Mapping, Planning, and Sample Detection Strategies for Autonomous Exploration

Arun Das
Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1
e-mail: adas@uwaterloo.ca
Michael Diu
Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1
e-mail: mdiu@uwaterloo.ca
Neil Mathew
Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1
e-mail: nmathew@uwaterloo.ca
Christian Scharfenberger
Systems Design Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1
e-mail: cscharfenberger@uwaterloo.ca
James Servos and Andy Wong
Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1
e-mail: jdservos@uwaterloo.ca, andy.wong@uwaterloo.ca
John S. Zelek and David A. Clausi
Systems Design Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1
e-mail: jzelek@uwaterloo.ca, dclausi@uwaterloo.ca
Steven L. Waslander
Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1
e-mail: stevenw@uwaterloo.ca

Received 15 December 2012; accepted 8 October 2013

ABSTRACT: This paper presents algorithmic advances and field trial results for autonomous exploration and proposes a solution to perform simultaneous localization and mapping (SLAM), complete coverage, and object detection without relying on GPS or magnetometer data. We demonstrate an integrated approach to the exploration problem, and we make specific contributions in terms of mapping, planning, and sample detection strategies that run in real-time on our custom platform. Field tests demonstrate reliable performance for each of these three main components of the system individually, and high-fidelity simulation based on recorded data playback demonstrates the viability of the complete solution as applied to the 2013 NASA Sample Return Robot Challenge.

1 INTRODUCTION

Exploration is an important and active area of research in field robotics, as vehicles capable of autonomous exploration have the potential to significantly impact a wide range of applications such as search and rescue operations, environmental monitoring, and planetary exploration. Search and rescue robots have been used in many recent tragedies, such as the 2001 World Trade Center collapse (Murphy, 2004), Hurricane Katrina in 2005 (Murphy et al., 2008), and the 2011 Tohuko tsunami and earthquake (Guizzo, 2011). Currently, the majority of robots used for search and rescue missions are teleoperated. They are effective to an extent, but fall short in their ability to reduce the workload of the field operator in time-critical disaster scenarios (Liu and Nejat, 2013). Vehicle autonomy in such scenarios can vastly enhance the rescue efforts of human operators. Advances in autonomous exploration are also of importance to robots tasked with climate change monitoring, forest resource monitoring, wildlife population monitoring (Dunbabin and Marques, 2012), and biogeological surveying (Wettergreen et al., 2008), as improvements in vehicle autonomy allow for scientific data collection in settings such as the deep sea, volcanoes, and other locations that are unsuitable for human exploration. Autonomous exploration capabilities are also desirable for Lunar and Martian missions as teleoperation becomes cumbersome due to large transmission delays. The currently active Mars

Science Laboratory (MSL) uses semiautonomous systems to perform local drivability analysis, path planning, and feature tracking onboard, however object identification and long-range exploration missions remain mostly a manual process (Howard et al., 2012; Kim et al., 2009).

For this work, we define the *exploration problem* as simultaneously performing coverage of an unknown environment, mapping the area, and detecting objects of interest. There are three main challenges present in a complete solution to the exploration problem. First, the approach should maintain a globally consistent map over long distances with mainly relative measurement information and intermittent absolute measurements, such as GPS and magnetometers. Although the extensive use of GPS is typical in rover applications, GPS becomes unreliable in many situations due to multipath and occluded sky-view. Furthermore, the use of magnetometer information is unreliable on vehicles with variable current draw and in environments with large sources of magnetic interferences, as could be the case in a disaster scenario. Second, the solution should reliably identify potential objects of interest at as great a range as possible to minimize the time spent sweeping an environment for candidate objects, as well as identify objects of interest in varying lighting and environmental conditions. Finally, a method to plan an efficient search path over a terrain with unknown obstacles and contours is required. As the terrain is not known ahead of time, the solution should not require full map knowledge and allow for replanning as the location is explored.

Existing approaches to the exploration problem largely treat the mapping, object detection, and planning components as separate. While successful autonomous mapping and planning in outdoor environments has been demonstrated (Bogdan Rusu et al., 2009; Broggi, Medici, Zani, Coati, and Panciroli, 2012; Kümmerle, Hahnel, Dolgov, Thrun, and Burgard, 2009; Urmson et al., 2008), the approaches generally consider local path computation and obstacle avoidance without addressing additional planning objectives such as coverage and search. General coverage approaches (Arkin, Held, and Smith, 2000; Zheng, Jain, Koenig, and Kempe, 2005) assume a known map, and often do not consider the notion of robust coverage in partially known environments. The objective of coverage is to generate a path to pass a given sensor footprint over all areas of a search environment. Given the dependency on the sensor footprint, it is clear that the navigation algorithms cannot be developed in isolation from object detection, as the design of the coverage path is inherently related to the maximum range at which the object detection can reliably perform. Thus, the development of a complete autonomous rover system requires careful consideration for each of the algorithms to ensure the overall effectiveness of the system. Integration of mapping, planning, and object detection has been explored, to varying degrees, as part of the RoboCup Rescue challenge (Balakirsky et al., 2007), however the competition focuses primarily on urban search and rescue and multirobot coordination, whereas our goal is a single-robot solution to the exploration problem in a larger, sparser, and unstructured environment, such as a forest.

In this work, we present contributions to each of the algorithmic challenges of exploration, as well as the considerations that allow for high performance of the integrated system. The large computational burden of existing three-dimensional (3D) laser localization and mapping methods is relaxed by first segmenting scan data to remove the ground plane completely, and then performing class constrained iterated closest point matching on a binned 2D representation of the remaining points. Complete coverage of the environment is efficiently guaranteed through convex polygonal decomposition of the free space around obstacles followed by a graph-based optimal path computation through the polygonal space. While searching the environment, real-time object detection is enabled through a novel salient object detection method, which accurately identifies those images that actually contain an object of interest, thereby greatly reducing the computational burden generated when processing three high-resolution camera streams simultaneously. We further define a specific set of object feature descriptors that provide robust sample classification, reducing the need for deviations from the search path. Each of the three methods is demonstrated to work reliably using field data collected in an outdoor forested environment as part of development for the 2013 NASA Sample Return Robot Challenge (NSRRC). Finally, we demonstrate the interdependence between the core components and highlight the design decisions that affect the complete approach. It should be noted that although the presented work is applied to the 2013 NSRRC, the methodology can be applied to general terrestrial missions in GPS-denied environments as well. The sample detection approach only requires the presence of salient objects, the mapping approach can operate reliably in any environment where there are sufficient geometric features for scan registration, and the planning algorithm can reliably guarantee coverage so long as the map remains globally consistent.

The autonomous rover designed for the 2013 NSRRC includes a Velodyne HDL-32E LIDAR, a Microstrain 3DM-GX1-25 inertial measurement unit (IMU), and three Point Grey Firefly MV 1.3 MP cameras, and it runs all algorithms on two Intel Core i7 laptops with 8 GB of RAM. The software architecture for the vehicle is depicted in Figure 1, including the connections between the three main components described in this work. During execution, updates to the drivability map trigger a planning event, which computes an efficient route for complete search coverage using the desired sensor footprint and outputs a sequence of waypoints to be traversed by the robot. A local controller translates waypoints into local trajectories and consistently performs path tracking in accordance with the sensor footprint set by the sample detection block. When an object is identified, the

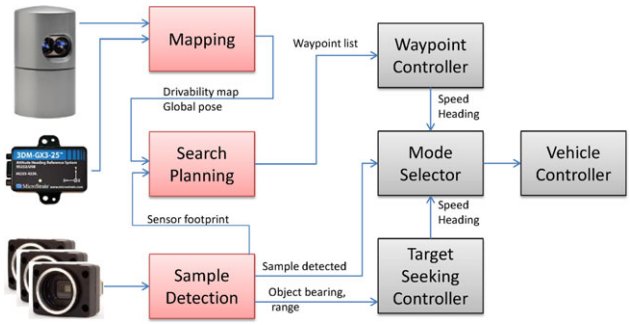


Figure 1. Software architecture for the 2013 NSRRC robot. The three red blocks of Mapping, Search Planning, and Sample Detection are the focus of this work as the solution to the autonomous exploration problem.

rover uses visual servoing to approach the object for further investigation before continuing on the coverage path.

This paper is divided into four main sections. First, Section 2 presents our SLAM algorithm tailored to produce a 2D drivability map from sparse laser data. The sample detection strategy, including presence detection, localization, and classification stages, is presented in Section 3, and the coverage path-planning method is defined in Section 4. Finally, in Section 5, a system demonstration on the 2013 NSRRC robot is presented, highlighting the interactions between the components of the system.

2 MAPPING

To perform high-level mission autonomy tasks such as vehicle path planning, obstacle avoidance, and exploration, a strategy to perform simultaneous localization and mapping (SLAM) is required. In a GPS- and magnetometer-denied environment, the error in the pose of the vehicle can grow unboundedly due to imprecise odometry and sensor noise. Thus, specific measures should be taken to relocalize against previously visited features. SLAM in a GPS- and magnetometer-denied environment also arises in indoor environments, which are generally well structured and allow for accurate localization and mapping using camera- or laser-based approaches (Engelhard, Endres, Hess, Sturm, and Burgard, 2011; Grisetti, G., Stachniss, C., and Burgard, 2007). It should be noted that although the proposed solution does not rely on GPS and magnetometer sensors, the information can be integrated to improve the solution when reliable measurements are available.

In outdoor settings, stereo vision has been successfully used for real-time SLAM by augmenting stereo feature matching with sparse bundle adjustments, and it is able to provide accurate pose information in rough outdoor environments over large trajectory lengths with relatively low position drift (Konolige, Agrawal, and Sol, 2011). While camera-based techniques have the advantage of providing extremely long-range bearing measurements, they suffer from poor range and field of view when compared to laser-based approaches, thus limiting the possibilities for autonomous navigation and requiring frequent stops to collect additional sensor data.

The use of LIDAR has been proposed to overcome the field of view and point cloud density limitations of stereo vision (Fong et al., 2008; Borrmann, Elseberg, Lingemann, Nüchter, and Hertzberg, 2008), with recent extensions and experimental results demonstrating consistent mapping results over areas as large as 60 × 100 m (Tong, Barfoot, and Dupuis, 2012). The ILRIS high-accuracy laser scanner used requires stationary data collection, and can produce highly detailed and accurate 3D point cloud maps of the environment, at significant computational cost, and leads to a stop-and-go approach similar to the Mars rover platforms, despite far greater computational capabilities onboard the testbed vehicles. The success of laser-based SLAM approaches is further demonstrated by autonomous driving platforms, such as the DARPA Urban Challenge vehicles or the Google driverless car. In these cases, however, a flat ground assumption or localization relative to a known map is used to generate drivability maps (Kammel et al., 2008; Levinson, 2011; Urmson et al., 2008), which are not feasible approaches for autonomous exploration.

A large, unstructured environment, such as a forest, makes laser scan registration-based approaches especially difficult since a typical point cloud from a laser scanner such as the Velodyne-HDL32E sensor is sparse and relatively noisy. Current state-of-the-art scan registration algorithms generally make assumptions about point cloud data that are not valid for our application. For example, 3D iterative closest point (ICP) methods requires a high point data density in order to provide accurate correspondences from a nearest-neighbor search (Nüchter, 2009). Generalized ICP (G-ICP) improves ICP by using the underlying surface structure of the point cloud to reject poorly corresponding points (Segal, Haehnel, and Thrun, 2009). The use of G-ICP requires the computation of surface normal information, which is difficult to perform accurately with noisy, unstructured point cloud data such as those generated from grass, trees, shrubs, or rubble. The Normal Distributions Transform models the point cloud as a set of Gaussian distributions, and performs a nonlinear optimization to align the two point clouds based on an algorithm-specific scoring function (Magnusson, Duckett, and Lilienthal, 2007; Stoyanov, Magnusson, Andreasson, & Lilienthal, 2012). This approach is conceptually suitable for an unstructured area, however in practice the algorithm suffers from poor convergence (Das and Waslander, 2012) and is not suitable for real-time implementation on a system with limited computation power.

A promising approach is the use of multilevel surface (MLS) maps (Kümmerle et al., 2009; Pfaff et al., 2007; Triebel, Pfaff, and Burgard, 2006), which model a point cloud with a

collection of patches. The patches are generated by binning the point cloud data into fixed sized columns, parallel to the height axis of the vehicle. The height information from the point cloud is used to create patches within each column, where each patch models surfaces at differing heights. The orientation of the patches can then be used to classify points as traversable or nontraversable. This classification system allows scan registration algorithms to constrain point correspondences between scans. Although conceptually attractive, the MLS mapping approach is not able to operate in real-time on a system with low computation resources. Real-time MLS mapping has been implemented on a full-sized Volkswagen vehicle (Kümmerle et al., 2009), however this platform possesses significantly more computational power when compared to what is available on our our platform.

2.1 Sparse Point Cloud SLAM Approach

The proposed method, *sparse point cloud SLAM* (SPC-SLAM), enables 2D SLAM to be performed in large, unstructured environments using sparse point clouds. It should be noted that although the presented solution is well suited to operate in open, unstructured environments, the approach would work well on any navigable terrain where there are sufficient environmental features for scan registration. To generate drivability maps in real time, the 3D point cloud is compressed into a 2.5D representation using a modified MLS map approach. While the general MLS approach models all drivable surfaces in the environment, the SPC-SLAM method treats the ground terrain as the only drivable level, and retains distributions of the obstacle points in order to perform outlier rejection for ICP scan registration. Since only a 2D drivability map is required, scans are first rotated based on vehicle pitch and roll estimates available from an extended Kalman filter (EKF), and then ground points are segmented and removed using Gaussian process regression. Ground segmentation is justified since the ground points contribute little to the localization accuracy, compared to the majority of the natural features in the environment such as trees and buildings. The remaining nonground points are then used to generate a 2D top-down map of the environment.

In the global map, each grid cell stores an average position of the planar projection of nonground points located within that bin in the global frame. In implementation, this point average is selected to be the 2D mean of the x-y point components. Using the mean points is a more robust approach than a naive occupancy grid, since the mean of the x-y points provides a better measure of location of the true model points within the cell. Accurate modeling of the points is especially important when performing ICP-based registration, as the ICP algorithm attempts to minimize the Euclidean distance between corresponding points. The height, or z-components, of the points located within each bin is used to generate a Gaussian distribution that models the height of the obstacles within the bin. This allows the global map to maintain a sense of the obstacle elevation within each bin, which is used to perform outlier rejection for the ICP algorithm.

