream-based segmentation in which one label is classifying multiple semantics and discard them. 5) Map the remaining abstract labels directly to semantic terrains by assigning each abstract label to the terrain with the most traversal evidence. If multiple abstract labels map to the same semantic, merge their underlying feature representations. In the case where an abstract label does not have any traversal evidence, keep it in the representation, but do not semantically map it to any terrains. Overall, the discarding, assignment/merging, and keeping of abstract labels results in the refinement of the underlying representation, which can improve the overall perception.

Step 1: Unsupervised Segmentation & Classification: We use a simple nearest neighbor classifier to assign unsupervised labels to all pixels within the images from a demonstration's image stream. As noted previously, each label within initial label set M is represented by a N dimensional feature vector. First, a set of superpixels is generated for each image I_j within the demonstration's image stream. Then, using the same feature extractors that were used to train M, a feature vector of dimension N describing the superpixel is obtained. Lastly, each superpixel is matched to the closest label within M as measured by it's L2 (Euclidean) distance, resulting in the final unsupervised label classification.

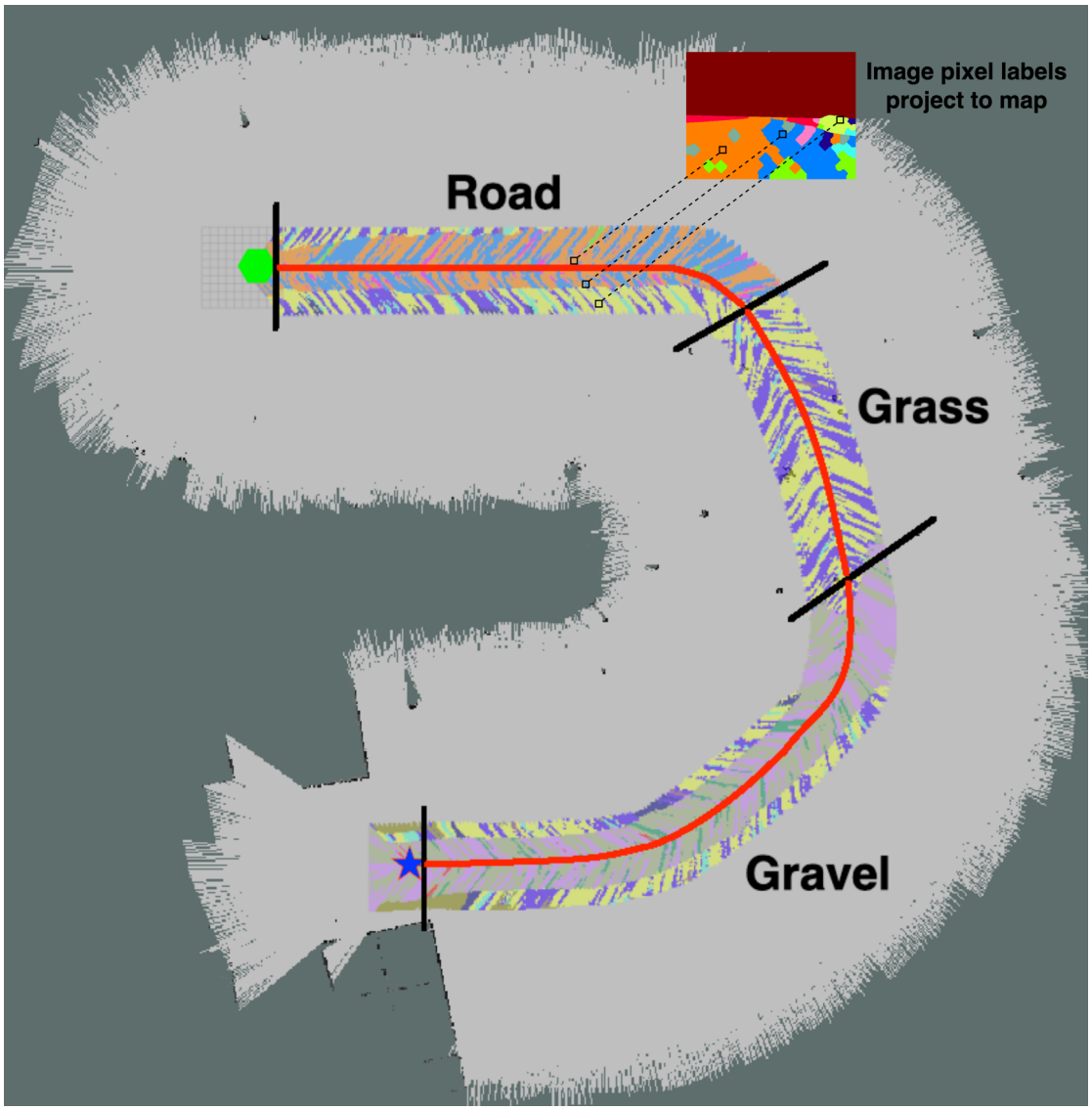

Figure 4.4: Example of simulated over-segmented terrain output. The robot starts at the green hexagon, and follows the red trajectory up until the blue star. During the lifetime of the robot's trajectory, it traverses 3 main terrains as indicated by the hatch marks- road, grass, and gravel. The sub-trajectories within each hatch mark predominately traverse over a particular set of unsupervised labels which are projected onto the robot map from the generated unsupervised segmentation images. Namely road (blue, orange), grass (yellow, dark purple), and gravel (tan, light purple, green). Traversed labels provide evidence for label refinement.

Step 2: Terrain Projection: Classified images from the demonstration data stream are then paired with their corresponding robot pose in the global reference frame of the robot's trajectory during teleoperaton and are projected onto the robot's global map (represented by an N by N grid). Fig. 4.4 shows an example of the projected perception map obtained when traversing the environment from a camera based pinhole camera model terrain projection system contained within the ROS autonomy stack. In this illustration, the robot's trajectory starts at the green hexagon in the top left of the image, follows the red path, and ends up in the bottom left of the image at the blue star. Colors around the demonstration trajectory correspond to pixels being projected from the classified images, specifically each label within the initial unsupervised label set capturing terrain features M is assigned a unique color identifier *. Following the robot trajectory, notice the black hatch marks, these are sub-trajectories where the major terrain being traversed changes- initially the robot is traversing road up until the first hatch mark, where it traverses grass up until the next hatch mark, and finally traverses gravel until the trajectory is completed. Within each sub-trajectory, the labels traversed over in the map output by the terrain projection system change. The robot is mostly traversing the two labels in M represented by orange and blue in the first region, the two labels represented by yellow and dark purple in the second, and the three labels represented by light purple, tan, and dark green in the third. Using the projected perception map alongside the robot's trajectory in the same reference frame, we can count how often trajectory ξ_j moves through each class in M and use this as evidence to reduce over-segmentation. Lastly, rather than a particular (x, y) grid cell in the projected perception map being marked as a distinct label disjoint from the set of possible labels, every grid cell may contain multiple labels. The reason for this is twofold: (i) Multiple contrasting superpixels may classify pixels which project to the same grid cell, and (ii) sequential classified frames

*Note that in the first and third regions, other colors appear on the side of the robot, this is because there is grass alongside the road and gravel in this particular operational environment.

project on a subset of grid cells with previous frames and as a result each grid cell is projected on multiple times.

Step 3: Evidence Accumulation: Grid cells within the projected map and the underlying labels the robot traversed provide a weakly supervised signal, which we refer to as traversal evidence. This evidence is used to define the context required to complete the autonomous task by: (i) identifying models using the same label to classify two distinct terrains, what we refer to as noise, and (ii) identifying labels to be merged due to over-segmentation. To accumulate terrain label evidence we count the number of times a trajectory state, s_i, intersects with the abstract classes in M. Each s_i maps to a grid cell in the projected perception map. Recall that it is possible for a grid cell to be associated with multiple labels from M so a trajectory state may provide evidence for more than one abstract label class. Thus, the evidence that abstract class, m, could represent terrain class Φ given trajectory ξ is represented as:

$$evidence(m) = \sum_{\forall s_i \in \xi} \mathbb{1}(m \cap s_i) \tag{4.1}$$

where $\mathbb{1}$ is an indicator function that determines if a label class and trajectory state fall into the same grid cell. After computing $evidence(m)$ for all labels for each trajectory in the demonstration dataset for a particular terrain Φ, each element is normalized as follows:

$$\sigma(m) = \frac{evidence(m)}{\sum_{j=1}^{|M|} evidence(m_j)} \tag{4.2}$$

yielding a value in the range $[0 - 1]$.

Step 4: Identifying Conflicting Evidence: After measuring the evidence for each unsupervised label across all terrains, certain labels may overlap. That is, certain labels may have non-zero traversal counts across multiple demonstrations across different terrains. This is referred to as label noise, labels with high amounts of overlap are considered noisy

as that label is unable to accurately predict terrains distinctly. Some reasons label noise may occur includes: poor feature extraction techniques, poor learned representations, and data complexity (highly dimensional, complex scenes, lighting and occlusion). Consequently, Since noisy labels lead to high amounts of overlap, we quantify the "noisiness" using entropy. Specifically, for each abstract label m the normalized Shannon entropy is calculated over a vector $\rho =< \sigma(m)_{\Phi_1}, \sigma(m)_{\Phi_2}, ..., \sigma(m)_{\Phi_k} >$, capturing overlap evidence across all demonstrations:

$$H(\rho) = \frac{-\sum_{m=1}^{|\rho|} \rho_m \log_2 \rho_m}{\log_2 |\rho|} \tag{4.3}$$

yielding a value in the range $[0 - 1]$. Large amounts of overlap across all terrains results in entropy values close to 1 while no overlap results in entropy values of 0, thereby quantifying how often a label is traversed across multiple terrains (semantics). Labels with high entropy are considered noisy, and such labels should be discarded from the model since they will provide bad classification results if kept. On the contrary, label noise may also simply occur due to misclassification, therefore some level of noise is to be expected. In order to balance the tradeoff between acceptable and unacceptable levels of noise, the threshold value ϵ captures the amount of acceptable risk. Therefore, the initial set of models in M are trimmed based on the specified threshold value ϵ. For each $m \in M$, if $H(\rho)$ is over ϵ the model is discarded, otherwise it is kept. The chosen value of ϵ depends on the amount of misclassification the system designers believe is acceptable, and may be determined by evaluating the models acceptable misclassification rate that still leads to overall mission success.

Step 5: Label Assignment and Terrain Representation: The remaining $m \in M$ after discarding those with high entropy which have non-zero traversal evidence are used to reason about semantic terrain label assignment. Each $m \in M$ is mapped to the terrain Φ

with the most evidence:

$$\tau(m) := \underset{\forall \Phi \in \mathbf{\Phi}}{\operatorname{argmax}} \ \ \sigma(m) \tag{4.4}$$

Finally, the set of feature vectors, $\mathcal{T}$, for all abstract labels assigned to the same terrain Φ_i are merged into a representation for the terrain. Specifically, a weighted average of the feature vectors is taken to represent this high level semantic:

$$\hat{\Phi}_i(\mathcal{T}) = \frac{\sum_{i=1}^{|\mathcal{T}|} L_2(\mathcal{T}_i, \operatorname{argmax}(T)) * \mathcal{T}_i}{\sum_{i=1}^{|\mathcal{T}|} \mathcal{T}_i} \tag{4.5}$$

Labels in M that do not contain any traversal evidence, are not mapped to a terrain and their underlying feature representation is unaltered.

After the label assignment is complete, a new set of class models exists which represents a combination of semantic terrain labels and a set of unsupervised abstract labels. Although the set of remaining unsupervised abstract labels does not provide semantic meaning for the navigation task, these learned concepts may still be valuable because they are representing something our sparse supervisory signal could not help explain. This includes semantics where it is not possible to have traversal evidence, such as the sky, or obstacles. To conclude, the resulting hybrid set of labels can then be used to produce perception output for an autonomous vehicle for terrain-aware navigation.

4.5 Evaluation - Simulation

4.5.1 Overview

This section evaluates the impact of terrain evidence obtained from human demonstrations to refine a perception representation. Namely we evaluate two main properties, (i) the ability to identify and merge over-segmented labels to produce a refined representation, and (ii) the ability to identify and discard poor representations. This is done by replacing stage 1 (Sec. 4.4.1) with a 3D graphics engine which simulates over-segmentation in the perception system. Over-segmentation is simulated agnostically by utilizing ground truth semantic segmentation images provided automatically for all camera frames in the simulator. This experiment setup enables us to test the methodology alone, without any influence of the chosen unsupervised perception algorithm. As a result, this experiment demonstrates the theoretical best case performance achievable. Thereby answering the question on whether or not human demonstrations can be used as a weak supervisory signal to refine an unsupervised perception model.

4.5.2 Perception Simulation

A simulator is constructed to capture common sources of error found in unsupervised semantic segmentation algorithms. Namely, (i) noisy labels with poor interclass performance and thereby fail to reliably classify a single semantic, (ii) over-segmented representations which fail to cluster/merge intraclass labels representing the same semantic, and (iii) misclassification errors where segments are incorrectly labeled. First the simulation's design and how each error source is simulated is presented. Followed by an outline of subsequent experiments to follow in this section.

