Online Learning Techniques for Improving Robot Navigation in Unfamiliar Domains

Boris Sofman

April 27, 2009

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Robotics

Abstract

Many mobile robot applications require robots to act safely and intelligently in complex unfamiliar environments with little structure and limited or unavailable human supervision. As a robot is forced to operate in an environment that it was not engineered or trained for, various aspects of its performance will inevitably degrade. Roboticists equip robots with powerful sensors and data sources to deal with uncertainty, only to discover that the robots are able to make only minimal use of this data and still find themselves in trouble. Similarly, roboticists develop and train their robots in representative areas, only to discover that they encounter new situations that are not in their experience base. Small problems resulting in mildly sub-optimal performance are often tolerable, but major failures resulting in vehicle loss or compromised human safety are not. We propose a series of online algorithms to enable a mobile robot to better deal with uncertainty in unfamiliar domains in order to improve performance, better utilize available resources and reduce risk to the vehicle. We will validate these algorithms through extensive testing onboard large mobile robot systems and will argue how such approaches can increase the reliability and robustness of mobile robots, bringing them closer to the capabilities required for many real-world applications.

Contents

1 Introduction

The vision of autonomous mobile robots revolutionizing the way we live our lives dates back almost a century. It is easy to justify their appeal: robots could automate many of the tasks that are too dirty, dangerous or dull to be suitable for humans and handle others at which they are simply more productive and accurate than humans can ever hope to be. These systems could automate tasks in diverse domains such as transportation, military reconnaissance and supply routes, agricultural tasks, space exploration, border and property patrolling and working in toxic environments.

The current state of autonomous robot navigation, however, has yet to fulfill these expectations. Almost all real-world uses of unmanned ground vehicles (UGVs) operate either in highly structured or controlled environments, where the state of the world is fully known at all times and plans can be precisely executed with minimal uncertainty or risk, or with extensive human involvement (see Figure 1).

Figure 1: Example real-world applications of mobile robotics: from left, the Kiva Systems inventory management robot, the KUKA industrial robot and the iRobot Packbot used for remote bomb disposal. As with most current real-world applications of robotics, these systems do not have to deal with complicated robot perception problems since they operate in controlled and structured environments or through tele-operation.

In reality, for mobile robots to be truly useful, they need to be robust enough to be able to sense and operate in partially unknown and unstructured environments. While many autonomous UGVs have advanced to a level where they are competent and reliable a high percentage of the time in many environments [1, 2, 3, 4, 5, 6], most of these systems are heavily engineered for the domains they are intended to operate in. Any deviation from these domains often results in sub-optimal performance or even complete failure. Given the cost of such systems and the importance of safety and reliability in many of the tasks that they are intended for, even a relatively rare rate of failure is unacceptable. In many domains that are prime candidates for mobile robotic applications, the risk of catastrophic failure, however small, is a primary reason why autonomous systems are still under-utilized despite already demonstrating impressive abilities.

While roboticists do everything they can to equip robots with capabilities relevant to a wide range of domains, in order for real-world applications of UGVs to increase, they must be able to adapt to changing and unfamiliar aspects of their environments. The uncertainty throughout the UGV development and deployment process can lead to degraded performance through several challenging circumstances:

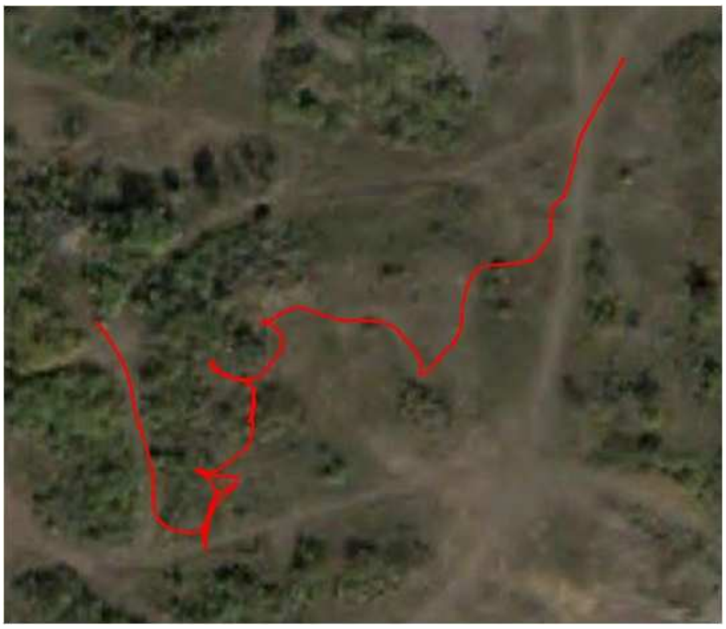


Figure 2: UGVs navigating with a limited perception range will often execute highly suboptimal paths. The path above was executed by a large UGV whose goal was to navigate to the goal at the left portion of the environment using an onboard perception system with a range of approximately 15 meters. Due to its limited visibility range, the UGV lost significant time navigating through the heavy vegetation, even requiring a human intervention at one point.

Developmental Uncertainty. Roboticists equip UGVs with powerful sensors and data sources to deal with uncertainty, only to discover that the UGVs are able to make only minimal use of this data and still find themselves in trouble. Overhead imagery data, for instance, has the potential to greatly enhance autonomous robot navigation in complex outdoor environments. In practice, reliable and effective automated interpretation of imagery from diverse terrain, environmental conditions, and sensor varieties proves to be challenging. As a result, a system that needs to perform reliably across many domains without major re-engineering must rely on only the subset of available information that generalizes well across many domains. Due to the data accuracy, consistency and density required for such features, this often limits a system to utilizing only onboard sensor data within a short proximity to the vehicle. This can lead to highly suboptimal behavior, often putting the UGV in dangerous situations such as the one shown in Figure 2.

In many difficult domains it is important to be able to interpret terrain well at a distance. Finding out about world as early as possible allows a UGV to construct more globally efficient paths while reducing risk. It is therefore important for a system to be able to take advantage of all potentially useful data sources and extend the range of its perception system by adapting to changing conditions without the necessity of human-supervised retraining.

Deployment Uncertainty. Similarly, roboticists develop and train their robots in representative areas, only to discover that they encounter new situations that are not in their experience base. If the perception system does not extend well to situations that were not represented during training, encounters with novel situations can lead to unexpectedly poor performance or even complete failures. Since it is impossible to prepare for the unexpected, one must assume that such situations will arise during real-world operation. To mitigate this risk a UGV must be able to identify situations that it is likely untrained to handle *before* it experiences a major failure. This problem therefore becomes one of novelty detection: how a robot can identify when perception system inputs differ from prior inputs seen during training or previous operation in the same area. With this ability, the system can either avoid novel locations to minimize risk or stop and enlist human help via supervisory control or tele-operation. For maximal impact such a system must be well-suited for online use as the system must incorporate feedback received through its continued experience and human aid.

Autonomy system limitations. At times, even the most robust autonomous systems are simply unfit to handle a given situation. Fortunately, in many cases an autonomous UGV is able to seek occasional help from a remote operator. The key is knowing *when* to ask for this help as human time is often valuable and limited. If we treat the human as an oracle with a high query cost, reasoning about the possible impact of human involvement in a situation can avoid a large number of unnecessary human queries. Since novel situations pose the greatest risk, if resolving such uncertainty through human help has a high potential benefit, a system can stop and call for remote help based on the assumption that humans are better able to negotiate new situations than autonomous vehicles. Furthermore, a learning system can observe the performance of the autonomous vehicle in particular situations and compare that performance to remote human-control performance in similar situations. When the vehicle encounters such situations in the future, it can then invoke whichever expert demonstrated better performance: the remote human or autonomous vehicle. When used in conjunction with a novelty detection system, such an approach can safeguard a UGV while maximizing the benefit from human availability throughout autonomous navigation.

We propose to address each of these challenges by presenting a series of online approaches that allow a UGV to adapt to the uncertainty of unfamiliar domains where human aid is limited or unavailable. For each problem we present relevant existing techniques followed by our approaches and an examination of how they address the limitations of similar techniques. We propose to implement and test these algorithms on large outdoor mobile robots and argue how the added ability to navigate safely and reliably in diverse real-world environments may facilitate the deployment of such robotic systems years earlier than otherwise possible.

2 Related Work

Operating under uncertainty is a central requirement of most non-trivial mobile robot tasks. The obvious initial approach for dealing with such uncertainty is with sensors and a perception system to interpret the incoming sensor data. These perception systems are often trained extensively on data representative of the environment the robot is to operate in, allowing them to parse meaningful information from this sensor data and identify degrees of traversability for the environment. A variety of sophisticated planning algorithms such as D* are adept at handling streams of traversal information updates and modifying paths appropriately in real-time

[7].

In existing commercial robots, sensor systems can range from the simple bump and react mechanism of the iRobot Roomba vacuum cleaner to the much more sophisticated line of industrial material handling robots from Seagrid Corporation that construct and operate on three-dimensional occupancy maps. Within the research domain, such examples are varied and numerous, including the systems mentioned earlier [1, 2, 3, 4, 5, 6].

All of these systems make assumptions about the types of environments they need to operate within. The limitation of such robots thus becomes their dependency on their pre-trained perception systems. While they often allow robots to operate within varieties of controlled or predictable environments, such techniques only work for limited ranges from the robot. Additionally, these systems will inevitably struggle when the environment sufficiently changes. Since robot behavior can become unpredictable, in many applications it is critical to detect such situations and apply vehicle safeguarding techniques.

With current techniques there is no sure way for a vehicle to autonomously deal with all situations it may encounter using only a pre-trained perception system. One way to address this problem is to revert to tele-operation for some or all of the mission. Full tele-operation is prohibitively expensive for many applications due to the degree of required human attention and communications bandwidth. In cases where tele-operation is employed a portion of the time, a policy must determine under which conditions the robot or the human are to take control. Just as in the medical domain, false positives are preferable to false negatives due to the potentially high cost of mistakes. Stentz et al. rely on this idea in their system of semi-autonomous tractors [8]. These tractors would execute a specified task and stop when they detect that something *may* be an obstacle and send an image over a wireless link to a human supervisor. The human can then either attend to the problem or tell the robot to proceed if the obstacle was a false alarm. This allows the system to function autonomously a large portion of the time while remaining cautions and allowing a human to quickly intervene in questionable situations. Since the time required to verify an obstacle is minimal, this allows a single operator to oversee several machines. Effective safeguarding techniques are one of the keys to applying robotic technologies to various outdoor industrial tasks [9].

An extensive discussion on related work in hybrid tele-operation control schemes is presented in section 7.1. As discussed in that section, many systems are designed to ask for help only *after* they find themselves in trouble. In many domains, robots do not have this luxury: a significant mistake may result in the end of the mission or the loss of lives. We argue that online learning techniques are the correct way to deal with such situations and present a series of algorithms to deal with the difficulties described previously.

3 Problem Statement

In this thesis we consider the problem of a highly robust autonomous mobile robot whose mission is to navigate through complex, natural terrain to a specified destination. We assume that this terrain does not have a known structure and is not engineered to support navigation as in the case of highway or urban driving and that all obstacles can be treated as static. Since this terrain is either previously untraversed or has potentially changed since the last traversal, the robot is equipped with a variety of near and far range laser and camera based onboard sensors to perceive its environment, as well as the possible availability of overhead imagery and elevation data. We assume the robot's location is known to within several meters of its true position throughout navigation.

The robot's perception system is able to take advantage of ample computing resources to interpret the terrain through sophisticated algorithms that can be trained prior to this navigation on other terrain. However, there is no guarantee that training terrain will be representative of terrain encountered during a mission, even though there is an expectation of strong overlap.

We assume, however, that a non-expert human may be available for a nominal percentage of operation time *during* navigation to provide remote tele-operation support at the request of the robot. Other than this limited remote assistance, the robot must deal on its own with any environmental, seasonal or temporal differences that complicate its mission.

A combination of three metrics of performance are reasonable in such a domain:

Safety. The safety of the vehicle is of highest priority. At all times during testing, a human operator supervises the robot behavior and is ready to intervene if the vehicle poses a risk to itself or its environment. The occurrences of such human interventions are tracked and are viewed *extremely* unfavorably since they represent a high possibility of critical failure or unacceptable environmental impact.

Human assistance required. Human time is considered expensive and therefore should be limited to as small of a portion of total operation time as possible.

Time and distance traveled. As with most navigation tasks, we want to reach our destination while minimizing the overall time and distance of travel.

The approaches we present in this thesis are intended to allow a mobile robot to deal with the described domain while improving its performance under the above-mentioned metrics.

4 Technical Approach

4.1 High-Level Approach

We deal with the common scenario where a UGV must act safely and intelligently in a complex natural environment with little structure and limited or unavailable human involvement. As a UGV's autonomy system is forced to operate in an environment that it was not engineered or trained for, various aspects of its performance will inevitably degrade. We present a series of online algorithms to deal with the challenges discussed earlier and enable a UGV to better deal with uncertainty in order to improve performance and reduce risk to the vehicle.

We begin in Section 5 by proposing an online, probabilistic model to effectively learn to use potentially powerful but domain specific features by leveraging other features that, while perhaps otherwise more limited, generalize reliably. This technique allows the system to cap-

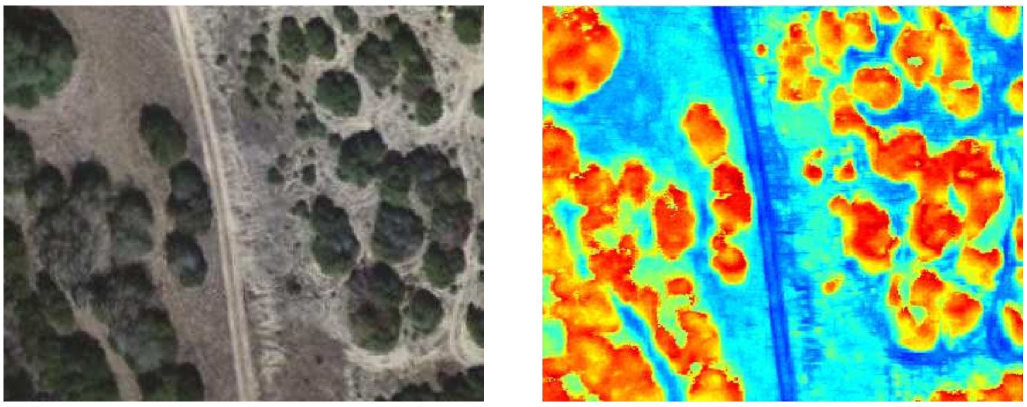


Figure 3: Sample results of online terrain traversal cost predictions from the Spinner UGV operating in Texas scrub lands using the system described in Section 5. 0.35 m resolution color overhead imagery used by our online learning algorithm (left) and corresponding predictions of terrain traversal costs (right). Traversal costs are color-scaled for improved visibility. Blue and red correspond to lowest and highest traversal cost estimates, respectively.

ture maximal benefit from common data sources such as overhead imagery without the need for human training as in other overhead interpretation techniques (sample results are shown in Figure 3). This efficient, self-supervised learning method allows a UGV to significantly extend the effective range of its onboard perception system in unfamiliar domains.

Small problems resulting in mildly sub-optimal performance are often tolerable, but major failures resulting in vehicle loss or compromised human safety are not. In Section 6 we consider the problem of online novelty detection for situations where even a well-adapting near-range perception system is potentially unfit to accurately evaluate the environment. We propose an online novelty detection algorithm that operates on onboard perception system based features and also has anytime properties allowing it to deal reasonably with time-critical situations. We also consider the problem of change detection, a location-dependent version of novelty detection where the goal is to identify when areas of the environment have significantly changed from a previous traversal. This capability is well-suited for UGVs tasked with navigating a route repeatedly as in the case of supply routes or patrolling. Such systems would be the first safeguards for UGVs by identifying potentially dangerous situations during traversal such as the situation shown in Figure 4. These approaches explore the idea of *deep memory*, the ability to be able to efficiently represent and utilize all previously encountered data.

Finally, in Section 7 we discuss an online approach to candidate selection where a system must choose online between one of several operating modes and explore several applications of such a system relevant to mobile robot navigation. We pursue this problem using an on-line, reinforcement learning approach. The system's goal is to learn to interpret available onboard and overhead sensor data in order to maximize its cumulative performance over the course of operation. This inevitably becomes a trade-off between exploring situations that will allow it to learn more about the world and exploiting those that appear to be optimal based on past experiences. In the case of learning to trade-off between autonomous and human tele-operation control, such a capability would enable a single operator to assist many UGVs, ensuring peak

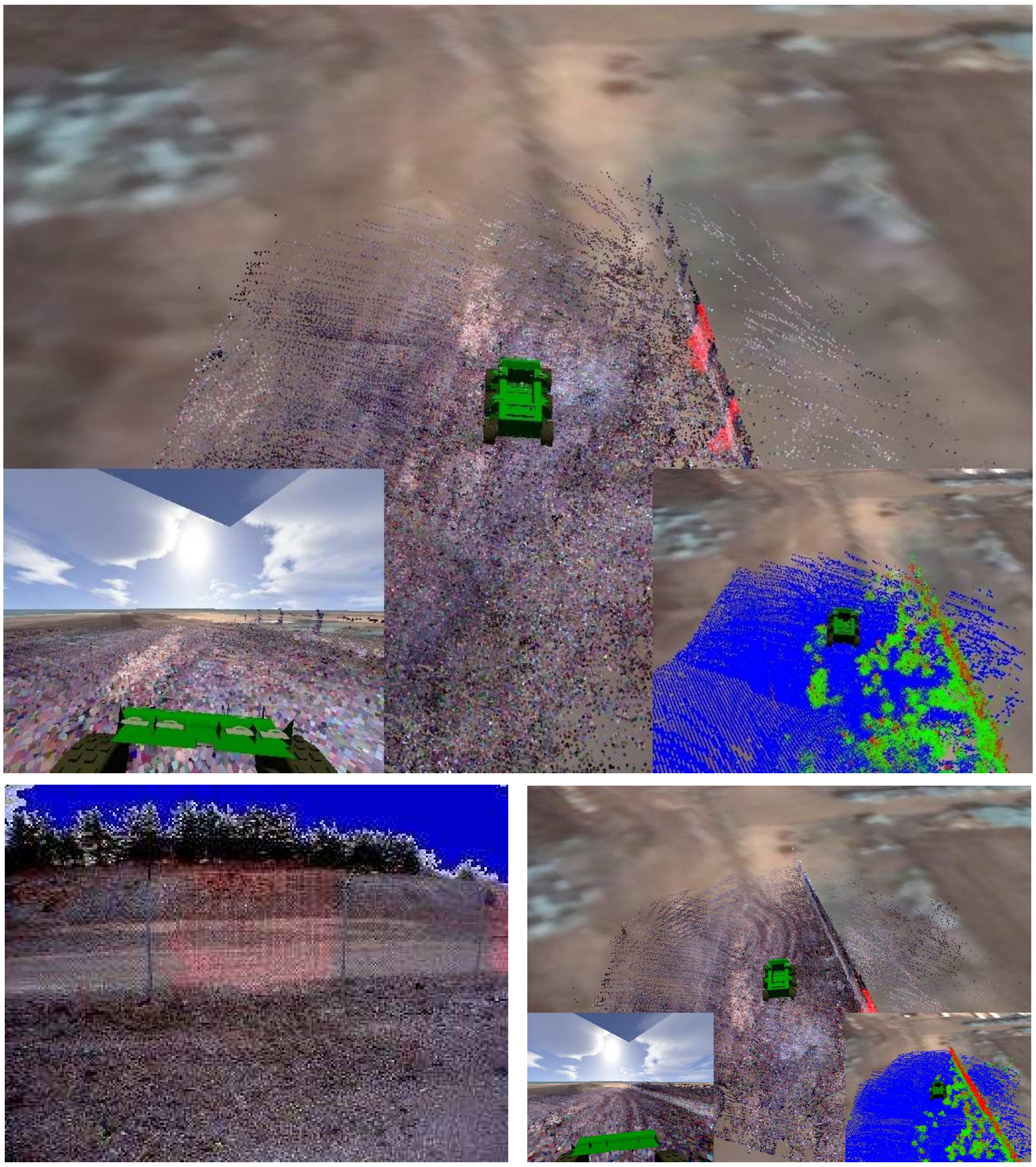


Figure 4: Sample result from online novelty detection algorithm onboard the Crusher UGV operating in western Pennsylvania. Chain-link fence was detected as novel (top and left, novelty shown in red) with respect to the large variety of terrain and vegetation previously encountered. After an initial stretch being identified as novel, subsequent portions of the fence are no longer flagged (right) due to the algorithm's online training ability. As with all future similar images, insets within the top image show a first-person view (left inset) and the classification of the environment by the perception system into road, vegetation, and solid obstacle in blue, green and red respectively (right inset).

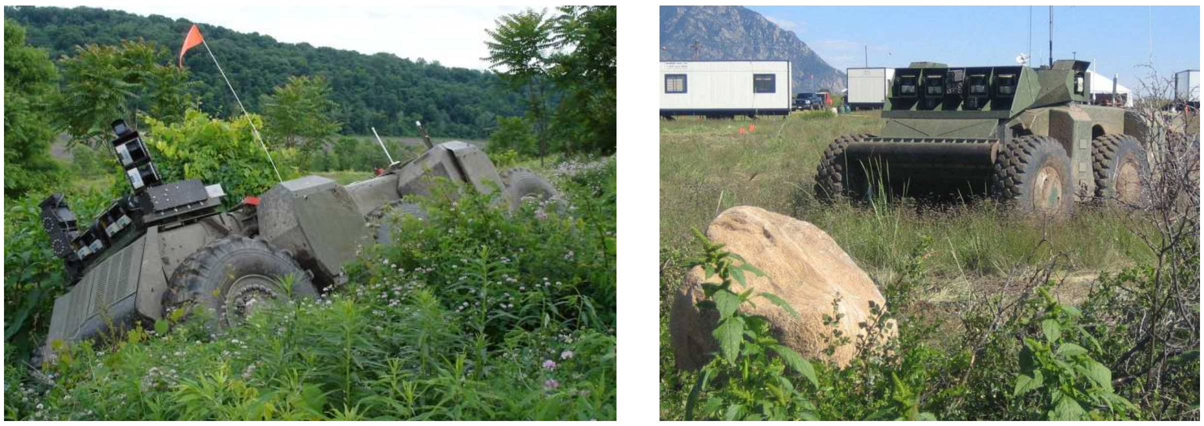


Figure 5: Spinner (left) and Crusher (right) robots used for experimentation throughout thesis work. The natural terrain shown here is representative of the domains they operate within.

performance for the entire team with minimal human involvement.

We concede that the complexity and unpredictability of the real world forces robotic systems to have to adapt online. The key is to identify the feedback that allows online training, whether it is one part of the system serving as an expert to train another or a human operator serving as the expert at opportune times. Such techniques then enable robots to adapt to and improve their performance in diverse environments with minimal human involvement and greatly expand the effectiveness and potential real-world applications of robotic systems.

4.2 System Architecture

It is important to introduce the system specifications on the UGVs we operate with as we will be referring to many aspects of these systems throughout later chapters.

This work is focused on improving autonomous navigation in complex outdoor environments where a mission will often consist of a desired destination that must be reached in a reasonable amount of time, with the possible availability of varying quality overhead data. The lack of structure such as lane markings on a road or the flat drivable surfaces and binary traversability assumptions of indoor environments lays much of the complexity of this problem on the UGV's perception system. It must now be able to not only identify the presence or absence of obstacles, but also accurately interpret the relative risks to safety and progress of each situation. Bushes and rocks may share similar qualities as seen by the perception system but differentiating between the two is vital to safe navigation.

4.2.1 Perception System

The Spinner and Crusher UGVs of the UPI Program that were used throughout our testing to date (shown in Figure 5) are intended for operation in complex, outdoor environments, performing local sensing using a combination of ladar and camera sensors [10]. Figure 6 outlines the high level data flow within the Spinner and Crusher autonomy systems.