A classification system is used to constrain the scan registration algorithm to compute point correspondences only between similarly classified points. Classification-based scan registration approaches have been successfully implemented in the MLS mapping technique, as the MLS approach classifies cells as drivable, nondrivable, or unknown. More general point cloud segmentation and classifications are also possible, and allow for the application of ICP correspondences to be constrained between segments of similar proximity, shape, and relative position between scans (Douillard et al., 2012). The *class-constrained ICP* (CC-ICP) methods have been shown to improve point correspondences and convergence rates, however general 3D point cloud segmentation is computationally expensive. To improve the robustness of our approach, the remaining obstacle points after the ground points have been removed are classified as *ground-adjacent* or *non-ground-adjacent*.

Ground adjacency classification is well suited for our application, as it ensures that edge features are maintained in the drivability map. Since the ground segmentation is already required to compute drivability, the determination of the nonground points requires no additional computation, and to further classify the points based on ground adjacency is computationally inexpensive. To localize the vehicle, a 2D CC-ICP registration is performed between x-y components of the local obstacle points and the cell mean components from the global map. The z-components for the points are compared to the Gaussian height distributions maintained as part of the global map, and a point-to-point correspondence for the CC-ICP registration is permitted only if the z-component for the local point is within one standard deviation for the Gaussian distribution of the corresponding cell.

To integrate new information into the global map, keyframes are employed (Bachrach, Prentice, He, and Roy, 2011; Davison, Reid, Molton, and Stasse, 2007; Hartley and Zisserman, 2004), which allows for the generation of a pose graph that is used to improve the global consistency of the map. A confidence score for the map is computed based on the amount of discrepancy between the global map and the current obstacle point cloud, which allows for the integration of new information into the map only when required, mitigating drift accumulation in the global map. To ensure a scan registration is valid, the CC-ICP registration score is computed using both the x-y component data and the height data of the point cloud. The score based on the x-y data is generated using the CC-ICP residuals, as this gives a measure of how well the CC-ICP algorithm converged. The score based on the height information is computed using the amount of overlap between the height distributions of the cells in the global map and distributions generated from

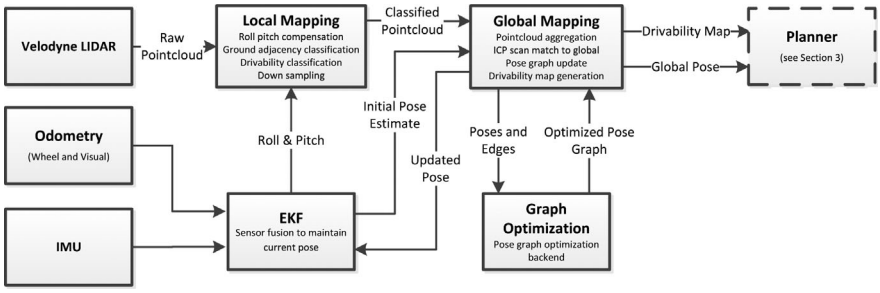


Figure 2. Block diagram of the SPC-SLAM mapping algorithm. The local mapping node uses the point clouds from the Velodyne LIDAR and the attitude estimate from the EKF to compensate the scan for the roll and pitch of the vehicle. The global mapping node then aggregates the information from the local maps, which is used to generate a drivability map for the path planner. The global mapping node also generates a pose graph that is updated by the graph optimization node.

the height components of the points in the registered point cloud. If this score falls below a user-defined threshold, the match is rejected. Vertices for the pose graph are added when the map uncertainty score grows above a user-defined threshold and new information is integrated into the global map. Additional edges for the pose graph are constructed by performing CC-ICP registration between the newly added vertex and its K-nearest-neighbor vertices, and edges for the pose graph are rejected based on the CC-ICP registration score. Once a loop closure is detected, the network graph optimizer g2o is used to optimize the pose graph represented by the stored keyframes (Kümmerle, Grisetti, Strasdat, Konolige, and Burgard, 2011). Finally, the global map is regenerated using the updated pose graph.

The SPC-SLAM system consists of four main nodes, as presented in Figure 2. The local mapping node is responsible for transforming, segmenting, and classifying the point cloud data from the Velodyne LIDAR. The EKF maintains an accurate estimate of the current robot pose, fusing information from all sources. The global mapping node aggregates keyframes into a global map based on the discrepancy score and generates the pose graph through CC-ICP scan registration. Finally, the graph optimization node is used to update the global pose graph and regenerate a drivability map, which is then passed to the global planner in order to perform coverage planning.

2.2 Ground Segmentation and Drivability Analysis

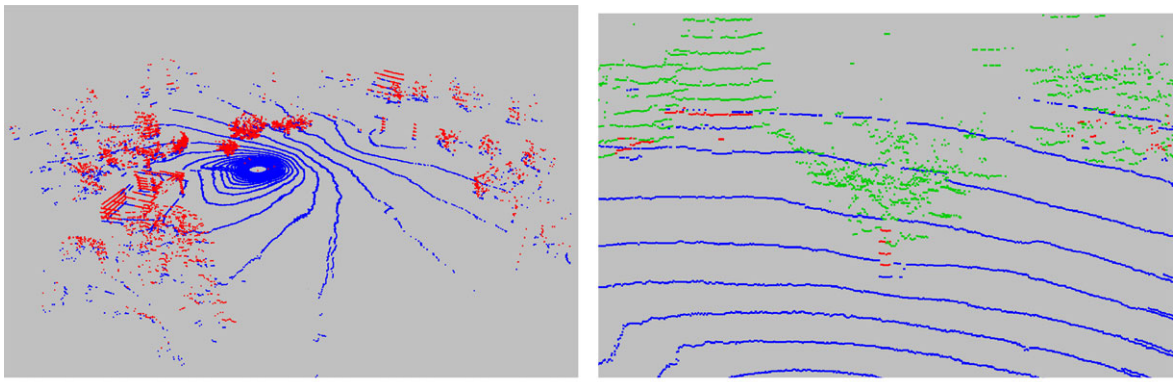
The ground segmentation algorithm used for SPC-SLAM was introduced by Tongtong, Bin, Daxue, Bo, and Qixu (2011). The goal of ground segmentation for the 3D point cloud is to assign each point as either belonging to the ground or not ground. Figure 3(a) presents the results of the GP-based ground segmentation approach on a typical point cloud from the Velodyne HDL-32E in a sparse forested environment containing a few additional manmade structures.

Once the points have been classified as obstacle or ground, they are further classified as drivable or not-drivable for the vehicle. For each sector, obstacle points where the difference in the the z-components between the obstacle point and the GP modelled ground point is less than the height of the vehicle, are classified as nondrivable points. The drivability classification is later used to generate a drivability map for the vehicle. Figure 3(b) illustrates the classification results for a typical point cloud in a forested area. As is visible in Figure 3(b), it is important to classify obstacle points as drivable versus nondrivable. Many features can include overhanging sections that the vehicle is capable of driving underneath. The drivability classification allows for the traversable sections to be accurately modeled, which is imperative for path-planning purposes.

2.3 Ground Adjacency Classification

Obstacle points are then further classified as either *ground-adjacent* and *non-ground-adjacent*. To classify the points, a local map is required. The SPC-SLAM local map partitions a plane in $\mathbb{R}^2$ that is orthogonal to the gravity vector for the vehicle orientation at the current time-step. Assume that the obstacle point cloud has been transformed to compensate for the roll and pitch angle of the vehicle, which can be directly provided by the EKF estimator. A point cloud is defined as the set of points $P = \{p_1, \ldots, p_{N_P}\}$, where $p_i \in \mathbb{R}^3$ for $i \in \{1, \ldots, N_P\}$. A point $p_j \in P$ consists of three components, $p_j = \{p_j^x, p_j^y, p_j^z\}$, which refer to the x, y, and z components of the point, respectively. Denote the local map as a set of fixed sized partitions, $\Lambda = \{l_1, \ldots, l_{N_\Lambda}\}$, where $l_i \subset \mathbb{R}^2$ is a cell in the local map, which contains N_Λ total cells. The cells are nonoverlapping, or $\Lambda = \cup_{i=1}^{N_\Lambda} l_i$ and $l_i \cap l_j = \emptyset$, $\forall i, j$, where $i \neq j$. With the cell partitions, define the points from point cloud P whose projection falls within cell l_j as

$$\gamma_j = \{p \in P : (p^x, p^y) \in l_j\}. \quad (1)$$



(a) Ground segmentation (b) Drivability segmentation

Figure 3. Point cloud segmentation results using the Gaussian process method, which includes drivable obstacle points (green), nondrivable obstacle points (red), and ground points (blue). (a) Ground segmentation only. (b) Ground and drivability segmentation.

To classify the points within a cell, a collection of nearest-neighbor cells is determined and evaluated. For the local map, $\chi_i \in \mathbb{R}^2$ and $\chi_j \in \mathbb{R}^2$ denote the geometric mean, or the centroid of the cells l_i and l_j, respectively. Define the distance between two cells l_i and l_j, using the distance function $d : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}$ between their respective centroids, as

$$d(\chi_i, \chi_j) = \left|\left|\chi_i - \chi_j\right|\right|_q . \tag{2}$$

When $q = 2$, the function d is the Euclidean distance between the centroids of the cells, however a distance based on the Manhattan norm, $q = 1$, or infinity norm, $q = \infty$, can also be used. For implementation, the infinity norm is used in order to extract a square subgrid of cells as the nearest neighbors. Using the definition for the distance between grid cells given in Eq. (2), the nearest neighbors, $\Phi \subseteq \Lambda$, for a cell ℓ_i can be given as

$$\Phi = \{\ell_j \in \Lambda : d(\chi_i, \chi_j) < \delta_\Lambda\} \backslash \ell_i, \tag{3}$$

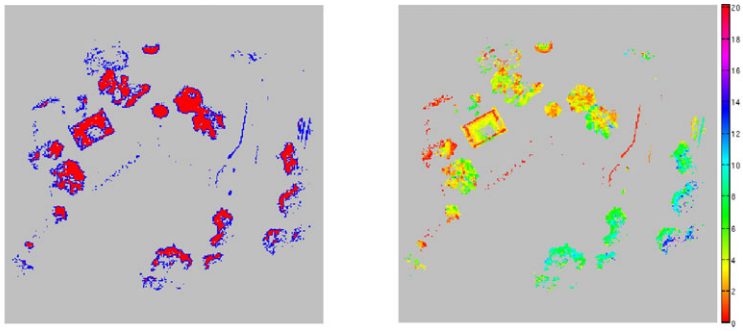
where δ_Λ is a user-defined threshold that determines the size of the neighborhood of l_j to consider. To classify the points within a cell as ground-adjacent or non-ground-adjacent, the nearest-neighbor cells for a target cell are evaluated. A counter variable I_Λ is used to count the number of ground-adjacent cells in the neighborhood. If at least β_Λ of the cells are ground-cells (i.e. they contain no points), then the points in the target cell are classified as ground-adjacent.

2.4 Registration to Global Frame

The goal of the global map is to maintain the global representation of the environment through the aggregation of point cloud data as the vehicle traverses the environment. The global map for the vehicle is generated from a 2D, top down perspective of the environment. Similar to the local map, the global map is represented by a collection of cells $\Theta = \{g_i \ldots g_{N_\Theta}\}$. A cell, g_i, consists of the mean value of the x-y projection of the points aggregated within the cell, or $\mu_i^{xy} \in \mathbb{R}^2$. The cell also maintains a distribution of the height information aggregated within the cell. Denote $\mu_i^z \in \mathbb{R}$ as the height mean, or mean of the z-components of the points aggregated within the cell. For ground points, the height value is set to zero. Finally, the standard deviation of the height points is maintained within the cell, and is denoted by $\sigma_i^z \in \mathbb{R}$.

To register the local map to the global map using CC-ICP, the classification of ground-adjacent and non-ground-adjacent points is performed for the cells in the global map in an analogous manner to the point classification in the local map. The nearest neighbors are identified through the inspection of the cells within a neighborhood of size δ_Θ of a target cell. A counter, I_Θ, is used to count the number of nearest-neighbor cells that are ground cells, where $\mu_i^z = 0$. If at least β_Θ of the cells are ground-cells, then the mean point of the target cell, μ_i^{xy}, is classified as ground-adjacent. Using ground adjacency classification of the global map, Θ, the model point clouds for the ground-adjacent points, P_{GA}^M, and the model point cloud for non-ground-adjacent points, P_{NGA}^M, can be generated.

In SPC-SLAM, the classified obstacle points from the local map are registered into the global frame using a CC-ICP scan registration technique. The CC-ICP scan registration algorithm seeks to find the parameters $T = [t_x, t_y, t_\theta] \in \mathbb{SE}(2)$, such that the Euclidean distance between corresponding points of a model point cloud and a scene point cloud which has been transformed by T is minimized. The ICP method treats nearest-neighbor points as correspondences for each iteration of the minimization. For CC-ICP, the points in the classified scene sets P_{GA}^S and P_{NGA}^S may only correspond with the classified points in the model set P_{GA}^M and P_{NGA}^M, respectively. To further improve correspondences, the height values maintained in the cells of the global map are used to reject correspondences from the local map that do not agree with the modeled height distributions from the global map. If the z-component of a point from an obstacle point cloud



(a) Ground adjacency classification for the global map (b) Mean cell height for the global map

Figure 4. Classification for the global map. (a) Classification of ground-adjacent (blue) versus non-ground-adjacent (red) cells. (b) Visualization of the cell mean height values (in meters).

that has been transformed by parameter estimate T is not within one standard deviation of the height distribution for the cell in which it is located in the global map, the point correspondence is rejected. The strategy of using constrained nearest-neighbor correspondences with height-based outlier rejection results in the contributions for the CC-ICP minimization coming from the most likely point-pair correspondences. Figure 4 illustrates an example of the ground adjacency classification for the global map, and provides a visualization of the aggregated height distributions within each cell.