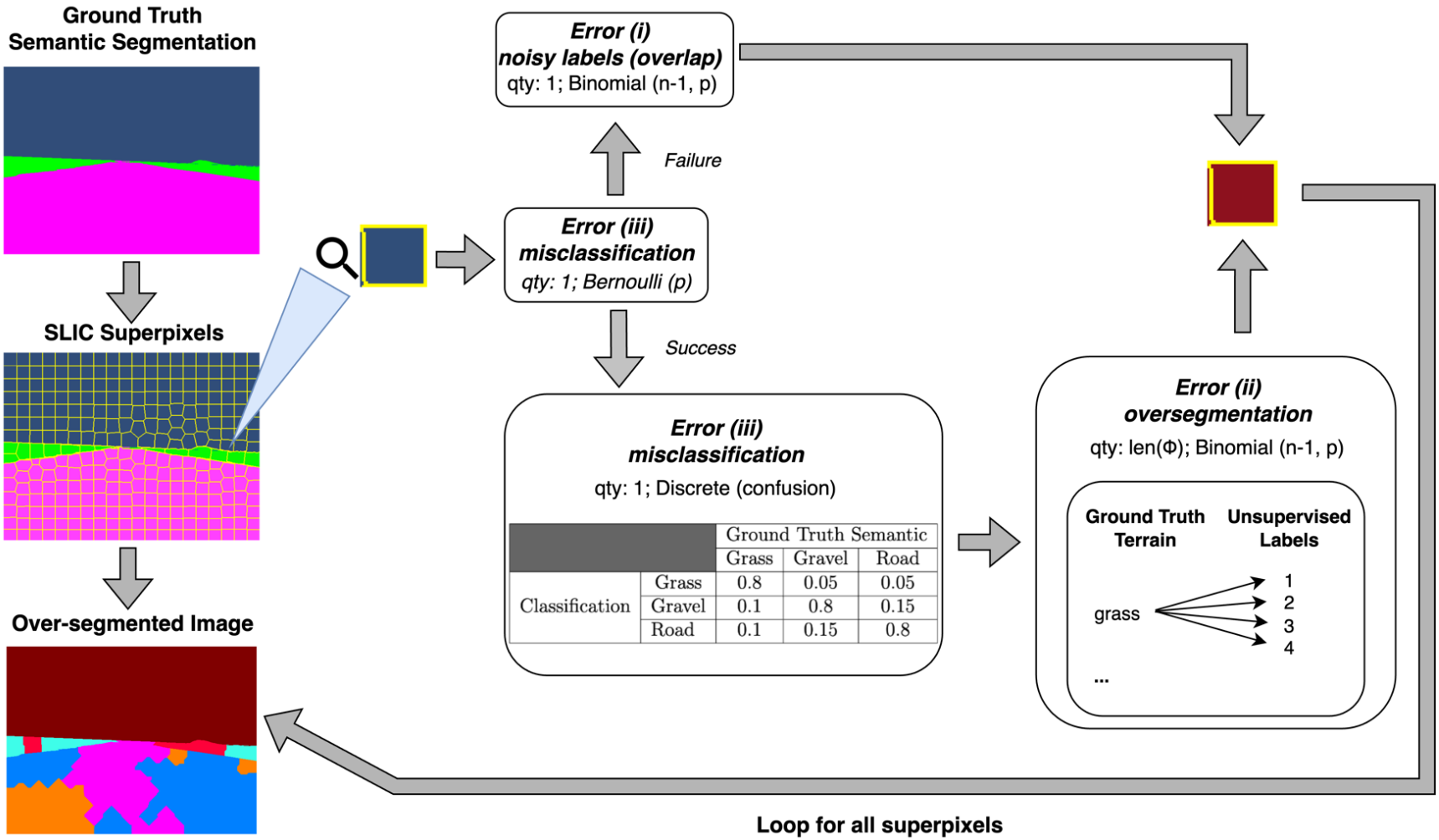

Figure 4.5: Architecture diagram of the over-segmentation simulator. Over-segmented images are generated using ground truth semantic segmentation images available in the simulation. First, the ground truth image is segmented into SLIC superpixels. Each superpixel is then assigned an unsupervised label dependent on result of the computational graph simulating three sources of error commonly found in unsupervised segmentation algorithms. The resulting stream of over-segmented images is used as a representative simulator of a stream based unsupervised segmentation model.

3D Graphics Engine: Perception simulation is enabled by the unity 3D graphics engine in which a ClearPath Warthog wheeled mobile robot is operating in a semi-structured outdoor environment. The graphics engine provides both raw image output and ground truth semantic segmentation images at 30 frames per second from the viewpoint of the robot's cameras. Specifically, the graphics engine produces ground truth images using six semantic labels- grass, gravel, road, sky, building, and unlabeled. The simulated robot runs the autonomous system software stack based on Robot Operating System (ROS) as outlined in Chapter 2.1.

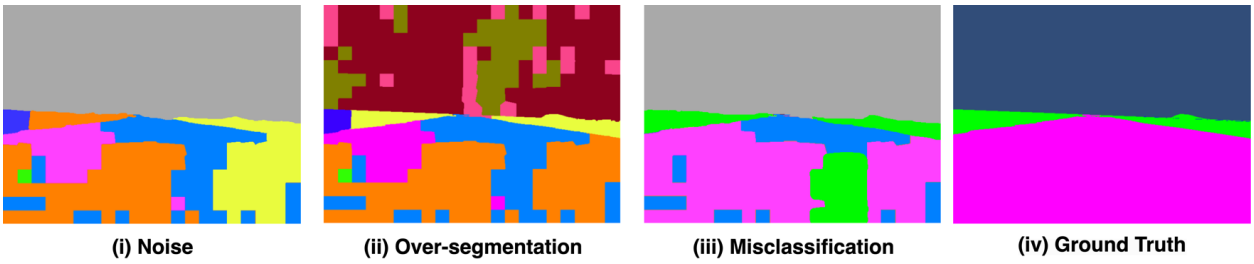

Figure 4.6: Illustration of the different sources of error commonly found in unsupervised perception algorithms alongside the ground truth (column 4). A noisy model contains label(s) classifying over multiple semantics (column 1; yellow, orange). Over-segmented models learn more labels than there are semantics (column 2; olive, burgundy). Misclassification of semantics occurs due to poor model performance (column 3; lime green).

Error (i)- noisy labels (Figure 4.6, column 1): After learning, labels may be present in the representation which contain noise. A noisy label has high interclass similarity- commonly classifying many segments spanning multiple semantic classes (see the yellow and orange labels in Figure 4.6 which are both classifying grass and road with the same label). Such labels worsen the quality of the perception system and therefore should be discarded. Although noisy labels can be thought of as misclassification, they are distinct in that rather than the label mostly classifying one class and occasionally misclassifying as others, noisy

labels commonly misclassify across all classes almost uniformly. We simulate this with a Bernoulli distribution which is sampled for each superpixel. If the trial is a success, the superpixel is classified from the set of ground truth labels. Conversely, if the trial is a failure, the superpixel is classified from a set of over-segmented labels containing noise. As a result, this formulation enables both interclass, and intraclass variability among labels.

Error (ii)- over-segmentation (Figure 4.6, column 2): Since several unsupervised algorithms are based on clustering techniques, the number of clusters (labels) often does not match the number of semantics in the scene because there is often no supervisory signal. There may be high intraclass similarity, as seen in the top half of the image in Figure 4.6 column 2 where the olive, burgundy, and dark pink labels are all classifying the same semantic, sky. If the number of labels learned is too low, it is called under-segmentation, conversely if it is too high, it is called over-segmentation. There is a tradeoff between over-segmentation and under-segmentation. Over-segmented representations provide a large, often complicated (no clear semantic meaning to humans) feature basis that explains the current operational environment. Under-segmented representations provide a small, limited feature basis which fails to distinctly classify key properties in an image. We explicitly simulate over-segmentation and focus on over-segmentation, as it is the primary problem the methodology aims to solve. Moreover, this is based on the philosophical viewpoint that it is better to have an over-segmented model and then refine upwards, leading to fewer, than the other way around. Mathematically, it is easier to merge labels together rather than create new ones.

For each ground truth semantic class, a discrete probability distribution is constructed where the number of possible outcomes is determined by the desired number of possible over-segmentation labels, denoted as the over-segmentation factor. The choice of distribution is up to the experimenter, however we chose the binomial distribution $B(n, p)$, where n represents the over-segmentation factor, and p represents the bias towards certain over-segmented labels.

Moreover, the binomial distribution can be used to create "discrete normal" distributions in which the mass of the distribution is distributed normally among the over-segmented labels. As each ground truth label is represented by its own separate binomial distribution, each may also have its own unique over-segmentation factor. Lastly, since each label is perfectly contained relevant to its parent ground truth label, high intraclass similarity occurs. That is, if a ground truth parent class has a high over-segmentation factor, there is high intraclass variability, resulting in many unsupervised labels to segment off into their own distinct groups even though they all belong to ground truth class.

Error (iii)- misclassification (Figure 4.6, column 3): It is well known that computer vision based perception algorithms are not able to perfectly classify everything. Dependent on the model architecture, and the training data used, classification performance varies greatly. Furthermore, when a model does fail to classify, the model's interclass performance is often not uniform. A model may commonly misclassify one class for another (looking at column 3 in Figure 4.6, notice how the green label incorrectly classifies road as grass). In the machine learning community, this performance is captured in a confusion matrix. A confusion matrix is a nxn matrix where n is equal to the number of possible classification labels. On one dimension is the ground truth result, and the second, the model's prediction. The diagonal of this matrix captures accuracy to correctly classify its own class while other members represent how certain labels misclassify one class for another.

This simulator takes in a confusion matrix as input to simulate misclassification. A discrete probability distribution is constructed from the confusion matrix, and is sampled in relation to the current ground truth label. In this setup, interclass variability is possible at the misclassification step, and as a result, when misclassification is paired with error ii, intraclass variability is also present.

Perception Simulation Workflow: Using ground truth semantic segmentation, the perception errors aforementioned are simulated in a bottom up manner for each image in an image stream as the robot is moving in its operational environment. A graphical overview can be seen in Figure 4.5. First, the image is segmented into a set of superpixels. Each superpixel is then assigned a final label as follows. A sample from a Bernoulli distribution $Ber(p)$ is taken. If the trial is a success, a subsequent sample is taken from a discrete probability distribution where possible outcomes of the distribution represent ground truth labels, and probabilities represent a perception models classification accuracy as defined by a confusion matrix (error iii). Lastly a sample is taken from the ground truth label's binomial distribution $B(n, p)$ to obtain the final unsupervised label (error ii). Note that the final labels are unsupervised, and therefore do not have any semantic meaning. If the Bernoulli trial is a failure, a sample is taken from a discrete uniform distribution, where its members represent the set of labels classifying multiple semantics (error i).

Experiment Framework: The following framework is applied to all subsequent experiments in this section, differing only on the parameters of the perception simulator. First, an initial sequence of images are collected where the simulated robot traverses each major semantic terrain present in the operational environment- grass, gravel, and road. Specifically, subsequent experiments use a ROS bag containing 1646 sequential images over 53 seconds (an overview of the robot's trajectory can be seen in Figure 4.4). Since this data is collected in a graphics engine, both raw images and their underlying ground truth is collected. Then, using the perception simulator, every ground truth image is over-segmented according to Figure 4.5. The resulting images are fed back into the graphics engine alongside a simulated autonomous system ROS software stack.

As previously mentioned in Section 4.4.2, part of this stack contains a terrain projection system which outputs a series of occupancy grid terrain maps from semantic images. The

resulting terrain maps are used as the traversal evidence acting as the input to Equation 4.1. Following through with the rest of the subsequent methodology, a refined representation is provided as output. The refined representation is then evaluated.

4.5.3 Experiment 1: Capability to identify and merge labels

In the first experiment, the methodology is evaluated on its ability to utilize traversal evidence to refine an initial perception representation and attempt to alleviate error (ii); over-segmentation. For this experiment, only over-segmentation is simulated, and therefore noisy labels (error i), and misclassification (error iii) are not considered. This setup is chosen to evaluate the theoretical maximum performance of the proposed methodology. If (i) the set of class labels produced by the over-segmented model produces perfect segment boundaries, and (ii) each of its labels never misclassifies with respect to the ground truth, then by utilizing resulting traversal evidence in the proposed methodology, the resulting representation should match the ground truth exactly. Now, although the assumptions are unrealistic for real-world perception models, this experiment verifies if human traversal evidence can produce a refined label set which maps to semantics present in the scene.