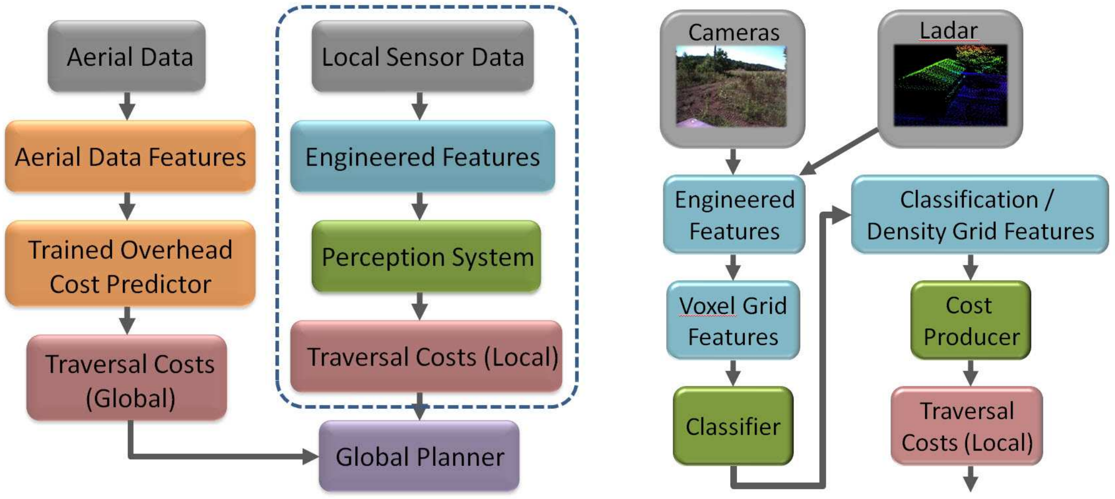


Figure 6: High-level system data flow (left) and a more detailed data flow of the perception system within the dotted box (right).

The perception system assigns traversal costs by analyzing the color, position, density, and point cloud distributions of the environment [11, 12]. A large variety of engineered features that could be useful for this task are computed in real-time (see Figure 7) and the local environment is segmented into columns of voxels (see Figure 8) in order to capture all potentially relevant information. This vertical voxelization approach is effective for mobile robots since the presence of specific features at certain vertical positions is highly relevant to their impact on traversal cost. For example, solid objects at wheel height are likely to be small rocks while similar features higher off of the ground are more likely to be trees or man-made objects. Each voxel, tagged with its corresponding features, is passed through a series of classifiers and combined with additional density-related features to create a compact set of intermediate features for each location in the world that is more suitable for traversal cost computation. The system then interprets these features through hand-tuned or learned methods to create a final traversal cost for that location in the world that can be used for path planning purposes.

These traversal costs are interpreted as relative measures of mobility risk (our robot works with traversal costs in the range of 16 to 65535). For example, the robot's on-board perception system assigns traversal costs of 16 (the minimum) to roads while grass is assigned a traversal cost of 48, implying the robot would be willing to take a detour of three times the distance in order to stay on a road as opposed to driving over grass. Meanwhile, dense vegetation is often assigned traversal costs of over 10000 in order to encourage the robot to traverse elsewhere except under extreme necessity. The robot re-plans its global path in real-time by finding minimum cost paths through the environment using the D* algorithm [13].

An extensive log playback and processing infrastructure allows us to develop and optimize many of our approaches offline.

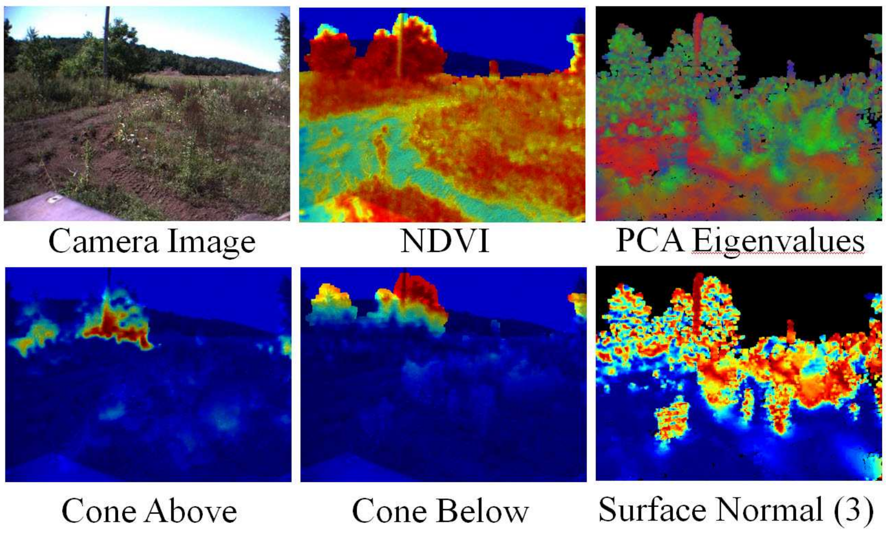


Figure 7: Example raw engineered features from the UGV's perception system. NDVI (normalized difference of vegetation index) is a useful metric for detecting vegetation [11].

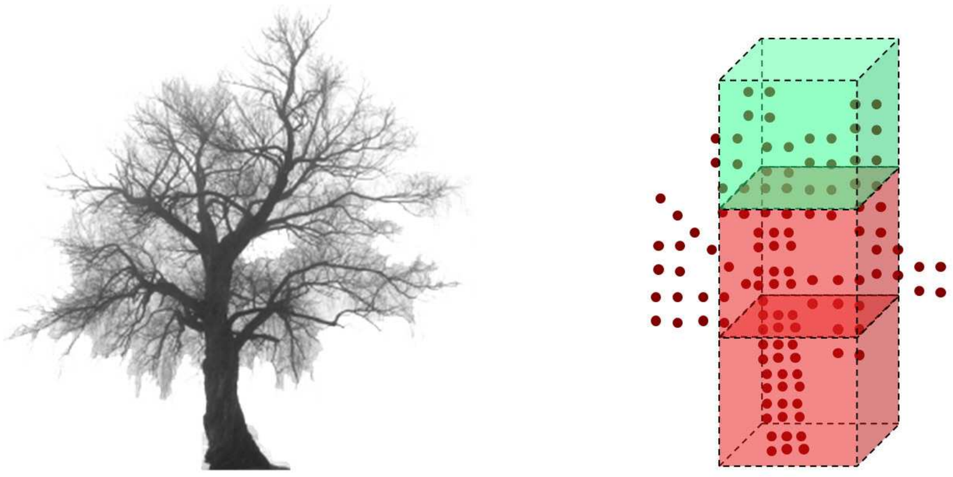


Figure 8: Illustration of the perception system's voxelization of vertical columns within the environment and subsequent classification. The voxels here are actually much smaller within the system but are enlarged for demonstration purposes. In the perception system, each voxel is a $20\ cm^3$ cube. Due to the size of the vehicle, 10 voxels in the vertical direction are computed at each location in order to include all potentially relevant information.

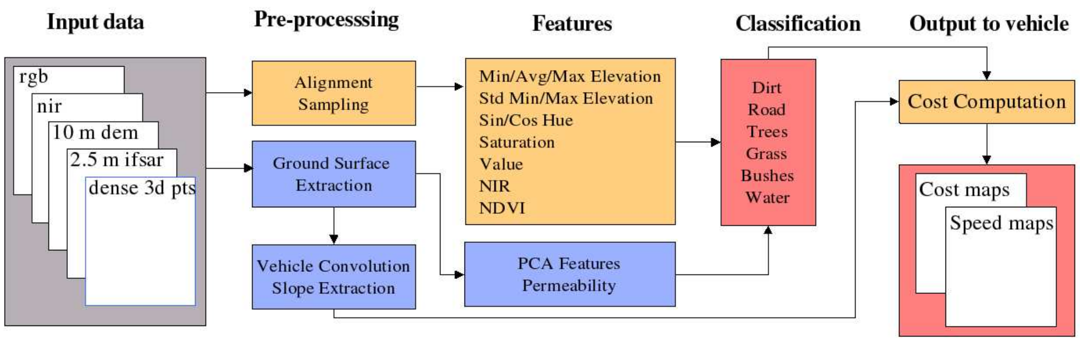


Figure 9: Overview of overhead data processing system: from raw data to cost maps.

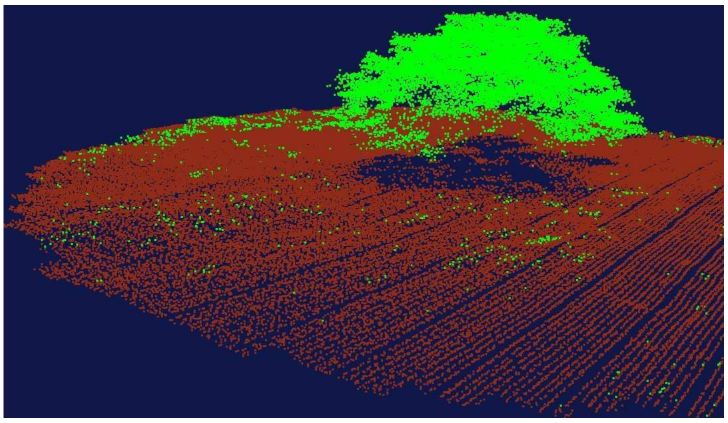


Figure 10: 3-D points are classified as ground (brown) or non-ground (green).

4.2.2 Overhead Data Usage

Improvements in both path safety and efficiency can be achieved by taking advantage of prior overhead data when available. Prior data can come from a variety of sources. Low resolution imagery (1 meter resolution gray scale) and topographic data (10 to 30 meter resolution) are already available for most of the world. In addition, higher resolution imagery and dense three-dimensional (3-D) data can be collected commercially upon request. Traversal cost maps for the environment can be produced from this data a priori using a hand-trained system or human demonstration for aiding online global path planning. The robot's onboard perception system is then used during navigation to fine-tune prior traversal estimates and adjust for areas of limited aerial visibility or changes in the environment from the time data was gathered.

In the human-supervised overhead cost prediction system, traversal costs are computed from a combination of semantic and geometric data as described in Figure 9 [14]. Semantic information of the terrain is obtained through supervised classification using features extracted from imagery and 3-D data [15]. A neural network with one input node for each feature, one hidden layer, and one output node for each desired classification category is used. Each terrain class is assigned a traversal cost by a human operator designed to mimic the behavior of the onboard perception system. Mobility analysis is then performed using the ground surface recovered from 3-D data (see Figure 10) or an available elevation map and traversal costs are

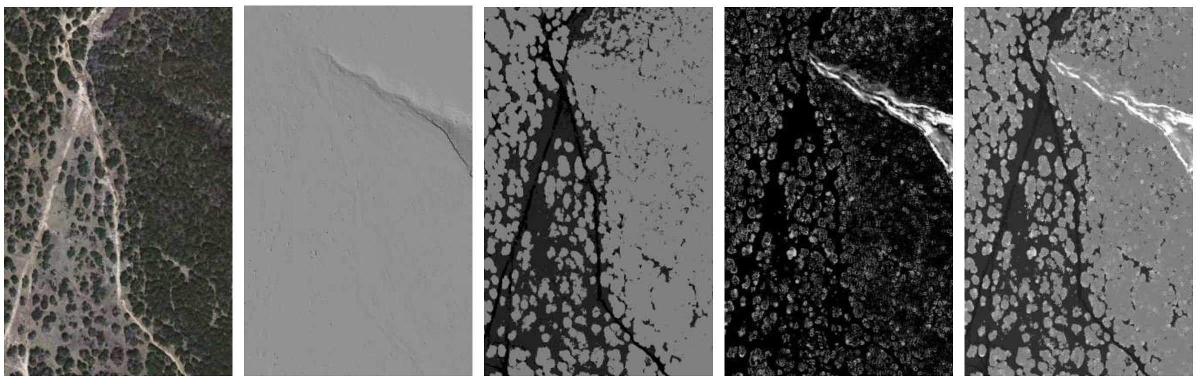


Figure 11: Sample results of cost map production for generally flat Texas terrain with a large vegetation variety. From left to right: an image of the environment, extracted ground surface, classification cost map, mobility cost map and final traversal cost map (sum of classification and mobility based cost maps). Each image is 368×595 meters.

assigned to reflect the capabilities of the vehicle based on computed parameters such as roll, pitch and ground clearance. Traversal costs are independently computed using the results of terrain classification and vehicle mobility analysis for each location in the world and summed to produce final traversal costs as shown in Figure 11.

Another approach used for taking advantage of overhead data sources is to have the system learn from demonstrated behavior [16]. Once provided with examples of how a domain expert would navigate based on the data, an imitation learning approach can learn mappings from raw data to cost that reproduce similar behavior. Because it is often difficult to pick and hand-tune traversal costs to achieve acceptable behavior, this approach produces cost functions with less human interaction and often better performance.

While these techniques for interpreting overhead data have been shown to significantly improve navigational performance, the requirement of manual retraining each time a new environment is encountered is a notable limitation that we address in this thesis.

4.2.3 E-Gator Platform

Due to the completion of the UPI Program, the on-board testing for proposed work will transition to the E-gator vehicles of the CTA project (see Figure 12). We are in the process of upgrading the autonomy system of the E-gator platform. This will be a significant effort but will greatly expand its perception and planning capabilities. The existing tilting SICK sensor will be supplemented with a high-performance camera system which will allow us to port to this system a subset of the UPI perception capabilities described in Section 4.2.1.

While the E-gator is in many ways a less capable platform than Spinner or Crusher, it requires significantly less overhead for deployment and testing. This will also allow us to test in more diverse environments including more urban settings.

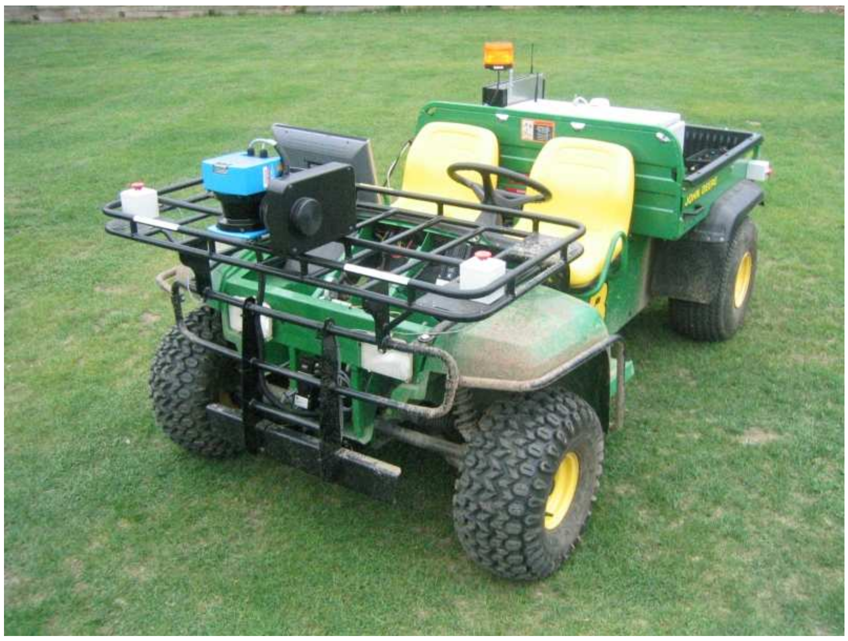


Figure 12: E-gator autonomous vehicle to be used for remaining experiments.

5 Onboard and Overhead Robot Perception in Unfamiliar Domains

Autonomous robot navigation in unstructured natural environments has been demonstrated extensively in a large variety of terrain, sensor payload and mission scenarios. Even though powerful at sensing, modeling, and interpreting the environment, these systems required significant tuning of parameters, either by hand or supervised training, to best adjust their algorithms to the local environment where the tests are conducted.

This highlights a common problem that arises in mobile robotics where potentially powerful sensor data and features are often difficult to take advantage of because they are situation or location specific. As mentioned previously, outdoor robot navigation can benefit from the now widespread availability of high quality overhead imagery and elevation data from satellite and aircraft. With this overhead data, many of the difficulties associated with autonomous robot operation can be alleviated, even with the coarsest of terrain resolution. Systems can then dispense with myopic exploration and instead pursue routes that are likely to be effective. Unfortunately, features computed from such sources can vary greatly due to diversity in terrain, environmental conditions and sensor varieties. This often invalidates pre-trained systems so the necessity for frequent manual retraining reduces the appeal of such approaches.

In much the same way, the complete use of onboard sensor data (such as ladar), if interpreted correctly, can help a robot make better decisions and increase traversal speed through improved knowledge of the environment. Unfortunately, as the complexity of environments increases, a robot's perception system must be engineered to evaluate its environment by first computing a set of intermediate features such as ground slope, object density, and vegetation

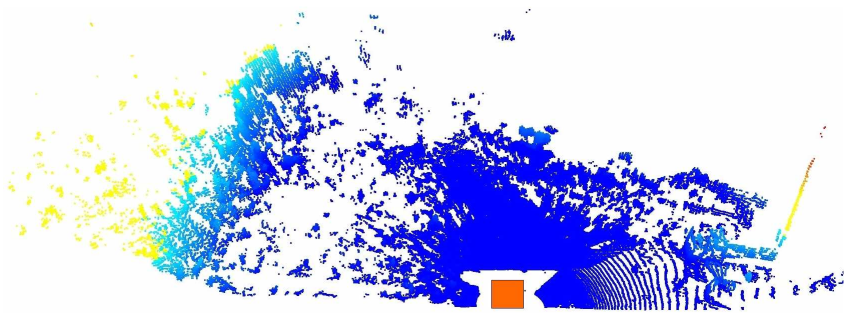


Figure 13: Typical ladar response from vehicle's perception system. Ladar points are color coded by elevation with lowest points appearing in blue and highest points appearing in yellow. Vehicle position is shown by the orange square. Notice the large drop in ladar response density (especially on the ground) as distance from the vehicle increases. Large objects such as the trees on the left generate ladar responses even at far ranges but are difficult to interpret through fixed techniques across different environments.

classification. While such features are consistent and generalizable, fixed techniques that successfully interpret onboard sensor data across many environments begin to fail past short ranges as the density and accuracy necessary for such computation quickly degrade and features that are able to be computed from distant data are very domain-specific (see Figure 13). This limits the effectiveness of such perception systems to a proximity of about 15 meters even though sensor data is often available at much higher ranges.

Building systems that can reliably interpret these scope-limited features is far from easy as even the smallest variations in lighting, season, terrain or even sensor calibration can have significant effects on data. Additionally, such estimates must be well calibrated with other onboard perception estimates of the terrain, or system performance may suffer. Developing approaches that can leverage these potentially powerful resources on an autonomous robot can significantly improve the versatility of many unmanned ground vehicles by allowing them to traverse highly varied terrains with increased performance.

One way to address such limitation is through on-line self-supervised learning where the autonomous system adjusts itself via perception and interaction with the environment. We propose to address the problem of learning and inferring between two heterogeneous data sources that vary in density, accuracy and scope of influence. The objective is to generalize from one data source, viewed as a reliable estimate, to be able to work with another, which may be high performance (e.g., long range or high accuracy) but difficult to generalize to new environments. We frame the problem as a simple, linear probabilistic model for which inference results in a self-supervised online learning algorithm that fuses the estimates from the two data sources. We also explore the advantages of this framework including reversible learning, feature selection, data alignment capabilities, reliable use of multiple estimates, as well as confidence-rated predictions [17].

5.1 Related Work

When navigating in an environment without full knowledge, robotic systems primarily rely on on-board perception systems. There are cases, however, when it is not possible to get an adequate understanding of the environment from the vehicle-based view of terrain without sacrificing speed or path optimality. For example, a vehicle navigating at high speeds in off-road environments may be unable to react to negative obstacles such as large holes and cliffs without the prior availability of overhead data. Even when the vehicle can safely navigate an environment, aerial sensing can dramatically improve path planning performance by detecting large obstacles such as buildings, forests and bodies of water as well as areas of preferable terrain such as roads.

Even low resolution prior data can provide significant improvement to vehicle performance, as demonstrated by numerous simulations in [18]. In [19], low resolution (25-30 meter) elevation maps were used to aid long distance planning. [20] demonstrated the extraction of features (roads, trees, water, etc.) from aerial surveys, for later correlation by an autonomous vehicle.

Similar research conducted by Charaniya et al. classified terrain into roads, grass, buildings, and trees using aerial LiDAR height data, height texture, and signal reflectance, achieving classification rates in the range of 66%-84% [21]. Cao et al. attempted to identify man-made objects from overhead imagery [22]. Knudsen and Nielson attempted to classify buildings using a previously available GIS database and RGB information for an environment [23].

The DARPA PerceptOR program contained an important prior data component. Aerial LiDAR data was used to predict vehicle roll and pitch over stretches of terrain, as well as to detect vegetation [24, 25]. This information was used to generate prior cost maps for use in global planning.

Within the UPI Program, several overhead data processing techniques were used extensively to achieve safe and efficient navigation over many kilometers (see Section 4.2) [14, 15, 16].

While such overhead data interpretation techniques proved crucial to successful navigation in some difficult environments, they were limited by the fact that they required human training or demonstration prior to use in new domains. Also, as discussed earlier, training onboard perception systems for general far-range use is infeasible due to the large variability in the features generated past short distances from the robot. As a result, such systems are often engineered or trained to perceive the environment in close proximity to the robot where features more often generalize well across many domains. In such cases the large portion of sensor data that cannot be interpreted through such techniques is often discarded.

Our approaches is one of several to use self-supervised learning to deal with the common robotics situation where data sources that, while highly relevant, are domain specific. For instance, camera imagery can potentially detect unpaved road in the desert significantly farther than some ladar-based systems can. Unfortunately, such data can prove very resistant to automated interpretation. In particular, classifiers that prove to be powerful indicators of road in a particular area often do not generalize to new conditions. Detecting such roads from a distance in a self-supervised manner proved to be a crucial component in Stanford Racings winning

Grand Challenge entry [26]. A similar version of this approach using reverse optical flow has also been proposed [27] and subsequently extended to off-road navigation [28].

Similarly, [29] presents a self-supervised approach for estimating terrain roughness from laser range data. Sensor, terrain and vehicle varieties contribute large errors that must be considered when making estimates. This system learns to model error by providing its own labels of terrain roughness in real-time from actual shock measured when driving over the target terrain. The vehicle in effect capitalizes on its ability to measure terrain roughness from the vehicle's inertial sensors to its range sensors.

Others have built systems that optimize parameters online using real-time performance as feedback. For example, such a technique allowed an adaptive motion planning system on an autonomous excavator to learn to perform at levels approaching skilled operators [30].

Numerous other research efforts have taken advantages of such techniques for obstacle avoidance using monocular camera [31, 32], estimating the depth from monocular imagery [33], automated learning of noise parameters in Kalman filters [34], terrain traversability classification [35], slip prediction [36], and estimating ground height from an autonomous tractor [37].

5.2 Approach

5.2.1 Formalization

We approach the problem of leveraging the powerful, but difficult to generalize, features in a Bayesian probabilistic framework using the notion of scoped learning [38]. The scoped learning model admits the idea of two types of features: “global” and “local”. Global features are generally useful, and their predictive power extends well to new domains, while local ones, which, although often very powerful, typically generalize poorly and are more difficult to take advantage of in a consistent way. These local features have *scope* that is limited to one particular domain. We wish to apply our system to extend the scope of such features to many possible domains. For our canonical problem of learning to leverage the extended range of overhead and far-range sensor data, these names may prove counter-intuitive, so we refer to them instead as *general* and *locale-specific* features. From this point on, ”global” and ”local” will refer to the proximity to the robot. Features generated from dense, vehicle-based ladar perception serve as our general features, while features generated from overhead based imagery and elevation data and far-range sensor data serve as our locale-specific features. The latter are particularly valuable to mobile robots because of their extended range and widespread availability.

Model. The scoped learning approach is a simple probabilistic model (shown graphically in Figure 14) that captures this notion of features that have scope. The outer plate L represents in graphical model notation that there are independent locales in which the model will be applied [39]. These correspond to new areas of the world in which our robot will operate.

Within the plate, we see a sequence of locale-specific features and corresponding general feature-based estimates. At each point in the sequence, we wish to make predictions about c (either all or a subset of them.) Here, c is the true variable we wish to predict, and $\tilde{c}$ is an estimate of that variable coming from the general features, while $\mathbf{x}$ are our locale-specific features. The parameters β common to the locale (plate) capture the relationship between locale-specific features and the variables of interest c. The length of our sequence is n.

This learning model captures the idea of self-supervised learning [40] in a Bayesian framework and extends the idea to integrate both the general feature-based estimates and the self-supervised locale-specific estimates. Driven by our application, we are particularly interested in the online *regression* case[1] where the goal is to learn to infer the true continuous values c_i in an online fashion as general feature-based estimates $\tilde{c}_i$ become available. We choose a simple model for c as a function of the k locale-specific features $\mathbf{x} = (x^1, \ldots, x^k)$ by modeling the distribution for c given $\mathbf{x}$ as a Gaussian with mean a linear function of $\mathbf{x}$ and with a variance of σ_l^2 :

$$E(c|\beta, \mathbf{x}) = \beta^T \mathbf{x} \tag{1}$$

We assume that the estimates from the general feature-based predictors have Gaussian noise and thus are distributed:

$$\tilde{c} \sim \text{Normal}(c, \sigma_g^2)$$

We take the σ_g and σ_l to be hyper-parameters lying outside the locale-specific plate.