2.5 Global Map Integration

To define when to include a new key frame in the global map, a map confidence score is proposed. Denote the point cloud $\bar{P}$ as the 3D point cloud that has been transformed into the global frame, using the transform parameters T obtained through CC-ICP registration of the 2D x-y point components. Denote the Gaussian height distribution of a cell g_i in the global map as $\mathcal{N}_i^{\Theta}(\mu_i^z, (\sigma_i^z)^2)$, and the height distribution for the set of transformed scene points, γ_i, contained within cell g_i as $\mathcal{N}_i^{\gamma}(\bar{\mu}_i^z, (\bar{\sigma}_i^z)^2)$. Using the height distribution contained within a cell and the height distribution generated using the transformed points, a score can be defined that quantifies the divergence between the two Gaussian distributions,

$$\zeta_i = \exp\left(-\frac{1}{2}\frac{(\mu_i^z - \bar{\mu}_i^z)^2}{(\sigma_i^z)^2 + (\bar{\sigma}_i^z)^2}\right). \tag{4}$$

Denote the points of $\bar{P}$ that are located with a cell of the global map, g_i, as γ_i. For each point in the set γ_i, a residual distance to the mean point value associated with the global map cell g_i can be calculated. Denote the total residual score for the cell κ_i as

$$\kappa_i = \begin{cases} 0 & \text{if } \mu_i^z = 0, \\ \sum\limits_{p \in \gamma_i} \| p^{xy} - \mu_i^{xy} \| & \text{otherwise,} \end{cases} \tag{5}$$

where $p^{xy} \in \mathbb{R}^2$ denotes a vector consisting of the x and y components of the point p. Note that a score of zero is given if the cell is a ground cell, as this implies there are no obstacle points to obtain a CC-ICP residual from. Finally, denote N_o as the total number of nonground cells in the global map that contain points from the transformed point cloud $\bar{P}$, and denote N_n as the total number of ground cells in the global map that contain points from the transformed point cloud. Note that N_n is the number of cells where there is a discrepancy in the global map, and the transformed point cloud suggests that the cells should be updated as obstacles. The larger N_n becomes, the less confidence there is in the global map. To integrate new information into the map, two conditions must exist. The *map uncertainty condition* is given as

$$\frac{N_n}{N_n + N_o} > \epsilon_n, \tag{6}$$

where ϵ_n is a user-defined parameter that controls the amount of discrepancy required before considering integrating new information into the map. To ensure that the scan registration converged to a correct solution, a score is computed based on the CC-ICP residuals and the difference in height distributions for each cell. The *registration uncertainty condition* is given as

$$\sum_{i=0}^{N_o} \left[\frac{\kappa_i}{\delta_d N_o} + \left(1 - \frac{\zeta_i}{N_o}\right)\right] < \epsilon_o, \tag{7}$$

where δ_d is the diagonal distance for a cell in the global map, and ϵ_o is a user-defined parameter that controls the required quality of registration required in order to integrate information into the global map. If the map uncertainty and

registration uncertainty conditions are satisfied, the information from the transformed point cloud $\bar{P}$ is integrated into the global map.

Intuitively, the map uncertainty condition defines a ratio of the number of cells in the global map where new information is suggested to be added by $\bar{P}$, to the total of cells which are occupied by $\bar{P}$. In the case in which sections of the environment are revisited, N_n will tend toward zero, as the map within the sensor range has been previously explored. The registration uncertainty condition is computed using the sum of two scores based on the CC-ICP residual normalized using the diagonal cell dimension, and the normalized discrepancy in height between the cells in the global map and the transformed point cloud $\bar{P}$. The map and registration uncertainty conditions are suitable for SPC-SLAM because they provide a robust method to control key frame insertion based on the quality of the scan registration and they are computationally inexpensive to compute.

The integration process simply consists of updating the means and standard deviations associated with each cell in the global map, g_i, using the points γ_i. To further improve robustness, cells are not updated until a required log-odds ratio of occupancy is achieved. The occupancy update is the standard Bayesian occupancy grid map update, and the method used for this work is described in detail in Thrun et al. (2005).

2.6 Graph SLAM and Global Consistency

To maintain a globally consistent map, a graph SLAM framework is used. Denote a vertex in the pose graph as $v \in \mathbb{SE}(2)$, and the set of all vertices, V_p, where each vertex defines a 2D vehicle pose at a particular time. Denote an edge between two vertices, v_i and v_j, as $e_{ij} \in V_p \times V_p$, and the full pose graph as $G_p(V_p, E_p)$. For the graph SLAM problem, each edge represents a constraint between its vertices. In SPC-SLAM, each constraint on edge e_{ij} is imposed by performing a 2D CC-ICP scan registration between the point clouds associated with the vehicle poses at vertex v_i and vertex v_j. When edges are added such that their configuration results in an overconstrained pose graph, a graph relaxation optimization is performed. Intuitively, the optimization can be viewed as finding the values for the vertices such that the sum of errors imposed by each edge constraint is minimized.

For SPC-SLAM, the graph is initialized with a fixed vertex at the vehicle's start position, which establishes the SLAM coordinate frame. A new vertex is added to the graph when new information is added into the global map, as described in Section 2.5. To ensure the vehicle does not travel too far without adding a vertex, a vertex is forced to be added if it has traveled a distance of $\delta_{\max}$ since the last added vertex. The forcing of a vertex is done to ensure that there is adequate overlap in the point clouds to perform a CC-ICP registration when constructing an edge. To keep the number of vertices in the graph manageable, a vertex is not added if there is another vertex that is within a distance of $\delta_{\min}$. Each time a vertex is added to the graph, edges are added by performing CC-ICP scan registration to the vertices which are the K-nearest neighbors to the current vehicle position. Once the edge constraints are added, the full graph optimization is performed using the g2o backend. Finally, using the optimized pose graph, the global map is regenerated using the point clouds associated with each vertex in the updated graph.

2.7 Drivability Map Extraction

To obtain a drivability map for the path planner, an additional global map is maintained. Every time new information is added into the global map, the drivability map is also updated using only the points classified as nondrivable, as described in Section 2.2. The map update takes place when new information is integrated into the map due to map uncertainty and also when the map is regenerated after a pose graph optimization. An example of the drivability map is presented in Figure 5. The black-colored cells represent the nondrivable obstacle information. The light gray cells represent overhanging features such as trees and archways that the vehicle can safely traverse under. An aerial photo of the mapped area is also provided for reference.

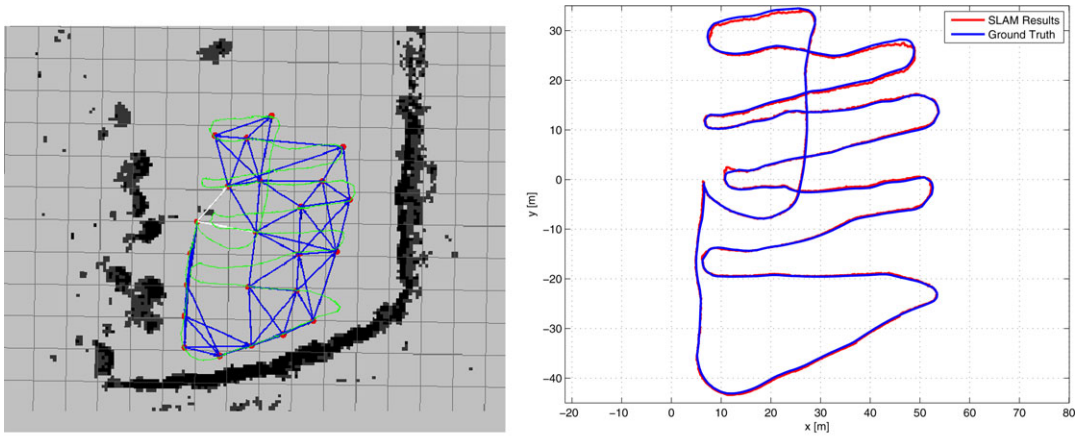
2.8 Mapping Results

An experiment to validate the SPC-SLAM approach is carried out in Waterloo Park, which is adjacent to the University of Waterloo campus. The test location is a field that measures approximately 60 m by 60 m, is mainly sparse, but contains trees and shrubs along the outer perimeter. The experiment consists of a test case where the vehicle drives a sweep path, which is typical of the operating conditions, as generally sweep paths are required to ensure coverage. The experiment is performed in real-time using C++/ROS onboard the vehicle. For the experiment, the global map cell size is selected to be 0.5 m, the map uncertainty score threshold is set to $\epsilon_n = 0.3$, and the registration uncertainty is selected as $\epsilon_o = 0.1$. The distance required to force the addition of a key frame is set to $\delta_{\max} = 20$ m and the minimum distance between key frames is set to $\delta_{\min} = 10$ m. The number of nearest-neighbor edges to connect upon a vertex insertion is set to three. The vehicle maintains a constant velocity of 1 m/s throughout the duration of the experiment. The accuracy of the vehicle path is validated using a Trimble S3 robotic total station, which is capable of collecting measurements to a tracking prism at a rate of approximately 1 Hz with millimeter-level accuracy.

Figure 6 presents the map results of the vehicle traversing the sweep path in the open field and returning to its starting position. The high localization accuracy of SPC-SLAM is illustrated by the strong overlap between the ground truth

(a) Aerial photo of test area (b) Driveabilty map

Figure 5. An example of a global drivability map. (a) Ariel photo of the mapped area. (b) Global map with nondrivable cells colored in black, and traversable obstacles that the vehicle can pass under colored in gray.



(a) (b)

Figure 6. Generated map and ground truth results for the sparse field experiment. Red denotes graph SLAM vertices, blue lines denote graph edges, green denotes the instantaneous vehicle path, and black denotes the map. (a) Generated map and pose graph. (b) Resulting vehicle trajectory overlaid with the vehicle ground truth determined using a robotic total station.

results and the SLAM trajectory depicted in Figure 6(b). The mean error over the entire run is calculated to be 0.274 m and the maximum error is determined to be 1.264 m, demonstrating that the SPC-SLAM approach provides accurate localization in sparse, outdoor environments. Finally, the final loop error, or the error between the SLAM solution and the ground truth once the vehicle returned to the starting point, is 0.224 m, which illustrates that SPC-SLAM is capable of providing a viable return-to-home solution.

For the presented experiment, the average execution time for the local mapping node is 27.9 milliseconds per iteration, the average execution time for the global mapping node is 48.3 milliseconds per iteration, and the average run time for the graph optimization node is 6.3 milliseconds per iteration. In comparison, performing 3D scan registration between two point clouds generated by the Velodyne scanner using methods such as G-ICP or NDT requires approximately 2–3 s per iteration. The increased computation is especially problematic when generating edges for the pose graph, as multiple scan registrations are performed in succession. It is clear that the SPC-SLAM algorithm is able to achieve real-time performance, especially when taking into consideration that the average point cloud generation frequency for the Velodyne HDL-32E laser scanner is approximately 10 Hz, or 100 milliseconds per scan.

The performed field test validates that the SPC-SLAM algorithm sufficiently meets the requirements to operate in a typical rover mission. The approach is able to generate globally consistent maps for path planning and can localize the vehicle in 2D with sufficient accuracy for return-to-base

capability. All computation is able to take place online, in real-time, using only the hardware onboard the vehicle.

3 SAMPLE DETECTION

In this section, we present our approaches and implementation of visual sample detection on an autonomous rover. Our major design goal is the development of reliable object detection for cameras monitoring the close and far surroundings of a mobile robot. A maximized detection coverage can be achieved by increasing the field of view of the cameras, i.e., the sensor footprint. However, increasing the sensor footprint results in a loss of resolution since objects would occupy fewer pixels in the image plane, increasing the difficulty of their localization and classification. This is particularly critical for objects being far away from the robot, and may lead to poor detection performance. On the other hand, increasing the image resolution leads to an increased processing time and may result in poor object detection with rapid robot movements. Hence, a light weight object detection approach is of great interest, one that can process images with up to 15 frames per second to handle rapid robot movements and can cope with low-resolution detections in order to maximize sensor footprint.

Most common approaches for visual object detection rely on scanning high-resolution images with a sliding window and matching to *a priori* known feature descriptors, both in controlled indoor and outdoor scenarios. Ekvall *et al.* (Ekvall, Kragic, and Jensfelt, 2007; Sjoe, Lopez, Chandana, Jensfelt, & Kragic, 2009) proposed a robot system that autonomously detects predefined objects in domestic indoor environments using single cameras. Other approaches take advantage of high-resolution, multicamera pan-tilt-zoom solutions for indoor object detection (Coates and Ng, 2010) and detect objects based on sliding-window approaches. In addition, a boosted decision tree classifier trained on a dictionary of small patches and HoG-features (Dalal and Triggs, 2005). All these approaches benefit from well controlled illumination conditions in indoor scenarios. In outdoor scenarios, however, the appearance of objects in terms of color or albedo strongly depends on uncontrollable illumination conditions that cannot be properly trained for in advance for the classification stage.

Recent autonomous rover work in the field of outdoor sample detection has focused on rock field and individual rock detection. Examples include the finding, characterization, and surveying of several rock types under variable lighting conditions, such as the APIC (Pugh, Tyler, and Barnes, 2010; Pugh, Tyler, Barnes, Labrosse, and Neal, 2011) and the OASIS (Castaño et al., 2007; Castaño et al., 2008; Estlin et al., 2012) systems. These works mainly focus on taking high-resolution close-up pictures of rocks and follow a common detection scheme: (i) object segmentation from the ground, (ii) novelty detection and feature extraction from the targeted objects, and (iii) map generation for path-planning tasks based on the detected object locations. In addition, the rover of the "Life in the Atacama project" (Thompson and Wettergreen, 2005) uses multiple cameras and an Expectation Maximum-based approach to detect interesting objects. All of these approaches benefit from high-resolution images and close-up views, which are not always available.

While the indoor object detection approaches take advantage of *a priori* known backgrounds, training the background in unknown environments is more challenging. Analog environments are used to simulate and train potential backgrounds, but there is no guarantee that they represent the environment properly. Hence, running object detection algorithms on images taken from objects located in unknown and untrained environments might confuse classifiers and cause false-positive detection rates. In addition, previous works benefit from multiple views [e.g., Thompson and Wettergreen (2005)], or rely on high-resolution images that facilitate unique feature extraction and descriptor generation. However, advanced feature extraction methods relying on SIFT, SURF, FAST, or CENSURE features followed by bag-of-word approaches for robust classification are no longer feasible in low-resolution images or when objects are far away from the robot. Also, object detection based on sliding window approaches in high-resolution images results in a wide field of view for rapid terrain coverage, but it also results in a computational burden too large for real-time object detection and classification on mobile robots.

To address these challenges and to make object detection more efficient, saliency maps have been introduced to determine potential regions of interest in natural images. By inhibiting regions containing background, saliency approaches can guide object detection toward regions of interest. They act in a purely bottom-up manner without considering information about target objects at all [see, e.g., Cheng, Zhang, Mitra, Huang, and Hu (2011); Itti, Koch, and Niebur (1998); Jain, Wong, and Fieguth (2012); Rutishauser, Walther, Koch, & Perona (2004); Walther and Koch (2006)], or use features of target objects (Mitri, Frintrop, Pervolz, Surmann, & Nuchter, 2005; Oliva, Torralba, Castelhano, and Henderson, 2003; S. et al., 2005) or scene context (Im and Cho, 2006; Torralba and Sinha, 2001) for saliency computation. Saliency maps have been successfully applied on mobile robots to localize and classify objects more efficiently (Mitri et al., 2005; Rudinac and Jonker, 2010; Sjoe et al., 2009; Xu et al., 2009b; Yu, Mann, and Gosine, 2009).