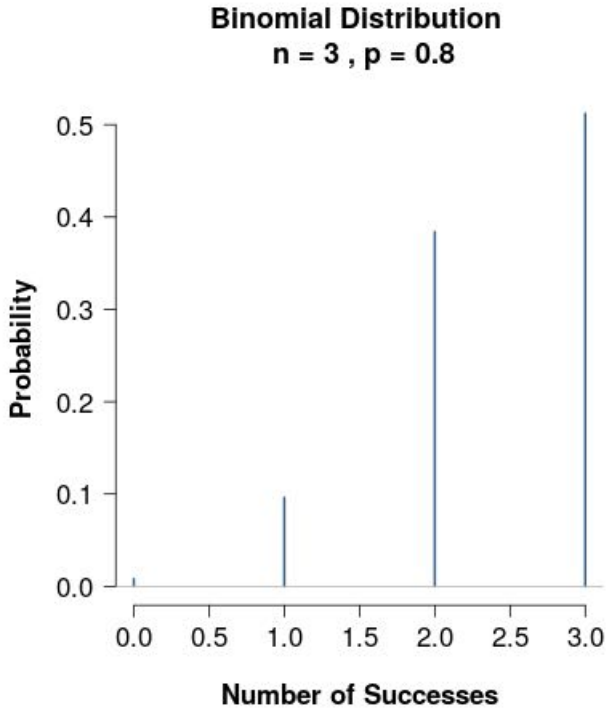

Figure 4.7: PMF of a Binomial distributon where $n = 3$ and $p = 0.8$

The six semantic classes- grass, gravel, road, sky, building, and unlabeled- are each assigned a unique probability distribution for over-segmentation. Namely, each terrain (grass, gravel, road) is given an over-segmentation factor of 4, while all other labels are given a value of 1. Note that non-terrains are not given an over-segmentation factor because the proposed methodology does not make use of that information when performing merging. Each terrain follows a binomial distribution $B(n, p)$ where $n = 3$ and $p = 0.8$, a visualization is depicted in figure 4.7. This choice of p results in a biased distribution where the 2 of the 4 labels take up most of the probability mass. This corresponds to real world behavior in relation to unsupervised semantic segmentation models, where certain labels occur more frequently than others when presented with the same semantic.

Experimental Results:

Traversal evidence for a given abstract label relative to the possible parent terrains is shown in Table 4.1.

Table 4.1: Experiment 1. Traversal Evidence (feature counts)

Label	**Terrain**		
	Grass	**Gravel**	**Road**
1	0	0	**4**
2	0	0	**8**
3	0	0	**108**
4	0	0	**152**
5	0	**13**	0
6	0	**51**	0
7	0	**91**	0
8	0	**140**	0
9	**17**	0	0
10	**103**	0	0
11	**146**	0	0
12	**191**	0	0
13	0	0	0
14	0	0	0
15	0	0	0

In this setup, labels 1-4 correspond to grass, 5-8 correspond to gravel, 9-12 correspond to road, 13 to sky, 14 to building, and 15 to unlabeled. Figure 4.8 illustrates the robot's point of view with the future trajectory overlayed. As the robot moves through space on its trajectory over the road, it traverses the blue, dark blue, and orange pixels, thereby collecting traversal evidence for labels 10, 11, and 12. This traversal evidence informs label merging.

Figure 4.8: Experiment 1. Column 1: Raw image; Column 2: Illustration of the robot's unsupervised over-segmented image with it's future trajectory (red) as it traverses the road (orange, blue, dark blue). Column 3: Ground truth image.

As per the methodology, once traversal evidence is obtained, first the entropy for each abstract label is calculated, and used to determine if any labels should be discarded, based on the threshold parameter $\epsilon = 0.8$. Since noisy labels (error i) and misclassification error (error iii) are not present, there is no overlap (divergent evidence) among the over-segmented labels in the resulting traversal evidence.

Table 4.2: Experiment 1. Abstract Label Entropy (1-8)

	Abstract Label							
	1	2	3	4	5	6	7	8
Entropy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 4.3: Experiment 1. Abstract Label Entropy (9-15)

	Abstract Label						
	9	10	11	12	13	14	15
Entropy	0.00	0.00	0.00	0.00	UNDEF	UNDEF	UNDEF

Table 4.2 and 4.3 shows entropy values (Equation 4.3) for each abstract label. Labels 1-12, which all correspond to terrains, have an entropy of 0.00, while 13-15 have undefined entropies. These values are expected due to the parameters set for the over-segmentation simulator, namely, there is no overlap since errors (i) and (iii) are not simulated. Moreover, note that a value of undefined for the entropy score means that no traversal evidence was present for a particular label, which is consistent as 13-15 represent sky, obstacle, and unlabeled. As a result of these entropy scores, no labels are discarded.

Next, each abstract label is mapped to the parent terrain for which it shares the most evidence with. Utilizing the traversal evidence for each abstract label as seen in Table 4.1, remaining labels are merged, resulting in a refined representation.

Table 4.4: Experiment 1. Quantity of labels

	Ground Truth Class		
	Grass	*Gravel*	*Road*
Initial	4	4	4
Discarded	0	0	0
Semantically Mapped	**4**	**4**	**4**
Not Semantically Mapped	0	0	0
Total Label Quantity	**1**	**1**	**1**

As seen in Table 4.4 initially, the over-segmented model starts with 12 terrain labels- 4 for grass, 4 for gravel, 4 for road. After utilizing traversal evidence and the proposed methodology, the refined model ends up with 3 labels, namely, 1 for each terrain semantic. This quantitatively verifies that by using the proposed methodology, human demonstrations do act as a supervisory signal which can be used to refine a over-segmented perception model. Moreover, initially labels were not mapped to any specific semantic class, giving them no semantic meaning. However, after utilizing the proposed methodology, each label is mapped directly to a semantic class. For example, before label 1 did not mean anything to a human, now it refers to road.

In Figure 4.9 the ground truth, over-segmented image, and refined image are stacked horizontally for the first frame in the same trajectory as shown above in Figure 4.4. Due to the parameters of the perception simulator, all segment boundaries are correct for both the over-segmented and refined models. The difference lies in the quantity of labels and final classification of segments; the over-segmented model has a higher number of labels and segments.

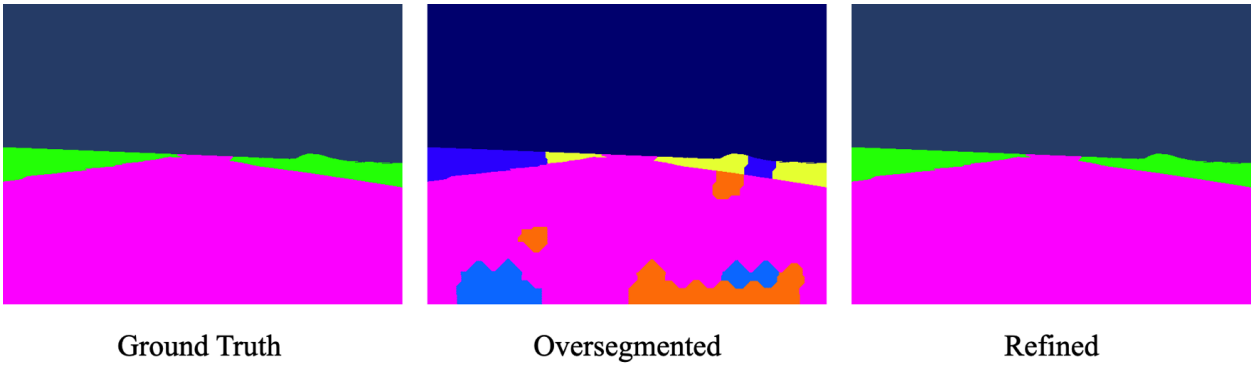

Figure 4.9: Experiment 1. Left: Ground truth. Middle: Initial over-segmented model. Right: Refined model.

Looking at the frame from the over-segmented model (Figure 4.9 column 2), the addition of extra segments and labels does not provide anymore information about the scene. As a result such labels should be merged to provide a more meaningful representation. Note that although each semantic has an over-segmentation factor of 4, not all 4 are present in every single frame- this is a result of the Bernoulli distribution chosen for over-segmentation where 2 out of the 4 labels take up the majority of the probability mass. Using the methodology for refinement results in a representation that exactly follows the ground truth (Figure 4.9 column 3). This qualatatively demonstrates that human demonstrations can be used as a weak supervisory signal to refine and merge similar labels contained within unsupervised over-segmented model while simultaneously giving the resulting labels semantic meaning.

4.5.4 Experiment 2: Capability to identify and discard poor labels

In the second experiment, the methodology is evaluated on its ability to utilize traversal evidence to refine an initial perception representation and attempt to alleviate error (i); noisy labels. For this experiment, over-segmentation (error ii) and noisy labels are simulated (error i), and therefore misclassification (error iii) is not considered. As in the previous experiment, segmentation boundaries are perfect. When a poor label is sampled in the first step of the perception simulator, that label has both inter and intraclass variability. When a good label is sampled, the label only has intraclass variability respective to its ground truth class. This setup is chosen to evaluate two aspects of the methodology. Namely, (i) if noisy labels with high interclass similarity (single labels classifying multiple semantics) are identified and discarded while keeping the ones with low interclass similarity (single labels classifying one semantic), and (ii) if the remaining label set after discarding merges to a meaningful representation.

The initial Bernoulli flip is given a success probability of 0.8, therefore the poor representations should be selected roughly 20% of the time. A secondary Bernoulli distribution is used to sample 2 poor labels, with a success probability of 0.5 such that each poor label is equally likely to occur.

The six semantic classes- grass, gravel, road, sky, building, and unlabeled- are each assigned a unique probability distribution for over-segmentation. Namely, each terrain (grass, gravel, road) is given an over-segmentation factor of 4, while all other labels are given a value of 1. Once again, non-terrains are not given an over-segmentation factor because the proposed methodology does not make use of that information when performing merging. Each terrain follows a binomial distribution $B(n, p)$ where $n = 3$ and $p = 0.8$. This configuration results in 17 possible labels, where 1-11 represent terrain 12-15 represent non-terrain and 16-17 represent noisy labels.

Table 4.5: Experiment 2. Traversal Evidence (feature counts)

Label	**Terrain**		
	Gravel	**Grass**	**Road**
1	0	0	**3**
2	0	0	**14**
3	0	0	**54**
4	0	0	**95**
5	0	**4**	0
6	0	**13**	0
7	0	**36**	0
8	0	**58**	0
9	**3**	0	0
10	**24**	0	0
11	**50**	0	0
12	**68**	0	0
13	0	0	0
14	0	0	0
15	0	0	0.0
16	48	23	50
17	75	25	59

Traversal evidence for a given abstract label relative to the possible parent terrains is shown in Table 4.5. Figure 4.10 illustrates the robot's point of view with the future trajectory overlayed. As the robot moves through space on its trajectory over the road, it traverses the blue, dark blue, and green pixels. However, also note that the dark blue and green pixels appear on both the road and in the grass, therefore creating conflicting evidence for labels 16 and 17. This is an example of label noise, signaling that such labels should be thrown out if their entropy is above the accepted threshold.

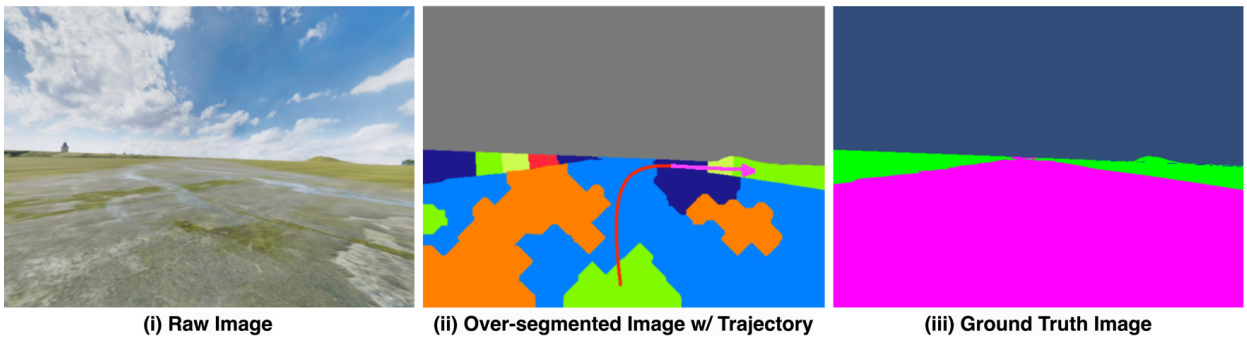

Figure 4.10: Experiment 2. Column 1: Raw image; Column 2: Illustration of the robot's unsupervised over-segmented image with it's future trajectory as it traverses the road (trajectory: red; pixels:blue, dark blue, green) and moves to the grass (trajectory: pink; pixels: dark blue, yellow, green), thereby creating conflicting evidence (pixels: green, dark blue). Column 3: Ground truth image.

As per the methodology, once traversal evidence is obtained, first the entropy for each abstract label is calculated, and used to determine if any labels should be discarded based on the threshold parameter $\epsilon = 0.8$.

Table 4.6: Experiment 2. Abstract Label Entropy (1-8)

	Abstract Label							
	1	2	3	4	5	6	7	8
Entropy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 4.7: Experiment 2. Abstract Label Entropy (9-17)

	Abstract Label								
	9	10	11	12	13	14	15	16	17
Entropy	0.00	0.00	0.00	0.00	UNDEF	UNDEF	UNDEF	**0.97**	**0.99**

Table 4.6 and 4.7 shows entropy values (Equation 4.3) for each abstract label. Once again, labels 1-12, which all correspond to terrains, have an entropy of 0.00, while 13-15 have undefined entropies. Distinct to this experiment, labels 16 and 17 have high entropy values. Specifically, since they are higher than $\epsilon = 0.8$, they are discarded.