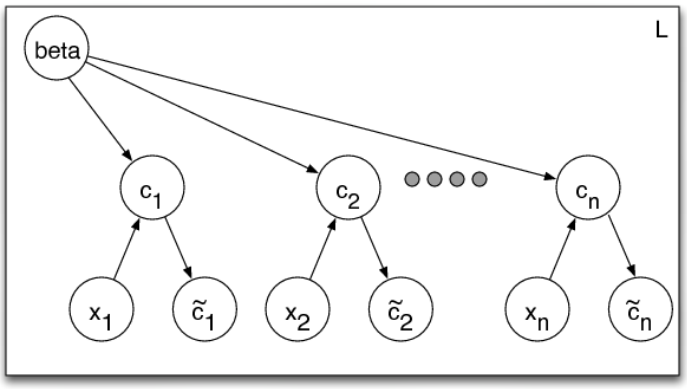


Figure 14: Graphical depiction of the scoped learning model. Hyper-parameters, including priors on the locale-specific parameters β and noise variances, lie outside the plate indexed by L and are not depicted.

To clarify this model using a real-world example, consider the case of learning online to predict traversal costs from overhead imagery data. In this scenario, c_i are the true traversal costs for locations in the world and $\tilde{c}_i$ are the general feature-based estimate for traversal costs of those location coming from the UGVs onboard perception system. These estimates are gathered online and are only available for a small subset of the locations in the world. The locale-specific features $\mathbf{x}_i$ are computed from overhead imagery for the environment and include color and texture based features as discussed later. In the case of overhead imagery, features $\mathbf{x}_{1\ldots n}$ are available for all location, but we do not have a way to interpret these features

[1]The original scope learning work [38] was developed in the context of classification using discrete features, generative descriptions of those features, and in batch.

a priori. We assume that c_i is a linear function of $\mathbf{x}_i$ governed by some weights β. As we traverse through the environment and observe additional perception system-based estimates $\tilde{c_i}$, we can refine online our best estimate of the mapping from overhead imagery-based features $\mathbf{x}$ to c.

Inference. We develop the inference for the model in an online fashion. Given a new data point $\tilde{c_i}$ estimating the true variable c_i, our goal is to compute new estimates[2] of the variables c_j, assuming we have already seen data $D = \{\mathbf{x}_{1\ldots n}, \tilde{c}_{1\ldots i-1}\}$. We can compute this by integrating over the uncertain parameters β which describe the relationship between the true variable and the local features.

$$p(c_j|\tilde{c_i}, \mathbf{x}_i, D) = \int d\beta \; p(c_j|\beta, \tilde{c_i}, \mathbf{x}_i) p(\beta|\tilde{c_i}, \mathbf{x}_i, D)$$

We can compute the required distribution over β as:

$$p(\beta|\tilde{c_i}, \mathbf{x}_i, D) \propto p(\beta|D) \int dc_i \; p(\tilde{c_i}|c_i) p(c_i|\beta, \mathbf{x}_i)$$

In our linear-Gaussian model, this can be understood as revising the posterior distribution from $p(\beta|D)$ in light of a Gaussian likelihood that takes into account noise from both general and locale-specific features.

Our computation of the posterior distribution $p(\beta|\tilde{c_i}, \mathbf{x}_i, D)$ is as follows. We first initialize our distribution to the prior distribution $p(\beta)$. Then, for every training example i, we multiply our distribution by $p(\tilde{c_i}|\beta, \mathbf{x}_i)$. Since the prior distribution and $p(\tilde{c_i}|\beta, \mathbf{x}_i)$ are normal, the posterior distribution is also normal. We use the notation $\hat{\beta}$ to represent the mean of the posterior distribution and V_β to represent the variance. Thus, computing $p(\beta|\tilde{c_i}, \mathbf{x}_i, D)$ is performing a self-supervised learning using a Bayesian linear regression model with noise variance $\sigma_l^2 + \sigma_g^2$.

We use our current estimate of the posterior distribution when we want to predict a future outcome c_j. We are interested in predictions in two cases: first, when we have no general feature-based estimate $\tilde{c_j}$ for a particular c_j, and second, when such an estimate is available. In the first case, the predictive distribution $p(c)$ has mean $c_p = \mathbf{x}^T\hat{\beta}$ and variance $\sigma_p^2 = \sigma_l^2 + \mathbf{x}^T V_\beta \mathbf{x}$ [41]. When we also have an estimate $\tilde{c_j}$, inference combines these two estimates:

$$p(c_j) = \text{Normal}(\sigma_p'^2(\frac{c_p}{\sigma_p^2} + \frac{\tilde{c_j}}{\sigma_g^2}), \sigma_p'^2)$$

where

$$\sigma_p'^2 = \frac{1}{\frac{1}{\sigma_p^2} + \frac{1}{\sigma_g^2}}.$$

We note that it is possible to compute the posterior distribution in batch, but we prefer to maintain an estimate of the posterior distribution as we receive general feature-based cost estimates so that we may immediately apply our algorithm to new data.

[2]We assume that our prior on β is a priori independent of the features $\mathbf{x}$ so that inference will remain the same even in the case where the features become available in some sequence.

5.2.2 Advantages of the Bayesian Learning Approach

Using the online Bayesian scope learning model provides a number of important benefits.

Confidence Rated Prediction. The variance estimate provided by our algorithm for the probability of each c can be used as a metric of confidence in the prediction. If a situation arises in which we must choose which one of several predicted outcomes to trust, we could simply use the one with the smallest variance.

Learning of the Hyper-Prior and Feature Selection. Our algorithm depends on a number of hyper-parameter terms that may be chosen based on data from multiple locales. We discuss ways to choose the noise variance terms σ_l and σ_g in section 5.2.1 and the prior distribution on parameters β in section 5.3.3. The prior distribution $p(\beta)$ is an isotropic Gaussian that is independent for each weight, and each weight is dependent on a shared hyperparameter α that moderates the strength of our belief over the values our weights β might take. That one α value controls the inverse variance of each weight β. We can modify our prior $p(\beta|\alpha)$ to consist of K hyperparameters, with each α_k independently controlling the inverse variance of each weight and define hyperpriors over all the α_k values. We can then use Tipping's hyperparameter re-estimation algorithm to do feature selection (see Algorithm 1) [42]. In this way, we can both automate feature selection and bias our algorithm to prefer certain features for new locales [43].

Algorithm 1 Hyper-parameter re-estimation procedure

```
1: given: Initial values for all $\alpha_k$ and $\sigma_l^2$
2: while $\alpha$ has not converged do
3:     Compute the mean $\hat{\beta}$ and covariance $V_\beta$ of the distribution of our weights $\beta$
4:     For all $K$ features, $\gamma_k \leftarrow 1 - \alpha_k V_{\beta kk}$
5:     For all $K$ features, $\alpha_k \leftarrow \frac{\gamma_k}{\hat{\beta}_i^2}$
6: end while
7: return $\alpha$
```

Once our α_k values have converged, we can remove any feature x_k with a corresponding optimal α_k value that tends toward ∞. Since an α_k value controls the inverse variance of each weight β_k, an α_k value that tends toward ∞ implies that the mean of the weight β_k value tends toward 0. Thus, we can remove that feature x_k since its weight β_k value of 0 would remove its contribution to predicting the output $\tilde{c}$.

Reversible Learning. A problem that often arises in online learning is the handling of multiple estimates of a particular quantity. For instance, in our canonical example, our general feature-based estimates $\tilde{c}_i$ may improve as we get closer and denser laser readings of the terrain. It is not appropriate to treat these as independent training examples: while they may differ in their variance, they are generally highly correlated. Neither is it useful to simply take the first estimate available: often this is a poor substitute for all the data. In our model, since we maintain an exact posterior distribution that lies inside the exponential family, we may effectively *remove* the effects of training on a data point by dividing out the likelihood term we had used to include it in the posterior [41]. In this way, we always have an estimate of the

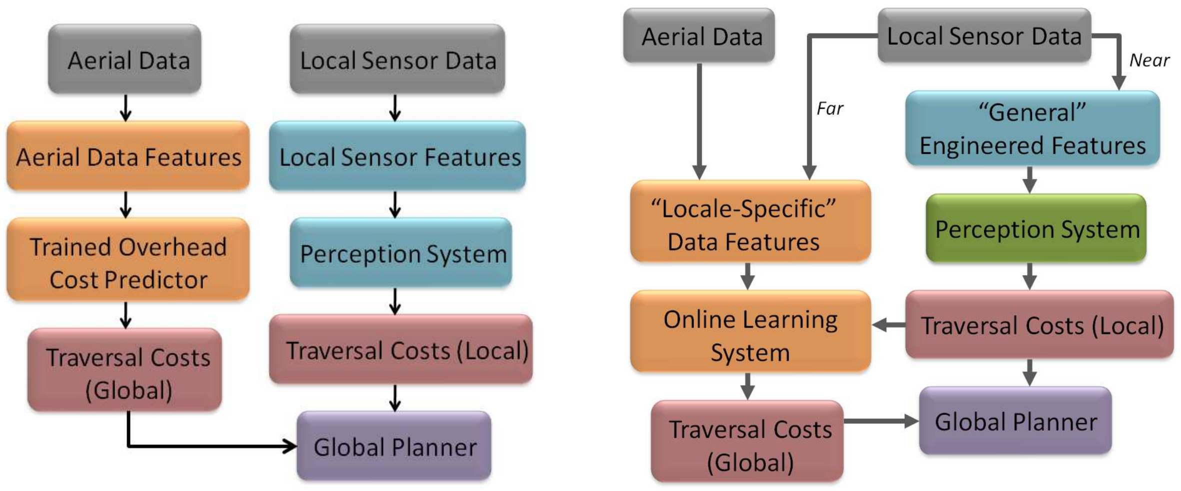


Figure 15: In the baseline system (left), aerial data can only be used after the cost prediction module is trained either through human labeling or demonstration. Utilizing the online learning framework (right) allows the perception system to learn mappings from locale-specific data features (overhead or far-range sensor data) to locally computed terrain traversal costs (computed from general features) and make prediction elsewhere in the environment.

posterior distribution of β using the current best estimate $\tilde{c}_i$. Minka has developed an alternate use of this "removal trick" for approximate inference [44].

5.3 Application to Mobile Robotics

We demonstrate this approach in the context of long range navigation onboard our rugged, all-terrain UGV during various field tests in complex natural environments. These environments in western Pennsylvania and Texas consisted of a variety of vegetation ranging from grass and short bushes to large, dense forests as well as other diverse obstacles such as large rocks, hills, dunes and ditches.

The information flow within the baseline system discussed in Section 4.2 is modified as shown in Figure 15. As the UGV traverses an environment, it utilizes its on-board perception system and these difficult to interpret features (computed from overhead imagery and elevation data or far-range sensor data) to learn the mapping from these features to computed terrain traversal costs in order to predict traversal costs elsewhere in the environment where only overhead data or far range sensor data is available, effectively extending the range of the vehicle's local perception system and allowing more effective navigation of the environment.

5.3.1 Terrain Traversal Cost Prediction

We chose to predict traversal cost rather than intermediate results such as slope, density, or presence of vegetation because traversal cost is the metric that most closely governs a vehicles navigation strategy through an environment. Our robots perception system is proficient at effectively assessing terrain traversal costs, so it is desirable to be able to mimic its predictive

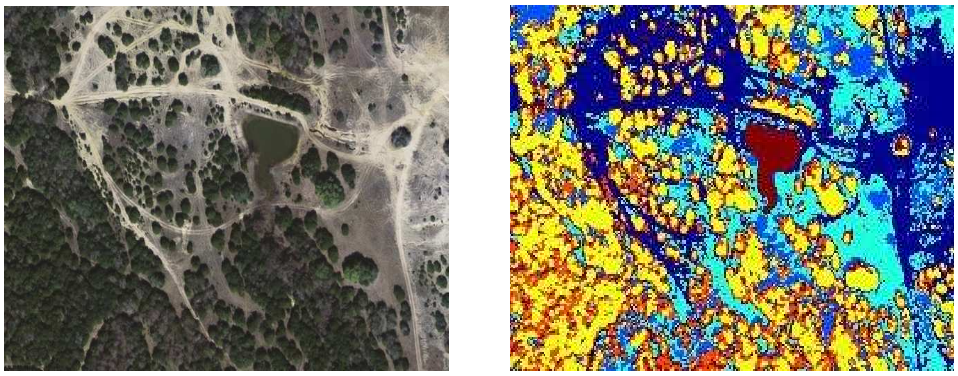


Figure 16: Sample clustering results from using the Gaussian Mixture Model algorithm on generated features. Overhead color imagery data used to generate features (left) and resulting clustering into six clusters (right). Membership features were generated by computing the fractional degree of membership of each pixel in each cluster.

abilities. We therefore use estimates from the robots perception system to evaluate the accuracy of traversal cost predictions.

Overhead Data Features. A set of feature maps for the vehicle's environment was generated from each overhead data source for use as inputs to the algorithm (these are our *locale-specific* features as defined in Section 5.2.1). In our implementation, HSV (hue, saturation, value) features were used to represent color imagery data while the pixel intensity of the black and white imagery data was used as a single feature. Raw RGB (red, green, blue) color data was inadequate for our approach due to its sensitivity to illumination variations. A hue (in degrees) of α was represented by the pair of values, $sin(\alpha)$ and $cos(\alpha)$ to address the continuity problem at hue values close to $0°$.

A feature containing the maximum response to a set of ten Gabor filters at various orientations centered at each pixel was also generated to capture texture in each type of imagery. Additional features for the black and white imagery data were generated by computing the means and standard deviations in intensity within windows of five meters around each pixel. Additional elevation-based features were computed as described in Section 4.2.2 and [14] when such data was available. All features were rescaled to the $[-1, 1]$ range and a constant feature was also included.

Finally, clustering of all previously computed features was performed that helped the algorithm to overcome the limitations of a linear model and easier identify patterns in the feature input space. Clustering of the input data was done through Gaussian Mixture Models in order to generate membership features by assigning each data point a fractional degree of membership in each output cluster (see Figure 16) [45]. Six clusters were chosen for our implementation.

The characteristics of an environment change with varying conditions. However, even outdated overhead data can be useful since most distinct areas in an environment will maintain uniformity in their characteristics despite these variations. By relaxing restrictions on the recency of overhead data, our algorithm further increases its impact on improving robot navigation.

Far-Range Sensor Data Features. Ladar and camera data were used to create the far-range sensor features used for training (once again, these are our *locale-specific* features as defined in Section 5.2.1). Ladar points in the environment were tagged with color values from the camera sensors by computing the pixel the ladar would appear in within the camera image. Color features were computed just as with overhead data. Additionally, the positions of the ladar points were used to compute the maximum vertical point spread and standard deviation of point heights. In order to incorporate contextual information about the neighbors of a location, similar features were computed from the location's neighbors within a small window. A constant feature was included as well. As a robot travels toward a location, features for that location are computed regularly (to account for variations in features due to distance from the robot) and stored to be used as possible future training examples.

Finally, it should be noted that the *lack* of ladar data in an area in front of the robot can serve as a confirmation of free space since ladar hits are rarely available on road or grass past about 30 meters due to the angle with respect to the sensor. We identify such free space by tracing away from the robot until a sensed object is encountered (up to 40 m away) and setting a tracing feature in all encountered cells. This features is the sole feature for such areas.

In some applications, subsets of features may not always be available. For example, in the case of far-range sensor data, some laser data may fall outside of the field of view of the onboard cameras making color-based features unavailable for those locations. In such a scenario, an independent instance of the learner can be trained simultaneously for each potentially possible set of features. New examples can then be sent to the learner that is trained to handle its available feature set.

Such techniques may be used to generate features from any combination of data sources gathered through a variety of methods.

5.3.2 Training and Prediction

Because traversal costs act as distance ratios as described in Section 4.2.1, errors in traversal cost estimates in low-cost areas are more detrimental than similar errors in high-cost areas. An error of 100 to an area of extremely high traversal cost would have negligible effect, while the same error at an area of desirable terrain would radically change the behavior of the robot.

In order to work with the linear model used by our algorithm, we deal with traversal costs within our algorithm on a logarithmic scale, converting from the normal traversal cost space for the purposes of training and prediction. The Gaussian error assumption embedded in our probabilistic model is a much better approximation when we measure error on this scale. Unlike in the regular traversal cost space, small errors in the log space lead to small errors in the traversal distance ratios.

Training examples are constructed from $\mathbf{x}_i$, the vector of feature values from either overhead data or far-range sensor data, and $\tilde{c}_i$, the average of all traversal cost estimates that have been calculated within the corresponding area. As with many robotic systems, the performance of our robot's on-board perception system quickly degrades as the distance from the robot increases (due to the lowered accuracy and density of sensor data), so the quality of a training example is measured by its proximity to the robot. Rather than struggling to decide at which point to utilize an example for training, the reversible learning capabilities of our algorithm allow us to maintain an optimal level of predictive abilities by ensuring that only the highest quality data available impact its state. As the robot approaches locations that had previously been used for training, obsolete examples are *unlearned* in favor of higher quality training examples available for those areas. Estimates greater than 12 m from the robot are ignored since such estimates are very unreliable and would only corrupt the quality of training in cases where they cannot be replaced with better estimates.

An example of this training process can be seen in Figure 17. As the vehicle explores more of the environment, the greater sample of training data allows it to more accurately interpret the locale-specific data sources. Notice how the shadow from the tree at the top right is initially estimated incorrectly as a very high-cost area (Figure 17c) but as the robot explores more of the environment, it begins to recognize its error and lower its estimate (Figure 17d).

As the algorithm acquires more training data, its predictive performance improves, allowing it to revise previously made traversal cost estimates. The algorithm specifies a degree of confidence for each prediction based on the similarity of the example to past training data (as indicated by the variance estimate), so predictions in which the algorithm lacks confidence can be ignored in favor of an alternative source of predictions or a default value. Notice how in Figure 17b the algorithm is able to identify its estimates for the trees in the environment as areas of low confidence (shown in blue) until the robot first encounters the tree below its starting position and is able to refine estimates for similar areas.

5.3.3 Applications of Trained Algorithm

This algorithm can be used in a variety of ways to aid in unmanned ground vehicle navigation, both in real-time on-board a robot and offline once it has been trained. In the case of online terrain traversal cost prediction, the algorithm can be used to periodically update traversal cost estimates within a region around the robot where features have been computed so that a real-time path planning algorithm such as D* can revise the vehicle's global path to account for the changes. As shown in the following section, the use of this algorithm to extend the robot's field of view results in significantly shorter and more intelligent paths.

When using overhead data, one can make traversal cost predictions for a large area without ever having to traverse or acquire training examples from that area beforehand since the predictive state of the algorithm can be captured at any time by the vector β at that moment. Instead, as long as identical features are computed for the two areas, one can simply drive through a representative area for a short period of time in order to train the algorithm to make predictions in a much larger area. We will also show that a priori traversal cost maps produced by this technique can be more accurate than even those produced from hand-trained techniques that utilize superior data sources.

It is important when using overhead data that the data be aligned with the estimated position of the robot. Even slight mis-registration can significantly hinder the performance of algorithms such as ours that are sensitive to such errors. An advantage of using an online

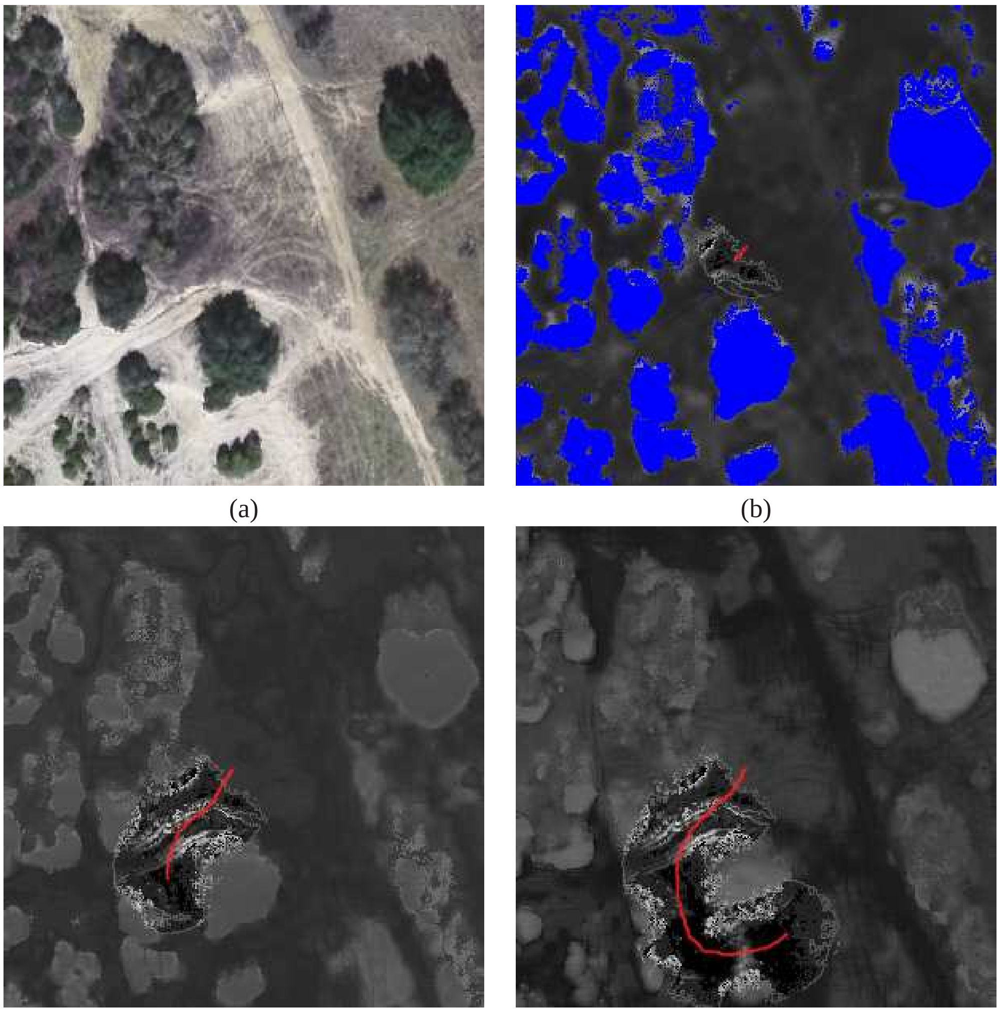

(a) (b) (c) (d)

Figure 17: Training progress of online learning algorithm using overhead color imagery data for traversal of environment shown in (a) is shown in (b) - (d). Dimensions of shown areas are 150 m $\times$ 150 m. Accumulated ground truth traversal costs computed by the robot's on-board perception system and vehicle path (shown in red) are overlaid on estimated traversal costs generated by the algorithm. Lower costs appear as darker colors and predictions that the algorithm lacked confidence in (due to insufficient representative training examples) are shown in blue.

Bayesian linear regression model is the ability to detect and correct such misalignments with relative ease.

When predicting a new traversal cost c_j, the model creates a predictive distribution $p(c)$ with a mean μ_p and a variance σ_p^2. Evaluating the predictive distribution at the traversal cost c_i of a training example gives the probability of having seen that traversal cost given its corresponding feature vector. We can use the probability of having seen all of our data, $p(\tilde{c}_1, \ldots, \tilde{c}_n)$, to detect map misalignments between overhead data sources and estimated vehicle positions by searching through a space of potential alignments for the one that maximizes the probability of the data.

Since $p(\tilde{c}_1, \ldots \tilde{c}_n)$ can be computed via the chain rule as the product of the predictive distributions evaluated at every $\tilde{c}_i$ used for training, the log data probability is the cumulative sum of $-\log \sigma_p - \frac{(y-\mu_p)^2}{\sigma_p^2}$ for every predictive distribution. After all examples have been received, we compute the average log data probability over all training examples and use this to compare against other alignments. As shown in the following section, correcting such misalignment produces traversal cost maps with better defined obstacles that more accurately reflect the true environment.

Note that for the results in the following section, we chose the hyper-parameter for noise variance σ_l^2 with ML-II and chose an isotropic Gaussian with high variance for the prior on β based on observations from previous robot traversals [41].

This approach allows increased versatility of many UGVs by allowing them to take advantage of data sources that are often ignored while improving performance and removing the necessity of human involvement and parameter engineering.