Finally, robots performing search tasks in large areas rarely see objects of interest. However, none of the aforementioned approaches has addressed or solved the more general issue of whether there are interesting objects in input images at all. Moreover, most saliency approaches tend to highlight regions with irrelevant information such as background or shadows in images containing

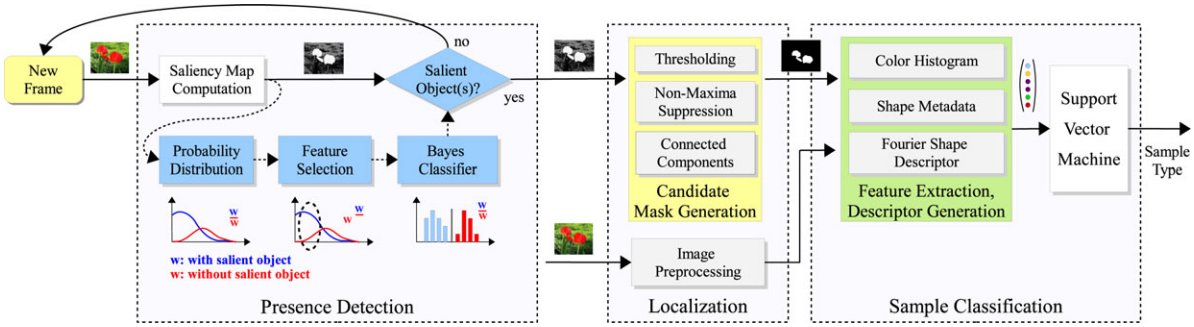


Figure 7. Overall three-stage classifier cascade framework for presence-dependent sample detection and classification in low-resolution images.

background only. Recent object detection approaches avoid the extraction of the background by segmenting saliency maps using thresholds [see, e.g., Xu et al. (2009b); Xu, Chenkov, Kuhnlenz, and Buss (2009a)], and presume that background leads to low saliency values and interesting objects to large saliency values. However, finding appropriate thresholds for suppressing background images is difficult since strongly cluttered background or shadows may lead to strong saliency values (see Figure 9). Hence, an approach that quickly prunes background images while maintaining a low false negative rate would avoid the execution of expensive object detection on pure background images.

3.1 Contributions and System Architecture

There are several major barriers in the path to real-time sample and object detection in ground vehicles. We make contributions to three of them here. First, computational complexity in performing object detection is lowered by the introduction of saliency-based presence detection that quickly prunes uninteresting background images and enables the robot to focus on images that are crucial to the task. Second, we address the challenge of robustly localizing and segmenting objects using very small, noisy pixel patches that are typical of the imagery acquired in outdoor scenarios with uncontrolled illumination conditions and large detection ranges. Due to limited computational resources, we also propose the use of simple yet robust features for object detection together with an image-processing pipeline that makes limited use of powerful but expensive algorithms such as bilateral filtering. Third, the introduction of a more domain-invariant technique, the saliency-based presence detection, substantially reduces the difficulty of adapting to untrained backgrounds that commonly confuse classifiers and cause high false positive detection rates.

This section introduces a novel framework to robust object detection and sample classification in low-resolution images captured in outdoor scenarios under uncontrolled lighting conditions. The underlying goal of the proposed scheme is to take advantage of the statistical characteristics of saliency values in an efficient manner for sample detection on mobile robots. The overall architecture of the proposed approach can be broken down into three main stages (see Figure 7): (i) probability-based presence detection of interesting objects in input images, (ii) object localization to generate masks of candidate objects within a frame, and (iii) sample classification based on a support vector machine using color histogram and shape metadata.

The first layer—*Presence Detection*—uses histogram features extracted from saliency maps to judge the presence of interesting objects in images. This is a newly introduced concept that effectively removes geometrical information such as object location in saliency maps and overcomes the limitations of approaches extracting interesting objects based on thresholds. It can also be viewed as a *domain adaptation* scheme that maintains the overall object classification performance, which might degrade abruptly in novel and untrained environments.

The second layer—*Localization*—localizes and segments objects in very small and noisy pixel patches by applying thresholding on saliency maps, followed by non-maxima suppression and connected labeling stages. The extraction of objects in saliency maps together with advanced image-enhancement techniques such bilateral filtering makes the approach feasible for a wide range of outdoor scenarios with unknown backgrounds. By using the F1-measure to select appropriate thresholds for segmenting candidate objects, we address the challenge of reducing the number of both false positive and false negative detections at the same time. The second layer outputs a binary decision and a bounding box for potential candidate objects.

The third and final stage, *Classification*, is very fast and uses a Support Vector Machine along with a Histogram Intersection Kernel to assign a k-class label (where k is the number of classes) and probability output to each candidate object. A careful selection of color and shape features makes

Figure 8. Samples to be retrieved in NASA SRRC. From left to right, these are referenced in the text as "Hook", "Ball", "Tube", "Stone", and "Wood".

Table I. Competition samples from the NASA Sample Return Robotic Challenge 2012. Potential confusers are highlighted.

Sample	Color	Potential False Positives
Hook (cylinder with hook)	White	Flowers, glare reflected from other surfaces
Tennis Ball	Pink	Tube, flowers
Tube (PVC pipe)	Fluorescent orange	Flowers
Stone/Rock	Yellow	Sunlight; Flowers; Sand; Dry grass
Wooden Cube	Brown	Trees, Ground, Grass, Buildings

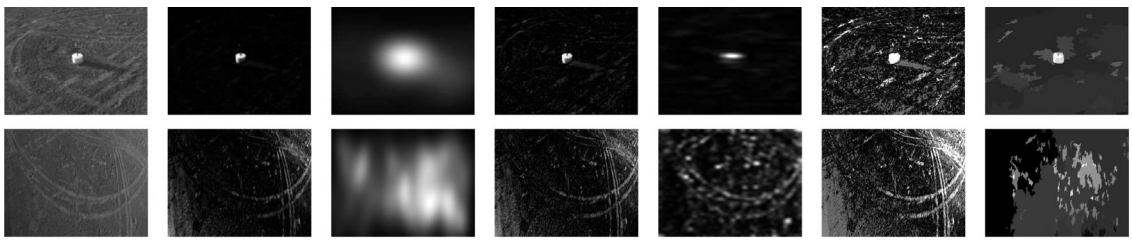
this approach feasible for detecting very small objects containing 200 pixels only, and it also overcomes issues with color shifts caused by uncontrolled and changing illumination conditions. The proposed architecture follows a general design principle of using a cascade of increasingly more expensive operations that run on successively fewer pixels or subregions. A detailed description of each stage is provided in the following sections.

The development of our work is evaluated and tested within the NASA Sample Return Robot Challenge where ten objects drawn from five object classes are randomly scattered throughout a large 80,000 m^2 outdoor environment. For test purposes, we have chosen five representative samples to train and test on, which are shown in Figure 8 and described further in Table I.

3.2 Methodology

3.2.1 Presence Detection

In this section, we introduce a novel approach that detects the presence of interesting objects in images. A histogram of saliency values (PDF) from state-of-the art saliency maps is constructed to effectively remove geometrical information. By exhibiting the largest variance across training samples using PCA, the probabilities of the saliency values that best discriminate between object and background images are stacked into a feature vector. A binary classification approach is then applied to robustly predict the existence of interesting objects in images. Finally, feature vectors extracted from saliency maps of new input images are fed into a trained classifier to quickly discard background images and to better enable a robot to focus on object images. Since various state-of-the-art saliency approaches produce saliency maps of different quality, we discuss and evaluate our presence detection scheme based on six state-of-the-art saliency approaches. These approaches are frequency-tuned [FT (Achanta, Hemami, Estrada, and Susstrunk, 2009)], visual attention [IT (Itti et al., 1998)], histogram contrast and global contrast [HC, RC (Cheng et al., 2011)], luminance contrast [LC (Zhai and Shah, 2006)], and spectral residual [SR (Hou and Zhang, 2007)]. These approaches are selected based on the criteria (i) low computational complexity [FT (Achanta et al., 2009)], (ii) spectral, contrast, and region-based saliency determination [SR (Hou and Zhang, 2007), LC (Zhai and Shah, 2006), HC/RC (Cheng et al., 2011)], and (iii) frequent use in robotics [IT (Itti et al., 1998)]. Figure 9 shows example saliency maps for object and background images.



(a) Source (b) FT (c) IT (d) LC (e) SR (f) HC (g) RC

Figure 9. Example object and background images obtained from the cameras attached to our robot, along with the saliency maps computed from six state-of-the-art saliency approaches, which are frequency tuned [FT (Achanta et al., 2009)], visual attention [IT (Itti et al., 1998)], luminance contrast [LC (Zhai and Shah, 2006)], spectral residual [SR (Hou and Zhang, 2007)], and histogram contrast and global contrast [HC, RC (Cheng et al., 2011)].

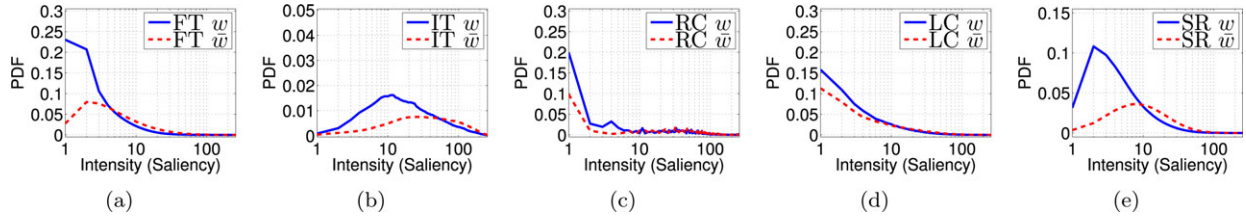


Figure 10. Probability distribution functions pdf of salient values from saliency maps of five saliency approaches which are FT, IT, RC, LC and SR. We computed the saliency maps for both images with (w) and without interesting object ($\bar{w}$). We chose logarithmic axes and a range of the salient values between [1, ..., 256] for illustration purposes (The area under each curve is 1.0).

Saliency Histogram Feature Extraction

A careful examination of the saliency histograms obtained from several saliency approaches (see Figure 10) shows that the PDFs are suitable to identify images containing interesting objects or background only. Let $i, i \in \{w, \bar{w}\}$ be two types of input images, where w denotes images with, and $\bar{w}$ images without, interesting objects, and Ω_i is the corresponding image class. A normalized saliency map $\mathbf{S}$ is a $u \times v$ image scaled from $[0, 1]$, using $m = 256$ discrete saliency values. Then, we propose a classifier that uses the probabilities $p(s = j|\Omega_i)$, $j \in \{1, \ldots, m\}$ of saliency values s_j as the entries of an m-dimensional feature vector $\mathbf{X}_i = [x_{i,1}, \ldots, x_{i,m}]^t$ $\mathbf{X}_i \in \mathbb{R}^{m \times 1}$, such that

$$\mathbf{X}_i = \text{PDF}(\mathbf{S}|\Omega_i) \text{ with } x_{i,j} = p(s = j|\Omega_i) \tag{8}$$

with PDF$(\mathbf{S}|\Omega_i)$ the probability distribution function of $\mathbf{S}$, and $x_{i,j}$ the jth entry of the feature vector $\mathbf{X}_i$. The saliency histograms PDF$(\mathbf{S}|\Omega_i)$ of several saliency approaches show that a limited number of saliency values s_j and their probabilities provide useful information to identify the presence of objects in images. As an example for FT-based saliency maps, Figure 10(a) shows that the probabilities $p(s=1|\Omega_i)$, $p(s=2|\Omega_i)$, and $p(s=3|\Omega_i)$ of the saliency values 1, 2, and 3 might be suitable for robust classification, whereas the probabilities of other saliency values might lead to poor classification results. To automatically select the saliency values and their occurrence probabilities that best discriminate between object and background images, we first build a feature matrix $\mathbf{M}$, $\mathbf{M} \in \mathbb{R}^{i \cdot N \times m}$,

$$\mathbf{M} = [\mathbf{X}_{w,1}, \ldots, \mathbf{X}_{w,N}, \mathbf{X}_{\bar{w},1}, \ldots, \mathbf{X}_{\bar{w},N}], \tag{9}$$

where N is the total number of training samples (training vectors), $i \in \{w, \bar{w}\}$ the two different image types, and m the dimension of a feature vector $\mathbf{X}$. Applying PCA (Pearson, 1901) to $\mathbf{M}$ provides the projection matrix $\mathbf{P}_\mathbf{M} = [\mathbf{E}_1, \ldots, \mathbf{E}_m]$, $\mathbf{P}_\mathbf{M} \in \mathbb{R}^{m \times m}$, containing the m eigenvectors (principal components) $\mathbf{E}$, $\mathbf{E} \in \mathbb{R}^{m \times 1}$ associated with the corresponding m eigenvalues λ. To obtain the saliency values with the highest variance across the training data, we determine d, $d << m$ principal components by selecting their corresponding eigenvalues $\lambda_u, u \in [1, \ldots, m]$ that represent 95% of the variance of the data. Using the projection matrix $\mathbf{P}_\mathbf{d} = [\mathbf{E}_1, \ldots, \mathbf{E}_d]$, $\mathbf{P}_\mathbf{d} \in \mathbb{R}^{m \times d}$, we then transform every feature vector $\mathbf{X}$ to $\mathbf{Z}$ as follows:

$$\mathbf{Z} = \mathbf{P_d}^T \cdot \mathbf{X}. \tag{10}$$

Table II. Average and standard deviation of the number of selected features (out of 256) after applying PCA on saliency maps obtained from six state-of-the-art saliency approaches.

Saliency Approach	FT	IT	HC	RC	LC	SR
No. Components, mean	3	9	90	72	29	3
No. Components, std	0	1	4	2	2	0

Table II presents the number of principal components after applying PCA to the feature vectors obtained from saliency maps of six state-of-the-art saliency approaches. We used 10-fold cross-validation to determine the average and the standard deviation of the number of principal components.