Next, each abstract label is mapped to the parent terrain for which it shares the most evidence with. Utilizing the traversal evidence for each abstract label as seen in Table 4.5, remaining labels are merged, resulting in a refined representation.

Table 4.8: Experiment 2. Quantity of labels

	Ground Truth Class			
	Grass	*Gravel*	*Road*	Noisy
Initial	4	4	4	2
Discarded	0	0	0	**2**
Semantically Mapped	**4**	**4**	**4**	0
Not Semantically Mapped	0	0	0	0
Total Label Quantity	**1**	**1**	**1**	**0**

As seen in Table 4.8 initially, the over-segmented model starts with 14 terrain labels- 4 for grass, 4 for gravel, 4 for road, and 2 noisy labels. After utilizing traversal evidence and the proposed methodology, the refined model ends up with 3 labels, namely, 1 for each terrain semantic. This verifies that by using the proposed methodology, (i) poor labels were identified and discarded, and (ii) of the remaining labels, accurate label merging occurred.

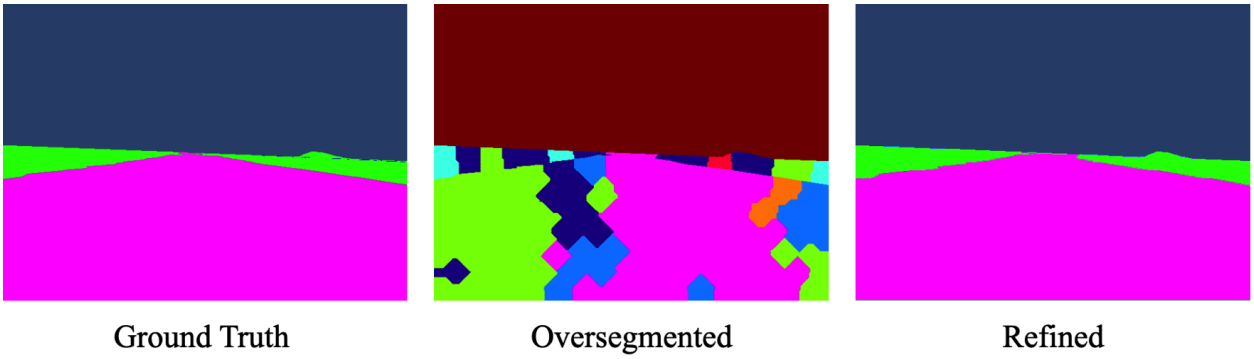

Figure 4.11: Experiment 2. Left: Ground truth. Middle: Initial over-segmented model with noisy labels. Right: Refined model.

In Figure 4.11 the ground truth, over-segmented image, and refined image are stacked horizontally for the first frame in the same trajectory as shown above in Figure 4.4. Once again, due to the parameters of the perception simulator, all segment boundaries are correct for both the initial over-segmented model and the subsequent refined model. The over-segmented model not only has a higher number of labels and segments, but also contains labels with poor interclass accuracy. In the over-segmented model, two labels (dark blue, lime green) are used to represent two different semantic concepts, grass and road, and therefore should be discarded. Using the methodology for refinement in the simulation without any misclassification results in a representation that is exactly like the ground truth. This further verifies that with the use of human demonstrations as a weak supervisory signal (i) poor labels can identified and discarded, and (ii) of the remaining labels, accurate label merging occurs.

4.5.5 Experiment 3: Robustness to All Noise Sources

In the third experiment, the methodology is evaluated on its ability to utilize traversal evidence to refine an initial perception representation and attempt to alleviate errors- (i); noisy labels and (ii) over-segmentation while subject to error (iii) misclassification. This experimental setup is chosen to demonstrate real world applicability.

Table 4.9: Confusion Matrix

		Ground Truth Semantic		
		Grass	Gravel	Road
Classification	Grass	0.8	0.05	0.05
	Gravel	0.1	0.8	0.15
	Road	0.1	0.15	0.8

The perception simulator's initial Bernoulli distribution has the value $p = 0.8$, meaning that poor labels with inter and intra class variability are chosen roughly 20% of the time. A separate Bernoulli with $p = 0.5$ is constructed to represent 2 poor labels. This distribution is sampled if the previous Bernoulli distribution draws a failure. If the previous Bernoulli is a success, a sample is taken from a discrete distribution based on confusion matrix seen in Table 4.9 to obtain the parent ground truth class. Given the parent ground truth class, a sample from its respective binomial distribution is taken where $n = 3$ and $p = 0.8$ to obtain the final over-segmented label. The initial model then utilizes the methodology in Section 4.4 to obtain a refined model.

Table 4.10: Experiment 3. Traversal Evidence (feature counts)

Label	Terrain		
	Gravel	Grass	Road
1	0	0	**4**
2	0	0	**6**
3	6	4	**54**
4	19	2	**75**
5	0	**0**	0
6	3	**10**	3
7	0	**16**	5
8	4	**45**	4
9	**0**	0	0
10	**21**	0	0
11	**59**	6	12
12	**82**	13	15
13	0	0	0
14	0	0	0
15	0	0	0
16	32	34	50
17	40	24	44

Traversal evidence for a given abstract label relative to the possible parent terrains is shown in Table 4.10. Specific to this experiment, note that many labels overlap (orange) a given parent terrain due to misclassification being simulated. As per the methodology, once traversal evidence is obtained, first the entropy for each abstract label is calculated, and used to determine if any labels should be discarded based on the threshold parameter $\epsilon = 0.8$.

Table 4.11: Experiment 3. Abstract Label Entropy (1-8)

	Abstract Label							
	1	2	3	4	5	6	7	8
Entropy	0.00	0.00	0.57	0.58	UNDEF	0.68	0.39	0.34

Table 4.12: Experiment 3. Abstract Label Entropy (9-17)

	Abstract Label								
	9	10	11	12	13	14	15	16	17
Entropy	UNDEF	0.00	0.70	0.76	UNDEF	UNDEF	UNDEF	**0.97**	**0.99**

Table 4.11 and 4.12 shows entropy values (Equation 4.3) for each abstract label. Labels 3,4,6,7,8,11,12,16,17 have non-zero entropy values, 13-15 have undefined entropies due to no traversal evidence, and labels 16 and 17 have high entropy values. Specifically, since labels 16 and 17 are higher than $\epsilon = 0.8$, they are discarded while the remaining are kept. Note that labels 6, 11, and 12 have relatively high entropy values, if the risk acceptance

level was lowered to $\epsilon = 0.6$ these labels would be discarded, showing the importance of accurately assessing and accepting the appropriate level of overlap among demonstrations. Furthermore, note that the non-zero entropy values are due to overlap occurring from the simulated misclassification.

Next, each abstract label is mapped to the parent terrain for which it shares the most evidence with. Utilizing the traversal evidence for each abstract label as seen in Table 4.10, remaining labels are merged, resulting in a refined representation.

Table 4.13: Experiment 3. Quantity of labels

	Ground Truth Class			
	Grass	*Gravel*	*Road*	Noisy
Initial	4	4	4	2
Discarded	0	0	0	**2**
Semantically Mapped	**3**	**3**	**4**	0
Not Semantically Mapped	1	1	0	0
Total Label Quantity	**2**	**2**	**1**	**0**

As seen in Table 4.13 initially, the over-segmented model starts with 15 terrain labels- 4 for grass, 4 for gravel, 4 for road, and 2 unkowns. After utilizing traversal evidence and the proposed methodology, the refined model ends up with 5 labels, namely, 2 for grass, 2 for gravel, and 1 for road. With respect to the 2 labels which contain terrain but are not semantically mapped- this result occurs due to the low probability each label has of appearing due to it's binomial distribution. Specifically, given the binomial distribution with parameters $n = 3$ and $p = 0.8$ the probability of obtaining 0 successes is 0.008. Therefore,

these labels occur rarely in the resulting robot's map and are never traversed. As a result, these labels are never directly mapped to a terrain. In the real world, this behavior may occur as labels that are learned and appear in the initial representation, may not transfer and be relevant to the new current operational environment. Most importantly, utilizing the methodology, the refined perception model is still able to accurately discard, merge, and semantically map the overall label set. All while being subjected to all three sources of unsupervised perception error.

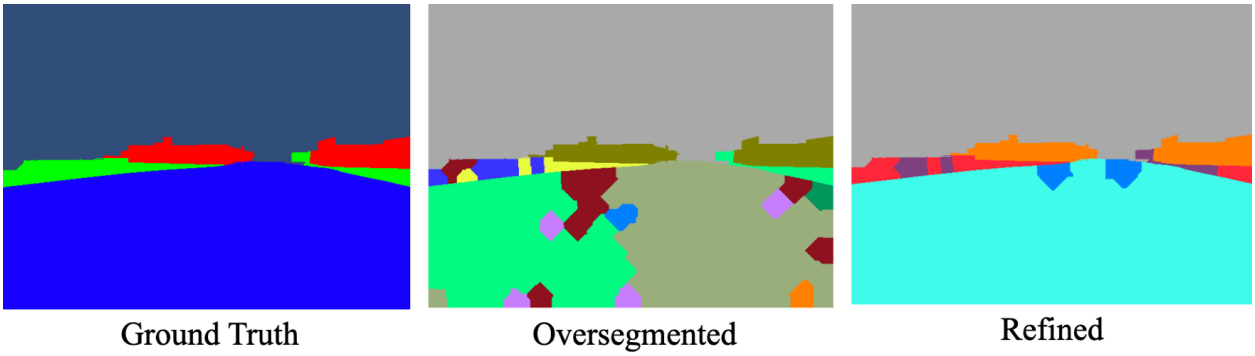

Figure 4.12: Experiment 3. Left: Ground truth. Middle: Initial over-segmented model with noisy labels. Right: Refined model.

In Figure 4.12 the ground truth, over-segmented image, and refined image are stacked horizontally for the $1250th$ frame in the same trajectory as shown above in Figure 4.4. This frame was chosen to show how the refined model does not perfectly match the ground truth. Using the methodology for refinement results in a representation that mostly matches the ground truth, except it can be seen in column 3 that grass and gravel remain slightly over-segmented. This is because two of the unsupervised labels, namely label 5 corresponding to grass and 9 to gravel have not been semantically mapped. While the two poor labels were identified and discarded, not all of the remaining labels were able to be semantically

mapped due to not having any traversal evidence. Therefore, it can be concluded that when considering all types of unsupervised segmentation model noise, one may still obtain a refined model, however there is a limit to the level of weak supervision human demonstrations can provide.

4.5.6 Summary of Findings

The previous experiments in this section result in the following takeaways- namely the proposed methodology has the capability to: (i) identify noisy labels and discard them, and (ii) identify labels with high intraclass similarity and merge them together. Furthermore, initial labels provided by the perception model did not contain any semantic meaning as they were obtained using unsupervised data. Initial labels mathematically represented learned features of the operational environment as vectors, which does not mean anything to a human. After utilizing human demonstrations, a refined label set is provided that maps unsupervised labels to semantics present in the scene which are understandable by a human.

4.6 Evaluation - Real World Robotics Dataset

This section evaluates the proposed methodology's ability to accurately refine a perception representation tailored for terrain-aware robotic navigation in real world unstructured environments. We use the Robot Unstructured Ground Driving Dataset (RUGD) [82]- this dataset contains video sequences of a teleoperated ground robot traversing various unstructured terrains. Specifically, we use the "Trail-3" ROS bag to train an initial unsupervised semantic segmentation model and then perform terrain inference using the bag's trajectory as a human demonstration. While RUGD provides ground truth labeled frames, it is only used during evaluation- and never used during training.

4.6.1 Evaluation Metrics

We evaluate each experiment by measuring the 3D segmentation accuracy [88], over-segmentation entropy, and under-segmentation entropy [31]. Traditional scores in supervised semantic segmentation, such as Mean IoU are not sufficient since initial models are never directly mapped to semantic concepts, and neither are labels in the refined models.

3D Segmentation Accuracy: While pixel-level accuracy represents the *area* of correctly classified pixels for a given frame, supervoxel-level accuracy represents the *volume* of correctly classified pixels over a sequence of frames. For a given image-stream, each ground truth label has a corresponding supervoxel T_i, thereby producing the set of supervoxels T. Similarly, each abstract label output from the unsupervised segmentation algorithm has a corresponding supervoxel S_i, producing the set of voxels S. For each T_i, a secondary set $\bar{S}$ is constructed quantifying which abstract labels overlap the most with the current ground

truth label according to the following rule:

$$\begin{aligned} \forall S_j \in S; S_j \in \bar{S} \iff |S_j| \cap |T_i| > |S_j| \cap |T_k| \\ : \forall T_k \in T \wedge T_k \neq T_i \end{aligned} \tag{4.6}$$

From the aforementioned sets, the 3D segmentation accuracy is measured as:

$$\text{ACC}(T_i) = \frac{\sum_{j=1}^{|\bar{S}|} |V(T_i)| \cap |V(\bar{S}_j)|}{|V(T_i)|} \tag{4.7}$$

where the volume $V(\cdot)$ of a given supervoxel G_i, namely $V(G_i)$, is computed as the proportion of pixels belonging to the given supervoxel and the total number of pixels in the image-stream.