5.4 Work To Date

When dealing with overhead data, our algorithm will be referred to as MOLL (Map On-Line Learning)[3] and when dealing with far range sensor data, our algorithm will be referred to as FROLL (Far-Range On-Line Learning). All imagery data was gathered from satellite on average several months prior to traversal and all elevation data was gathered by surveying from helicopter. Both MOLL and FROLL were run on a 1.8 GHz processor with 2 GB of memory. Because of the large amount of aerial data potentially required by MOLL, we implemented a paging system so that only currently relevant areas of overhead data are kept in memory.

5.4.1 Field Test Results

The algorithm was tested with both overhead data and far-range sensor data in real-time onboard our unmanned ground vehicle to measure its impact on navigation performance. The test environments contained a large variety of vegetation, various-sized dirt roads (often leading through narrow passages in dense vegetation), hills, and ditches. The vehicle traversed series of courses defined by series of waypoints by using only its on-board perception system

[3]This algorithm at times has also been referred to as OOLL (Overhead On-Line Learning)

Table 1: Statistics for Course Traversals With and Without the Online Learning Algorithm

	Without Algorithm	With MOLL	With FROLL
Total Traversal Time (sec)	1370	1001	898
Total Distance Traveled (m)	1816	1682	1575
Average Speed (m/s)	1.33	1.68	1.75
Number of Interventions	1	0	0

for navigation. It then traversed the same courses with the help of MOLL, first with 40 cm resolution overhead imagery and elevation data to supplement the on-board perception system with traversal cost updates computed within a 75 m radius once every 2 seconds and then with FROLL used to interpret and make predictions from far-range sensor data every 1 second. The algorithm was initialized for each course with no prior training (see Figures 18 and 19 for sample results). Notice how in Figure 18 the MOLL path shows how the robot learned to avoid the dense area of trees after its initial encounter with the area immediately chose to follow the road to the goal. The FROLL path shows how the robot chose a similar path to the baseline system but was able to give up on dead ends quicker and was able to avoid the large detour at the end of the course due to its extended perception range.

As shown in Table 1, our algorithm allowed the vehicle to complete the courses using MOLL in 27% less time while traversing 7% less distance and with FROLL in 34% less time while traversing 13% less distance. Additionally, while we were forced to manually intervene during the tests with only the perception system in order to correct the vehicle's heading when it became trapped in heavy vegetation and could not escape on its own, no manual interventions were necessary when using our algorithm.

While it appears from the shown statistics that FROLL overall resulted in more effective paths than MOLL, we found that with MOLL, the vehicle chose to drive further distances on more preferable terrain in order to avoid difficult or dense areas that presented a larger possibility of damage to the sensors or the need for human intervention. Such an instance an be seen in Figure 19a.

Figure 19b shows a situation where MOLL can be most useful. Since each traverse began with an untrained algorithm, the MOLL course was given an additional waypoint at the right of the image in order to give it an opportunity to train on a small sample of the environment. As seen in Figure 19d, this small amount of training allowed it to identify the wall of trees that heavily hindered the progress of the vehicle in the traverses using only the perception system or FROLL.

In general, we found that both techniques not only improved the quality of the paths chosen by the vehicle but also allowed higher speed navigation by increasing the time the vehicle had to react to upcoming obstacles and identifying safer terrain such as roads. The demonstrated quantitative impact of these algorithms, however, cannot accurately capture their potential impact on overall performance. Rather, it is important to understand the types of situations that a UGV may encounter for which these algorithms will be most beneficial. If obstacles in an environment are fairly sparse or largely known prior to navigation, extending the range of the

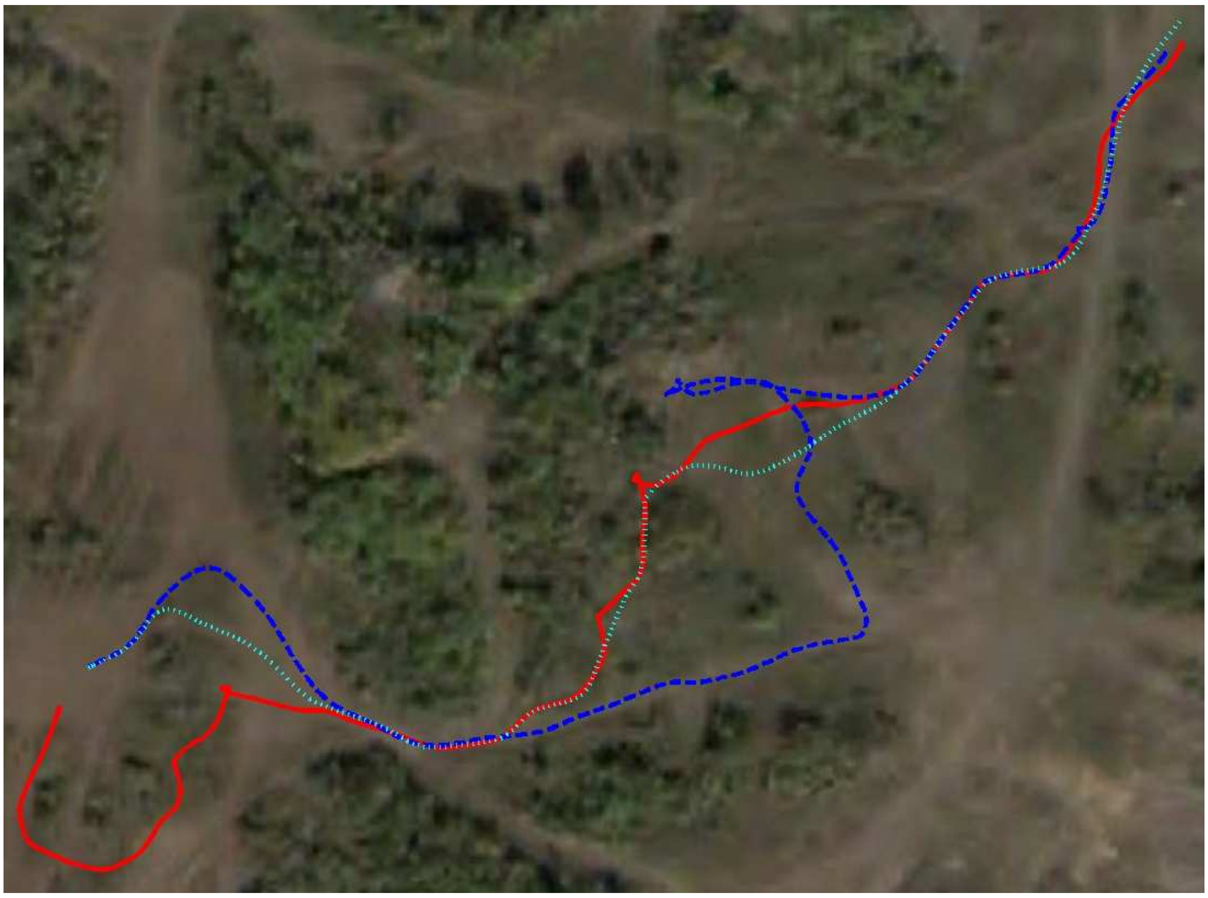


Figure 18: Comparison of paths executed by our robot for shown course when using only on-board perception (in solid red), and with MOLL (in dashed blue) and FROLL (in dotted cyan) used in real-time on-board the robot. Course started at the top right and ended at the bottom left.

perception system will not significantly improve performance metrics. On the other hand, if the environment is full of dense obstacles, hazards and cul-de-sacs with less visible routes towards the goal then the degree of benefit when using such approaches can be arbitrarily large.

5.4.2 Field Test Data Post-Processing Results

MOLL was also used to produce a priori traversal cost maps for a multi-kilometer course over a large area of complex terrain with heavy vegetation and elevation obstacles defined by a series of GPS waypoints (see Figure 20). The algorithm was trained for about 7 minutes using two types of overhead imagery data by driving the vehicle by remote control at about 5 meters per second through the training course outlined by the red box in Figure 20a. The trained algorithm was then used off-line to generate a traversal cost map and plan an initial path through the much larger course. Closeups of generated traversal cost maps and resulting planned paths are shown. For comparison, we also included the resulting path from a traversal cost map generated by a supervised learning algorithm with human-picked examples from the actual course and features generated from both overhead imagery and high-density elevation

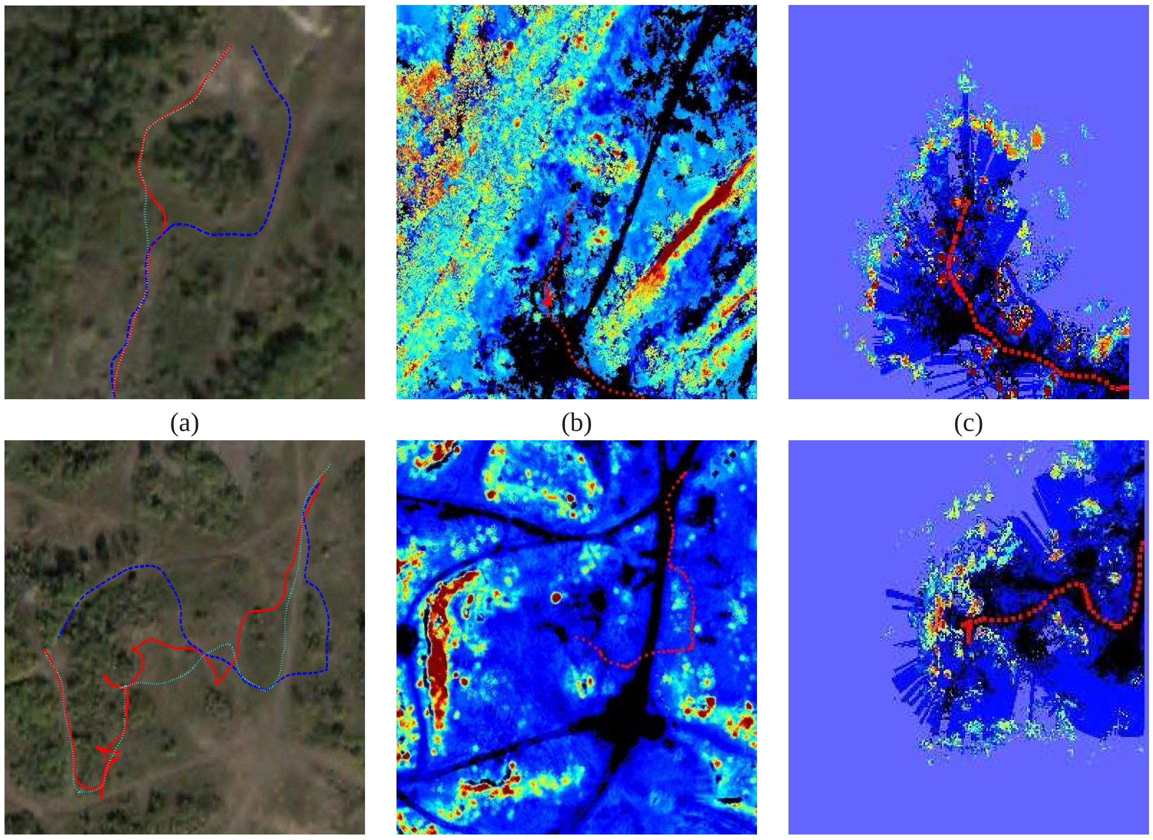

(a) (b) (c)
(d) (e) (f)

Figure 19: Comparison of paths executed for shown situations when using only on-board perception (in solid red) and with MOLL (in dashed blue) and FROLL (in dotted cyan) are shown in (a) and (d). In (a) the course started at the bottom and ended at the top and in (d) the course started at the top right and ended at the left. Predictions of terrain traversal costs for the environment by our algorithm at the times the vehicle chose to avoid the large obstacles in front of it are shown for MOLL in (b) and (e) and for FROLL in (c) and (f). Traversal costs are color-scaled for improved visibility. Blue and red correspond to lowest and highest traversal cost areas, respectively, with roads appearing in black. In (a) the MOLL path chose to travel slightly further on road in order to avoid the more difficult passage to the left while the FROLL path was able to detect the opening to the left much sooner than the baseline path. In (b) MOLL helped the vehicle avoid the area of dense trees by executing a path that is 43% shorter in 73% less time.

Table 2: Types of Overhead Data Used by Overhead Online Learning (MOLL) and Hand-Trained Algorithms Used To Produce Prior Cost Maps

Algorithm	Data Used	Resolution
MOLL (color)	Color imagery	0.35 m
MOLL (B & W)	Terraserver B & W imagery	1.0 m
Human-Supervised	Color imagery Elevation	0.35 m < 0.2 m

data (see Table 2 for a description of data sources) [14].

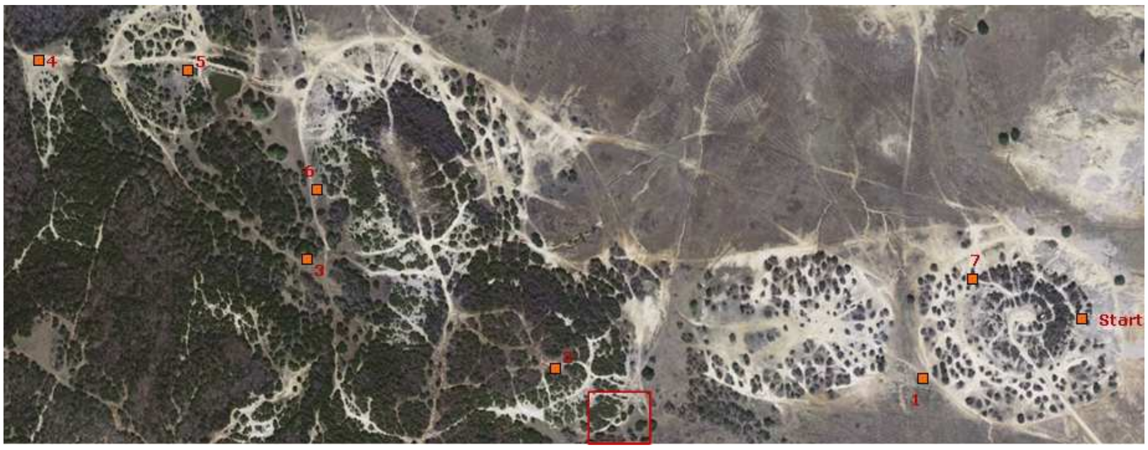
We evaluated the performance of MOLL against the human-trained technique by accumulating all the traversal costs generated by the vehicle's on-board perception system during a traversal of the course shown in Figure 20 and comparing those costs to the estimates from each of the generated prior cost maps. The average absolute error in traversal cost (on a log scale as described earlier) for each method is shown in Figure 21 as a function of training time. This result shows that MOLL is competitive with respect to the human-trained algorithm using only imagery data after only a short period of training. However, it should be pointed out that maintaining a tight correspondence from traversal costs assigned by the human-trained algorithm to those assigned by the perception system was difficult to strictly enforce. This highlights another advantage of the online learning approach over a human-trained approach: by relieving the need for manual manipulations of traversal cost assignment strategies, the entire system is more adaptable to changes in both the environment and the perception system.

During post-analysis of this test, we discovered that the overhead imagery data and the estimated position of the vehicle were in fact misaligned by about 1.5 m. While this result shows that our algorithm is robust to such map misalignment, this article also demonstrates how our algorithm can be used to detect such errors in alignment in order to achieve optimal performance.

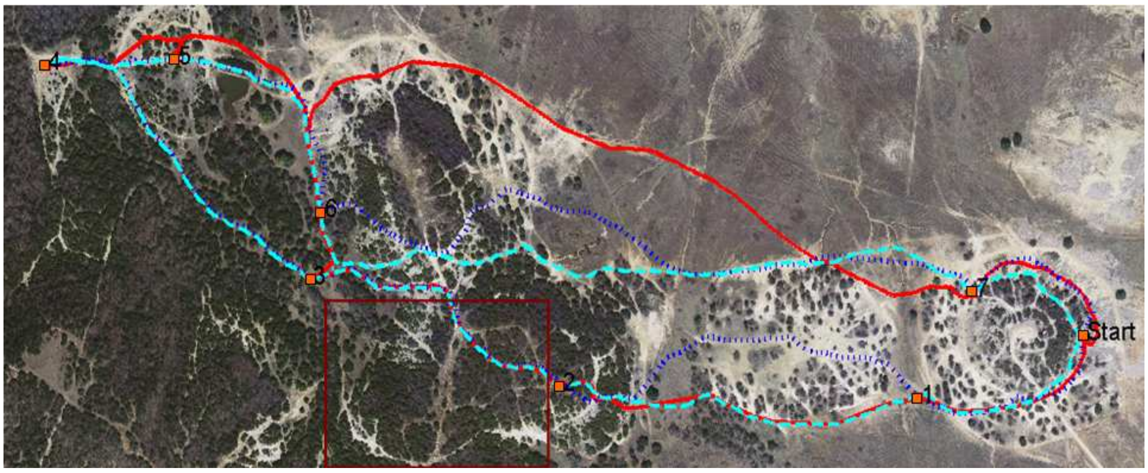
Performance of FROLL was similarly evaluated by comparing its estimates to all computed perception costs during a course as a function of training time. Only estimates that had not been used for training yet were included in this calculation (so as not to use the training set for testing). The results can be seen in Figure 22. Notice how after only three minutes of training the algorithm has mostly converged to its final predictive performance.

5.4.3 Offline Map Alignment

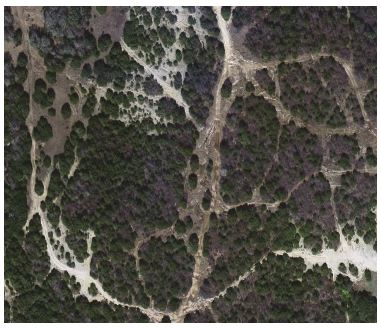
We applied our map alignment technique to a manually misaligned log of perception data and overhead imagery features. A brute force search across all potential map alignments in 0.35 m increments in the four cardinal directions detected a misalignment of 3.85 m west and 4.9 m north. Such a search is too computationally expensive to be performed in real-time but was completed in about an hour through offline processing. Computed probabilities of observed perception data and the corresponding improvement in traversal cost estimates can be seen in Figure 23. As expected, correcting the misalignment improved the definition of obstacles in the traversal cost maps and resulted in a stronger correspondence with the actual environment,



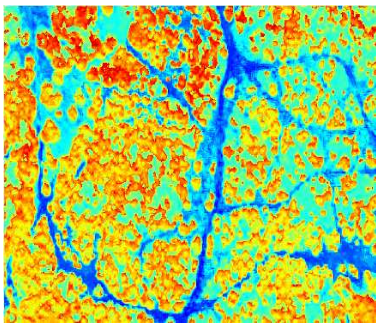
(a)



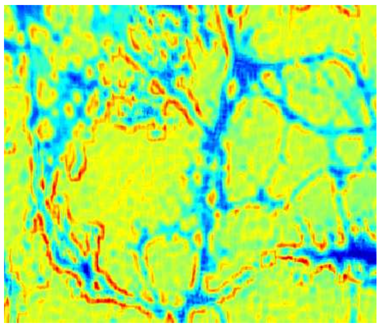
(b)



(c)



(d)



(e)

Figure 20: Circular course with the GPS waypoints for which a priori paths were planned is shown in (a). MOLL was trained during a short traversal of the training course outlined in the red box. Shown area is 2000 m $\times$ 750 m. A priori paths generated by a human-trained algorithm (solid red), MOLL using color imagery data (dashed cyan), and MOLL using black and white imagery data (dotted blue) are shown in (b). Traversal cost maps produced by MOLL for the closeup area in (c) using overhead color imagery and black and white imagery are shown in (d) and (e) respectively. See Table 2 for description of data sources. Traversal costs are color-scaled for improved visibility where blue and red correspond to lowest and highest traversal cost areas respectively.

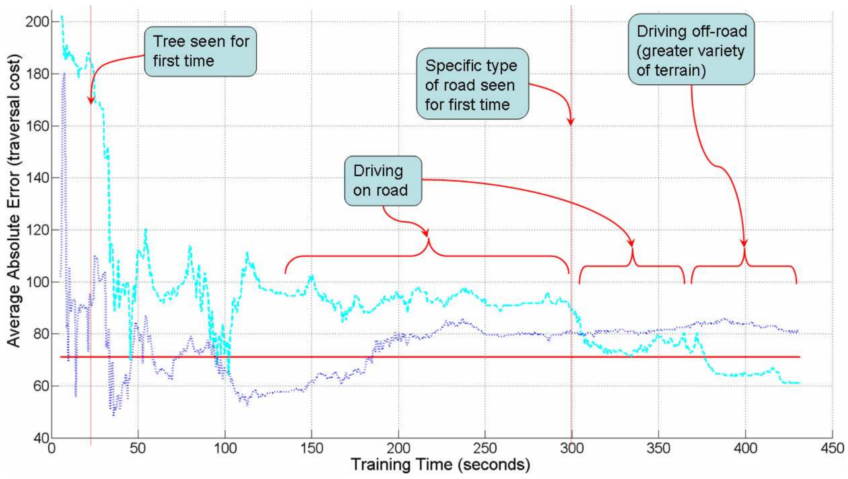


Figure 21: Average absolute error between log scale traversal costs computed by the robot's on-board perception system over the course of a multi-kilometer traverse of terrain and traversal cost estimates computed using three techniques: human-trained supervised learning algorithm using high resolution imagery and elevation data (solid red) and MOLL using only color imagery (dashed cyan) and black and white imagery (dotted blue) as a function of training time by driving in a similar environment. See Table 2 for a description of data sources. The erratic performance for the first few minutes of training are due to the large effects of new training examples when so little previous data was available. In the case of MOLL with black and white imagery, the initial sample of terrain happened to correspond well to the rest of the course. MOLL training takes longer with color imagery due to a greater number and variety of features.

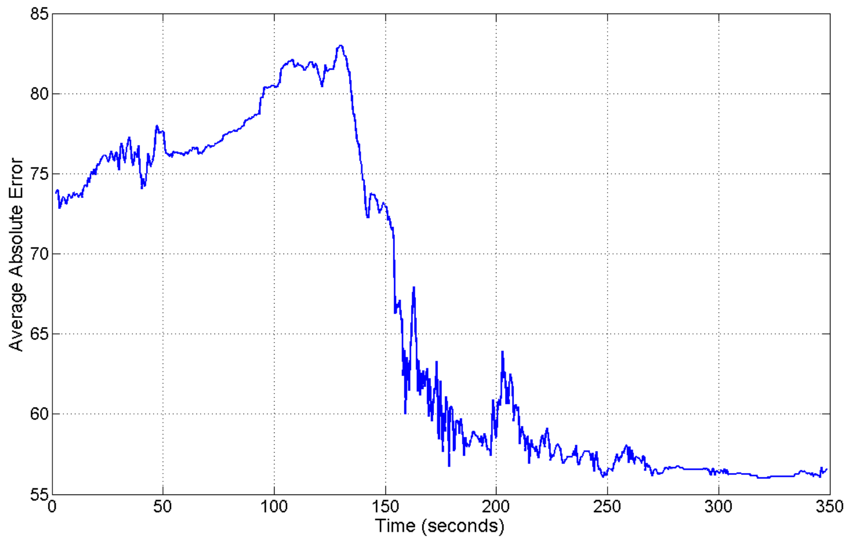


Figure 22: Average absolute error between log scale traversal costs computed by the robot's on-board perception system and traversal cost estimates generated by FROLL as a function of training time.

correctly showing that the traveled path is clearly on the road.

5.4.4 Feature Selection

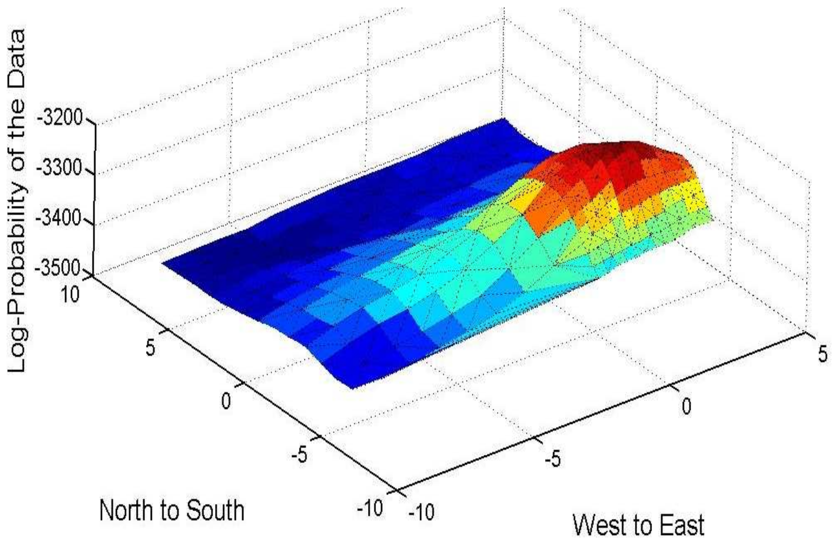
We applied our feature selection imagery to a set of 33 overhead imagery and elevation data based features. While these features were all relevant to the environment, we assumed that many of them were redundant and therefore selected the 14 most important from the set. Figure 24 depicts the accuracy of cost predictions as a function of training time using this reduced set of features compared to the original set. As expected, the smaller set of selected features resulted in a decreased training time.