Classification

After applying PCA to reduce the dimensionality of the feature space, we propose to train and use a Bayes classifier to classify an input image either as an object image (w) or a background image ($\bar{w}$). Therefore, we model the probability distribution $p(\mathbf{Z_o}|\Omega_i)$ of observations $\mathbf{Z_o}$ (training and test data) as multivariate normal, i.e., $p(\mathbf{Z}|\Omega_i) \sim \mathcal{N}(\mu_i, \Sigma_i)$ (Duda, Hart, and Stork, 2000):

$$p(\mathbf{Z_o}|\Omega_{w_i}) = \frac{1}{(2\pi)^{(d/2)}\det(\Sigma)^{(1/2)}} \cdot \exp\left[-\frac{1}{2}(\mathbf{Z_o} - \mu)^T \Sigma^{-1}(\mathbf{Z_o} - \mu)\right] \tag{11}$$

with μ a d-component mean vector, and Σ the corresponding d-by-d covariance matrix. To distinguish between the two classes Ω_i, $i = \{w, \bar{w}\}$, we use the minimum-error-rate discriminant function, which can be written as follows (Duda et al., 2000):

$$g(\mathbf{Z_o}) = \ln\frac{p(\mathbf{Z_o}|\Omega_w)}{p(\mathbf{Z_o}|\Omega_{\bar{w}})} + \ln\frac{P(\Omega_w)}{P(\Omega_{\bar{w}})} \tag{12}$$

with the prior probabilities $P(\Omega_w)$ and $P(\Omega_{\bar{w}})$, and we derive the final discriminant function $g_i(\mathbf{Z_o})$ for a class Ω_i as

follows:

$$g_i(\mathbf{Z_o}) = -\frac{1}{2}(\mathbf{Z_o} - \mu_i)^T \Sigma_i^{-1}(\mathbf{Z_o} - \mu_i) - \frac{d}{2}\ln(2\pi) - \frac{d}{2}\ln\det(\Sigma_i) + \ln P(\Omega_i). \quad (13)$$

Using PCA and the Bayes classifier, we can robustly identify the presence of interesting objects in input images based on saliency maps. As later discussed in Section 3.3, we obtained an F1-score of 92.0% for the FT saliency maps, and an F1-score of 82.1% for saliency maps obtained from the IT saliency approach. We used 10-fold cross-validation for testing and equal prior probabilities $P(\Omega_w) = P(\Omega_{\bar{w}}) = 0.5$.

3.2.2 Localization

Once the Presence Detection module has identified candidate images, the Localization module uses a chain of image-processing techniques to extract complementary image features such as connected component (blob) information. The module outputs object centroids with bounding boxes as its final output, which is used as a mask for feature extraction by subsequent stages. First, the saliency map S is binarized into S_b using the threshold $\Delta = \nu \cdot S_{\max}$, $\nu \in (0, 1]$, with $S_{\max}$ the maximum saliency value. To identify a threshold that reduces the number of false positives and avoids the miss of valid objects, the F1 measure has been chosen to determine an appropriate parameter ν. Based on the F1 metric, $\nu = 0.86$ has been determined as an optimal choice (see Section 3.3.2, Figure 12). Nonmaximum suppression is then run on binarized saliency maps to further improve region extraction, followed by connected components detection and blob area and bounding box determination. This results in a series of candidate subregions that are used to mask regions containing potential objects. We also remove blobs with a pixel area below A_{low} or bounding box with dimensions larger than $B_{X\max}$, $B_{Y\max}$. We then transform the masked image regions into the YCbCr domain and perform bilateral filtering (Tomasi and Manduchi, 1998) and sharpening on each color channel. To better obtain the object boundaries for mask refinement, we further apply Canny edge detection separately on the Y, Cb, Cr channels, and combine the edges using logical OR, yielding a subregion mask. Finally, blob finding is performed on each subregion and blob properties are computed. The remaining blobs are all reported to the next stage, which performs classification based on features of the subregions.

3.2.3 Feature Extraction and Classification

As a next step, a classification stage is used to identify interesting objects with high confidence and to prune background images or task-irrelevant yet visually salient objects, identified by Presence Detection and Localization. For classification, color and shape features are chosen to overcome difficulties such as small bounding box size of objects, occluded objects, high levels of lighting, distance, and pose variability. Color and shape feature provide a mixture of computationally inexpensive, dimensionless, and discriminative features, and include separate Cr and Cb channel color histograms, seven non-color-related shape metadata features such as the ratio of area over perimeter, solidity, eccentricity of an ellipse that is fit to the blob, albedo, perimeter, major axis length, and extent. It should be noted that it is possible to compute local color histograms and directly search for objects with the target color distributions, bypassing the entire saliency, presence detection, and localization steps. However, multiple object classes may have colors that may be seen in the natural environment. In addition, lighting and glare in the outdoor environment can drastically change the sensed color. This motivates us to choose a visual saliency approach rather than color alone to prelocate objects that stand out from their background environment.

Once the features have been extracted, we feed them into a support vector machine (SVM) for final classification due to its excellent performance with high-dimensional problems, its fast prediction times, automatic tuning, its ability to support nonlinear similarity measures via kernels, and its continuous probability value outputs. Multiple one-vs-one classifiers are built to handle the multiclass classification. We select the Histogram Intersection Kernel K_{HIK} (Grauman and Darrell, 2005) [see Eq. (14)] for classification due to its good performance for both the color histograms of the Cb and Cr channels and the shape metadata.

$$K_{\text{HIK}}(E, F) = \sum_{j=1}^{r} \min\left(E_j, F_j\right). \quad (14)$$

In addition, we evaluated the performance of different kernels in Section 3.3.3. Finally, if we denote the (mis)labeling of a subregion R_X of a true class X instead as class Y and denote this by $C(R_X) = Y$, then we observe that the importance of a misclassification $C(R_{o_1}) = o_2$, one object class as another, is not as severe as $C(R_{o_1}) = \text{BG}$ (false negative), or $C(R_{\text{BG}}) = o_2$ (false positive), where BG denotes background.

3.3 Results

To investigate the potential of our presence detection-based approach for robustly detecting objects and classifying samples in input images, we extensively evaluated our combined detection and classification scheme on four main measures. The first and most important measure is the overall classifier precision, and is used for both evaluating our presence detection and classification schemes. High classification performance is necessary for defining a planning coverage sweep pitch on mobile robots to ensure that all explored areas have been properly searched for potential objects. Additionally, we define the detection and

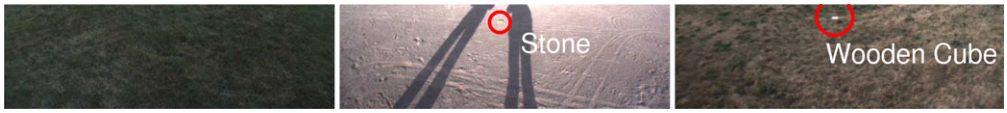


Figure 11. Our three test environments: Green Grass, Sand, and Patchy Brown Grass. Each cross-validation fold trains on two of these environments, and tests on the third.

localization performance based on the F1-score as our second measure. Unlike past works which seek to characterize and image rock fields and individual rocks, the figure of merit here is that we are able to localize the object sufficiently. For the final two measures, we quantify the savings in computational complexity and increases in Precision/Recall obtained by different steps in our object detection system. In general, Precision is the more important evaluation metric of the two since it represents the number of false-positive detections requiring further investigation by the robot.

We use two different training and testing datasets for evaluating the performance of our detection approach. For evaluating presence detection, we use 1,000 test images collected from our rover in different outdoor environments taken under different illumination conditions, and perform 10-fold cross-validation to obtain the mean detection performance and its standard deviation. To evaluate the entire framework, our training and testing dataset is a partial set of the test set used for presence detection, and contains 482 images collected from our rover in three environments (see Figure 11)—Green Grass, Sand, and Patchy Brown Grass, containing six classes of objects. We use the objects provided by the NASA sample return competition as illustrative examples—Hook (Hk.), Stone (Stn.), Tube (Tb.), Ball, Wood (Wd.), and background (NoObj.). We perform threefold *leave-one-environment-out* cross-validation by successively training on all images of two environments and testing on the third environment. This is a more challenging test environment that stresses the system's ability to adapt to scenes with very different textures, colors, and lighting, compared to regular cross-validation. This is also another important difference to existing test procedures since such measures are not regularly undertaken in similar object detection studies in robotics.

3.3.1 Performance Evaluation Presence Detection

In this section, we describe the experiments and discuss the results obtained for our presence detection approaches. To study the influence of different saliency maps on the performance of presence detection, we conducted experiments based on six saliency approaches, which are FT (Achanta et al., 2009), IT (Itti et al., 1998), HC and RC (Cheng et al., 2011), LC (Zhai and Shah, 2006), and SR (Hou and Zhang, 2007). Image noise does not influence the training and classification process since it can be suppressed by all these approaches to a certain degree.

We determined the classification performance of our presence detection approach. We used 10-fold cross-validation to determine the mean classification performance and the standard deviation, applied to thousands of test images containing both simple and complex background imagery. We captured these images using wide-angle cameras attached to our robot. Table III illustrates Precision, Recall, and F1-measures. It can be seen that the quality of the produced saliency maps strongly affects the classification performance. Our approach performs best on saliency maps from the frequency-tuned saliency approach with a mean F1-measure of $\approx 92.0\%$ and a standard deviation of 3.00%; the saliency maps obtained from the HC- and RC-based saliency approaches might be unsuitable for presence detection due to their low mean F1-measure scores and high standard variations. Table III also illustrates that the IT-based saliency approach that is commonly used in the field of robotics is also well suited for detecting the presence of interesting objects in input images. The poor classification rate for both the HC- and RC-based saliency maps is due to the fact that both HC- and RC-based approaches show some limitations when producing saliency maps of images containing textured background. Finally, Table IV illustrates the

Table III. Performance of presence detection. The performance has been evaluated for six state-of-the art saliency approaches, which are FT (Achanta et al., 2009), IT (Itti et al., 1998), HC and RC (Cheng et al., 2011), LC (Zhai and Shah, 2006), and SR (Hou and Zhang, 2007) based on 1,000 test images captured in outdoor scenarios.

	FT		IT		HC		RC		LC		SR	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
Precision	**0.91**	**0.09**	0.78	0.23	0.59	0.39	0.63	0.21	0.70	0.23	0.81	0.14
Recall	**0.92**	**0.02**	0.87	0.02	0.58	0.25	0.70	0.15	0.72	0.22	0.78	0.09
F1-measure	**0.92**	**0.03**	0.82	0.04	0.59	0.30	0.66	0.18	0.71	0.22	0.79	0.11

Table IV. Presence Detection performance on the six-class NSRRC dataset.

Estimated Class	True Class: Object	True Class: Background	Totals
Object	218	23	241
Background	13	228	241
Totals	231	251	482

presence detection performance applied to our second data set containing 482 test images. Based on the results obtained from the second data set for presence detection, we next compute the localization and classification performance of our overall object detection and classification approach.

3.3.2 Localization Results

The first step for localizing potential target objects is a binarization of the saliency maps using a static threshold $\tau_{\text{thresh}} = \nu \cdot S_{\max} \nu \in (0, 1]$. To find an appropriate value for parameter ν, we conducted experiments using ground-truth data from our data set as well as Precision, Recall, and F1 measures. Figure 12 illustrates the Precision, Recall, and F1 curves resulting from varying parameter $\nu \in (0, 1]$, and demonstrates that introducing Presence Detection is a tradeoff involving an increase in Precision by several percentage points ($\approx$ 2–5%) over a wide operating band, at a small ($\approx$ 1–2%) cost in Recall, and an almost unchanged F1 ratio. For object detection in general, it is important to obtain high Precision to reduce the number of false positives, but also to obtain high Recall to avoid objects not being detected at all. Hence, we set parameter ν to 0.86 according to the highest F1 value of the F1-measure graph of Figure 12. Experiments show that objects were never observed with saliency values smaller than $0.3 \cdot S_{\max}$. Hence, we also ignore all saliency values smaller than $0.3 \cdot S_{\max}$. For the nonmaximum suppression and connected component labeling, we set the minimum blob area allowed, A_{low}, to $A_{\text{low}} = 5$ pixels to reduce the effects of noise, and we set the largest bounding box allowed for candidate blobs, $B_{X\max}, B_{Y\max}$, to $B_{X\max} = 12\%$ and $B_{Y\max} = 30\%$ of the image width and height, respectively.

Table V. Localization results on our six-class dataset, using threefold CV as described in the text. Localization is able to provide a low absolute false negative rate (the sum of the bolded entries), which is obtained by a bias toward higher false-positive rates.

	No PD		With PD	
Total input images		482		482
Input images input into Localization block		482		241
No. of objects correctly localized	222		210	
No. of objects incorrectly localized	0		0	
No. of false positives	149		18	
No. objects reported by Localization		371		228
True background images rejected		102		5
No. of objects missed by Localization		9		8
Total images processed by Localization		482		241

We then conducted experiments to determine the performance of the proposed localization approach. We define a correct localization as one where the reported object centroid of a target object region is at most within $\pm$20 pixels of the hand-annotated ground truth. Our experiments (see Table V) showed that the tradeoff between false positives and false negatives is correctly tilted toward generating more false positives; in the "With PD" case, 18 of 251 (7.2%) background images were detected as false positives, compared with 8 of 231 objects (3.5%) detected as false negatives. The final classification stage then

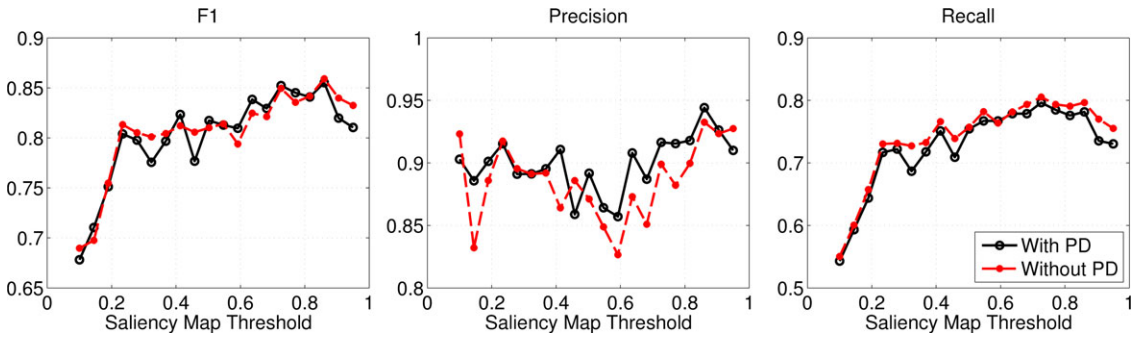


Figure 12. F1, Precision, Recall, with and without Presence Detection (PD). It can be seen that introducing Presence Detection is a tradeoff involving an increase in Precision by several percentage points ($\approx$ 2–5%) over a wide operating band, at a small ($\approx$ 1–2%) cost in Recall, and essentially unchanged F1 ratio.

Table VI. Overall results for the three-stage classifier using Presence Detection (PD), and features from color histogram (C) and shape metadata (SM). All results use SVM with the histogram intersection kernel.

	Prec.	Rec.	F1
PD, C + SM	**0.944**	0.782	**0.855**
No PD, C + SM	0.933	**0.796**	**0.859**
PD, C	0.888	0.775	0.827
No PD, C	0.816	0.790	0.802
PD, SM	0.620	0.611	0.615
No PD, SM	0.610	0.603	0.606

attempts to further filter false positives out by using a set of complementary features that Localization does not utilize.