Over and Under Segmentation Entropy: 3D Segmentation accuracy in of itself is not a sufficient metric, as an algorithm may obtain a low measure (0.0) if all voxels mostly overlap with a given T_i or a high measure (1.0) with many small voxels. Over-segmentation may provides a deeper explanation of an image, but often catches details that have high intraclass similarity, often complicating scene understanding and resulting in a large number of output classes. In contrast, under-segmentation may provide a simple explanation, but does not provide enough information about the relevant children concepts relevant to the task. Therefore, we consider two additional measures, over-segmentation entropy and under-segmentation entropy. As setup, probability distributions over pixels and voxels are constructed. The probability a pixel in stream D is a member of a given supervoxel G_i is measured as:

$$P(G = G_i) = \frac{|V(G_i)|}{|V(D)|} \tag{4.8}$$

The joint probability of two voxels T_i and S_j is:

$$P(T = T_i, S = S_j) = \frac{|V(T_i)| \cap |V(S_j)|}{|V(D)|} \tag{4.9}$$

The conditional probability of two voxels T_i and S_j is:

$$P(T = T_i|S = S_j) = \frac{P(T = T_i, S = S_j)}{P(S = S_j)} \tag{4.10}$$

Using these distributions, the over-segmentation entropy is measured as:

$$\mathcal{H}(S|T) = -\sum_{S,T} P(T, S) \log P(S|T) \tag{4.11}$$

Similarly, the under-segmentation entropy is measured as:

$$\mathcal{H}(T|S) = -\sum_{S,T} P(T, S) \log P(T|S) \tag{4.12}$$

Where lower values of over and under segmentation entropy indicate better performance.

4.6.2 Quantitative Results

We use the Robot Unstructured Ground Driving Dataset (RUGD)- this dataset contains video sequences of a teleoperated ground robot traversing various unstructured terrains. The frames that make the video sequences are used to train an initial unsupervised semantic segmentation model, while the teleoperated demonstration is used to provide evidence for label refinement. Specifically, an initial USSL model using a combination of color, and local binary pattern [79] features is trained on the Trail-3 dataset. Demonstration evidence in the corresponding ROS bag is split across the four main terrains present in the dataset, resulting in four distinct trajectories, thereby constructing Φ. Namely, *asphalt* is traversed over in images $0 - 55$, *mulch* from $121 - 210$, *gravel* from $341 - 440$ and *grass* from $552 - 580$. These subsets of images relative to the robot's global map reference frame are fed into the aforementioned terrain projection system to obtain terrain occupancy grids which are used as

label evidence. Using the label evidence according to the proposed methodology (Sec. 4.4), we obtain 3 distinct USSL models varied by their entropy acceptance threshold ϵ, namely $\epsilon = 0.8$, $\epsilon = 0.4$, and $\epsilon = 0.2$- each resulting in subsequent smaller label sets M. These values were chosen to show the impact ϵ has on the final label set.

Table 4.14: Number of USSL Labels

	USSL Model			
	Original	Refined: $\epsilon = 0.8$	Refined: $\epsilon = 0.4$	Refined: $\epsilon = 0.2$
# Labels	44	24	9	**7**

Label refinement reduces the overall label set: Using the proposed methodology, all refined models obtain a smaller label set M compared to the initial number of labels (Table 4.14). Once again showcasing that human demonstrations paired with image projection do in fact provide a weak supervisory signal that can be used to refine an unsupervised semantic segmentation model's initial over-segmented label set.

Table 4.15: 3D Segmentation Accuracy

		USSL Model			
		Original	Refined $\epsilon = 0.8$	Refined $\epsilon = 0.4$	Refined $\epsilon = 0.2$
Terrain	Mulch	31.21	**88.13**	45.86	50.96
	Grass	58.91	**80.22**	37.80	73.89
	Asphalt	17.79	8.64	45.26	**46.38**
	Gravel	59.71	45.62	**63.66**	0.00

Label refinement leads to higher accuracy: The best segmentation accuracy for

each respective terrain occurs in one of the refined models. Moreover, the refined model with $\epsilon = 0.4$, has the lowest range among accuracy scores, indicating that it is able to classify all terrains present most reliably. Note the 0.00 value for gravel for the refined model with $\epsilon = 0.2$, this indicates that no label supervoxel mostly overlapped with the gravel ground truth label. This is a sign that the label set has been refined too much, therefore there is a limit to how much merging can be done before the representation becomes unable to adequately explain its operating environment.

Table 4.16: Under-segmentation and over-segmentation

		USSL Model			
		Original	Refined $\epsilon = 0.8$	Refined $\epsilon = 0.4$	Refined $\epsilon = 0.2$
Metric	USE	6.252	6.391	4.523	**3.986**
	OSE	1.334	1.227	1.000	**0.987**

Label refinement provides more accurate segmentation boundaries: As ϵ decreases, and therefore the number of labels, as does the under and over segmentation entropy. This indicates that the segment boundaries obtained from the refined models are more accurate across the image-stream with respect to the ground truth labels. Although $\epsilon = 0.2$ achieves the best score, that does not make it the best model since it failed to accurately identify all terrains as demonstrated in Table 4.15. The chosen value for ϵ depends on the system designers risk acceptance, and the tradeoff between having an under-segmented or over-segmented model. If a higher level of risk acceptance is allowed (epsilon increases) the refined model will contain a higher number of labels with the tradeoff of being over-segmented compared to when a low level of risk acceptance is allowed, resulting in less labels and under-segmentation.

4.6.3 Qualitative Results

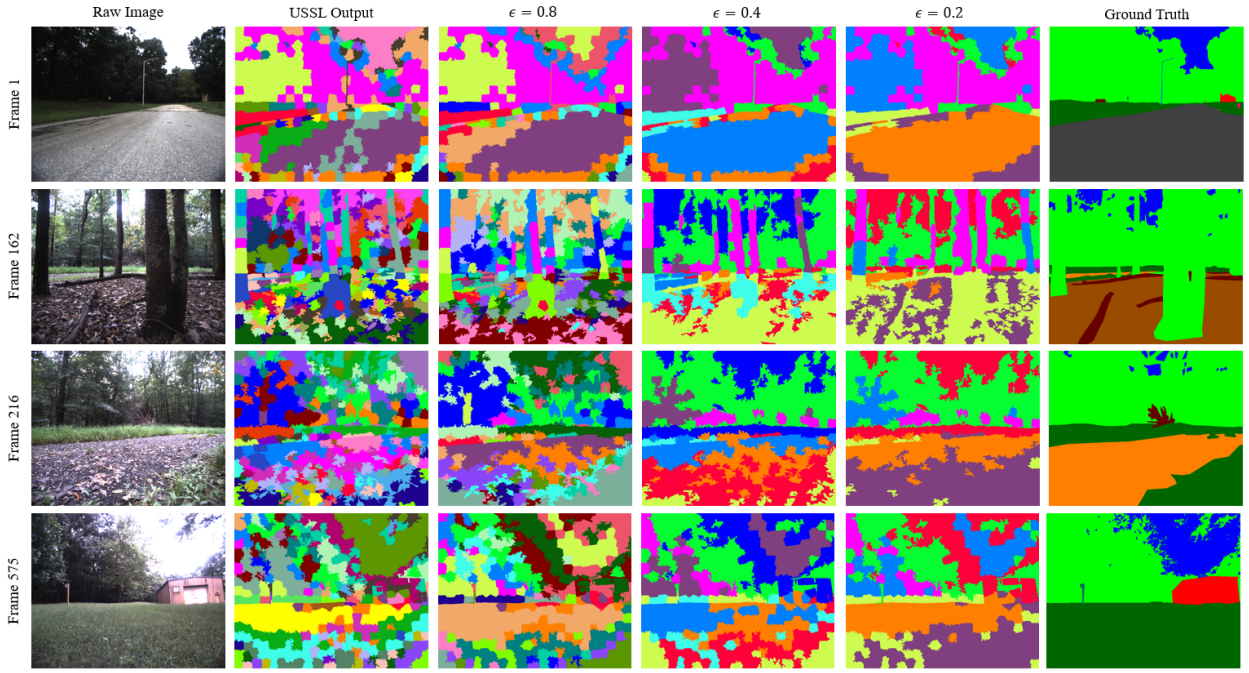

Figure 4.13: Qualitative segmentation results at frames where each main terrain appears. Rows: Segmentation output among models compared to the raw image and ground truth. Columns: Model unsupervised segmentation output across frames.

A qualitative comparison of the segmentation results produced by the original USSL model and our refined model visually shows reduced over-segmentation and improved label consistency (Fig. 4.13). Progressing through the columns for each row, we see considerably less over-segmentation as ϵ decreases. This establishes the proposed methodology's ability to merge abstract classes together using human demonstrations.

Moreover, there is a limit on ϵ before it leads to undesired segmentation, in column 5 the same label is being used to classify the road, gravel, and portions of the grass. In order to decide the final refined model to be chosen, there is a tradeoff between over and under segmentation. To elaborate, the acceptance of an over or under segmented final label set is

dependent on the type of autonomous behavior one would like to obtain. If system designers simply want to distinctly classify traversable terrains from the un-traversable terrain such as the horizon line between terrain and sky they may want to chose a lower risk acceptance as it results in fewer labels, and having more perception information does not help the overall task. Contrarily, if system designers would like to autonomously navigate with a clear preference over terrains, and terrains sub-types, such as the different types of road, grass, and gravel, they may want to chose a high risk acceptance as it results in a higher number of labels, which although may be noisy, may enable the autonomous capability desired.

4.7 Evaluation - Autonomous Waypoint Navigation

As previously hinted, the combination of inverse reinforcement learning and unsupervised semantic segmentation enables autonomous waypoint navigation using only unsupervised data, and human demonstrations. After an initial unsupervised semantic segmentation model is trained and refined, the resulting representation may be used as a reward basis for autonomous traversal. This differs from the work in Chapter 3 which required a *supervised* perception system.

4.7.1 Experiment Setup

This experiment utilizes the refined model obtained in Experiment 3 (Section 4.5.5) as the reward basis for inverse reinforcement learning. Specifically, the labels present in the refined model are used for parameter w in Chapter 3, Equation 3.3 in inverse reinforcement learning. Two reward models are trained utilizing 3 human demonstrations where the demonstrator stays mostly on the road, traverses grass when there is no other option, and avoids gravel. The first reward model uses maximum entropy inverse reinforcement learning, while the other uses risk averse Bayesian reward learning. Their relative performance to perform

autonomous waypoint navigation in an unstructured environment via a 3D graphics engine is then evaluated.

4.7.2 Experimental Results

Using this representation, a reward model for terrain traversal is learned on 3 human demonstrations for two techniques Maximum Entropy inverse reinforcement learning, and risk averse Bayesian reward learning with a uniform prior and an acceptable risk factor of $\epsilon = 0.8$ yielding the following reward weights seen in Table 4.17:

Table 4.17: Reward weights among semantic terrains

Reward Model	**Feature Weight**				
	Grass	Gravel	Road	U1	U2
(i) Maximum Entropy IRL	-0.20	-0.51	0.38	-0.63	-0.50
(ii) Risk Averse Bayesian Reward Learning	-0.43	-2.00	0.99	-2.00	-2.00

Note that U1, and U2, correspond to unknowns. This is because U1 and U2 where never directly mapped to any terrain, although they are terrains. When comparing the weights between the two models, both reward models learn weights which match the demonstrator's preferences over terrains. Namely, road achieves the highest reward in both models, followed by grass, and finally gravel. Moreover, both reward models learn that the unknown features have the highest negative reward. This is due to the fact that the unknown terrains rarely occur in the resulting demonstration maps, and they are never traversed. The difference between the two models is in the relative difference between rewards. Namely, RABRL, assigns high negative reward to gravel, U1, and U2 since their uncertainty is above the determined threshold. Moreover, the relative difference of rewards between road and grass is higher on the RABRL model.

To evaluate autonomous navigation performance, the refined model is ran online, performing semantic segmentation on images output by the simulated robot's camera in the 3D graphics engine. The autonomous system's terrain projection module takes the semantic images as input, and outputs a set of occupancy grids for each of the perception model's terrain labels by projecting pixels to the ground plane. Then the learned reward weights are used to assign costs to occupancy grids. The robot then performs autonomous waypoint navigation using a search based planner which finds the least cost path to a goal using the produced cost map.

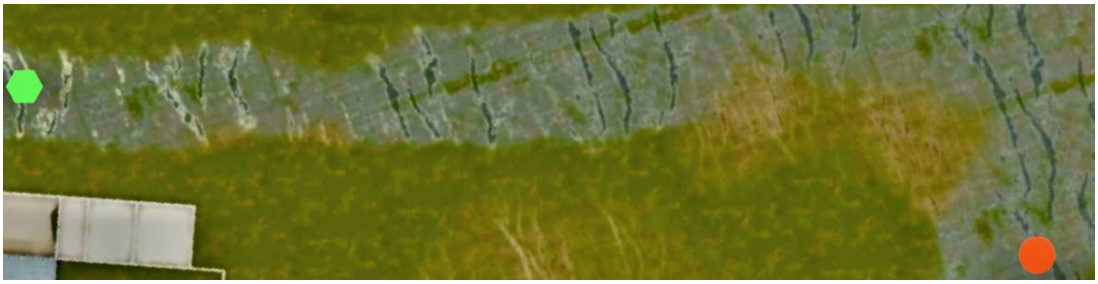

Figure 4.14: Birds eye view of the test scenario for autonomous operation in an unstructured environment containing 3 terrains, grass (green), gravel (brown), and road (gray). Green hexagon: start. Red circle: end.