6 Anytime Online Novelty Detection

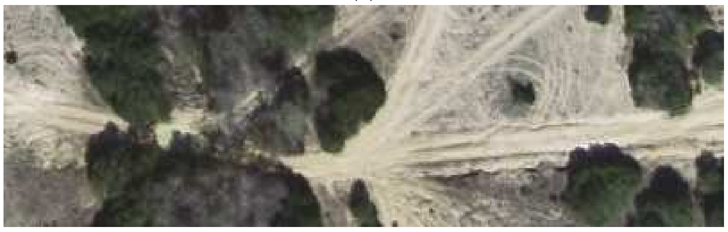
Safe traversal of UGVs can be facilitated by effective novelty detection capabilities. Two common limitations of novelty detection systems are particularly relevant to the mobile robotics domain. Autonomous systems often need to learn from their experiences and continually adjust their models of what is normal and what is novel. For example, if human feedback were to confirm that a certain type of environment selected as novel is actually safe to handle with the existing autonomy system or demonstrate to the system the proper way to handle the situation (as in [46]), the model no longer needs to identify such inputs as novel. Most novelty detection approaches, however, build a model of the normal set of examples a priori in batch in order to detect novel examples in the future but are unable to update that model online without retraining.

Furthermore, existing novelty detection techniques see diminished performance when using high-dimensional feature spaces, particularly when some features are less relevant, redundant, or noisy. These qualities are particularly common in features from many UGV perception systems due to the variety of sensors and uncertainty about how these features relate to novelty. For example, the relevance of camera-based features such as color and texture of an area of the environment to novelty (or similarity metrics in general) is difficult to understand as subsets of the features could contain redundant information or be mostly irrelevant. It is therefore important for novelty detection techniques to be resilient to such feature properties.

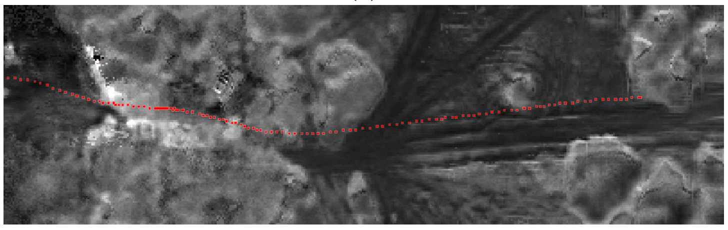
We present an online approach that addresses these common problems with novelty detection techniques. We approach the problem of novelty detection as one of online density estimation where seen examples generate an influence of familiarity in feature space. When prior class information is available, we show how using Multiple Discriminant Analysis (MDA) for generating a reduced dimensional subspace to operate in rather than other common techniques such as Principal Components Analysis (PCA) can make the novelty detection system more robust to issues associated with high-dimensional feature spaces [45]. In effect, this creates a lower dimensional subspace that truly captures *what makes things novel*. Additionally, our algorithm can be framed as a variant of the NORMA algorithm, an online kernelized SVM optimized through stochastic gradient descent, and therefore shares its favorable qualities [47]. Along with its anytime properties, this allows our algorithm to better deal with the real-time



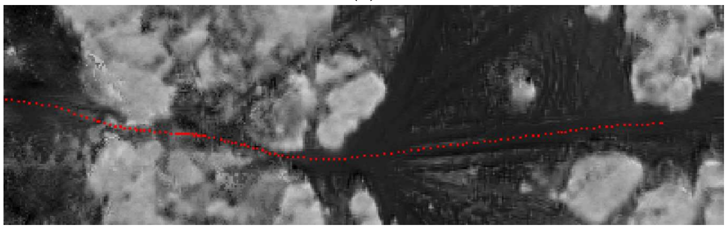
(a)



(b)



(c)



(d)

Figure 23: Example of how misalignment between overhead data sources and estimated vehicle position can be detected using our algorithm. Computed log probability of the perception system sensor data encountered over a 12.6 m $\times$ 12.6 m search space of alignment shifts is shown in (a). MOLL cost prediction for area shown in (b) before alignment correction and after correcting detected misalignment of 3.85 m west and 4.9 m north) appear in (c) and (d) respectively (best alignment is assumed to be that which produces the highest probability of seen perception data). Darker colors in the images correspond to lower traversal costs. The robot's path is shown in red.

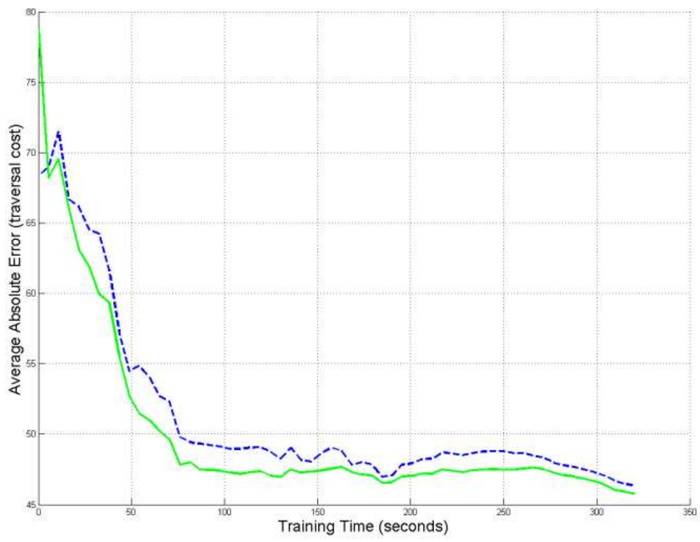


Figure 24: Average absolute error between log scale traversal costs computed by unmanned ground vehicle's on-board perception system over the course of a multi-kilometer traversal of terrain and traversal cost estimates computed before and after feature selection: results of using the full set of 33 features from both imagery and elevation data (dotted blue) and a subset of 14 features selected using the feature selection algorithm (solid green) are shown.

demands of online tasks.

While this work was targeted toward mobile robotics applications, the approaches here are more generally applicable to any domain which can benefit from online novelty detection.

6.1 Related Work

Novelty detection techniques (also referred to as anomaly or outlier detection) have been applied to a wide range of domains such as detecting structural faults [48], abnormal jet engine operation [49], computer system intrusion detection [50], and identifying masses in mammograms [51]. In the robotics domain some have incorporated novelty detection systems within inspection robots [52, 53].

Novelty detection is often treated as a one-class classification problem. In training the system sees a variety of "normal" examples (and corresponding features) and later the system tries to identify input that does not fit into the trained model in order to separate novel from non-novel examples. Instances of abnormalities or novel situations are often rare during the training phase so a traditional classifier approach cannot be used to identify novelty in most cases.

Most novelty detection approaches fall into one of several categories. Statistical or density estimation techniques model the "normal" class in order to identify whether a test sample comes from the same distribution or not. Such approaches include Parzen window density estimators, nearest neighbor-based estimators, and Gaussian mixture models [45]. These techniques often use a lower-dimensional representation of the data generated through techniques such as PCA.

Other approaches attempt to distinguish the class of instances in the training set from all other possible instances in the feature space. Schölkopf et al. [54] show how an SVM can be used for specifically this purpose. A hyper-plane is constructed to separate the data points from the origin in feature space by the maximum margin. One application of this technique was document classification [55]. A noticeable drawback of this approach is that it makes an inherent assumption that the origin is a suitable prior for the novel class. This limitation was addressed by [56] by attracting the decision boundary toward the center of the data distribution rather than repelling it from the origin. A similar approach encloses the data in a sphere of minimal radius, using kernel functions to deal with non-spherical distributed data [57]. These techniques all require solutions to linear or quadratic programs with slack variables to handle outliers.

Another class of techniques attempts to detect novelty by compressing the representation of the data and measuring reconstruction error of the input. The key idea here is that instances of the original data distribution are expected to be reconstructed accurately while novel instances are not. A simple threshold can then be used to detect novel examples. The simplest method of this type uses a subset of the eigenvectors generated by PCA to reconstruct the input. An obvious limitation here is that PCA will perform poorly if the data is non-linear. This limitation was addressed by using a kernel PCA based novelty detector [58]. Benefits of more sophisticated auto-encoders, neural networks that attempt to reconstruct their inputs through narrow hidden layers, have been studied as well [59].

Online novelty detection has received significantly less attention than its offline counterpart. Since it is often important to be able to adjust the model of what is considered novel in real-time, many of the above techniques are not suitable for online use as they require significant batch training prior to operation. While Neto et al. [52] replaced the use of PCA for novelty detection with an implementation of iterative PCA, performance was still largely influenced by the initial data set used for training. Marsland proposed a unique approach that models the phenomenon of habituation where the brain learns to ignore repeated stimuli [53]. This is accomplished through a clustering network called a Grow When Required (GWR) network. This network keeps track of firing patters of nodes and allows the insertion of new nodes to allow online adaptation.

Markou and Singh have written a pair of extensive survey articles detailing many additional novelty detection applications and techniques [60, 61].

The performance of the above-mentioned novelty detection approaches, however, quickly deteriorates as the number of less relevant or noisy features grows. The disproportionately high variance of many of these features make it difficult for many of these algorithms to capture an adequate model of the training data and their effects quickly begin to dominate more relevant features in making predictions. Our algorithm addresses this crucial limitation in cases where class information is available within the training set while still being suitable for online use.

6.2 Approach

6.2.1 Formalization

The goal of novelty detection can be stated as follows: given a training set $D = \{\mathbf{x}\}_{1...N} \in \mathcal{X}$ where $\mathbf{x}_i = \{x_i^1, \ldots, x_i^k\}$, learn a function $f : \mathcal{X} \rightarrow \{novel, not\text{-}novel\}$. In the online scenario, each time step t provides an example $\mathbf{x}_t$ and a prediction $f_t(\mathbf{x}_t)$ is made.

We perform online novelty detection using the online density estimation technique shown in Algorithm 2 [62]. All possible functions f are elements of a *reproducing kernel Hilbert space* $\mathcal{H}$ [63]. All $f \in \mathcal{H}$ are therefore linear combinations of kernel functions:

$$f_t(\mathbf{x}_t) = \sum_{i=1}^{t-1} \alpha_i k(\mathbf{x}_i, \mathbf{x}_t) \tag{2}$$

We make the assumption that proximity in feature space is directly related to similarity. Observed examples deemed as novel are therefore remembered and have an influence of familiarity on future examples through the kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$. A novelty threshold, γ, and a learning rate, η, are initially selected. For each example $\mathbf{x}_t$, the algorithm accumulates the influence of all previously seen novel examples (line 5). If this sum exceeds γ then the example is identified as novel and is remembered for future novelty prediction (line 7). Non-novel examples do not need to be stored as they have no impact on future novelty computations (even though a coefficient of 0 is assigned in line 9 for clarity, these examples do not need to be stored). We suggest simply using the Gaussian kernel with an appropriate variance σ^2:

$$k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}} \tag{3}$$

Algorithm 2 Online novelty detection algorithm

```
 1: given: A sequence of features $S = (\mathbf{x}_i)_{1...T}$; a novelty threshold $\gamma$; a learning rate $\eta$
 2: outputs: A sequence of hypotheses $\mathbf{f} = (f_1(\mathbf{x}_1), f_2(\mathbf{x}_2), \ldots)$
 3: initialize: $t \leftarrow 1$
 4: loop
 5:    $f_t(\mathbf{x}_t) \leftarrow \sum_{i=1}^{t-1} \alpha_i k(\mathbf{x}_i, \mathbf{x}_t)$
 6:    if $f_t(\mathbf{x}_t) < \gamma$ then
 7:       $\alpha_t \leftarrow \eta$
 8:    else
 9:       $\alpha_t \leftarrow 0$
10:    end if
11:    $t \leftarrow t + 1$
12: end loop
```

6.2.2 Improved Dimensionality Reduction

Especially if the number of features is large, it may first be necessary to project the high-dimensional input $\mathbf{x}_t$ into a lower-dimensional subspace more suitable for novelty detection using distance metrics. The most common choice for this among dimensionality reduction (and novelty detection) techniques is PCA. PCA finds a linear transformation that minimizes the reconstruction error in a least-squares sense. If subsets of the features are redundant, noisy or are dominated disproportionally by a subset of the training set, however, applying techniques such as PCA, or any unsupervised dimensionality reduction technique for that matter, may yield disappointing results as precisely the most relevant directions for differentiation may be discarded in order to reduce reconstruction error of a less relevant portion of the feature space.

Rather than optimizing for reconstruction error, *discriminant analysis* seeks transformations that are efficient for discriminating between different classes within the data. Multiple Discriminant Analysis, a generalization of Fischer's linear discriminant for more than two classes, computes the linear transformation that maximizes the separation between the class means while keeping the class distributions themselves compact, making it useful for classification tasks [45].

We argue that when prior class information for the training set is available, using MDA to construct a lower dimensional subspace using labeled classes not only optimizes for known class separability but likely leads to separability between known classes and novel classes. In cases described earlier that result in poor performance when using PCA, MDA will largely ignore features that do not aid in class discrimination, instead focusing on the obviously differentiating features. The key observation here is that novelty detection is about encountering new classes, so by using discriminating ability as the metric for constructing a subspace, one can capture the combinations of features that make known classes novel *with respect to each other* and likely generalize to previously unseen environments, in effect capturing *what makes things novel*.

6.2.3 Framing as Instance of NORMA

The NORMA algorithm is a stochastic gradient descent algorithm that allows the use of kernel estimators for online learning tasks [47]. As with our algorithm, f is expressed as a linear combination of kernels (2). NORMA uses a piecewise differentiable convex loss function l such that at each step t we add a new kernel centered at $\mathbf{x}_t$ with the coefficient:

$$\alpha_t = -\eta l'(\mathbf{x}_t, y_t, f_t) \tag{4}$$

Our algorithm can easily be framed as an online SVM instance of NORMA using a hinge loss function as follows:

$$\begin{aligned} y_t &= \gamma && (5)\\ l(\mathbf{x}_t, y_t, f_t) &= max(0, y_t - f_t(\mathbf{x}_t)) && (6) \end{aligned}$$

Taking the derivative of (6), we get:

$$l'(\mathbf{x}_t, y_t, f_t) = \begin{cases} -1 & \text{if } f_t(\mathbf{x}_t) < \gamma \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

As before, the gradient of our loss is non-zero only when the accumulated contributions from stored examples are less than the novelty threshold γ, signifying that the example is novel. From (4) and (7) we then get:

$$\alpha_t = \begin{cases} \eta & \text{if } f_t(\mathbf{x}_t) < \gamma \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

This is equivalent to the update steps in lines 7 and 9 of Algorithm 2, showing that our algorithm can be framed as a specific instance of the NORMA algorithm.

NORMA produces a variety of useful bounds on the expected cumulative loss [47]. For novelty detection this directly relates to the number of examples that are expected to be flagged as novel. This means we are competitive with respect to the best $f \in \mathcal{H}$ in terms of representing our sample distribution with the fewest number of examples. This is to our advantage both from a computational perspective, since memory and prediction costs scale with the number of remembered examples, as well as performance since we want to minimize false positives that may be costly to handle.

6.2.4 Query Optimization

Without further measures, the potential number of basis functions stored by Algorithm 2 could grow without bound. NORMA deals with this issue by decaying all coefficients α_i and dropping terms when their coefficients fall below some threshold. This is unsuitable for our application since we do not want to repeatedly flag similar examples as novel. Instead, we propose a modified anytime version of our algorithm that ensures efficient and bounded computation (see Algorithm 3).

This algorithm takes advantage of the fact that familiarity contribution to new queries is often dominated by only a few examples. First, we can easily gain some efficiency by only processing stored examples until we have reached the novelty threshold (line 7). The key performance improvement, however, comes from the sequence optimization in line 17. For each prediction, the stored example that breaks the novelty threshold γ, or the new novel example itself, is moved to the front of the list as it is more likely to impact future queries[4]. This is a slight variation of the traditional problem of dynamically maintaining a linear list for search queries for which the move-to-front approach was proven to be constant-competitive, meaning no algorithm can beat this approach by more than a constant factor [64]. As well as allowing us to bound the number of stored examples (line 19), this gives our algorithm an anytime property by enabling it to as quickly as possible classify as much of the environment as possible as not

[4]Another variant is to move stored example $argmax_{j\in[1,i]} k(\mathbf{x}_t, \mathbf{x}_j)$ to the front of the list.

Algorithm 3 Online novelty detection algorithm with query optimization

```
 1: given: A sequence of features $S = (\mathbf{x}_i)_{1...T}$; a novelty threshold $\gamma$; a learning rate $\eta$; a
    maximum example storage capacity $N$
 2: outputs: A sequence of hypotheses f $= (f_1(\mathbf{x}_1), f_2(\mathbf{x}_2), \ldots)$
 3: initialize: $t \leftarrow 1$; ; $n \leftarrow 0$
 4: loop
 5:    $i \leftarrow 1$
 6:    $f_t(\mathbf{x}_t) \leftarrow 0$
 7:    while $f_t(\mathbf{x}_t) < \gamma$ and $i \leq n$ do
 8:       $f_t(\mathbf{x}_t) \leftarrow f_t(\mathbf{x}_t) + \alpha_i k(\mathbf{x}_i, \mathbf{x}_t)$
 9:       $i \leftarrow i + 1$
10:    end while
11:    if $f_t(\mathbf{x}_t) < \gamma$ then
12:       $\alpha_{n+1} \leftarrow \eta$
13:       $\mathbf{x}_{n+1} \leftarrow \mathbf{x}_t$
14:       $n \leftarrow n + 1$
15:       $i \leftarrow i - 1$     // i was incremented one extra time
16:    end if
17:    optimize sequence: Move $(\alpha_i, \mathbf{x}_i)$ to front
18:    if $n > N$ then
19:       Delete $(\alpha_i, \mathbf{x}_i)_{i>n}$
20:       $n \leftarrow N$
21:    end if
22:    $t \leftarrow t + 1$
23: end loop
```

At line 17, if $f_t(\mathbf{x}_t) =$ *not-novel*, i indexes the example that broke the novelty threshold. Otherwise, i indexes $\mathbf{x}_t$.

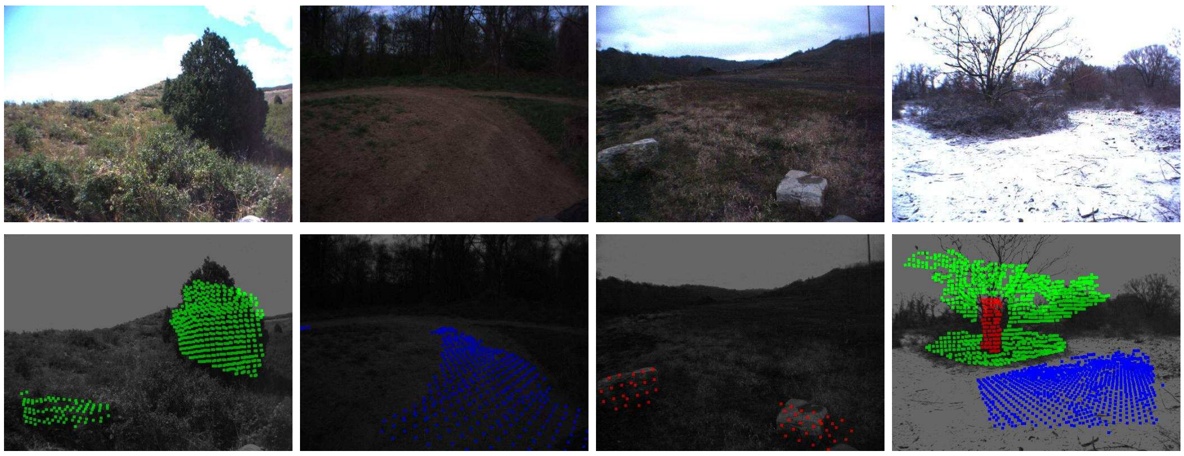


Figure 25: Examples of hand labeled class categories (bush, road / grass, rocks, tree trunk, tree branches, etc.)

novel. When this algorithm is unable to run to completion due to time constraints, it will fail intelligently by generating false positives but never potentially dangerous false negatives.

6.3 Application to Mobile Robotics

A natural application of this algorithm is to online novelty detection for a mobile robot. To perform novelty detection on the Crusher UGV we used subsets of the initial raw features as well as the intermediate classification and density features for each voxel as shown in Figure 6.

Due to the voxelization approach of the perception system discussed in Section 4.2.1, we are forced to deal with a relatively high-dimensional feature space (49 features) as well as with the associated issues described earlier.

6.3.1 Dimensionality Reduction for High-Dimensional Data

We address this problem by using MDA with an extensive library of hand-labeled examples across many environments and conditions to compute a lower dimensional subspace more suitable for density estimation as described in the previous section. Of the available classes, four were used to construct a three-dimensional subspace: road/dirt, rocks, bushes and barrels (see Figure 25). A fifth class of examples corresponding to various non-barrel man-made objects was withheld to verify the suitability of this subspace (see Figure 26).

Figure 27 shows the projection of all five classes onto the first three basis vectors computed by PCA and LDA using the first four classes[5]. The LDA projections clearly show better separation between the new set of man-made examples and the original four classes. As expected, the most overlap occurs with the barrel class as barrels share common properties with other man-made objects such as smooth surfaces, colors, etc. Since we would desire these new examples to be identified as novel relative to the rest of the classes, this separation implies that

[5]All features were initially rescaled to zero-mean, unit-variance.

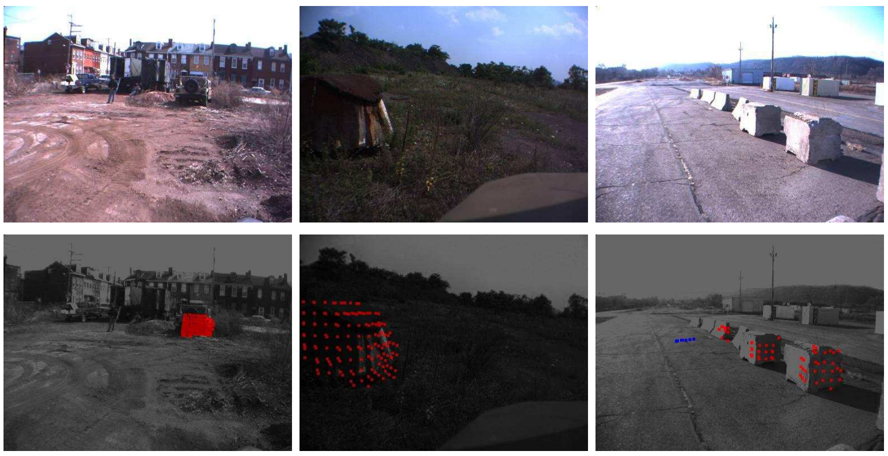


Figure 26: Sample hand labeled examples in the ’other man-made’ class used for validation of dimensionality reduction effectiveness. This category excluded instances of barrels which were used as a separate class.

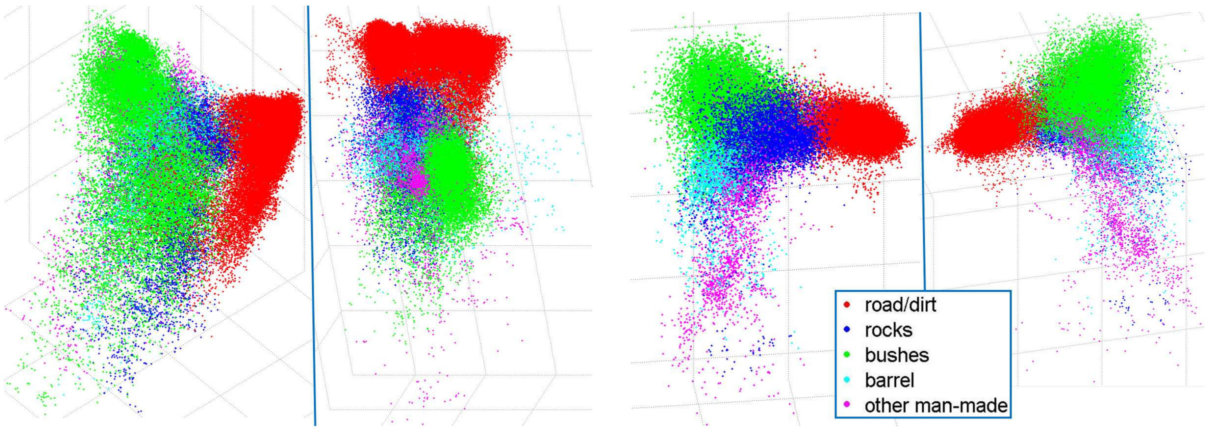


Figure 27: All data points projected onto the subspace defined by the first three basis vectors computed by PCA (left) and LDA (right). Only the first four classes were used to construct the subspaces (’other man-made’ class was withheld as a test class). The LDA-based projection clearly shows significantly more separation between the new man-made class and the known classes, implying a more suitable subspace for novelty detection.