3.3.3 Feature Extraction and Classification Results

We now present the experimental validation results for our design choices of features, kernels, fusion method, and classifier. Overall summary results are reported in Table VI using standard metrics in information retrieval—Precision, Recall, and F1 measure. The combination of Presence Detection (PD), using color (C) and shape metadata (SM) as introduced in Section 3.2.3, fused by simply concatenating the feature vectors (early fusion) and paired with the histogram intersection kernel (HIK), outperformed the other configurations tested in Table VI in Precision and tied for highest F1 values. Recall was about 1% lower than the "No PD" case, a good tradeoff for the computational savings, unnecessary rover movement, and time savings achieved. Unless otherwise stated, the subresults discussed in the following sections will be based on this configuration. A breakdown of precision scores per object class is shown in Figure 13. The two example classes most improved by the Presence Detection Stage, namely Stone and Wood, have the closest resemblances in terms of shape or color, respectively, to natural variations commonly found in the background environment.

Analysis of selected features. Several feature selection algorithms were run using the WEKA data mining tool (Hall et al., 2009) to select the features of the different shape metadata features with the highest impact to object classification. Removing weak features such as aspect ratio, pixel area, Fourier shape features, and minor axis length improved the classification performance by 3.0%, from 81.38% to 84.37%. However, we believe that more sophisticated shape extraction approaches, such as fitting candidate subregions to a 2D or 3D shape model, or curve completion techniques should be investigated.

Analysis of kernels and fusion methods. To find appropriate kernels for our application, we conducted extensive experiments to compare and evaluate the performance of three kernels, namely HIK kernel K_{HIK}, linear kernel K_{linear}, and RBF kernel K_{RBF} for classification (Grauman and Darrell, 2005). Experiments showed that the HIK has the highest classification performance using normalized shape metadata such as solidity and eccentricity, followed by the linear kernel K_{linear} and the RBF kernel K_{RBF}. For the color features, we found that the HIK kernel outperformed RBF, and also significantly outperformed the additive χ^2 kernel K_{χ^2}, which is commonly used for color histograms. In addition, normalization by double-centering the rows and columns was beneficial for the shape metadata, and the color histograms were normalized by their bin count. The nonhomogeneous nature of these features—a mixture of histogram data, continuous-valued shape metadata features, and basis coefficients—can affect the choice of similarity kernels, normalization methods, and classifiers. Hence, we also evaluated different strategies for feature fusion such as early, middle, and late feature fusion strategies. We define early fusion as simply concatenating the feature data together, before any kernels or classifiers are applied. An example of a middle fusion strategy is to separately compute kernels, e.g., the HIK for color and RBF for shape metadata, and then to apply a combining strategy on the kernels. The late fusion strategy uses the classifier *outputs*, such as the activation values of a neural network, or the probability estimates of a Bayesian classifier, in conjunction with a combining strategy. This late approach can be considered when early fusion would result in a high-dimensional problem, when training is very expensive and we wish to reuse the constituent classifiers as black boxes. We briefly report on the results of these strategies below.

Our experiments also showed that early fusion (feature vector concatenation) was found to give the highest classification performance in comparison to midlevel and late-level fusion. Midlevel fusion using a product combination of kernels for color and shape metadata lowered the F1 score by around 4.4%, from 80.0% to 84.4%. Among the

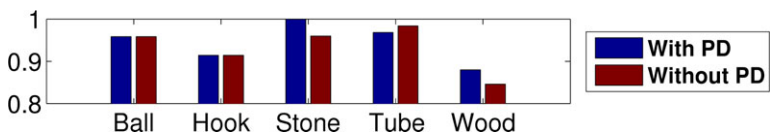


Figure 13. Per-class Precision, with and without presence detection (PD), at the operating point with highest F1. Improvements can be seen for Stone and Wood classes with PD, while Tube performs slightly worse. The average precision for objects is 94.42% (PD) and 93.30% (No PD).

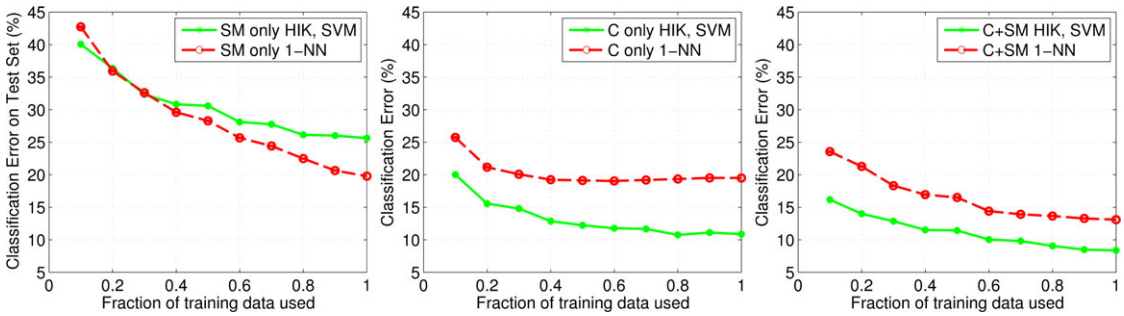


Figure 14. Learning curves using Presence Detection, with shape metadata (SM) only (left), color histogram (C) information only (center), and with fused color + shape metadata (C + SM, right). The SVM classifier is unable to generalize with shape metadata alone, but substantial gains are seen when fused with color information.

late fusion methods, we tested an additive combination of the probability values generated by separate color (p_C) and shape metadata (p_{SM}) classifiers, and also by taking the max combination, i.e., max(p_C, p_{SM}). The additive combination resulted in 2.5% lower classifier precision from 84.4% to 81.8%, while the product combination lowered precision by 3.4% from 84.4% to 80.9%. The learning curves in Figure 14 visually quantify the benefits of feature fusion. The performance of a support vector machine (SVM) using color features or shape metadata alone is compared to the one nearest-neighbor (1-NN) classifier. We use cross-validation to set the SVM cost parameter. It can be seen that SVM *underperforms* 1-NN after it has overfit to the data in the middle graph, using shape metadata only. When color and shape metadata are combined, however, the SVM classifier gains significant discriminative power, as seen in the right-hand side graph, and outperforms either feature alone. This ability to learn, generalize, and move beyond the performance of memorization is a prerequisite to adapt to new environments on which the system was not trained.

The low percentage of background training data in the "With PD" graph as seen in Figure 15 suggests that one way to boost performance with Presence Detection may be to increase the number of background images that it is trained on by using background images rejected in previous stages. The latter increases the number of background training images by $\approx 8\times$ and raises precision to 94.4% from using color and shape metadata (C + SM) features, compared to 89.4% without this step. The results in Table VI include this procedure. In addition, the right-hand side of the class distributions chart in Figure 15 (by object type) illustrates the central theoretical reason for the improvement in precision and recall of our three-level classifier cascade: with presence detection (PD), the classifier's workload involves far less classification of background images than without PD.

Finally, we conducted experiments to determine the execution time for presence detection, object localization, and sample classification. For this purpose, we chose several test scenarios containing no interesting objects at all, and scenarios with several target objects such as a tube, a hook, etc. The execution times for all scenarios were averaged over 480 test images with a resolution of 640 × 480 pixels. The overall execution time for the scenarios containing background images was 12 ± 3 ms for presence detection. For object images, the overall execution time was 12 ± 4.12 ms for presence detection, 10 ± 3.23 ms for localization, and 6 ± 2.45 ms for object classification. All the presence detection, localization, and object detection algorithms were executed on the hardware onboard the vehicle. In addition to the overall execution times, Figure 16 quantifies the computational savings of our approach using Presence Detection as a preprocessing step in comparison to approaches without presence detection. By reducing the number of images that the relatively expensive localization module must operate on, a very significant speedup can be achieved. However, the expected savings over the life of a long-autonomy mission would be much greater since the number of images containing interesting objects might be much lower than the number of images containing background only.

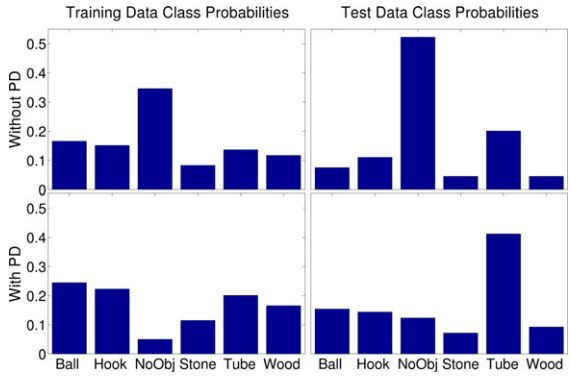


Figure 15. Training and Test Class distributions for one cross-validation fold ("patchy grass") seen by SVM classifier, with and without presence detection (PD).

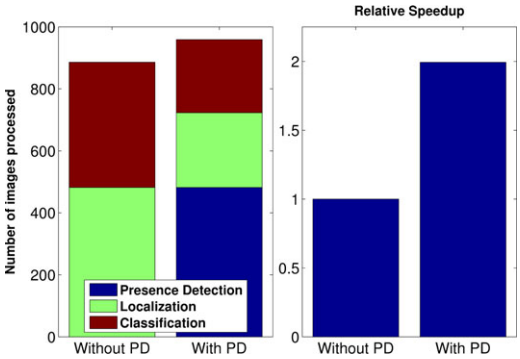


Figure 16. Time Savings using Presence Detection. By reducing the number of images that the relatively expensive localization module must operate on (left graph), a very significant speedup is achieved (right graph).

4 PATH PLANNING

The goal of the path-planning process is to address the needs of coverage, whereby a given environment is to be searched for samples with a guarantee that no areas will be left unvisited. The main cost metric of the optimization is the total path distance required to completely cover an area, given a finite sensor footprint that sweeps along the path.

The notion of coverage investigated in this work involves using the two perception modules, namely mapping and object detection as described in Sections 2 and 3, to navigate and search the environment. The mapping module generates a drivability map using a LIDAR with a large sensor footprint of approximately 75 m radius. The path planner must generate a path to search the environment as identified by the map by passing a smaller camera sensor footprint over every point in the search space. The planning process must be consistently updated as new map and visual information becomes available.

We solve the path-planning problem in two main stages. The first stage consists of transforming the map into a polygonal representation and decomposing it into a set of smaller polygons (sectors). The second stage involves generating an optimal coverage path through all the internal sectors. The two stages are each formulated as separate NP-hard problems, and polynomial approximation schemes are presented to achieve improvements over related work in the coverage planning literature.

Solutions to the *coverage planning problem* have been presented in a variety of application domains, including search and rescue (Moret, Collins, Saia, & Ling, 1997), domestic vacuum cleaning robots (Konolige, Augenbraun, Donaldson, Fiebig, & Shah, 2008), and automated milling tool path generation (Arkin et al., 2000). A common solution from the robotics domain is the Frontier Exploration method (Yamauchi, 1997), which assumes no knowledge of the map at initialization. Instead, a map is incrementally built via sensor feedback while commanding the robot to constantly drive toward the nearest boundary of unexplored space, defined as a frontier. The primary shortcoming of such an approach is the high probability of erratic overlapping paths that lead to excessive redundant sensor sweeps.

If the problem is relaxed by assuming a known map, a more efficient approach may be taken. For instance, Arkin et al. (2000) proposes a methodical "zigzag" sweeping pattern along a single major axis, applied across the entire map. The problem is formulated as a graph in which a vertex is created for each sweep collision with an obstacle or map boundary. A collision-free Eulerian tour of the graph can then be found. The heuristic solution presented in Arkin et al. (2000) results in a total path length that is bounded by 2.5 times the optimal. A major drawback of this approach is the need to circumnavigate obstacle and environment boundaries more than once.

The inefficiencies presented by a constant sweep direction can be mitigated using a method proposed by Moret et al. (1997). The authors propose a decomposition of the map into smaller sectors as it allows for variations in the sweep direction within each sector to better suit map geometry and minimize overall path length (Moret et al., 1997). The map is converted into a polygonal representation, and partitioning is performed at major vertices. The decomposed sectors are then represented as a graph, and the shortest coverage path through all sectors is computed using a Traveling Salesman path (TSP). The major assumption, however, is that no obstacles exist in the search space. The assumption simplifies the decomposition problem to one which can be solved using existing polynomial approximation algorithms (Greene, 1983; Keil, 1985).

In practice, obstacles and nonconvex boundaries in the polygonal representation of an environment can block lines of sight from a sensor. Therefore, any sectors derived from a decomposition must also be convex to guarantee sensor coverage. Given this additional constraint, the coverage path-planning problem can be transformed into the Watchman Problem (Ntafos, 1992), which finds the shortest path for a guard to fully monitor an area containing obstacles.

The Watchman problem is an NP-hard problem with its computational complexity dependent upon the number of regions present. One method of mitigating run-time in a real-time system is to reduce the number of regions required to represent the environment, the minimum of which is called the Optimal Convex Decomposition (OCD). Unfortunately, the OCD of a polygon with obstacles (or "*holes*") is itself an NP-hard problem (Keil, 2000).

To date, there have been various methods of suboptimal decompositions. The Trapezoidal cut method (Chazelle, 1987) slices the region in a constant direction from each obstacle and boundary vertex to form convex trapezoidal partitions. However, the large aspect ratios and often small areas make sweeping within such partitions inefficient. Another choice is Delaunay decomposition (Chew, 1989),

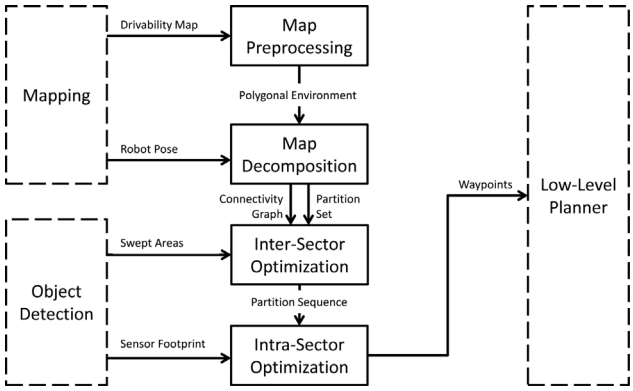


Figure 17. Coverage planning algorithm flowchart.

which forms small triangles around every vertex, optionally followed by aggregation algorithms. However, many more regions than are necessary are generated with Delaunay decomposition, which again leads to impractical search geometries.

More recently, a greedy decomposition approach (Vitus, Waslander, and Tomlin, 2008) displayed functionality similar to the OCD solutions by Greene and Keil, but it extended the capability to polygons with holes. The algorithm presents an approximation scheme that has been demonstrated to consistently outperform the Trapezoidal or Delaunay methods.