Each model is tested on its ability to autonomously navigate the mission in Fig 4.14, where the start location is represented by a green hexagon, and the goal location represented by a red circle. Each model attempts the mission 3 times. To quantitatively measure performance, we compare the trajectories obtained from autonomous traversal to an expert trajectory teleoperated by a human and considered to be optimal. The distance between the optimal trajectory and the autonomous trajectory is captured using the modified Hausdorff Distance [22], namely:

$$h(A,B) = \frac{1}{|A|} \sum_{a_i \in A} \min_{b_j \in B} (d(a_i, b_j)) \tag{4.13}$$

Where A represents the expert trajectory, B represents the autonomous trajectory, and d represents the distance between two (x, y) coordinates a and b. A lower score in this metric is better, meaning that there is less distance between the optimal teleoperated trajectory and the autonomous trajectory.

Table 4.18: Modified Hausdorff Distance for each reward model

Model	Modified Hausdorff Distance			
	Mean	Median	Best	Worst
(i) MaxEnt	7.401	7.785	**3.975**	10.459
(ii) RABRL	**5.363**	**5.313**	4.412	**6.363**

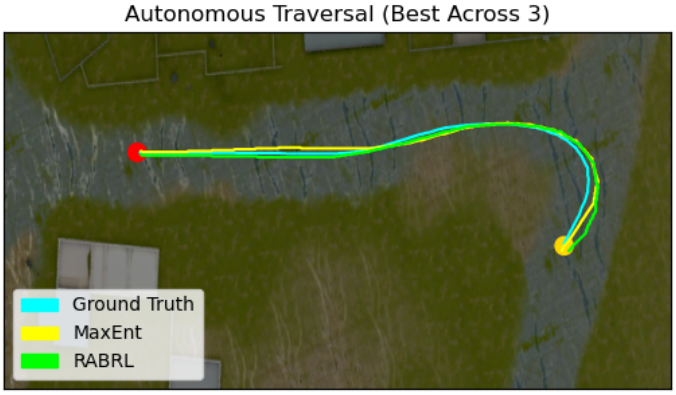

Figure 4.15: Birds eye view showing *best* case autonomous traversal performance.

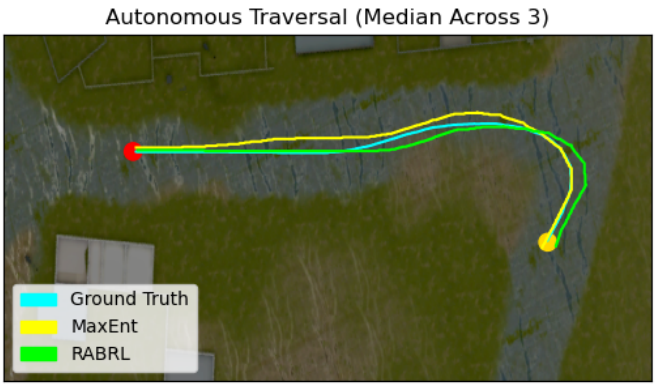

Figure 4.16: Birds eye view showing *median* case autonomous traversal performance.

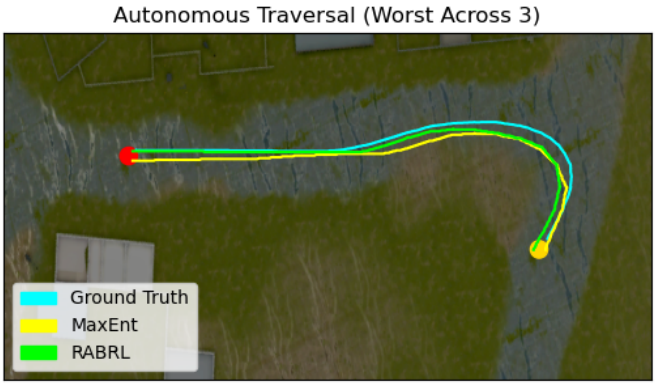

Figure 4.17: Birds eye view showing *worst* case autonomous traversal performance.

Comparing the two models, the risk averse bayesian reward learning (RABRL) model achieves a lower mean, median, and worst case modified Hausdorff Distance. The maximum entropy model achieves the lowest distance for the best autonomous trajectory compared to RABRL. Although quantitatively it seems that the weights of the RABRL model are more representative of the underlying environment's true reward when compared to the maximum entropy model as denoted by it's lower Hausdorff Distance, this is not the case qualitatively when considering the overall mission goals. When looking at the accompanying figures (Figure 4.15, Figure 4.16, and Figure 4.17), we can see that both models accurately follow the demonstrators preferences over terrain. That is, they stay on the road while avoiding the grass and gravel. Most importantly, both models are able to autonomously navigate the unstructured environment with low distances, backing up this dissertation's claim that an unsupervised segmentation model's labels can be used as reward basis for inverse reinforcement learning.

To conclude, supporting experimental results in this section show that one can achieve autonomous waypoint navigation which follows a demonstrators preferences using only unsupervised data. Using an unsupervised perception model as the reward basis for inverse reinforcement learning has large real-world impact. On the contrary, supervised perception models require labeled data representative of the operational environment. Besides cost and time, if the labeled data fails to capture all aspects of the underlying environment it will not be able to perform desired mission behavior. Using unsupervised data, engineers can quickly collect a sequence of images, train an initial perception model, refine it, and use it as a perception system. This enables one to drop a robot anywhere, and quickly train it to understand the current operational environment.

4.8 Conclusion

In contrast to dense pixel-wise semantic labeling, this chapter presents a methodology to incorporate human demonstrations as weak supervisory signals to an unsupervised semantic segmentation algorithm. First, the algorithm learns a set of abstract label representations across an unlabeled video stream resulting in an over-segmented set of labels. Second, a human teleoperates the robot to provide a small dataset of navigation demonstrations that traverse each high-level semantic concept present in the environment. Finally, the traversal evidence from the human demonstrations is used to map the over-segmented unsupervised abstract labels to a discrete set of high-level semantic labels to create a more unified representation that alleviates over-segmentation. Using the resulting unified class representations as the perception model, a perception subsystem is able to segment a subset of pixels to a high-level semantic label for every image in future video streams.

Supporting experimental results both in an over-segmentation simulator utilizing a 3D graphics engine, and on the RUGD dataset show that demonstrations which inform label merging for an initial over-segmented unsupervised model are able to learn a set of labels which directly map to every terrain in the scene. Moreover, the refined perception output may be used as the reward basis for inverse reinforcement learning, thereby enabling autonomous waypoint navigation with terrain preferences in unstructured environments.

4.9 Appendix- Unsupervised Semantic Scene Labeling

The first stage of our pipeline relies on an unsupervised stream-based segmentation algorithm to learn an initial partitioning of pixels into coherent groups. The output of unsupervised algorithms is often over-segmented, and later stages in our pipeline will refine this output (discussed in Sec. 4.4) to produce a perception model that provides high-level semantic segmentation output similar to supervised models but without ground truth pixel wise annotations for learning.

Although any unsupervised segmentation algorithm could be inserted into the first stage of our pipeline, we use the Unsupervised Semantic Scene Labeling (USSL) [80] algorithm with modifications to use pre-trained deep neural networks for feature extraction. USSL is selected for its ability to identify true class boundaries and looser segmentation requirements which result in an over-segmented image. As seen in Fig. 4.3, learned segment labels from USSL are not required to form a connected component of pixels, USSL has the ability to learn a single segment that is composed of pixels from disconnected regions in an image. This is beneficial in autonomous navigation, as the same terrain may be separated from another, such as a road with grass on both sides. A overview of the USSL algorithm follows.

USSL is an ensemble-based approach that leverages agglomerative (bottom-up) clustering to automatically learn the number of unique representative concepts within a data stream. The algorithm reduces noise introduced in unsupervised learning caused by the lack of explicit guidance on what to learn while discovering novel classes throughout time in the data stream without a-priori knowledge on the ground truth set of output class labels. If USSL is trained entirely online, initial global models are noisy at first, but become stable as more windows are processed. Conversely, if USSL is trained offline, it can learn a base set of output labels, and then when operating online, it can either continue to learn new output classes, or stay with the base set.

USSL classifies pixels in a bottom up manner using two main concepts, local and global label models. A local label model segments an image based on color similarity to previous frames in a window of images by agglomeratively clustering superpixels [2] to obtain a set of parent segments describing an image. Global label models encode the relationship among multiple local model parent segments across overlapping windows throughout time, producing a set of final output labels which are used to perform semantic segmentation. Next, we describe the main USSL components, including feature extraction, local labeling where superpixel groupings are discovered via agglomerative clustering, and global labeling, where local model clusters are composed into an ensemble.

Feature Extraction

USSL receives as input a stream of images $D = \{I_1, I_2, ..., I_n\}$. Each I_i is first processed into K superpixels using SLIC [2], then feature extraction is performed for each SLIC segment. USSL was originally tested [80] using generic low-level features, including histogram of oriented gradients [18], local binary patterns [60], and scale invariant feature transforms [50]. Rather than use generic low-level features, we extend USSL to leverage modern deep learning feature extraction approaches. Specifically, we evaluate the use of two deep convolutional neural networks for feature extraction, MobileNetV2 [69], and EfficientNet [73]. Both models where trained on the ImageNet dataset [19]. These specific architectures where chosen due to their focus on runtime efficiency as they will be residing on a simulated robotic platform with limited computational resources. For a given model, a feature vector for each superpixel, $\phi(k_i)$, of dimension 1280 is extracted using the penultimate (second to last) layer of the model architecture.

Local Labeling

Creation of an ensemble of local models is achieved by running the segmentation task on overlapping sliding windows. Windows are initialized every $p/2$ frames such that the first half of the frames overlap with the window before it, and the second half with the window after. For our experiments, we set $p = 6$ because the incoming data-stream is 6 frames per second. Local labeling within a window is responsible for grouping superpixels into clusters using proximity and feature similarity information via agglomerative clustering. The set of learned clusters within a window comprises a set of local models $M = \{m_1, m_2, ..., m_n\}$, where each m_i is the feature representation of an abstract output label. Local models are encoded as a single region adjacency graph (RAG) [63], $RAG = (V, E)$,such that each vertex v_i represents local model m_i and an edge e_j is created for each surrounding neighbor model.

During clustering, similarity between any two superpixels, k_i and k_j is computed as:

$$s(k_i, k_j) = \frac{1}{1 + \sqrt{[\phi(k_i) - \phi(k_j)]^2}} \tag{4.14}$$

yielding a numerical value in the range $[0, 1]$. Eq. (4.14) is evaluated for all pairwise combinations of feature vectors to obtain the similarity matrix S such that the (i, j) index represents the superpixel similarity score between superpixel K_i an K_j. Since both superpixels K and local models M are groups of pixels, they can be used interchangeably as arguments to this equation and the ones that follow. The nearest neighbor for each superpixel K_i is then calculated yielding S_{max}.

$$S_{max} = \underset{k_j \in K}{\operatorname{argmax}}\, s(k_i, k_j) \tag{4.15}$$

For each frame in the window, USSL automatically learns the merging threshold for halting agglomerative clustering in the RAG, thereby updating the set of local models. In

order to determine which superpixels are merged, a Gaussian history distribution h of feature similarities throughout the lifetime of the window. Each nearest neighbor local model and superpixel serve as samples to this distribution:

$$H = [s(k_{max}, m_{max}) \mid \forall\ (k_{max}, m_{max}) \in S_{max}] \tag{4.16}$$

The agglomerative clustering threshold is calculated as:

$$\beta = \begin{cases} -1, & \text{if } s(m_i, m_j) < \mu_h \\ 1, & \text{otherwise} \end{cases} \tag{4.17}$$

Where μ_h represents the mean of the history distribution. Local models are then iteratively merged within M. At each step, the next model to be merged is determined by:

$$l^* = \max_{m_i \in M} \quad \beta * s(m_i, m_j) \tag{4.18}$$

This process continues until l^* becomes negative. The steps defined by Eqs. (4.14) - (4.18) repeat for each frame in the window- updating the set of local models for each incoming frame throughout the lifetime of the window.