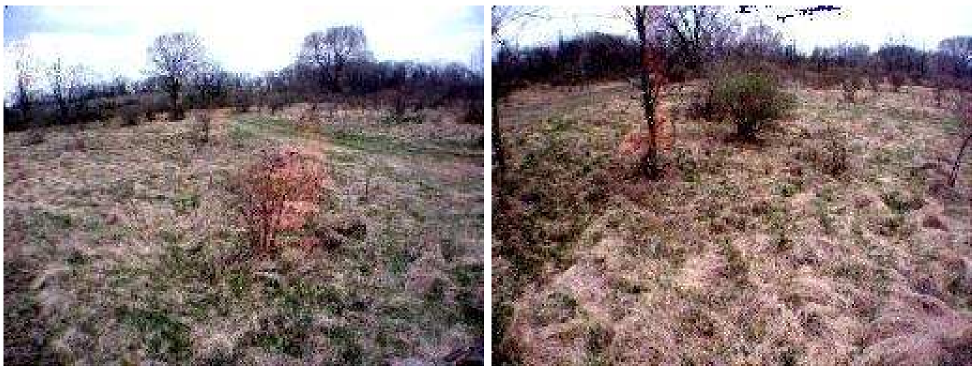


Figure 28: After initialization with no prior novelty model, various small vegetation was detected as novel (identified in red).

that this is a more suitable subspace for use as a similarity metric within a novelty detection system.

Because our algorithm is efficient for online use, the novelty model can start uninitialized or can be seeded with a sampling of examples used during training so that it can identify areas that are novel and potentially unsafe to handle with the current perception system.

6.4 Work To Date

Our novelty detection algorithm (with query optimization) was tested in real-time on logged data to evaluate its online novelty detection performance. The test environment traversed by the robot consisted of combinations of road, grass and dirt, a large variety of vegetation, a series of small barrels, several ditches, large heavily-sloped piles of rocks and a long chain-link fence.

We projected all examples into the three-dimensional subspace generated by MDA as described in the previous section from the first four hand-labeled classes (not using the non-barrel man-made objects class). To best exhibit the online novelty detection abilities of our algorithm, the model was initialized to contain no prior examples. As the environment was explored, perception system features were averaged into $0.8\ cm^2$ grid locations for use as online batches of examples. Those that were identified as novel relative to the current model (composed of everything previously identified as novel) were incorporated into the model as described earlier.

The vehicle's initial environment consisted of fairly open terrain with some light scattered vegetation scattered on both sides. As expected, instances of such vegetation were detected as novel the first few times they were seen (see Figure 28).

The vehicle then encountered areas of much denser, larger vegetation. Initially, a majority of such vegetation was identified as novel with respect to previous inputs (see Figure 29). As the vehicle continued navigating through similar vegetation, the model adapted and no longer identified such stimuli as novel (see Figure 30). Figure 31 demonstrates this learning process through a series of overhead images of this initial environment, identifying all future locations that are novel with respect to the *current* model. Output is shown at three points in time: near the beginning of navigation, just before initial encounters with dense vegetation and after sensing a small amount of dense vegetation. It is clearly visible how the system adapts quickly, causing future similar instances to no longer be flagged as novel.

Proceeding through the environment, the vehicle then encounters a series of plastic barrels (see Figure 32). As desired, the first several appear as novel with respect to the large variety of vegetation previously seen while later barrels are no longer novel due to their strong similarity to the initially seen barrels. Similarly, a long stretch of a chain-link fence is identified as novel late in the course (see Figure 4). Again, the initial portions of the fence triggered the novelty detection algorithm while later portions were no longer novel due to the algorithm's adaptation. Additional examples of novel instances identified during traversal appear in Figure 33.

Overall, the novelty detection algorithm was able to identify all major unique objects (vegetation, barrels, fence, etc.) with a relatively small amount of false positives due to effective adaptation to the environment. When PCA was used to create the feature subspace, the lack of separability between classes resulted in either unacceptably many false positives or false negatives, depending on parameter choices. As with any algorithm, the success of this approach is heavily dependent on the quality of features.

Computation time comparisons between the two algorithms on this course highlight the effectiveness of query optimization (see Figure 34). While the average computation time required per novelty query using Algorithm 2 grows with the number of stored examples, Algorithm 3 experiences temporary spikes in computation time as novel areas are encountered but query optimization allows the algorithm to quickly adapt its ordering of examples in order to maintain a bounded computation throughout navigation and allow effective anytime novelty prediction.

6.5 Proposed Work

6.5.1 Novelty Detection

To date, our results in online novelty detection have been qualitatively compelling but we have not derived any quantitative metrics of our algorithm's performance. Such a result would consist of ROC analysis that would describe the expected trade-off between false-positives and false-negatives for our approach dependent on the novelty threshold as well as justifying the choice of MDA over PCA for dimensionality reduction. A significant difficulty in measuring performances for such an online application is that we do not have a straightforward way to label training data and test performance. Rather than an object simply being novel or non-novel, there is a temporal component to online novelty detection that complicates the metric and is often subject to interpretation. Furthermore, novelty detection is performed on a cell-by-cell basis so predictions will need to be clustered for testing purposes for each independent area in the environment. If necessary for more robust performance, we will also explore non-linear dimensionality reduction alternatives to MDA.

An additional challenge we hope to address is the presence of several hand-picked parameters in this system, specifically the kernel bandwidth of each novel example and the threshold for novelty. We propose to further leverage the availability of hand-labeled classes during training to identify parameters that generalize well to new data.

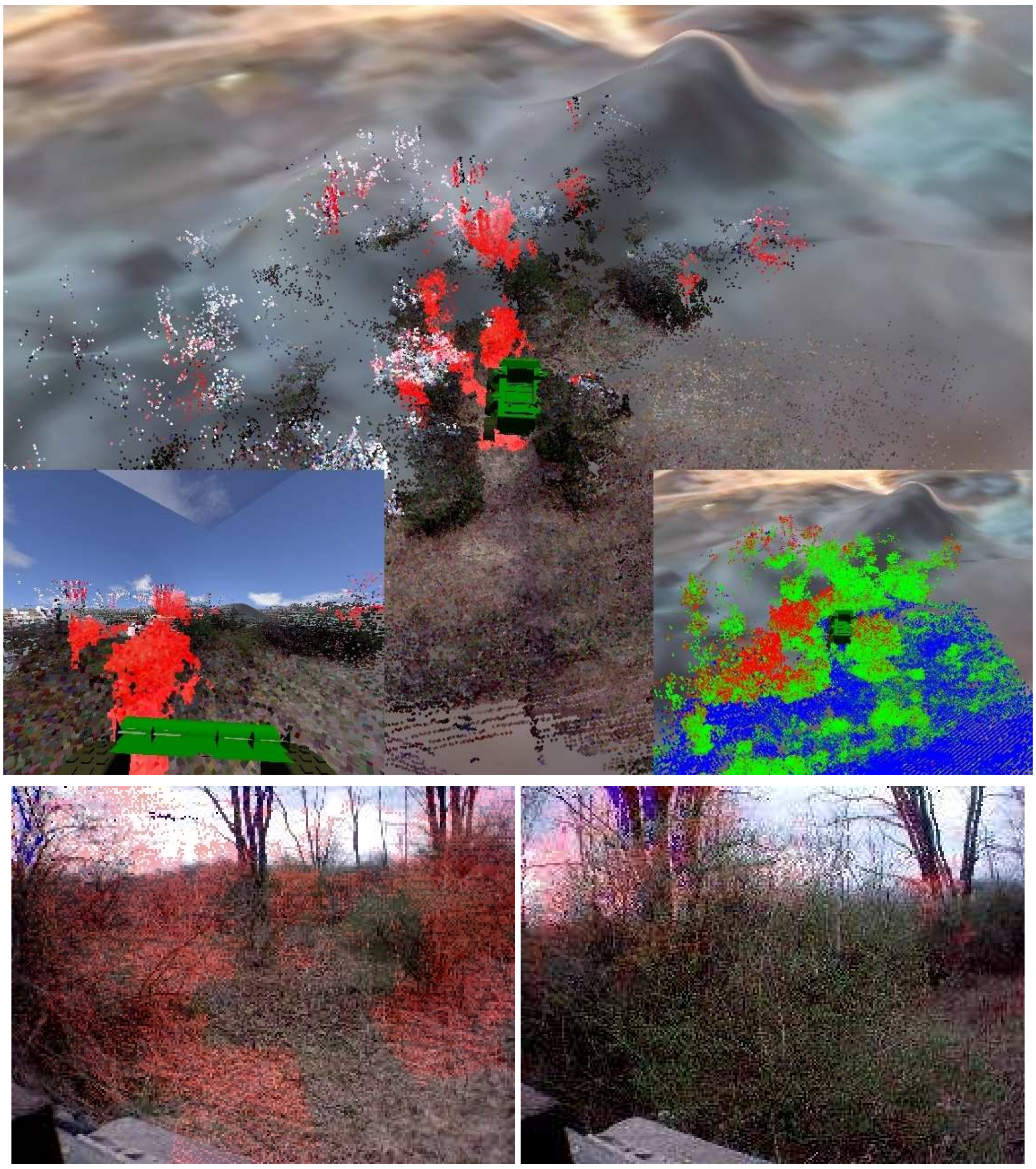


Figure 29: Initial encounter with larger and denser vegetation results in a significant amount of detected novelty (identified in red).

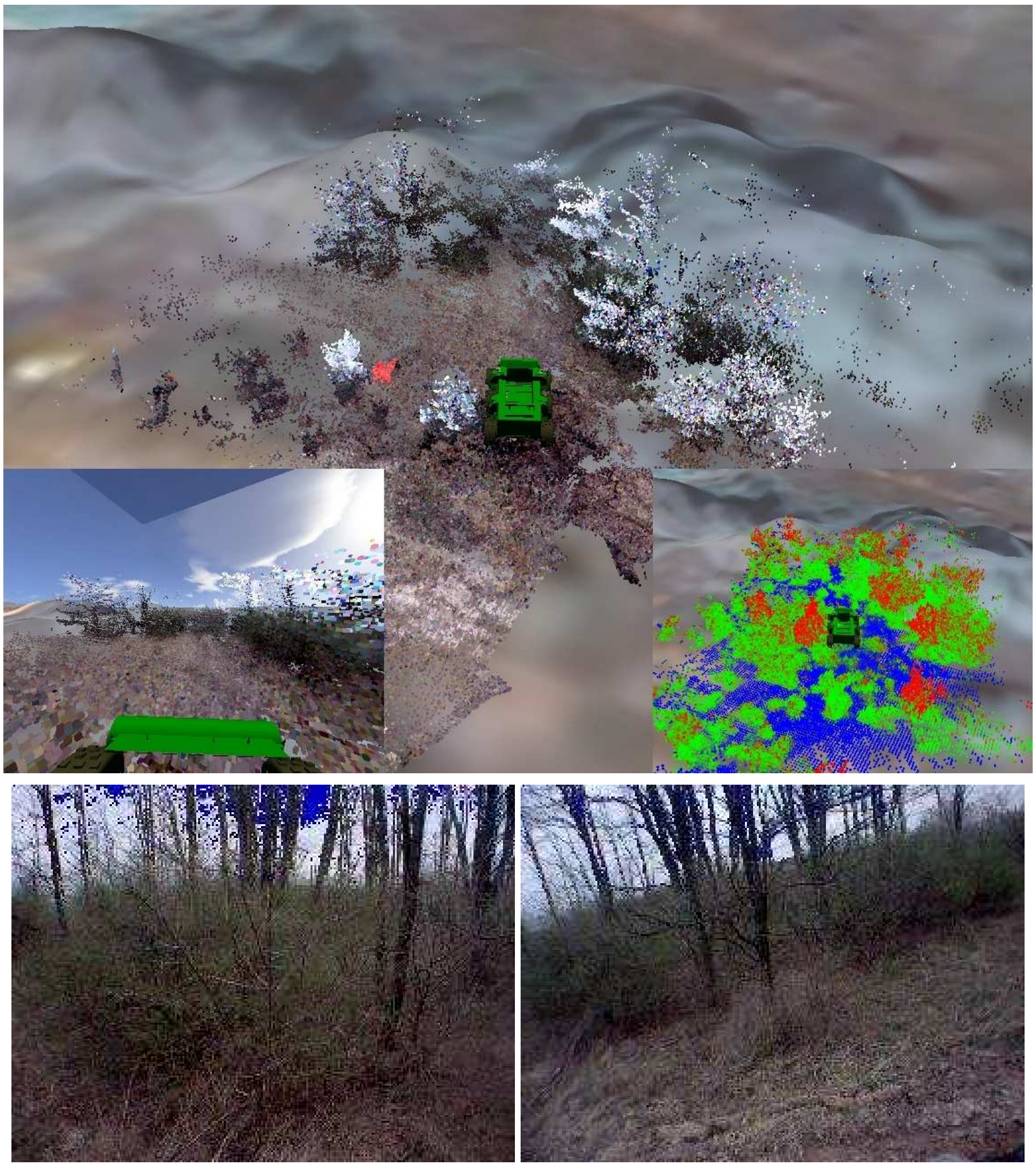


Figure 30: Similar vegetation as that shown in Figure 29 encountered a short time later. Notice how almost all vegetation is no longer novel due to similarity to previous stimuli.

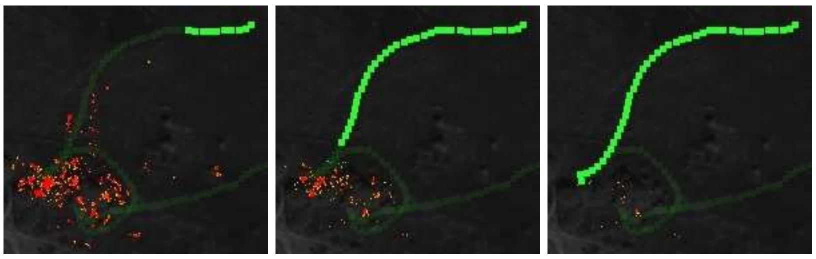


Figure 31: Novelty of all future perception input using current novelty model on vegetation-heavy terrain shown in Figures 28, 29 and 30 at three points throughout traversal. Robot's past and future path is shown in light and dark green respectively and novelty of terrain is indicated by a gradient from yellow (moderately novelty) to red (high novelty). Robot is initialized without a prior novelty model.

Furthermore, we will need to provide rigorous theoretical arguments as to why our online anytime approach has good regret properties while minimizing the amount of time necessary to call something novel. This will require analysis and formal justification for the various query optimization variants discussed previously.

Since to this point we have been working in real-time on logged data, our next step will be to implement this approach onboard the E-gator platform. We propose to conduct a series of experiments under various conditions to demonstrate the capabilities of this approach and the potential benefit of human involvement in such situations. Our primary metric will be the apparent improvement to vehicle safety measured by how well detected novel locations correlate to vehicle struggles or human interventions. We presume that in such situations, a human operator queried for help due to the detected level of novelty could have averted the poor performance or required intervention.

A variety of categories of objects will be used to rigorously test the performance of the novelty detection system. The most obvious include a variety of man-made objects such as vehicles, buildings and barriers such as fences. Unique vegetation that the vehicle may struggle with will also be explored. Of particular interest are objects that are of unique appearance but can be potentially navigated through by the vehicle. Examples of such objects include soft or malleable materials such as foam, paper / cardboard, inflatable objects, or situations that occlude sensing abilities such as smoke.

Two modes of operation will both be explored. In the first, the system will begin with no prior knowledge, causing everything to appear initially novel (as described in the initial results of Section 6.4). The goal here is to observe the instances and order of objects that are identified as novel and to ensure that future instance of those objects no longer appear novel. This is intended to test the online capabilities of the novelty detection system.

The second mode of operation is one where the novelty detection system is initially seeded with a representative sample of normal terrain (road, grass, rocks and common vegetation). Here the goal is to identify situations that are novel with respect to this training set, resulting in a simulated human query. We would track the percentage of test cases identified as novel (as well as false positives), as well as the number of human interventions averted compared to a system without novelty detection.

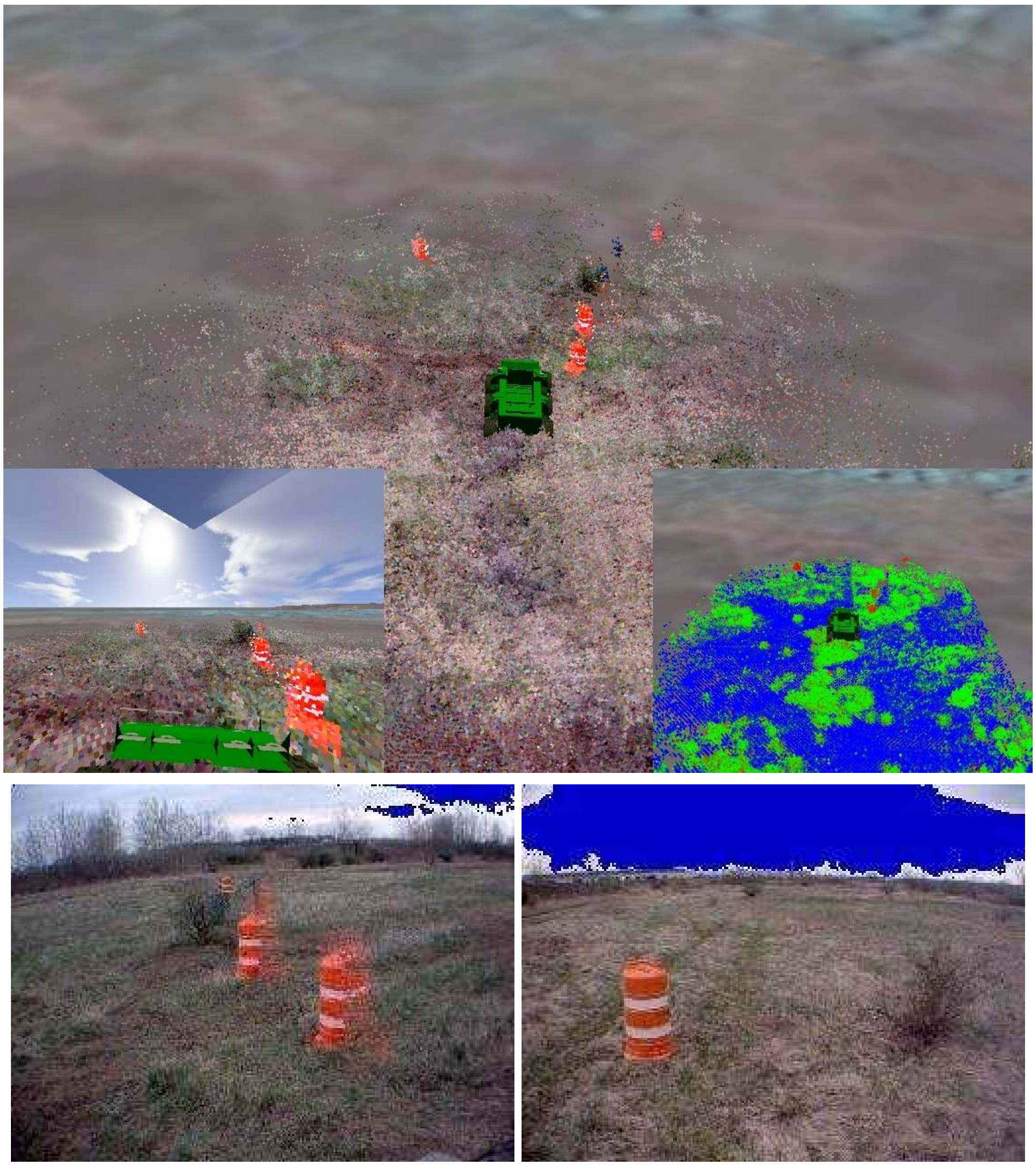


Figure 32: Series of barrels encountered later in the course. The initial barrels are detected as novel (red shade) even after significant exposure to a large variety of vegetation (top and left). Later barrels are no longer identified as novel due to online training.

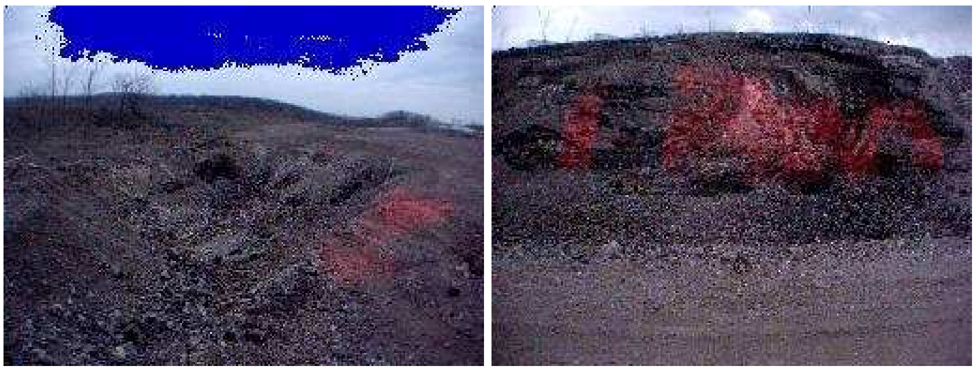


Figure 33: Additional examples of novel instances identified during later traversal (red shade): first encounter with a ditch (left) and a large, heavily-sloped pile of rocks (right).

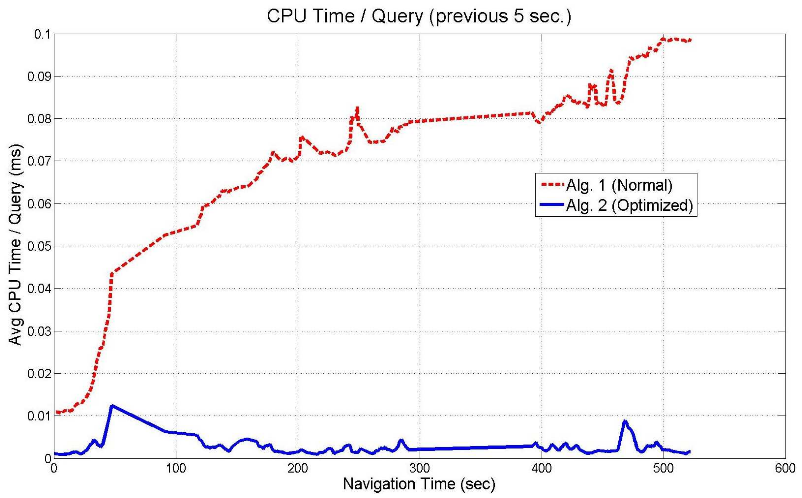


Figure 34: Average computation in milliseconds per novelty query on 3.2 GHz CPU for Algorithm 2 (dashed red line) and Algorithm 3 (solid blue line) over the previous 5 seconds throughout navigation. Computational complexity of Algorithm 3 remains bounded due to the order optimization step (line 17). These timings do not include feature computation and projection costs as they are identical under both algorithms.

While it's difficult to speculate about the potential performance of such a system, we will consider it successful if we can eliminate at least half of the human interventions on this particular platform by identifying a majority of novel situations that are hazardous to vehicle progress with relatively few false positives.

6.5.2 Change Detection

Change detection is a problem we believe is closely related to novelty detection. Whereas in the case of novelty detection we try to identify stimuli that are novel with respect to all previously encountered stimuli, in the case of change detection, we are interested in detecting when a significant change has occurred in a previously traversed location.

Such a technique is especially useful for repetitive navigation tasks such as supply routes and patrolling where such changes can pose navigational risks or locations of interest that a human should be aware of. In urban or work environments, this capability would serve as a safety system to reliably detect humans. For example, a human sitting in the path of the robot may appear identical to a bush or a boulder as seen by a perception system from a traversability standpoint. The goal of a change detection system is specifically to be able to discriminate between such cases.