4.1 Coverage Planning Approach

The coverage planning approach is depicted in Figure 17 and consists of four main steps. The first step is *map preprocessing*, in which inputs are collected from the two perception modules of the robot. A 2D drivability map and the current robot pose are obtained from the mapping stack while the sensor footprint is obtained from the Object Detection stack. The map is then preprocessed into a polygonal representation using an edge-tracing method to define boundaries. In the *map decomposition* step, the greedy cut algorithm is applied to decompose the polygonal map into a set of convex sectors to be explored. Next, the *intersector path optimization* step represents the adjacency relationships between the convex sectors on a graph-based roadmap. The problem of finding an optimal order of sectors to visit is an NP-hard problem that is transformed into a Hamiltonian path problem and solved using the well-known Lin-Kernighan heuristic algorithm (Lin and Kernighan, 1973). Finally, the *intrasector path optimization* refines the complete coverage path given the sequence of sectors to traverse from the previous step. The sweep pattern orientation as well as the points of entry and exit points in each sector are computed to minimize the path length.

An important feature of coverage planning algorithms is the need to guarantee complete coverage in a changing map. The drivability map is frequently updated as new information becomes available to the robot and it is not sufficient to simply regenerate a path at every map update. The replanning process retains a memory of previously visited regions to avoid retraversals and the update frequency is limited to one sector at a time. The output of the path-planning process is a set of waypoints that can be executed by low-level obstacle avoidance planners such as Wavefront (Barraquand, Langlois, and Latombe, 1992), trajectory rollout, or other graph-based planning techniques. The choice of the low-level planner is correlated with the dynamics of the robot. However, since the particular application requires the robot to operate at low speeds, both differential steer and swerve drive robots are able to execute straight line paths designed using a simple A* graph-based path planner. Overall, the presented approach is compatible with any application for which a 2D drivability map can be produced and SLAM problems are solved.

4.2 Map Preprocessing

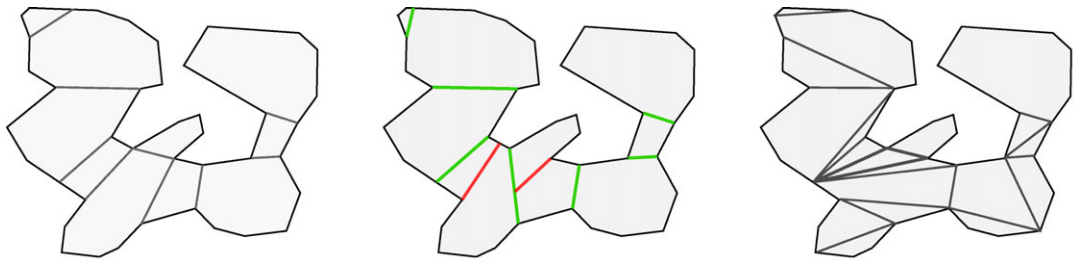
In map preprocessing, obstacle boundaries are dilated and converted into vector representations using Suzuki's border-tracing algorithm (Suzuki and Be, 1985). The boundary contours are then simplified to reduce the number of vertices at the cost of shape fidelity, since the eventual polygon decomposition stage is sensitive to the number of vertices in the system. The result is a set of polygons that represent the obstacle boundaries.

We then apply the Ramer Douglas Peucker (RDP) algorithm for boundary vertex reduction. In a review by Nguyen (Nguyen, Gächter, Martinelli, Tomatis, and Siegwart, 2007), RDP was found to be over twice as fast as the next best candidate in a comparison with incremental, line regression, RANSAC, Hough Transform, and Expectation-Maximization techniques. The RDP algorithm works by removing points less than a configurable orthogonal distance away from a line joining a start and end vertex. The remaining vertices are then iteratively used to form new lines toward the end vertex until all points have been inspected. The end result is control over the amount of detail along an obstacle border while maintaining the overall obstacle shape. It is assumed that the sensor footprint of the robot will cover any small areas no longer present due to smoothing.

4.3 Map Decomposition

Given a polygonal representation of the search space, the map decomposition problem can be stated as follows. Let an environment $\mathcal{E} \subset \mathbb{R}^2$ be bounded by a simple polygon, S, consisting of a set of vertices V and a set of directed edges E.

Each edge $e_{ij} \in E$ is defined as a line segment between two vertices v_i and v_j, where $i \neq j$. The vertices and edges



(a) Optimal Decomposition (b) Greedy cut (c) Greene's Sweep-line Approximation

Figure 18. (a) Optimal solution via Greene's Dynamic Programming algorithm. (b) Greedy cut solution returning the optimal number of partitions. Matching cuts are highlighted in green, single cuts are red. (c) Greene's swept-line approximation returning smaller, more jagged partitions.

in polygon S are listed in a clockwise (CW) fashion. The furthest Euclidean distance between any two vertices in S is defined as ξ. Obstacles (holes) are denoted by simple polygons that are internal to S. The set of all holes in the system is defined as H, and each hole, h_k, is indexed by $k \in \{1, \ldots, n_H\}$, where n_H is the cardinality of the set H. Each obstacle is also defined by a set of directed edges $e_{ij} \in E$ connecting vertices v_i and v_j, but stated in a counterclockwise (CCW) manner. The interior of the boundary excluding the obstacles represents the search space to be covered.

Let the interior angle between two adjacent edges of a polygon be defined as ψ_i, occurring at the vertex of intersection, v_i. A set of nonconvex vertices (NCV) is defined by

$$V_{\text{NCV}} = \{v_i \subset V : (v_i \in S \wedge \psi_i > \pi) \vee (v_i \in H \wedge \psi_i < \pi)\}. \tag{15}$$

4.3.1 Greedy Cut Decomposition Algorithm

The greedy cut decomposition is an approximation algorithm to the optimal convex decomposition of a polygon (Vitus et al., 2008). A sequential cut-based approach is taken that incrementally segments the map until all its constituents are convex. The main premise is that convex partitions can be formed by adding cuts emanating from each NCV, thereby dividing a nonconvex region into convex sectors (Chazelle and Dobkin, 1979).

A cut from an NCV to an existing edge or another cut is hereby referred to as a *single cut*, defined as an added edge to the system, $\{e_{i\Delta} : v_i \subset V_{\text{NCV}}\}$. It has been demonstrated that at most, a cut may eliminate two NCVs, one at each end point (Chazelle and Dobkin, 1979). Such two-NCV cuts are referred to as *matching cuts* and are defined as $\{e_{ij} : v_i \wedge v_j \subset V_{\text{NCV}}\}$. One condition is that all cuts must exist within the NCVs' cones of bisection, defined as an area bounded projection of the two edges connecting to an NCV.

The algorithm greedily searches all NCVs in the system to make matching cuts first, followed by single cuts for unmatched NCVs. Since greedy cuts are made until all NCVs are eliminated from the system, there is no preplanning or merging of partitions during the cut process. Instead, the combined set of all cut and boundary edges is used to trace the resultant partition boundaries. Starting with an arbitrary edge, a loop may be initiated and extended with an element within the edge set that forms the tightest convex turn with the previous edge. Edges are incrementally removed from the set once connected, and retracing a path back to the original start point signifies the completion of one sector. The process is repeated until no edges remain in the set. After cuts are applied to outstanding NCVs, the resulting edges define a set of n_S convex polygonal sectors.

The number of partitions returned, n_s, is upper bounded by a solution of only single cuts, which remove one NCV per cut, and lower bounded by a system consisting only of matching cuts, which remove two NCV per cut, and this interval must, by definition, also contain the optimal decomposition. The bounds on n_s are

$$n_s \in \left[\frac{|V_{\text{NCV}}|}{2} - |n_H| + 1, |V_{\text{NCV}}| - |n_H| + 1\right]. \tag{16}$$

The algorithm operates in $O(n \log n)$ time for the total number of NCVs in the matching cut and polygon identification stages, while a remaining single-cut stage runs in $O(n|E|)$, with the number of remaining unmatched NCVs.

Alternative NCV-elimination methods discovered in the literature are either slower [Greene's optimal Dynamic Programming algorithm, which operates in $O(n^4)$ time (Greene, 1983)], or results in too many polygons [the swept-line approximation method at $O(n \log n)$ time (Greene, 1983)]. Figure 18 shows the decomposition of a sample nonconvex polygon using the aforementioned algorithms, and highlights the near-optimal decomposition attained by the greedy cut.

4.4 Path Generation

Given the set of sectors to cover, the task of the path planner is to compute an optimal path ρ for the robot through the sectors such that sensor coverage is guaranteed over the entire searchable area. Each sector is entered and exited at points along its boundary and is searched using a *zigzag* sweep coverage pattern. The optimality of the path depends on the order in which sectors are visited, the structure of the coverage pattern within each region, and the entry and exit points used to link successive sectors. The solution approach involves a discrete problem formulation for which graph-based path-planning algorithms can be applied.

4.4.1 The Path Generation Problem

After the map decomposition step, the boundary polygon, S, is decomposed into a set of n_S convex polygonal sectors. Each sector, indexed by s_i, where $i \in \{1, \ldots, n_S\}$, represents a subset of the total traversable area to be covered by the robot such that $\cup_{i=1}^{n_S} \alpha(s_i) = \alpha(\mathcal{E})$, where $\alpha : \mathbb{R}^2 \rightarrow \mathbb{R}$ is a function that represents the area of a polygon.

Let each sector s_i, where $i \in \{1, \ldots, n_S\}$, be represented by a set of vertices V_{s_i}. The complete set of vertices in the map, $V = \cup_{i=1}^{n_S} V_{s_i}$ is the union of n_S collectively exhaustive but non-mutually-exclusive sets. It is clear that connected sectors share vertices. If sectors s_i and s_j are not connected, $V_{s_i} \cap V_{s_j} = \emptyset$. If they are connected, $V_{s_i} \cap V_{s_j} \neq \emptyset$ for $i \neq j$. The discrete formulation of the path generation problem enables an efficient graph-based approach to compute a search path ρ. The approximation scheme employed to compute the search path is dependant on a number of simplifications to the problem formulation that are detailed as follows.

Computing a path that passes through an entry and exit vertex in each sector vertex set V_{s_i} in the polygonal map presents a problem with a high level of computational complexity. Similar problems, such as the Generalized Vehicle Routing Problem (GVRP) (Goel and Gruhn, 2008), which involves finding a path passing through one vertex in each of many vertex sets, have been previously studied in the literature. However, for the coverage problem, defining sector sweep coverage costs between every pair of intrasector and intersector vertices is computationally expensive and unsupportable, given the temporal constraints of real-time operation. The problem can be mitigated by splitting the process into two smaller subproblems, namely an intersector path optimization and an intrasector path optimization.

Intersector path costs are minimized by finding a path that travels only between adjacent sectors, through shared vertices between them. In the case in which two consecutive sectors are not adjacent, the path travels along collision free lines between sector vertices. Restricting intersector paths in this manner is deemed suitable as it is equivalent to applying visibility graphs in polygonal maps, which have been shown to determine the shortest paths between any two vertices on the map (de Berg, van Kreveld, Overmars, and Schwarzkopf, 2000). The polygonal vertices and connecting edges in the map decomposition generate a similar visibility graph over the environment.

Intrasector sweep path costs can be minimized by optimally structuring the sweep path within a sector, given a sector's entry and exit points. By applying predetermined search patterns, sweep paths can be easily projected over large areas, and costs calculated rapidly for real-time path optimization.

The goal of the path generation problem formulation is to design a path such that for all $i \in \{1, \ldots, n_S\}$, the robot enters each polygonal sector s_i at a vertex $v_a \in V_{s_i}$, covers the sector using a sweep coverage pattern, and then leaves at a vertex $v_b \in V_{s_i}$ to continue its path.

4.5 Intersector Path Optimization

An unweighted, undirected *sector graph* $G_s = (V_s, E_s)$ is defined, where V_s is the set of vertices representing each of the n_S traversable sectors in $\mathcal{E}$. Each vertex $v_i \in V_s, i \in \{1, \ldots, n_S\}$ represents a sector s_i. An edge e_{ij} is added, between vertices v_i and v_j, representing sectors s_i and s_j if, for $i \neq j$, $V_i \cap V_j \neq \emptyset$ (i.e., the sectors have at least one common vertex).

The sector graph G_s is constructed under the assumption that the robot, when restricted to travel between adjacent sectors, will minimize the intersector travel costs. This is a reasonable simplification that allows a significant amount of edge pruning, given the assumption that every sector in the map is connected via a common vertex to at least one other sector. Figure 19 shows the construction of G_s over the polygonal sectors in the map decomposition. Assuming the robot only travels through shared vertices, the optimization over G_s can be described as the *Sector order decision problem*.

Sector visit order decision problem: *Does there exist a path ρ in G_s starting at v_0 such that $|v_i \cap \rho| >= 1 \quad \forall i \in \{1, \ldots, n_S\}$?*

The decision problem, as described, belongs to a known class of NP-hard problems that can be solved using a transformation to an NP-Complete Hamiltonian Path Problem. Since the graph G_s is connected, but not complete and thereby not guaranteed to be Hamiltonian, a reduction to the Hamiltonian path can be obtained using *metric closure* whereby for each pair of vertices v_i and v_j that are not connected in G_s, a temporary edge that represents the best shortest path between the vertices is computed using a shortest path algorithm such as A* or Djikstras (de Berg et al., 2000). The problem can then be solved using a variety of commercially available exact and heuristic traveling salesman problem (TSP) solvers (Lin and Kernighan, 1973).

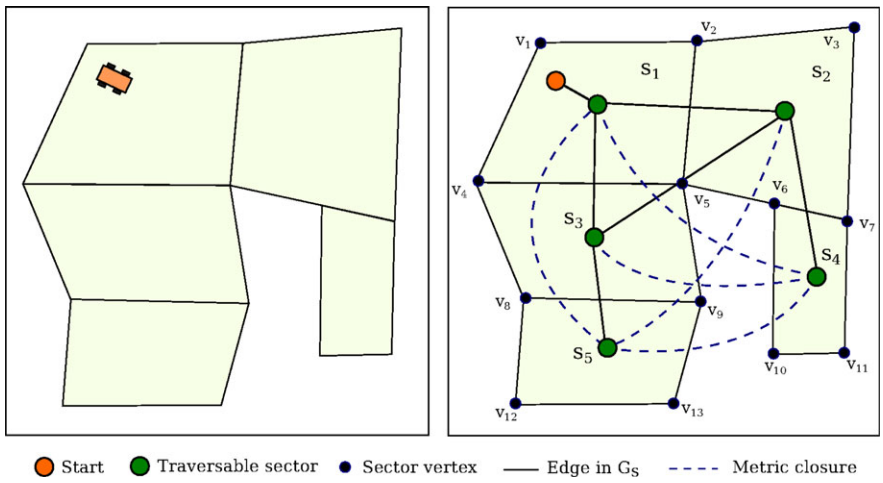


Figure 19. Graph-based formulation to optimize the intersector path generation.

4.5.1 Augmented Metric Closure

The sector graph G_s is transformed, using an augmented metric closure method into a graph on which the solution to the Hamiltonian path will provide the solution to the sector order visit problem.