Global Labeling

After all frames from a sliding window have been processed, the global labeling of USSL uses the evidence across the current local model ensemble to generate a global set of abstract output labels for the entire stream, $\hat{M} = \{\hat{m}_1, \hat{m}_2, ..., \hat{m}_n\}$. Each $\hat{m}_i$ from the most recent window is added to the global labeling graph, $G = (V, E)$. Each vertex, $v_i \in G$, represents a local label model indexed by window and label $v_i(w_k, \hat{m}_j)$. An edge $e_k \in G$ encodes that two local models from overlapping windows have at least one pixel in common. The edge

weight (α_{e_k}) represents the number of shared pixels between the two models. The underlying assumption is that if two local models from separate sliding windows share enough pixels in common, the underlying semantic concept learned across the two are the same. Ultimately, the set of connected components in G will define the final global models, $\hat{M}$, but edges with low evidence of similarity are first pruned to eliminate noise introduced in the local unsupervised model. The evidence score for vertex v_i is calculated as:

$$\epsilon_{v_i}(v_i(w_k, \hat{m}_j)) = \frac{\alpha_{e_{(w_k, \hat{m}_j)}}}{\sum_{e_k=1}^{|E|} \alpha_{e_k}} \tag{4.19}$$

Where $|\cdot|$ denotes set cardinality If $\epsilon(v_i(w_k, \hat{m}_j)) < \delta$ the edge is pruned. Each resulting group of connected components represents a single global label.

Chapter 5 Conclusion

5.1 Overview of contributions

This dissertation provides three main contributions:

1. Contribution 1 - Risk Averse Bayesian Reward Learning for Autonomous Navigation from Human Demonstration

2. Contribution 2 - Refining Unsupervised Semantic Segmentation Labels with Human Demonstrations

3. Contribution 3 - Unsupervised Reward Basis for Inverse Reinforcement Learning from Human Demonstrations

The claims of each contribution are verified with supplementary experimental results in their associated chapters. The breif summary of each contribution follows.

Contribution 1: In the first contribution, the method outlined in Chap. 3 is used to learn a reward model for a robot operating in an unstructured environment. The training environment contained 2 terrains, while the testing environment contained 3 terrains. The additional terrain in the testing environment is considered dangerous due to mission constraints. The resulting reward function leads to paths which match the demonstrator's terrain preferences while completely avoiding the third dangerous terrain. This is enabled by the use of a Bayesian posterior which quantifies uncertainty over the reward basis. This contribution provides evidence that uncertainty quantification facilitates autonomous systems to operate safely in environments beyond where they were trained.

Contribution 2: In the second contribution, the method outlined in Chap. 4 is used to refine an initial over-segmented, non-semantic feature representation to a smaller, semantic feature representation from human demonstrations. Supporting experimental results verify that there are three forms of perception refinement which are provided with the use of human demonstrations; (i) capability to identify and discard poor labels, (ii) capability to merge over-segmented labels together representing the same semantic concept, and (iii) capability to map unsupervised labels to semantic concepts, specifically terrain.

Contribution 3: In the final contribution, the methods from Chap. 3 and Chap. 4 are combined to enable autonomous waypoint navigation using only a sequence of unsupervised images and a limited set of human demonstrations. Experimental results verify the ability to learn control behaviors such as demonstrating a preference over terrains with the refined reward basis.

5.2 Future Work

While initial work [25] demonstrated the ability to follow terrain preferences from human demonstrations while avoiding the dangerous terrain, subsequent work [24] demonstrated the ability to utilize human demonstrations as a weakly supervised signal to refine unsupervised perception models and use this model as a reward basis for inverse reinforcement learning, a number of possible future contributions remain. Subsequent subsections outline future work that may be completed upon each major body of work presented in this dissertation, and other related lines of work in the field of autonomous system safety.

5.2.1 Risk Averse Bayesian Reward Learning from Human Demonstrations [25]

The primary appeal of Bayesian reward learning is that the resulting posterior can be used to quantitatively measure uncertainty. The original work subscribed to the risk averse view, where the model automatically penalized features with high uncertainty. However, in real world robotics scenarios this is not always the case, uncertainty may not always mean risky behavior. Suppose that instead of the unknown terrain being something dangerous such as mud or ice, the unknown terrain was actually beneficial, such as perfectly smooth racetrack quality asphalt. An alternative view of uncertainty is that the model is quantitatively describing to system designers what it knows, and what it doesn't know. With this lens, an active scenario may be considered, where the model identifies areas of high uncertainty to construct queries for system designers. The query would arise a simple *yes, no* question, *"Is this feature desirable?"*, and the answer is used to update the posterior's prior over beliefs. While active inverse reinforcement learning over a Bayesian posterior has been considered in other works [47, 47], they both focus on particular states, rather than the features that describe states.

Moreover, risk averse Bayesian reward learning involved either directly computing the

posterior over a small discrete set of reward functions, or estimating it using Markov chain Monte Carlo (MCMC) methods. Although MCMC methods do enable one to consider a larger, continuous reward space, training time is still limited by the size of the MDP, the number of features considered, and the number of demonstrations provided. At each reward function evaluation, the solution to a Markov decision process must be exactly computed to obtain the expected state visitation frequency over environmental features. Prior work in inverse reinforcement learning [28] enables the state visitation frequency to be estimated rather than exactly computed. The application of such a technique can be used to decrease training time. If training time can be decreased enough, one could train reward models in real time, as the robot is operating. Assuming the appropriate feature detectors exist for the desired behavior, real time training would enable an operator to drop a robot in a new environment, drive it around for a few minutes, and then be able to autonomously navigate that environment within a short period of time.

Furthermore risk averse Bayesian reward learning requires feature indicator(s) for terrain(s) whose reward weight(s) are unknown. Suppose a robot learns a reward function in an environment where only one unknown terrain is considered, if it later operates in an area where two unknown terrains are present, it will ignore it completely as it is not part of the underlying state representation considered for reward learning. Rather than seek to explicitly have feature detectors for all terrain features, one could feed in preprocessed sensor data as input to a deep neural network which outputs cost maps directly. Since deep neural networks are non linear, this also enables one to consider non-linear reward models. This has been done in traditional Maximum entropy inverse reinforcement learning settings [85] but much less has been done in the Bayesian setting [35]. However nonlinear, deep models often require a large number of demonstrations, so the ability to learn non-linear models from a small amount of demonstrations is a key area of interest.

Although this work explicitly considers the Maximum Entropy robot model [96], other robot models such as the ones outlined in [37] could be substituted for Eq. (3.9). In the maximum entropy model, all demonstrations are considered to be optimal, but that is not always the case in real world scenarios. For example, one may have an exact ranking over many trajectories, or may have certain groups of trajectories that are more desirable than others. Therefore exploring the autonomous performance obtained by learning on various demonstrations of trajectories is also a future research question to be answered. *Are certain models capable of learning more accurate reward model's with less demonstrations? Which types of demonstrations are the easiest to collect without combing through the data?*

Lastly, although this dissertation showed the methodology was able to learn accurate rewards if assumptions about being risk averse are correct, can we provide provable guarantees on the learned reward model? One path towards providing provable bounds could calculate credible intervals about measures of central tendency about each marginal reward weight. A different path may consider the use of side information in the form of temporal logic constraints provided as side information to the learning agent, such as in recent work [20]. However while this work explicitly provided provable empirical verification measures during learning, further work is required to make sure verification measures are upheld during model deployment.

5.2.2 Unsupervised Perception Model Refinement [24]

While the refinement methodology proposed in Chapter 4 does produce refined representations that aid autonomous traversal, some limitations remain. First, if the initial label set M is an overall poor representation with many noisy labels, the refined label set will not be any better. At best, all noisy labels can be identified and discarded, but if all labels are noisy, the resulting refined representation will result in all labels being discarded. Such behavior may occur from poor feature extraction techniques, poor learned representations, and data

complexity (highly dimensional, complex scenes, lighting and occlusion). This behavior was seen in Section 4.6. As such, future work seeks to replace the feature detectors used in Section 4.6 with off the shelf deep learning feature detectors trained on unstructured datasets such as RUGD [80].

Furthermore, while the refinement methodology is able to identify labels with high intra-class similarity to be identified and merged together, but those with high interclass similarity are only discarded. If a label has high interclass similarity, it may be more appropriate to "break the label up" into several labels. The proposed approach is to discard noisy labels, when perhaps they should be introspected more deeply. This limitation predominately comes from the focus on alleviating over-segmentation, and as such the methodology may be expanded to alleviate under-segmentation beyond discarding.

Moreover, since the weak supervisory signal obtained from human demonstrations directly maps labels to semantic classes, the resulting set of semantic segmentation images obtained by the refined model may be used to automatically label unsupervised data. As mentioned previously, ground truth data labeling is normally a time-consuming and expensive process. However, a framework based on the proposed methodology for refinement can be created for automatic data labeling of raw images. Subsequent labeled data could be used to train perception algorithms which only work with supervised data. This does have one caveat, in real world scenarios, resulting labeled data likely will not match the ground truth exactly. The level of accuracy of resulting labeled data is dependent on the initial algorithm performing unsupervised semantic segmentation. Therefore, further prepossessing steps will likely be required.

Lastly, although this work was able to incorporate human teleoperated demonstrations, future work seeks to incorporate secondary sources of supervisory information in order to

develop more accurate feature representations with sources such as human intervention and active human evaluation.

5.2.3 Autonomous Systems Safety and Formal Verification

While the research outlined in this dissertation focus on two main tasks: (i) responding to previously unseen environmental features, and (ii) finding an accurate perception representation of the current environment, future work seeks to take a broader approach to autonomous systems safety. An initial literature survey I conducted in 2022 [23] identified the current state of the art regarding approaches to autonomous system safety while simultaneously outlining the gaps in current literature. Broadly speaking, future work seeks to adapt autonomous system behavior learned in one environment to reliably and safely transfer to new environments, in the context of autonomous navigation in unstructured environments.

Towards this goal, the main research path seeks to bridge the gap between recent advancements made in formal verification of autonomous systems such as [51] [84] and real world autonomous systems. Currently, formal verification guarantees are often made in simulated environments modeling how an autonomous system is expected to act, however such guarantees fail to hold in the real world. *How can we get reliable real world verification measures of autonomous system performance?* Moreover, accompanying assumptions such as the capability to model exact robot dynamics, having perfect perception or mapping, or the capability to identify all possible future states a robot may encounter a priori are not possible in real world scenarios. *Can we relax these assumptions while still providing accurate verification measures?*

A second possible research path includes developing realtime runtime monitoring methods loosely based on [3] for a navigation system. System safety specifications can be provided in a formal language such as temporal logic which act as constraints during autonomous operation. If an autonomous system trajectory planner provides a path that violates these

specifications, the plan may either be sent back to the planner, or altered by the runtime monitor such that the plan does satisfy the specifications. For example, suppose a planner choses the shortest path to reach a goal, but the path provided requires the robot to operate at a high speed at a high pitch/slope grade which may cause the robot to flip due to its physical constraints. The navigation system, which is aware of such physical constraints, takes the plan as input and checks if every pose in the suggested plan either violates or satisfies the constraints. In this system configuration, the runtime monitor is planner agnostic, it takes plans as input, and verifies safety properties of the plan.

Bibliography

[1] Pieter Abbeel and Andrew Y. Ng. "Apprenticeship learning via inverse reinforcement learning". In: *Proceedings of the twenty-first international conference on machine learning*. ICML '04. Banff, Alberta, Canada: Association for Computing Machinery, July 2004, p. 1.

[2] Radhakrishna Achanta et al. "SLIC superpixels compared to state-of-the-art superpixel methods". In: *IEEE transactions on pattern analysis and machine intelligence* 34.11 (2012), pp. 2274–2282.

[3] Mohammed Alshiekh et al. "Safe reinforcement learning via shielding". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.

[4] Dario Amodei et al. "Concrete Problems in AI Safety". en. In: *arXiv:1606.06565 [cs]* (July 2016). arXiv: 1606.06565.

[5] Christophe Andrieu et al. "An introduction to MCMC for machine learning". In: *Machine learning* 50.1-2 (2003), pp. 5–43.

[6] Brenna D Argall et al. "A survey of robot learning from demonstration". In: *Robotics and autonomous systems* 57.5 (2009), pp. 469–483.

[7] Saurabh Arora and Prashant Doshi. "A survey of inverse reinforcement learning: Challenges, methods and progress". In: *Artificial Intelligence* 297 (2021), p. 103500.

[8] Saurabh Arora and Prashant Doshi. "A survey of inverse reinforcement learning: Challenges, methods and progress". In: *Artificial Intelligence* 297 (2021), p. 103500.

[9] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495.

[10] Michael Bain and Claude Sammut. "A Framework for Behavioural Cloning." In: *Machine Intelligence 15*. 1995, pp. 103–129.

[11] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation learning: A review and new perspectives". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828.

[12] Christopher M Bishop. *Pattern recognition and Machine Learning*. Springer, 2006. ISBN: 0387310738zs.

[13] Daniel Brown et al. "Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations". In: *International conference on machine learning*. PMLR. 2019, pp. 783–792.

[14] Daniel Brown et al. "Safe imitation learning via fast bayesian reward inference from preferences". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 1165–1177.

[15] Liang-Chieh Chen et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 801–818.

[16] Jang Hyun Cho et al. "Picie: Unsupervised semantic segmentation using invariance and equivariance in clustering". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 16794–16804.

[17] Jaedeug Choi and Kee-Eung Kim. "MAP inference for bayesian inverse reinforcement learning". In: *Advances in Neural Information Processing Systems*. 2011, pp. 1989–1997.

[18] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. 2005, pp. 886–893.

[19] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. 2009, pp. 248–255.