It is also important that this approach be able to deal with imperfect localization with errors of up to several meters as is common in many domains. Like novelty detection, this problem requires an efficient representation for large quantities of perception data. However, we will need to explore alternative data representations that are able to identify location-specific deviations while being robust to position uncertainty. If we consider the continuum between the use of location-independent features on one extreme and fully location-dependent features on the other, novelty detection lies at the first extreme while change detection lies closer to the other extreme (see Figure 35). The exact position on this spectrum is determined by position uncertainty during operation. We plan to develop algorithms and data structures that allow us to operate anywhere on this continuum in order to be able to perform both novelty detection and change detection using the same general approach.

Experiments to validate change detection capabilities will consist of baseline navigation of a courses followed subsequent navigation of the same courses with a variety of changes that are to be detected by the system. The system will need to detect situations such as movements or additions of obstacles, replacements of some objects with similar other ones and, most importantly, the presence of humans on the course that were not present previously, even if they take the place of similarly-sized obstacles. A successful demonstration of change detection will be able to identify such situations even if the perception system does not find any differentiation.

We will measure the system's performance while varying sensitivity parameters through a rigorous ROC analysis over a variety of courses. We hope to be able to identify almost all situations of new objects appearing where they were not present before. The key achievement will be to identify humans in scenes, especially when the perception system would not otherwise distinguish between them and previously seen obstacles.

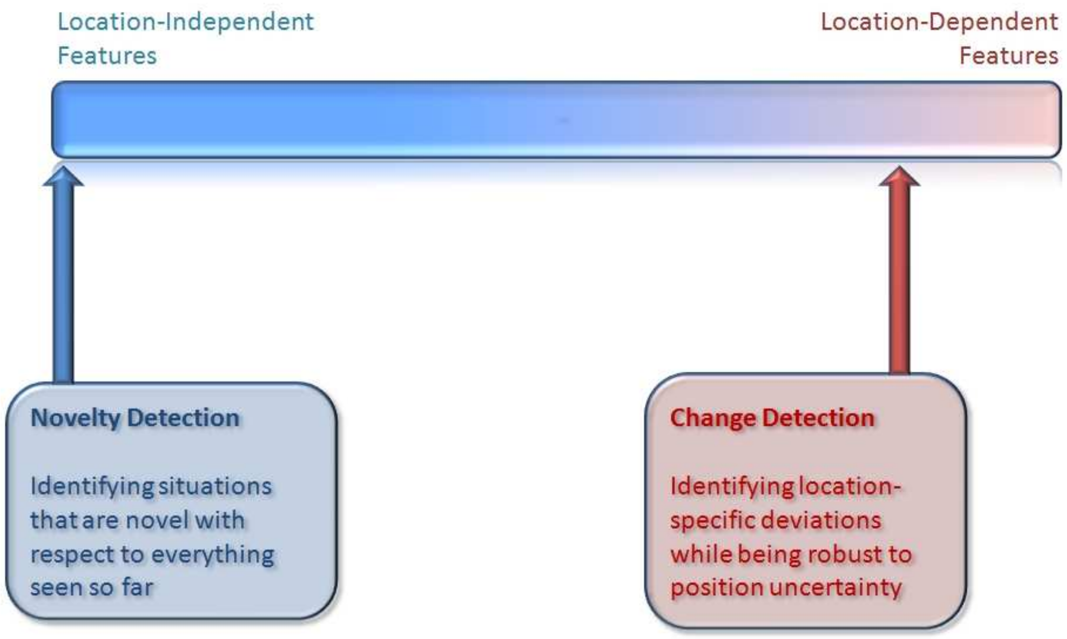

Location-Independent Features
Location-Dependent Features
Novelty Detection
Identifying situations that are novel with respect to everything seen so far
Change Detection
Identifying location-specific deviations while being robust to position uncertainty

Figure 35: Novelty detection and change detection both require an efficient representation of previously seen perception features and lie on a continuum between fully location-independent features and fully location-dependent features. Novelty detection does not take into account the locations of any features while change detection is largely location dependent but must be able to account for moderate positioning errors.

7 Online Candidate Selection

As we argued previously, many UGVs possess competent autonomy systems that are capable of dealing with a wide variety of situations. These systems are not perfect, however, and their performance degrades as characteristics of their environment begin to diverge from the environments used throughout training. In some domains it is therefore reasonable to assume that a human operator may be available for short periods of time to provide remote supervision or tele-operation. The responsibility of deciding how and when to use this remote operator assistance to improve performance and mitigate risk lies with the autonomy system.

The final portion of this thesis is devoted to a candidate selection system that observes the performance of the autonomous vehicle in particular situations and compares that performance to remote human-control in similar situations. When the vehicle encounters such situations in the future, it will be able to make a decision about which candidate will perform better.

It is important for such a system to be well-suited for online use. Not only is it unpredictable in advance how well the autonomy system will perform in novel situations, but human operator performance can also vary depending on factors such as bandwidth limitations, operator handicaps such as limited skill or familiarity with the interface, fatigue and weather conditions. To address such a scenario, we present a reinforcement learning based approach that learns to trade-off between multiple candidates while minimizing the potential penalties incurred when choosing the wrong system in each situation.

One way to take advantage of such a system is to combine limited human availability with the online novelty detection capabilities of Section 6. Since it is impossible to predict what a UGV may encounter, the key to success is for the UGV to seek help *before* it experiences a major failure. The human can then either inform the robot that it is safe to proceed or handle the novel situation himself, significantly reducing mission risk. Once the significant mission risk posed by novel situations is mitigated, the candidate selection system could be utilized to further optimize the systems navigational performance through selective human control. We call such a two-pronged approach the *Assisted-Autonomy Framework*. While we explore the novelty detection and candidate selection problems independently in this thesis, we do not plan to implement or evaluate the effectiveness of a combined approach.

7.1 Related Work

As robotic systems continue to play a larger role in our societies, there has been increased attention on how to optimize the interactions between humans and robots. This field is often referred to as Human-Robot Interaction, or HRI.

Many researchers have investigated approaches for heterogeneous human-robot teams. In such a scenario, humans and robots act as independent agents that must cooperate on a given task. Such techniques have been explored in domains such as the “Treasure Hunt” scenario [65, 66], robot soccer [67], forest fire monitoring [68], border patrol [69] and search and rescue assistance [70, 71, 72, 73].

We deal instead with the scenario where a human can contribute limited attention to improve a robotic system’s performance but is not himself an independent agent in the scenario. In this case, a robotic system operates somewhere on the spectrum between full autonomy, where there is no human involvement, and full tele-operation, where the human is in complete control at all times. Scenarios where the degree and methods of human interactions with robots within a system can be varied dynamically in order to optimize performance are often referred to as ones of *sliding autonomy* or *adjustable autonomy*. While most mobile robot systems tend to lie on one of the two extremes of this spectrum, effectively balancing autonomy with limited human involvement can lead to significant improvements in safety, efficiency and overall cost.

The extreme of full tele-operation is already common in many tasks such as remote bomb disposal or reconnaissance [74], operation in hazardous environments [75, 76] and robotic surgery [77]. On the other extreme, full autonomy has been heavily studying in the research community but often is unable to transition into real-world applications due to high reliability requirements and cost constraints. We argue that the way to optimizing the value and impact of robotic systems is to find a compromise on this spectrum that balances cost and development time with autonomous abilities, allowing these systems to be fielded years before otherwise possible.

In some scenarios where the human is the primary operator, the autonomy system is intended to aid by request or when it detects a dangerous situation. This is especially common in various driving assistance systems that are gradually becoming available in high-end vehicles. These include systems that detect drowsiness [78], provide parking assistance [79], automate cruise control [80], and provide situation aware brake assistance for collision avoidance [81]. Similar techniques have also been applied to trains, busses, semi-trucks, many forms of public transit [82], and aiding flight and air traffic control [83].

Similar approaches have been applied to surgery to decrease surgeon fatigue and improve performance. Through the use of force feedback and vision systems, researchers are developing a hybrid control scheme to perform basic subtasks in robotic-assisted laparoscopic surgery [84]. Such systems can automate repetitive tasks such as the cleaning-suction process, a simple yet tiring procedure that often limits durations of surgical procedures.

Much of the success in space exploration has been largely due to robotic technologies. The immense cost of rovers that are to operate on Mars or the moon makes it imperative that their safety is protected. While the space program has historically been conservative in introducing autonomous capabilities to robotic systems, NASA and the research community are gradually beginning to explore the potential benefits of limited autonomous operation. While tele-operation is possible in most cases, the distances over which commands must travel make the associated time delays troublesome. Commands sent to the moon would experience up to a 2.5 second delay while tele-operation of a rover on the surface of Mars can experience delays of up to 45 minutes.

A safeguarded tele-operation approach sharing control of the rover using a command fusion strategy was proposed for time-delayed remote driving [85]. In benign situations, users remotely drive the rover, while in hazardous situations, a safeguarding system running onboard the rover overwrites user commands to ensure vehicle safety and deal with the user's inability to evade obstacles effectively due to the time delay.

Specifying series of waypoints at once can partially alleviate this issue but would be still hindered by the operator's limited field of view at any given point in time. In an effort to increase performance through autonomy, NASA began utilizing navigational autonomy at times on its Mars Exploration Rovers, Spirit and Opportunity. In May 2005, NASA integrated a version of the Field D* algorithm into the navigation software of the rovers, enabling local and global planning [13, 86]. Such approaches have the potential to improve productivity while significantly reducing supervision requirements and personnel costs. Others are exploring the roles of adjustable autonomy in future space missions to allow humans to closely interact with robotic systems at whatever level of control is most appropriate [87].

The formulation for most of the situations described above is sometimes referred to as *user-based autonomy*: adjustable autonomy is driven by the need to support user control [88]. We instead deal with scenarios within the *agent-based autonomy* formulation where autonomy is the default mode of operation and an agent explicitly reasons about whether and when to transfer decision-making control to a human. Since interrupting the human often has high costs, complexity falls on defining an acceptable *transfer-of-control* strategy.

The Trestle system, for example, consists of three different robots that must work together to assemble a small structure from individual beams [89, 90]. These robots can either function autonomously or through tele-operation from a human. Through decision trees and Markov Decision Processes trained from prior performance data, the system was able to utilize the human intermittently to achieve a balance between human use and overall performance. However, in scenarios such as the one we address, prior performance information is often unavailable and the system must learn such models online. Furthermore, this system is able to request human assistance to correct failure, a luxury that is not available in many situations.

For single-agent systems some have suggested relinquishing control when there is an expectation of high benefit [91, 92] or the degree of uncertainty is high [93]. In scenarios where a single human must supervise multiple robots, adjustable autonomy becomes a requirement. Scerri et al. have extensively studied transfer-of-control strategies for large multi-agent teams using Markov Decision Processes [94]. Similar techniques have enabled NASA to replace a full-time multi-person operating staff for supervising satellite behavior with automated systems that signal for human help only when unexpected events occur [95].

Fong introduced the idea of *collaborative control* to allow the human and the robot to engage in dialog to exchange information, ask questions or to resolve differences, allowing a single human to supervise multiple robots simultaneously [96, 97]. This is accomplished through a set of approximately thirty defined queries enabled at specific situations. While this improved robot performance in situations that matched the pre-defined system specifications, the system would not extend well to novel situations due to its rigid definition and lack of online learning abilities.

Goodrich and Schultz have written an extensive survey article on the field of Human-Robot Interaction exploring many additional approaches and applications [98].

The key difference in our approach from the above-mentioned approaches is that we do not constrain the system by any pre-determined rules or models. Since it is not possible to prepare for all possible circumstances a robot may encounter, the ability to learn online the capabilities of each potential expert allows our systems to better adapt to more diverse and challenging environments.

7.2 Approach

7.2.1 Contextual Multi-Armed Bandit Setting

The candidate selection problem involves choosing an operator for each encountered situation from a set of candidate systems, in our case the autonomy system and the human tele-operator, whose performance we assume comes from some unknown distribution. It is therefore intuitive to frame this problem as an instance of the commonly studied *multi-armed bandit* problem [99, 100, 101]. Bandit problems are relevant to a wide range of domains such as statistics, economics and clinical trial decisions [102, 103, 104].

In the k-armed bandit setting, at each time step the world chooses k losses (or rewards), $l_1, \ldots, l_k$, and the player makes a choice of an arm $i \in \{1, k\}$ without knowledge of the hidden losses. The player then observers only the loss l_i corresponding to the chosen arm. Since the loss distributions are unknown, there is an inevitable conflict between minimizing the immediate loss and gathering information that will be useful for long-term performance. This is often referred to as the *exploration-exploitation trade-off* since we must choose between *exploring* our unknown loss distributions and *exploiting* the arm we currently believe to be best.

We deal with a more suitable variation of this setting called the *contextual bandits* setting where at each time step t the player also observes some contextual information x_t which can be used to determine which arm to pull [105, 106]. Since we can compute onboard perception or overhead features for areas that we encounter, it is reasonable for us to take these features into account when deciding on a candidate operator.

As is common with bandit problems, our goal is to minimize regret, the difference between the performance of the algorithm and that of the optimal algorithm in hindsight:

$$R = \sum_{t=1}^{T} (l_t - l_t^*) \tag{9}$$

where l_t^* is the loss incurred at each round by the optimal strategy.

7.2.2 Exploration-Exploitation Trade-off

Finding the right balance between exploration and exploitation when dealing with a bandit setting is one of the core problems in the field. For the standard multi-armed bandit problem, some simple approaches include:

ϵ-greedy strategy. The best known arm is selected for a proportion $1 - \epsilon$ of the time and a random arm is selected for a proportion ϵ [107].

ϵ-first strategy. A pure exploration phase is followed by a pure exploitation phase. For an experiment of length T, the exploration phase where a random arm is chosen occupies ϵT steps and the exploitation phase where the best arm is chosen occupies the remaining $(1 - \epsilon)T$ steps.

ϵ-decreasing strategy. Similar to the ϵ-greedy strategy, except that the value of ϵ decreases as the experiment progresses, resulting in higher exploration earlier in the experiment and more exploitative behavior later.

Approaches such as these are not well-suited our problem since they ignore the availability of contextual information. We therefore choose to deal with the exploration-exploitation trade-off through the use of confidence bounds. With a model that is able to supply confidence bounds, the widths of the confidence bounds reflect the uncertainty of the algorithm's knowledge. By choosing the candidate with the highest upper confidence bound at each time step, the algorithm elegantly trades off between exploration and exploitation. When uncertainty is high, choosing that candidate will provide information that will quickly reduce uncertainty in that region of the model. As we gain knowledge about each candidate, confidence bounds will shrink and we will choose the candidate with the highest expected performance. This approach was well-justified for the bandits setting and shown to have small regret [108].

7.2.3 Linear Optimization as Multi-Armed Bandits Problem

An algorithm that was very influential on our approach was a linear optimization analog of the k-armed bandits problem proposed by Dani et al. where rather than finitely many arms, the decision set is a compact subset $D \subset \mathbb{R}^n$ [109]. At each step, the algorithm must choose a decision $x_t \in D$, and each choice results in a loss $l_t = c_t(x_t)$ where c_t is assumed to be a fixed linear function with some amount of additional noise.

Their algorithm utilizes upper confidence bounds by maintaining an ellipsoidal region in which the optimal decision μ is contained with high probability. Suppose decisions $x_1, \ldots, x_{t-1}$ have been made, incurring corresponding losses $l_1, \ldots, l_{t-1}$. Then their estimate $\hat{\mu}$ to the true cost vector μ can be constructed by minimizing the square loss:

$$\hat{\mu} = \underset{v}{\operatorname{argmin}}\ \mathcal{L}(v),\ \text{where}\ \mathcal{L}(v) = \sum_{\tau<t}(v^T x_\tau - l_\tau)^2 \tag{10}$$

A natural confidence region for μ is then the set v of decisions for which $\mathcal{L}(v)$ exceeds $\mathcal{L}(\hat{\mu})$ by at most some amount β:

$$\{v | \mathcal{L}(v) - \mathcal{L}(\hat{\mu_t}) <= \beta\} \tag{11}$$

The confidence region at time t, B_t, is defined to be the ellipsoid that contains the region defined in (11). The decision at the next round is then the greedy optimistic decision:

$$x_t = \underset{x\in D}{\operatorname{argmin}}\ \min_{v\in B_t}(v^T x) \tag{12}$$

We propose to utilize a variation of this approach as described in the following section.

7.2.4 Formalization

We are dealing with the following online candidate selection problem. At each time step t, we get some contextual features x_t for our environment and must choose from one of k candidates to operate the robot for that time step[6]. These features will be generated from either on-board or overhead sources using similar methods to those described earlier and will contain information over a more broad area. For example, in the case of overhead features, one way to achieve this is to convolve the feature values for the environment with a Gaussian kernel in order to blur the data, in effect introducing an influence from surrounding areas into each location. The goal in such a setting is to minimize the amount of time spent dealing with the situation at that time step, measured by the period of time it takes the robot to enter and exit a 3 meter radius window around that location.

[6]In the case of choosing between a human and the autonomy system, $k = 2$. We discuss this problem in the more general case as it could also be applied to choosing between multiple autonomy systems, multiple human operators, etc.

After each selection, the algorithm observes the noisy feedback l_t^i of only the chosen candidate i. We assume a simple linear model for each l^i as a function of the contextual features x_t:

$$E(l_t^i|\mu^i, x_t) = \mu^i x_t \tag{13}$$

We assume the estimates have Gaussian noise and are therefore distributed:

$$\tilde{l}_t^i \sim Normal(l_t^i, \sigma^2) \tag{14}$$

Our problem differs from the linear optimization scenario of Dani et al. described above in that we do not choose $x_t \in D$ at each time step but rather receive a fixed x_t and must choose among our k candidates. We are therefore tracking k instances of the linear optimization problem in parallel, one for each candidate. This makes our confidence region problem simpler as we only have k alternatives to evaluate, the upper confidence estimate of x_t for each of the k candidates, rather than all hypotheses contained in the ellipsoid B_t. Bayesian Linear Regression as described in Section 5.2.1 is therefore an appropriate algorithm for maintaining estimates for each μ^i and generating upper confidence-based predictions.

7.3 Proposed Work

7.3.1 Candidate Selection

We propose to validate this candidate selection algorithm offline through two applications relevant to mobile robot navigation.

First, we plan to demonstrate how this approach can be used to manage the previously mentioned trade-off between human and autonomous control. While we do not have the system infrastructure to be able to trade-off online between tele-operation and autonomous vehicle control, we plan to simulate such an online scenario by using a pair of logged traversals of the same course by each candidate: a human tele-operator using a high-bandwidth camera system and the autonomy system. All locations where the path of the human driver and the autonomous driver are in sufficient proximity can be used as a training point for the system. As the algorithm chooses a candidate, the traversal time for the specified candidate that will be revealed to the algorithm.

We also plan to show how the same technique can be used to deal with scenarios where limited high-resolution overhead data is available to aid the robot in navigating through an environment. The Digital Terrain Elevation Data (DTED) level of an overhead elevation data set specifies its density of coverage. DTED level 3 overhead data is available for a majority of the world but is so sparse that in most cases it adds very little to the features that can be generated from overhead imagery. Meanwhile, higher resolution overhead data can be used to produce more accurate traversal cost estimates that the UGV can use for better prior path computation but often require expensive and time-consuming aerial surveying and a large amount of bandwidth if remotely supplied to the vehicle. In scenarios where there is either limited time to gather that data or limited bandwidth for wireless transmission of the data to the vehicle during navigation, our algorithm can be extended to allow the robot learn to identify the situations where it will most benefit from high resolution data in order to allocate it to areas that maximize its impact.

We plan to simulate such a scenario by utilizing a series of multi-waypoint logged runs from a previous field test on the same courses using DTED levels 3, 4 and 5 overhead data. The candidates for each waypoint in this case are the choice of density of overhead data for an area bounding that path segment. The candidate selection system therefore had the goal of learning a mapping from the average of feature values computed from readily available overhead imagery and DTED 3 data within the segment's bounding box to the average traversal speed for the vehicle over that segment of the path using each candidate type of data.

While DTED 5 data will almost always result in the best performance, we plan to deal with a scenario where high-density data is available for only a percentage of all path segments. At each step we plan to use a linear program to optimize the allocations of remaining data availability using the predicted performance on all remaining segments from the learned models for each candidate at that time. Selections at each step would then be based on the initial step of this locally computed optimal allocation. To avoid having to do integer programming, we would choose the candidate with the highest allocation at the first step.

We plan to measure the performances of our approach in each of these applications by tracking the cumulative and average regret of our algorithm compared to several simple approaches such as a random-candidate strategy.

7.3.2 Intelligent Uncertainty Resolution

An extension we would like to consider is an intelligent uncertainty-resolution technique to minimize the potential number of human queries during navigation. With a functioning novelty or change detection system, it is important to consider the benefit of resolving uncertainty, presumably through a human query. For example, if a novel object blocks a primary route to the goal and the next best alternative has a much higher cost, then the potential benefit of involving a human exceeds the cost (see Figure 36). Similarly, a candidate selection system choosing between autonomous and human control can avoid unnecessary human involvement in situations that can be relatively easily circumnavigated autonomously, allowing larger human utilization at more critical situations. By considering the possible impact on the global path of both possible extremes (trivially traversable and completely untraversable), we can choose to avoid novel or potentially difficult situations that cannot significantly improve our metrics.

Such an approach provides several key advantages. As mentioned previously, it enables us to better utilize human attention during hybrid control systems such as the candidate selection framework. Even without human involvement, a perception system can benefit from this technique when dealing with uncertainty. If a component of the system is not confident about it's prediction (for example, when a classifier is uncertain about whether an area contains vegetation or solid obstacles), rather than predicting a traversal cost that represents blend of the two possibilities, the system can choose to be conservative and label uncertainty with high cost un-

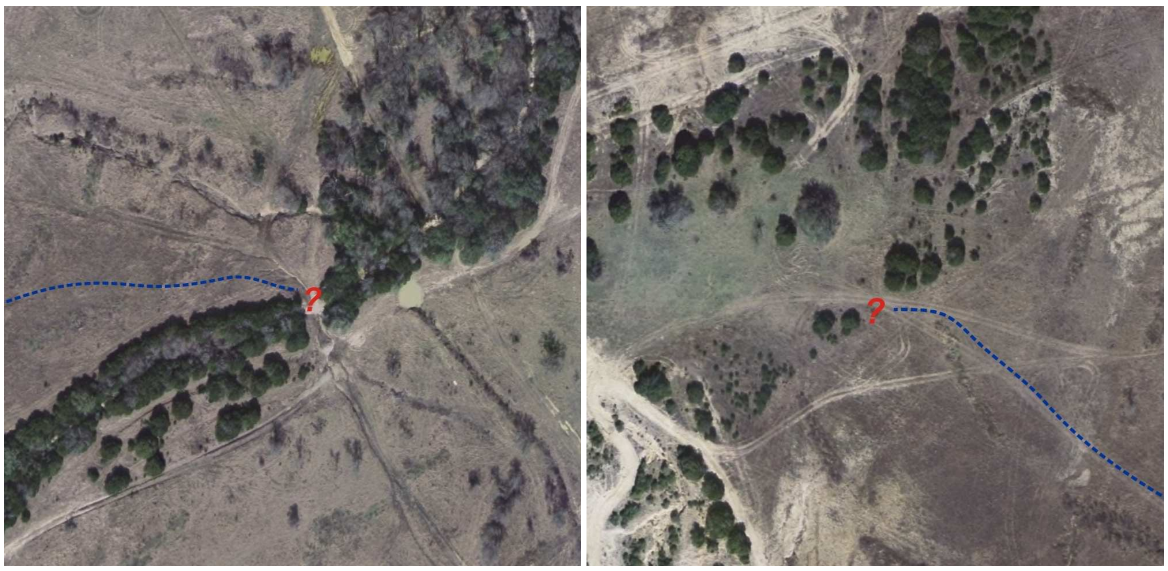


Figure 36: We plan to develop an uncertainty resolution technique that can distinguish between situations such as those above. In the left image, resolving the uncertainty at this pinch point through a human query can potentially save significant time and travel distance. In the right image, the outcome of a human query cannot significantly effect the path.

less there is a potentially high benefit from querying a human for help or attempting to navigate through the location.

Possibly of higher impact, such an approach would achieve the benefits of higher sensitivity within novelty or change detection systems without unacceptably high human time commitments. Because only a minority of situations will have the potential to significantly influence our relevant metrics, we can tolerate more false positives without the potential of heavily burdening a human supervisor.