[20] Franck Djeumou et al. "Task-guided IRL in POMDPs that scales". In: *Artificial Intelligence* 317 (2023), p. 103856.

[21] Carl Doersch, Abhinav Gupta, and Alexei A Efros. "Unsupervised visual representation learning by context prediction". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1422–1430.

[22] Marie-Pierre Dubuisson and Anil K Jain. "A modified Hausdorff distance for object matching". In: *Proceedings of 12th international conference on pattern recognition*. Vol. 1. IEEE. 1994, pp. 566–568.

[23] Christian Ellis, Maggie Wigness, and Lance Fiondella. "A mapping of assurance techniques for learning enabled autonomous systems to the systems engineering lifecycle". In: *2022 IEEE International Conference on Assured Autonomy (ICAA)*. 2022, pp. 28–35.

[24] Christian Ellis et al. "Refining Unsupervised Semantic Segmentation Labels with Human Demonstrations (under review)". In: *2024 IEEE/RSJ International Conference on Robotics and Automation (ICRA)*. 2024.

[25] Christian Ellis et al. "Risk averse bayesian reward learning for autonomous navigation from human demonstration". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021, pp. 8928–8935.

[26] Linus Ericsson et al. "Self-Supervised Representation Learning: Introduction, advances, and challenges". In: *IEEE Signal Processing Magazine* 39.3 (2022), pp. 42–62.

[27] Di Feng et al. "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges". In: *IEEE Transactions on Intelligent Transportation Systems* 22.3 (2020), pp. 1341–1360.

[28] Chelsea Finn, Sergey Levine, and Pieter Abbeel. "Guided cost learning: Deep inverse optimal control via policy optimization". In: *International conference on machine learning*. PMLR. 2016, pp. 49–58.

[29] Alberto Garcia-Garcia et al. "A survey on deep learning techniques for image and video semantic segmentation". In: *Applied Soft Computing* 70 (2018), pp. 41–65.

[30] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. "Unsupervised Representation Learning by Predicting Image Rotations". In: *International Conference on Learning Representations*. 2018.

[31] Haifeng Gong and Jianbo Shi. *Conditional Entropies as Over-Segmentation and Under-Segmentation Metrics for Multi-Part Image Segmentation. University of Pennsylvania Department of Computer and Information Science; Philadelphia, PA*. Tech. rep. USA: 2011. Technical Report MS-CIS-11-17, 2011.

[32] Junyao Guo, Unmesh Kurup, and Mohak Shah. "Is it safe to drive? An overview of factors, metrics, and datasets for driveability assessment in autonomous driving". In: *IEEE Transactions on Intelligent Transportation Systems* 21.8 (2019), pp. 3135–3151.

[33] Dylan Hadfield-Menell et al. "Inverse reward design". In: *Advances in neural information processing systems* 30 (2017).

[34] Borja Ibarz et al. "Reward learning from human preferences and demonstrations in Atari". In: *Advances in neural information processing systems* 31 (2018), pp. 8011–8023.

[35] Mahdi Imani and Seyede Fatemeh Ghoreishi. "Scalable inverse reinforcement learning through multifidelity Bayesian optimization". In: *IEEE transactions on neural networks and learning systems* 33.8 (2021), pp. 4125–4132.

[36] Lucas Janson, Tommy Hu, and Marco Pavone. "Safe motion planning in unknown environments: Optimality benchmarks and tractable policies". In: *arXiv preprint arXiv: 1804.05804* (2018).

[37] Hong Jun Jeon, Smitha Milli, and Anca Dragan. "Reward-rational (implicit) choice: A unifying formalism for reward learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4415–4426.

[38] Xu Ji, Joao F Henriques, and Andrea Vedaldi. "Invariant information clustering for unsupervised image classification and segmentation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9865–9874.

[39] Peng Jiang et al. "Rellis-3d dataset: Data, benchmarks and analysis". In: *2021 IEEE international conference on robotics and automation (ICRA)*. 2021, pp. 1110–1116.

[40] Dongshin Kim et al. "Traversability classification using unsupervised on-line visual learning for outdoor robot navigation". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* 2006, pp. 518–525.

[41] Jens Kober, J. Andrew Bagnell, and Jan Peters. "Reinforcement learning in robotics: A survey". en. In: *The International Journal of Robotics Research* 32.11 (Sept. 2013), pp. 1238–1274.

[42] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. "Revisiting self-supervised visual representation learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 1920–1929.

[43] Fahad Lateef and Yassine Ruichek. "Survey on semantic segmentation using deep learning techniques". In: *Neurocomputing* 338 (2019), pp. 321–348.

[44] Hsin-Ying Lee et al. "Unsupervised representation learning by sorting sequences". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 667–676.

[45] Jan Leike et al. "AI Safety Gridworlds". en. In: *arXiv:1711.09883 [cs]* (Nov. 2017).

[46] Maxim Likhachev. *Search-Based Planning Library*. `https://github.com/sbpl/sbpl`. (Visited on 11/24/2014).

[47] Manuel Lopes, Francisco Melo, and Luis Montesano. "Active learning for reward estimation in inverse reinforcement learning". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2009, pp. 31–46.

[48] Viktor Losing, Barbara Hammer, and Heiko Wersing. "Incremental on-line learning: A review and comparison of state of the art algorithms". In: *Neurocomputing* 275 (2018), pp. 1261–1274.

[49] Björn Lötjens, Michael Everett, and Jonathan P How. "Safe reinforcement learning with model uncertainty estimates". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2019, pp. 8662–8668.

[50] David G Lowe. "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2 (2004), pp. 91–110.

[51] Matt Luckcuck. "Using formal methods for autonomous systems: Five recipes for formal verification". In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 237.2 (2023), pp. 278–292.

[52] Kunal Menda, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. "EnsembleDAgger: A Bayesian Approach to Safe Imitation Learning". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 5041–5048.

[53] Joseph Victor Michalowicz, Jonathan M Nichols, and Frank Bucholtz. *Handbook of differential entropy*. CRC Press, 2013. ISBN: 978042907224i.

[54] Ryusuke Miyamoto et al. "Vision-based road-following using results of semantic segmentation for autonomous navigation". In: *2019 IEEE 9th International Conference on Consumer Electronics (ICCE-Berlin)*. 2019, pp. 174–179.

[55] Robin R Murphy. *Disaster robotics*. The MIT press, 2014. ISBN: 978026253465.

[56] Keiji Nagatani et al. "Emergency response to the nuclear accident at the Fukushima Daiichi Nuclear Power Plants using mobile rescue robots". In: *Journal of Field Robotics* 30.1 (2013), pp. 44–63.

[57] Andrew Y. Ng and Stuart J. Russell. "Algorithms for Inverse Reinforcement Learning". In: *Proceedings of the Seventeenth International Conference on Machine Learning*. ICML '00. 2000, pp. 663–670.

[58] Scott Niekum et al. "Learning grounded finite-state representations from unstructured demonstrations". In: *The International Journal of Robotics Research* 34.2 (2015), pp. 131–157.

[59] Yu-ichi Ohta, Takeo Kanade, and Toshiyuki Sakai. "An analysis system for scenes containing objects with substructures". In: *Proceedings of the Fourth International Joint Conference on Pattern Recognitions*. 1978, pp. 752–754.

[60] Timo Ojala, Matti Pietikäinen, and David Harwood. "A comparative study of texture measures with classification based on featured distributions". In: *Pattern recognition* 29.1 (1996), pp. 51–59.

[61] Takayuki Osa et al. "An Algorithmic Perspective on Imitation Learning". en. In: *Foundations and Trends in Robotics* 7.1-2 (2018), pp. 1–179. ISSN: 1935-8253, 1935-8261.

[62] Jitendra Parmar et al. "Open-world machine learning: applications, challenges, and opportunities". In: *ACM Computing Surveys* 55.10 (2023), pp. 1–37.

[63] Theodosios Pavlidis. *Structural Pattern Recognition*. New York: Springer, 1980, pp. 46–64.

[64] Martin L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. eng. Wiley series in probability and statistics. OCLC: 254152847. Hoboken, NJ: Wiley-Interscience, 2005. ISBN: 9780471727828.

[65] Morgan Quigley et al. "ROS: an open-source Robot Operating System". In: *ICRA workshop on open source software*. Vol. 3. 3.2. Kobe, Japan. 2009, p. 5.

[66] Deepak Ramachandran and Eyal Amir. "Bayesian Inverse Reinforcement Learning." In: *Proceedings of the International Joint Conference on Artificial Intelligence*. Vol. 7. 2007, pp. 2586–2591.

[67] Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. "Maximum margin planning". In: *Proceedings of the International Conference on Machine learning*. 2006, pp. 729–736.

[68] Stuart Russell. *Human compatible: Artificial intelligence and the problem of control*. Penguin, 2019.

[69] Mark Sandler et al. "Mobilenetv2: Inverted residuals and linear bottlenecks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[70] Adarsh Jagan Sathyamoorthy et al. "Terrapn: Unstructured terrain navigation using online self-supervised learning". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 7197–7204.

[71] Satinder Singh, Richard L Lewis, and Andrew G Barto. "Where do rewards come from?" In: *Proceedings of the annual conference of the cognitive science society*. Cognitive Science Society. 2009, pp. 2601–2606.

[72] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: an introduction*. Second edition. Cambridge, Massachusetts: The MIT Press, 2018. ISBN: 9780262039246.

[73] Mingxing Tan and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks". In: *International conference on machine learning*. PMLR. 2019, pp. 6105–6114.

[74] Yewteck Tan et al. "Risk-aware autonomous navigation". In: *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III*. Vol. 11746. SPIE. 2021, pp. 335–348.

[75] Marco Toldo et al. "Unsupervised domain adaptation in semantic segmentation: a review". In: *Technologies* 8.2 (2020), p. 35.

[76] Faraz Torabi, Garrett Warnell, and Peter Stone. "Behavioral Cloning from Observation". In: *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI-18*. July 2018, pp. 4950–4957.

[77] Alexander JB Trevor, John G Rogers, and Henrik I Christensen. "Omnimapper: A modular multimodal mapping framework". In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2014, pp. 1983–1990.

[78] Jessica Van Brummelen et al. "Autonomous vehicle perception: The technology of today and tomorrow". In: *Transportation research part C: emerging technologies* 89 (2018), pp. 384–406.

[79] Li Wang and Dong-Chen He. "Texture classification using texture spectrum". In: *Pattern recognition* 23.8 (1990), pp. 905–910.

[80] Maggie Wigness and John G Rogers. "Unsupervised semantic scene labeling for streaming data". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 4612–4621.

[81] Maggie Wigness, John G. Rogers, and Luis E. Navarro-Serment. "Robot Navigation from Human Demonstration: Learning Control Behaviors". en. In: *IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Queensland, May 2018, pp. 1150–1157.

[82] Maggie Wigness et al. "A rugd dataset for autonomous navigation and visual perception in unstructured outdoor environments". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 5000–5007.

[83] Maggie Wigness et al. "Using perception cues for context-aware navigation in dynamic outdoor environments". In: *Field Robotics* 1.1 (Oct. 2021), pp. 1–33.

[84] Tichakorn Wongpiromsarn et al. "Formal Methods for Autonomous Systems". In: *arXiv preprint arXiv:2311.01258* (2023).

[85] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. "Maximum entropy deep inverse reinforcement learning". In: *arXiv preprint arXiv:1507.04888* (2015).

[86] Markus Wulfmeier, Dominic Zeng Wang, and Ingmar Posner. "Watch this: Scalable cost-function learning for path planning in urban environments". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 2089–2095.

[87] Markus Wulfmeier, Dominic Zeng Wang, and Ingmar Posner. "Watch this: Scalable cost-function learning for path planning in urban environments". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 2089–2095.

[88] Chenliang Xu and Jason J Corso. "Evaluation of super-voxel methods for early video processing". In: *2012 IEEE conference on computer vision and pattern recognition*. 2012, pp. 1202–1209.

[89] Chenliang Xu, Caiming Xiong, and Jason J Corso. "Streaming hierarchical video segmentation". In: *European Conference on Computer Vision*. Springer. 2012, pp. 626–639.

[90] Hongshan Yu et al. "Methods and datasets on semantic segmentation: A review". In: *Neurocomputing* 304 (2018), pp. 82–103.

[91] Jiakai Zhang and Kyunghyun Cho. "Query-Efficient Imitation Learning for End-to-End Simulated Driving". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI'17. San Francisco, California, USA: AAAI Press, 2017, pp. 2891–2897.

[92] Shao Zhifei and Er Meng Joo. "A survey of inverse reinforcement learning techniques". In: *International Journal of Intelligent Computing and Cybernetics* 5 (3 2012).

[93] Zhi-Hua Zhou. "A brief introduction to weakly supervised learning". In: *National science review* 5.1 (2018), pp. 44–53.

[94] Fuzhen Zhuang et al. "A comprehensive survey on transfer learning". In: *Proceedings of the IEEE* 109.1 (2020), pp. 43–76.

[95] Brian D Ziebart. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University, 2010.

[96] Brian D Ziebart et al. "Maximum entropy inverse reinforcement learning." In: *AAAI*. Vol. 8. Chicago, IL, USA. 2008, pp. 1433–1438.