We plan to demonstrate the functionality of this uncertainty resolution technique through experiments in conjunction with the novelty detection system described previously. We will design courses with a variety of novel situations and objects as described earlier with varying impact on the optimal path. We plan to show that compared to a baseline novelty detection system we can achieve equivalent or better safety, time and distance metrics. We will show that this can be achieved using fewer simulated human queries even while running a more sensitive novelty detection system.

8 Schedule

Table 3 presents a summary of tasks proposed in this thesis, identifying ones which have been completed and proposing a date of completion for the rest.

As can be seen from this table, all perception system related online learning work has already been completed and the major remaining work is related to systems development and integration onto the E-gator platform, the *deep memory* approaches for novelty detection and

Table 3: Schedule of proposed work

Task	Date of Completion
Perception-Related Online Learning (Section 5)	
Core algorithm development	Done
Overhead Map Online Learning (MOLL)	Done
Far-Range Online Learning (FROLL)	Done
Extensions and testing onboard UPI vehicles	Done
E-gator platform development (Section 4.2.3)	Summer 2009
Online Novelty-Detection (Section 6)	
Core algorithm development	Done
Initial offline experiments	Done
Explore other potential *deep memory* approaches	Summer 2009
Novelty detection implementation onboard E-gator platform	Summer 2009
Change detection with positioning uncertainty	Fall 2009
Continued experiments	Fall 2009 / Spring 2010
Theoretical analysis and comparisons	Spring 2010
Online Candidate Selection (Section 7)	
Core algorithm development	Done
Operator selection experiments (on logged data)	Spring 2009
Overhead data selection experiments (on logged data)	Spring 2009
Uncertainty-resolution extension	Fall 2009
Thesis Writing	
Thesis proposal	May 2009
Thesis writing and defense	Spring / Summer 2010

change detection, and intelligent uncertainty resolution.

Algorithm development and testing is planned to continue throughout 2009, concluding with the thesis writing and defense in 2010.

9 Contributions

The contributions of this thesis will be a series of online probabilistic algorithms for enabling mobile robots to operate more effectively and safely in unfamiliar domains. We will argue how online techniques are necessary for overcoming the limitations of offline methods and how such techniques are vital to extending the real-world applications and impact of mobile robotics. We will also explore the idea of *deep memory*, the ability to represent and utilize large amounts of previously seen information, and how such techniques can be applied to the problems of novelty and change detection. Furthermore, we will demonstrate through both offline and online testing how combining the adaptive performance of our algorithms with the inherent mobility of a capable UGV can lead to more efficient navigation of complex environments. Finally, we will present theoretical justification for our approaches as well as a variety of extensions to our algorithms that will further their impact on the field of mobile robotics.

The combination of these techniques will improve the effectiveness and range of the robot's perception system, dramatically reduce the number of mission-ending errors by identifying potentially hazardous unfamiliar situations, reliably detect unexpected changes in previously traversed environments and allow better utilization of the availability of limited human assistance. Such capabilities will increase the reliability and robustness of mobile robot systems and will be a step towards enabling more UGVs to be fielded in real-world applications.

Acknowledgment

First, I would like to thank my two advisors, Tony Stentz and Drew Bagnell for their support, guidance and endless patience, as well as Chris Urmson and Larry Jackel for being part of my thesis committee and contributing insightful suggestions for the future direction of this work. I would also like to thank all of the members of the UPI project team, without whom this work would not be possible. Particular thanks go to David Bradley, David Silver and Cristian Dima who created many of the classification data sets and perception tools used extensively throughout this work, Cliff Olmstead whose exceptional visualization tools were used to generate portions of the results and Thomas Pilarski who was able to simultaneously juggle numerous constraints and research agendas on the project while still ensuring that everyone was able to accomplish their goals. I would also like to thank Dominic Jonak, Balajee Kannan, Freddie Dias, Marc Zinck, Bradford Neuman and Nisarg Kothari of the CTA project for their help in the development of the E-Gator platform. Finally, I would like to thank John Cole, Nicolas Vandapel and Ellie Lin with whom I collaborated heavily on the earlier portions of this work.

This work was partially sponsored by DARPA under contract Unmanned Ground Combat Vehicle - PerceptOR Integration (contract number MDA972-01-9-0005) and by the U.S. Army Research Laboratory under contract Robotics Collaborative Technology Alliance (contract number DAAD19-01-2-0012). 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 U.S. Government.

Boris Sofman is partially supported by a Sandia National Laboratories Excellence in Engineering Fellowship. The author gratefully acknowledges Sandia's Campus Executive Laboratory Directed Research and Development Program (LDRD) for this support. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DOE. No. DE-AC04-94AL85000.

References

[1] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. R. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[2] A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarski, P. Rander, S. Thayer, N. Vallidis, and R. Warner, "Toward reliable off road autonomous vehicles operating in challenging environments," *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 449–483, 2006.

[3] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Stanley: The robot that won the DARPA grand challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, June 2006.

[4] L. D. Jackel, E. Krotkov, M. Perschbacher, J. Pippine, and C. Sullivan, "The DARPA LAGR program: Goals, challenges, methodology, and phase I results," *Journal of Field Robotics*, vol. 23, no. 11-12, pp. 945–973, 2006.

[5] S. Goldberg, Maimone, and L. Matthies, "Stereo vision and rover navigation software for planetary exploration," in *Proceedings of the IEEE Aerospace Conference*, 2002.

[6] B. Bodta and R. Camden, "Technology readiness level 6 and autonomous mobility," in *Proceedings of the SPIE, Volume 5422, Unmanned Ground Vehicle Technology VI*, 2004.

[7] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *ICRA*, 1994, pp. 3310–3317.

[8] A. Stentz, C. Dima, C. Wellington, H. Herman, and D. Stager, "A system for semi-autonomous tractor operations," *Autonomous Robots*, vol. 13, no. 1, pp. 87–104, 2002.

[9] A. Stentz, "Robotic technologies for outdoor industrial vehicles," in *Proceedings of SPIE AeroSense*, 2001.

[10] A. Stentz, J. Bares, T. Pilarski, and D. Stager, "The crusher system for autonomous navigation," in *AUVSIs Unmanned Systems North America*, August 2007.

[11] D. M. Bradley, R. Unnikrishnan, and J. Bagnell, "Vegetation detection for driving in complex environments," in *ICRA*. IEEE, 2007, pp. 503–508.

[12] J.-F. Lalonde, N. Vandapel, D. Huber, and M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility," *Journal of Field Robotics*, vol. 23, no. 1, pp. 839 – 861, November 2006.

[13] D. Ferguson and A. Stentz, "Using interpolation to improve path planning: The field D* algorithm," in *Journal of Field Robotics*, vol. 23, no. 1. John Wiley & Sons, February 2006, pp. 79–101.

[14] D. Silver, B. Sofman, N. Vandapel, J. A. Bagnell, and A. Stentz, "Experimental analysis of overhead data processing to support long range navigation," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, October 2006, pp. 2443 – 2450.

[15] B. Sofman, J. A. Bagnell, A. Stentz, and N. Vandapel, "Terrain classification from aerial data to support ground vehicle navigation," Robotics Institute, Carnegie Mellon University, Tech. Rep., August 2005.

[16] D. Silver, J. D. Bagnell, and A. T. Stentz, "High performance outdoor navigation from overhead data using imitation learning," in *Robotics Science and Systems*, July 2008.

[17] B. Sofman, E. L. Ratliff, J. A. Bagnell, J. Cole, N. Vandapel, and A. Stentz, "Improving robot navigation through self-supervised online learning," *Journal of Field Robotics*, vol. 23, no. 1, December 2006.

[18] C. Urmson, "Navigation regimes for off-road autonomy," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, 2004.

[19] P. Tompkins, "Mission-directed path planning for planetary rover exploration," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, 2004.

[20] C. Scrapper, A. Takeuchi, T. Chang, T. Hong, and M. Shneier, "Using a priori data for prediction and object recognition in an autonomous mobile vehicle," in *Proceedings of the SPIE Aerosense Conference*, April 2003.

[21] A. P. Charaniya, R. Manduchi, and S. K. Lodha, "Supervised parametric classification of aerial lidar data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2004.

[22] G. Cao, X. Yang, and Z. Mao, "A two-stage level set evolution scheme for man-made objects detection," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2005.

[23] T. Knudsen and A. A. Nielson, "Detection of buildings through multivariate analysis of spectral, textural, and shape based features," in *Proceedings of IGARSS*, 2004.

[24] A. Stentz, A. Kelly, P. Rander, H. Herman, and O. Amidi, "Real-time, multi-perspective perception for unmanned ground vehicles," in *Proceedings of the Association for Unmanned Vehicle Systems International*, 2003.

[25] N. Vandapel, R. R. Donamukkala, and M. Hebert, "Experimental results in using aerial ladar data for mobile robot navigation," in *Proceedings of the International Conference on Field and Service Robotics*, 2003.

[26] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. R. Bradski, "Self-supervised monocular road detection in desert terrain," in *Robotics: Science and Systems*, G. S. Sukhatme, S. Schaal, W. Burgard, and D. Fox, Eds. The MIT Press, 2006.

[27] D. Lieb, A. Lookingbill, and S. Thrun, "Adaptive road following using self-supervised learning and reverse opticalflow," in *Proceedings of Robotics: Science and Systems*, June 2005.

[28] A. Lookingbill, J. Rogers, D. Lieb, J. Curry, and S. Thrun, "Reverse optical flow for self-supervised adaptive autonomous robot navigation," *International Journal of Computer Vision*, vol. 74, no. 3, pp. 287–302, Sep. 2007.

[29] D. Stavens and S. Thrun, "A self-supervised terrain roughness estimator for off-road autonomous driving," in *UAI*. AUAI Press, 2006.

[30] P. Rowe, "Adaptive motion planning for autonomous mass excavation," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, January 1999.

[31] Y. LeCun, U. Muller, J. Ben, E. Cosatto, and B. Flepp, "Off-road obstacle avoidance through end-to-end learning," in *Proceedings of the Neural Information Processing Systems*, 2005.

[32] P. Vernaza, B. Taskar, and D. D. Lee, "Online, self-supervised terrain classification via discriminatively trained submodular markov random fields," in *ICRA*. IEEE, 2008, pp. 2750–2757.

[33] J. Michels, A. Saxena, and A. Y. Ng, "High speed obstacle avoidance using monocular vision and reinforcement learning," in *Proceedings of the International Conference on Machine Learning*, 2006.

[34] P. Abbeel, A. Coates, M. Montemerlo, and A. Y. N. andSebastian Thrun, "Discriminative training of kalman filters," in *Proceedings of Robotics: Science and Systems*, 2005.

[35] D. Kim, J. Sun, S. Oh, J. Rehg, and A.Bobick, "Traversability classification using unsupervised on-line visual learning for outdoor robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006.

[36] A. Angelova, L. Matthies, D. Helmick, G. Sibley, and P. Perona, "Learning to predict slip for ground robots," in *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, 2006.

[37] C. Wellington, “Learning a terrain model for autonomous navigation in rough terrain,” Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, December 2005.

[38] D. Blei, J. Bagnell, and A. McCallum, “Learning with scope, with application to information extraction and classification,” in *Proceedings of the 2002 Conference on Uncertainty in Artificial Intelligence*, June 2002.

[39] M. I. Jordan and Y. Weiss, *Graphical models: Probabilistic inference*. MIT Press, 2002.

[40] S. M. Kakade and A. Y. Ng, “Online bounds for bayesian algorithms,” in *Advances in Neural Information Processing Systems 17*, L. K. Saul, Y. Weiss, and L. Bottou, Eds. Cambridge, MA: MIT Press, 2005, pp. 641–648.

[41] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*. Chapman and Hall/CRC, 2004.

[42] M. E. Tipping, “Bayesian inference: An introduction to principles and practice in machinelearning,” in *Proceedings of the Advanced Lectures on Machine Learning*, 2003.

[43] A. Y. Ng and M. I. Jordan, “Convergence rates of the voting gibbs classifier, with application to bayesianfeature selection,” in *Proceedings of the International Conference on Machine Learning*, 2001.

[44] T. Minka, “A family of algorithms for approximate bayesian inference,” 2001.

[45] R. O. Duda and P. E. Hart, *Pattern Classification*. John Wiley and Sons, 2000.

[46] N. D. Ratliff, D. Silver, and J. A. Bagnell, “Learning to search: Functional gradient techniques for imitation learning,” *Autonomous Robots, special issue on Robot Learning*, 2009.

[47] J. Kivinen, A. J. Smola, and R. C. Williamson, “Online learning with kernels,” in *NIPS*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. MIT Press, 2001, pp. 785–792.

[48] K. Worden, “Structural fault detection using a novelty meassure,” *Journal of Sound and Vibration*, vol. 201, no. 1, pp. 85–101, 1997.

[49] P. Hayton, B. Schölkopf, L. Tarassenko, and P. Anuzis, “Support vector novelty detection applied to jet engine vibration spectra,” in *NIPS*, T. K. Leen, T. G. Dietterich, and V. Tresp, Eds. MIT Press, 2000, pp. 946–952.

[50] J. Ryan, M.-J. Lin, and R. Miikkulainen, “Intrusion detection with neural networks,” in *Advances in Neural Information Processing Systems*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Eds., vol. 10. The MIT Press, 1998.

[51] L. Tarassenko, P. Hayton, N. Cerneaz, and M. Brady, "Novelty detection for the identification of masses in mammograms," in *Proceedings of the Fourth International IEEE Conference on Artificial Neural Networks*, vol. 409, 1995, pp. 442–447.

[52] H. V. Neto and U. Nehmzow, "Visual novelty detection with automatic scale selection," *Robotics and Autonomous Systems*, vol. 55, no. 9, pp. 693–701, 2007.

[53] S. Marsland, U. Nehmzow, and J. Shapiro, "On-line novelty detection for autonomous mobile robots," *Robotics and Autonomous Systems*, vol. 51, no. 2-3, pp. 191–206, 2005.

[54] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *NIPS*, S. A. Solla, T. K. Leen, and K.-R. Müller, Eds. The MIT Press, 1999, pp. 582–588.

[55] L. M. Manevitz and M. Yousef, "One-class SVMs for document classification," *Journal of Machine Learning Research*, vol. 2, pp. 139–154, 2001.

[56] C. Campbell and K. P. Bennett, "A linear programming approach to novelty detection," in *NIPS*, T. K. Leen, T. G. Dietterich, and V. Tresp, Eds. MIT Press, 2000, pp. 395–401.

[57] D. M. J. Tax and R. P. W. Duin, "Support vector domain description," *Pattern Recognition Letters*, vol. 20, no. 11-13, pp. 1191–1199, 1999.

[58] H. Hoffmann, "Kernel PCA for novelty detection," *Pattern Recognition*, vol. 40, no. 3, pp. 863–874, Mar. 2007.

[59] N. Japkowicz, C. Myers, and M. A. Gluck, "A novelty detection approach to classification," in *IJCAI*, 1995, pp. 518–523.

[60] M. Markou and S. Singh, "Novelty detection: a review - part 1: statistical approaches," *Signal Processing*, vol. 83, no. 12, pp. 2481–2497, 2003.

[61] ——, "Novelty detection: a review - part 2: neural network based approaches," *Signal Processing*, vol. 83, no. 12, pp. 2499–2521, 2003.

[62] B. Sofman, J. A. Bagnell, and A. Stentz, "Anytime online novelty detection for vehicle safeguarding," Robotics Institute, Carnegie Mellon University, Tech. Rep., April 2009.

[63] B. Schölkopf and A. J. Smola, *Learning with Kernels*. Cambridge, MA: The MIT Press, 2002.

[64] D. Sleator and R. Tarjan, "Amortized efficiency of list update and paging rules," *CACM: Communications of the ACM*, vol. 28, 1985.

[65] E. Jones, B. Browning, M. B. Dias, B. Argall, M. M. Veloso, and A. Stentz, "Dynamically formed heterogeneous robot teams performing tightly-coordinated tasks," in *International Conference on Robotics and Automation*, May 2006, pp. 570 – 575.

[66] M. B. Dias, B. Kannan, B. Browning, E. Jones, B. Argall, M. F. Dias, M. B. Zinck, M. M. Veloso, and A. Stentz, “Sliding autonomy for peer-to-peer human-robot teams,” in *10th International Conference on Intelligent Autonomous Systems 2008*, July 2008.

[67] B. Argall, Y. Gu, B. Browning, and M. M. Veloso, “The first segway soccer experience: Towards peer-to-peer human-robot teams,” in *In Proceedings First Annual Conference on Human-Robot Interactions*, March 2006.

[68] D. Casbeer, R. Beard, T. McLain, S.-M. Li, and R. Mehra, “Forest fire monitoring with multiple small UAVs,” *American Control Conference, 2005. Proceedings of the 2005*, pp. 3530–3535 vol. 5, June 2005.

[69] A. Girard, A. Howell, and J. Hedrick, “Border patrol and surveillance missions using multiple unmanned air vehicles,” *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 1, pp. 620–625 Vol.1, Dec. 2004.

[70] J. Casper and R. R. Murphy, “Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 33, no. 3, pp. 367–385, 2003.

[71] J. Scholtz, J. Young, J. L. Drury, and H. A. Yanco, “Evaluation of human-robot interaction awareness in search and rescue,” in *International Conference on Robotics and Automation*. IEEE, 2004, pp. 2327–2332.

[72] H. A. Yanco and J. L. Drury, “Rescuing interfaces: A multi-year study of human-robot interaction at the AAAI robot rescue competition,” *Auton. Robots*, vol. 22, no. 4, pp. 333–352, 2007.

[73] I. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, and S. Burion, “Human-robot teaming for search and rescue,” *Pervasive Computing, IEEE*, vol. 4, no. 1, pp. 72–79, Jan.-March 2005.

[74] B. Yamauchi, “Packbot: a versatile platform for military robotics,” in *Proceedings of SPIE*, vol. 5422, 2004, pp. 228–237.

[75] R. C. Coulter, A. Stentz, P. G. Keller, G. K. Shaffer, W. R. L. Whittaker, B. Brummit, and W. Burky, “A system for telerobotic control of servicing tasks in a nuclear steam generator,” Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-90-24, December 1990.

[76] R. C. Coulter, A. Stentz, P. Keller, and W. R. L. Whittaker, “A telerobotic solution for tool insertion tasks in nuclear servicing,” in *Remote Systems Session of the ANS Winter Meeting*, November 1991.

[77] G. Sung and I. Gill, “Robotic laparoscopic surgery: a comparison of the da vinci and zeus systems.” *Urology*, vol. 58, no. 6, pp. 893–8, 2001.

[78] R. Grace, V. Byrne, D. Bierman, J.-M. Legrand, D. Gricourt, B. Davis, J. Staszewski, and B. Carnahan, “A drowsy driver detection system for heavy vehicles,” in *Proceedings of the 17th Digital Avionics Systems Conference*, vol. 2, 2001, pp. I36/1 – I36/8.

[79] M. Wada, K. S. Yoon, and H. Hashimoto, “Development of advanced parking assistance system,” *Industrial Electronics, IEEE Transactions on*, vol. 50, no. 1, pp. 4–17, Feb 2003.

[80] A. Vahidi and A. Eskandarian, “Research advances in intelligent collision avoidance and adaptive cruise control,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 4, no. 3, pp. 143–153, 2003.

[81] J. McCall and M. Trivedi, “Driver behavior and situation aware brake assistance for intelligent vehicles,” *Proceedings of the IEEE*, vol. 95, no. 2, 2007.

[82] R. Bishop, “Intelligent vehicle applications worldwide,” *IEEE Intelligent Systems*, vol. 15, no. 1, pp. 78–81, 2000.

[83] C. Tomlin, G. Pappas, and S. Sastry, “Conflict resolution for air traffic management: a study in multiagent hybrid systems,” *Automatic Control, IEEE Transactions on*, vol. 43, no. 4, pp. 509–521, Apr 1998.

[84] A. Krupa, M. de Mathelin, C. Doignon, J. Gangloff, G. Morel, L. Soler, and J. Marescaux, “Development of semi-autonomous control modes in laparoscopic surgery using automatic visual servoing,” *Lecture Notes in Computer Science*, vol. 2208, pp. 1306–??, 2001.

[85] E. Krotkov, R. Simmons, F. Cozman, and S. Koenig, “Safeguarded teleoperation for lunar rovers: From human factors to field trials,” in *In Proc. IEEE Planetary Rover Technology and Systems Workshop*, 1996.

[86] J. Carsten, A. Rankin, D. Ferguson, and A. Stentz, “Global path planning on board the mars exploration rovers,” *Aerospace Conference, 2007 IEEE*, pp. 1–11, March 2007.

[87] G. A. Dorais, R. P. Bonasso, D. Kortenkamp, B. Pell, and D. Schreckenghost, “Adjustable autonomy for human-centered autonomous systems on mars,” Apr. 12 1998.

[88] R. T. Maheswaran, M. Tambe, P. Varakantham, and K. L. Myers, “Adjustable autonomy challenges in personal assistant agents: A position paper,” in *Agents and Computational Autonomy*, ser. Lecture Notes in Computer Science, M. Nickles, M. Rovatsos, and G. Weiß, Eds., vol. 2969. Springer, 2003, pp. 187–194.

[89] F. W. Heger and S. Singh, “Sliding autonomy for complex coordinated multi-robot tasks: Analysis & experiments,” in *Robotics: Science and Systems*, G. S. Sukhatme, S. Schaal, W. Burgard, and D. Fox, Eds. The MIT Press, 2006.

[90] B. P. Sellner, F. Heger, L. Hiatt, R. Simmons, and S. Singh, "Coordinated multi-agent teams and sliding autonomy for large-scale assembly," *Proceedings of the IEEE - Special Issue on Multi-Robot Systems,* vol. 94, no. 1, pp. 1425 – 1444, July 2006.

[91] E. Horvitz, A. Jacobs, and D. Hovel, "Attention-sensitive alerting," in *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI-99),* K. B. Laskey and H. Prade, Eds. S.F., Cal.: Morgan Kaufmann Publishers, Jul. 30–Aug. 1 1999, pp. 305–313.

[92] H. Hexmoor, "A cognitive model of situated autonomy," *Lecture Notes in Computer Science*, vol. 2112, pp. 325–, 2001.

[93] J. P. Gunderson and W. N. Martin, "Effects of uncertainty on variable autonomy in maintenance robots," in *In Workshop on Autonomy Control Software*, 1999, pp. 26–34.

[94] P. Scerri, D. V. Pynadath, and M. Tambe, "Towards adjustable autonomy for the real world," *Journal of Artificial Intelligence Research*, vol. 17, p. 2002, 2002.

[95] D. M. Brann, D. A. Thurman, and C. M. Mitchell, "Human interaction with lights-out automation: a field study," in *In Proceedings of the 1996 Symposium on Human Interaction and Complex Systems*, 1996, pp. 276–283.

[96] T. W. Fong, "Collaborative control: A robot-centric model for vehicle teleoperation," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2001.

[97] T. W. Fong, C. Thorpe, and C. Baur, "Multi-robot remote driving with collaborative control," *IEEE Transactions on Industrial Electronics*, 2003.

[98] M. A. Goodrich and A. C. Schultz, "Human-robot interaction: A survey," *Foundations and Trends in Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2007.

[99] H. Robbins, "Some aspects of the sequential design of experiments," *Bull. Amer. Math. Soc*, vol. 58, no. 5, pp. 527–535, 1952.

[100] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in applied mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[101] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning, 47*, vol. 2, no. 3, pp. 235–256, 2002.

[102] B. Ghosh and P. Sen, *Handbook of sequential analysis*. Marcel Dekker, 1991.

[103] T. Lai, "Adaptive treatment allocation and the multi-armed bandit problem," *The Annals of Statistics*, pp. 1091–1114, 1987.

[104] M. Weitzman, "Optimal search for the best alternative," *Econometrica: Journal of the Econometric Society*, pp. 641–654, 1979.

[105] C. Wang, S. Kulkarni, and H. Poor, "Bandit problems with side observations," *IEEE Transactions on Automatic Control*, vol. 50, no. 3, pp. 338–355, 2005.

[106] J. Langford and T. Zhang, "The epoch-greedy algorithm for contextual multi-armed bandits," *Advances in Neural Information Processing Systems*, 2007.

[107] C. Watkins, *Learning from delayed rewards*. Cambridge University, 1989.

[108] P. Auer, "Using confidence bounds for exploitation-exploration trade-offs," *The Journal of Machine Learning Research*, vol. 3, pp. 397–422, 2003.

[109] V. Dani, T. P. Hayes, and S. M. Kakade, "Stochastic linear optimization under bandit feedback," in *21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008*. Omnipress, 2008, pp. 355–366.