First, the *vertex graph* $G_v = (V_v, E_v, c_v)$ is defined, where V_v is the total set of all vertices in the polygonal decomposition of $\mathcal{E}$, and E_v is the set of edges where an edge e_{ij} between vertices v_i and v_j exists if there is a direct, collision-free path between the two vertices. The cost function $c_v(e_{ij})$ is the Euclidean distance between the vertices. This graph forms the basis of intersector route planning in the robot. The robot is constrained to move only along the edges defined on graph G_v during all intersector travel.

Next, the *augmented graph* $G_{sv} = (V_{sv}, E_{sv}, c_{sv})$ is defined, where the set of vertices V_{sv} is inherited directly from the vertex set V_s of the sector graph G_s. All edges in E_s are copied into E_{sv}. For every pair of sectors s_i and s_j, where $i \neq j$ and $e_{ij} \notin E_s$, a supplementary edge e_{ij} is added and the edge costs are defined using the function $c_{sv}(e_{ij})$ as follows.

For each edge e_{ij},

$$c(e_{ij}) = \begin{cases} 0 & \text{if } e_{ij} \in E_s, \\ \tau(s_i, s_j) & \text{if } e_{ij} \notin E_s, \end{cases} \tag{17}$$

where $\tau(s_i, s_j)$, defined as the shortest route cost between sectors s_i and s_j, is calculated on graph G_v by finding $v_l \in s_i$ and $v_k \in s_j$, such that they minimize the smallest Euclidean distance between the two sectors. Now the cost $\tau(s_i, s_j)$ between v_l and v_k is computed on G_v using an A* search and assigned to $c(e_{ij})$ on G_{sv}. The graph G_{sv} thus defined is a complete graph that can be used to compute the Hamiltonian path through the sectors using the Concorde LinKern solver, which is a freely available implementation of the Lin-Kernighan heuristic (Lin and Kernighan, 1973). The resulting path provides an order of sectors for the robot to visit, given the defined cost metric.

4.6 Intrasector Route Optimization

The optimal sector visit order was computed as a Hamiltonian path with the assumption that in transitioning between adjacent sectors, the robot will pass through a common vertex. When faced with nonadjacent sectors, it will follow the shortest path through their two closest vertices. Note that since adjacent polygons can share more than a single vertex, the chosen entry and exit points in each sector will significantly affect the final cost. Ideally, the path must seamlessly transition between sectors with minimal unnecessary travel.

In optimizing the coverage path within a sector, a sweep pattern is generated within a convex polygon. For a pair of entry and exit points denoted by v_a and v_b, a sweep pattern, with a pitch that is equal to the diameter of the sensor footprint of the robot, is generated. The orientation of the sweep is chosen to minimize the total sweep path length within the sector.

4.6.1 Graph Construction for Intrasector Path Optimization

From Figure 19, it is observed that the shortest route through all sectors travels through entry and exit vertices, which are shared between adjacent sectors in the Hamiltonian path. By formulating the intrasector cost metric between the vertices, the problem space for the intrasector path optimization can be defined.

A *sweep* graph $G_a = (V_a, E_a, c_a)$ is defined, where V_a is the set of vertices, E_a is a set of directed edges between them, and c_a is the cost function for the edges in E_a. The attributes of the graph are detailed as follows:

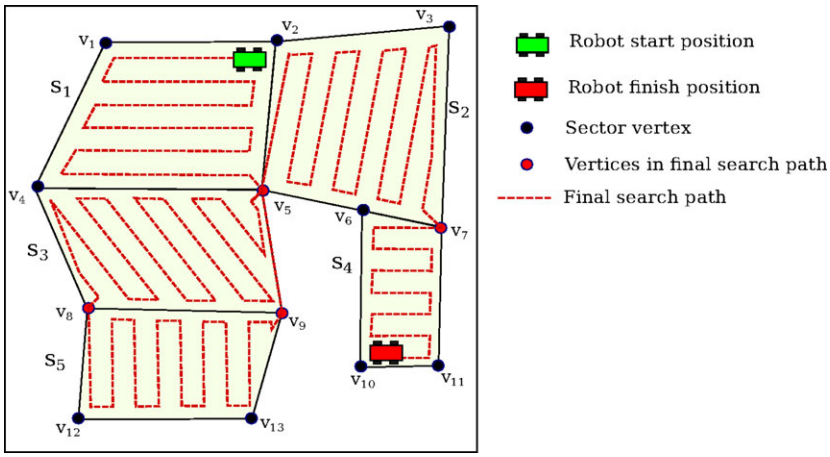


Figure 20. Final coverage path: The sector visit order generated is $\{x_1, x_3, x_5, x_2, x_4\}$. The path travels through intersector vertices in the order $\{v_5, v_8, v_9, v_5, v_7\}$. Observe that v_{dummy} between sectors s_5 and s_2 is replaced with the waypoints $\{v_9, v_5\}$.

Vertices: Define $n_S - 1$ disjoint vertex sets, $V_{a,1}, \ldots, V_{a,n_S-1}$. The set $V_{a,i}$ for $i \in \{1, \ldots, n_S - 1\}$ is given by the set of all shared vertices between sector s_i and s_{i+1} that are indexed according to the sector visit order. Shared vertices between every pair of sectors are defined independently, and if the same vertex is shared between more than two sectors, it is repeatedly defined in the formulation. In the case in which consecutive sectors s_i and s_j do not share any vertices, a dummy vertex is added to the empty set $V_{a,i}$. Any path through the graph is implicitly forced to pass through this dummy vertex. Once the path optimization is complete, the dummy vertex is replaced by a set of waypoints comprising the shortest path between the two disconnected sectors.

Another vertex set $V_{a,0}$ is defined to contain the start position v_0 of the robot in the map. The total vertex set in graph G_a is then $V_a = V_{a,0} \cup V_{a,1} \cup \cdots \cup V_{a,n_S-1}$.

Edges: We add a directed edge e_{lk} between vertex v_l and v_k, where $v_l \in V_{a,i}$ and $v_k \in V_{a,j}$ for some $i, j \in 1, \ldots, n_S - 1$, to E_a if $j = i + 1$. The edge e_{lk} represents a transition from the entering vertex v_l to the exiting vertex v_k in the sector s_i. v_k is subsequently the entering vertex in sector s_j.

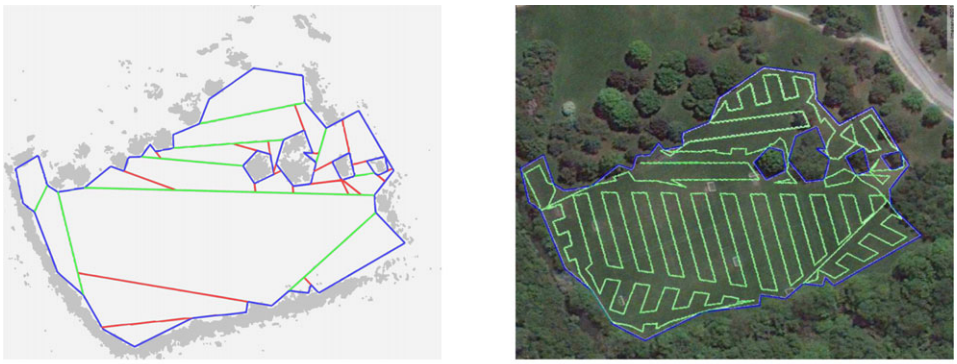
Edge Costs: Each edge $e_{lk} \in E_a$ is associated with a nonnegative cost $c_a(e_{lk})$ that is computed based on the distance traversed by the complete intrasector coverage path within sector s_i. The cost includes the sweeping pattern as well as travel to and from the entry and exit points. A zero cost is assigned to any edge e_{lk}, for which either v_l or v_k is a dummy vertex.

The A* shortest path search algorithm is now implemented on G_a and the solution is an ordered list of vertices in G_a that have a direct mapping to vertices in G_v on the polygonal map. In the A* solution path $\rho_a := \{v_0, v_1, v_2, \ldots, v_{n_S}\}$, every pair of consecutive vertices is an entry-exit pair for a visited sector. In the case in which a $V_{\text{temp}(i)}$ occurs in the path, it is replaced with the shortest path between the two sectors s_i and s_{i+1} with a cost of $\tau(s_i, s_j)$ as calculated on G_v. The intermediate waypoints between the two sectors are then injected into the dummy vertex placeholder. The path ρ_a is transformed into the final solution path ρ by injecting intermediate waypoints generated by the sweeping path in each sector s_i between vertices v_i and v_{i+1} indexed according to ρ_a. Using the above-mentioned methods, the theoretical final path obtained for the example in Figure 19 is shown in Figure 20.

4.7 Dynamic Replanning

The planning algorithm described thus far is capable of generating a compete coverage path over a static drivability map. However, in a partially known search environment, the planner must be able to dynamically adapt to a changing map and replan a coverage path as more information about the environment becomes available. Of particular concern in this situation is determining the replanning frequency and retaining a memory of previously visited regions of the map to minimize redundant search.

The approach used to solve this problem is consistent with the overarching planning method. Given that within each convex sector every point lies within the line of sight of the laser scanner, it can be safely assumed that the sectors near the robot contain no large occlusions in the current map. Therefore, a path plan for each sector can be executed without a global replan. Once each sector is searched, the robot decomposes the map with new information while blocking out sectors already visited. A new coverage path can now be planned to continue the search. This continues until all regions in the map are searched.



(a) Decomposition, rectangular region (b) Tour Path, rectangular region

Figure 21. Planning on sample drivability maps. On the decomposition, gray spots are obstacles, blue lines are boundaries, red lines are single cuts, and green lines are matching cuts. On the right, generated paths are shown overlaid on a satellite image.

Table VII. Greedy cut decomposition results.

	Total Area (m^2)	No. Obstacles	No. Sectors	No. Vertices	Run-time (s)
6 m pitch	10,943	4	32	100	0.55

4.8 Planning Results

The experimental results for path planning are presented in two segments. The following set of experiments demonstrate the solution quality and run-time performance of the planning algorithms on a single map compared with other methods described in literature. The second set of planning results are presented in Section 5, in conjunction with mapping to demonstrate dynamic replanning and real-time execution on a sample return robot.

The results demonstrating basic path planning are conducted in a centrally sparse region, with an approximate area of 11,000 m^2 that is bounded by shrubbery along a nearly rectangular perimeter. The proposed planning algorithm is validated by making comparisons to the Eulerian path approximation method by Arkin (Arkin et al., 2000). The main metric of comparison is total path length, and our coverage solutions are displayed in Figure 21. Table VII presents detailed decomposition results and collective processing times for the subroutines used in the planning algorithm, excluding initialization routines such as message handling and collision checks. For the experiments, a sweep pitch of 6 m is used to reflect the current sample detection camera range of the robot, and the onboard robot computers were used to measure execution times. Table VIII shows detailed path length results for the environment, including scenarios for various sweep pitches. In these comparisons, an exaggerated lower bound on the path length is found by dividing the area by the sensor footprint diameter (sweep pitch). Path lengths for the lower bound, Arkin's Eulerian path approximation, and the decomposition approach are presented.

It can be noted that the greedy algorithm consistently outperforms the Eulerian path approximation by an average of 25%, with the performance gap widening as pitch is enlarged. Given a larger pitch, the distance required to sweep an area is reduced, and the savings from setting sweep directions to avoid circumnavigating obstacles make up a larger percentage of the total path length. In comparing absolute path lengths, experimental results show a distance 1.1–1.6 times the lower bound for both test cases. These results demonstrate that the algorithm can successfully decompose a search region into convex search sectors, given a boundary and obstacle data derived from an online 2D drivability map. Execution times have been demonstrated to be under 3 s for the given environments, proving its viability for real-time applications. For further experimental analysis,

Table VIII. Path generation results, rectangular region.

			Greedy Cut and Path Generation		
Sweep Pitch (m)	Lower Bound (m)	Arkin (m)	Path length (m)	% Improvement	Run-time (s)
3	3,648	4,682	4,024	14.05	0.94
6	1,823	2,858	2,124	25.72	0.55
12	916	1,946	1,317	32.32	0.31

Figure 22. University of Waterloo rover for the NSRRC 2013.

Section 5 shows how by dynamically reacting to new obstacles, the algorithm is capable of generating tour paths that guarantee full sensor coverage to all areas of a changing map.

5 INTEGRATED EXPERIMENTAL RESULTS FOR THE SAMPLE RETURN PROBLEM

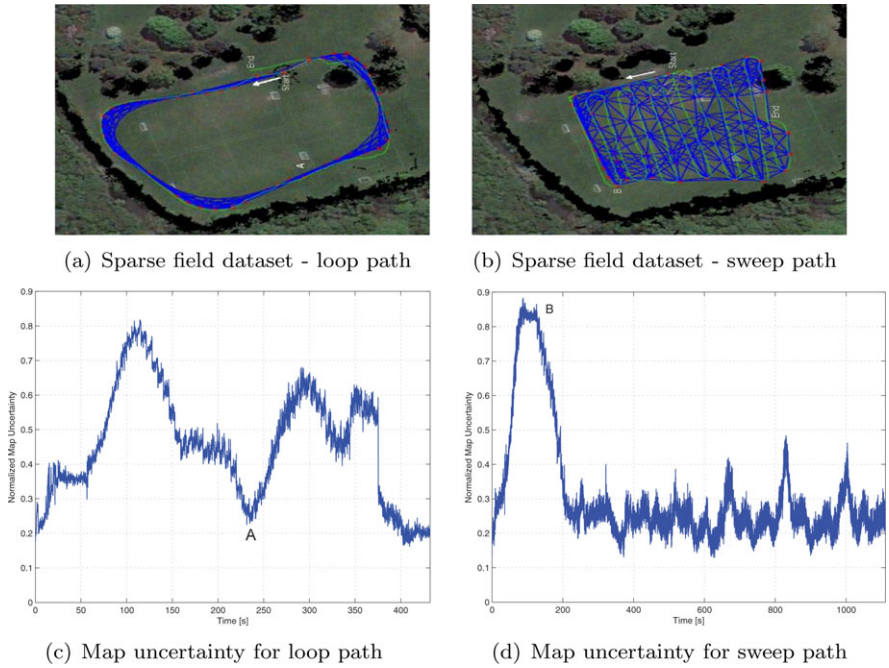
To spur innovation and expand on the state of the art, NASA has developed a Centennial Challenge for rover algorithm design, with a prize of $1.5 million USD. The first NASA Sample Return Robot Challenge (NSRRC) (Office of the Chief Technologist, 2013) was held in June of 2012 at Worchester Polytechnic Institute, and a second competition took place in June of 2013. The challenge requires competitors to develop an autonomous rover weighing less than 80 kg, capable of searching an unknown 80,000 m^2 area for 10 known samples in under 2 hours, at a maximum speed of 2 m/s. Some reference features from the environment are provided, as is a topographical map of the area, but neither GPS nor magnetometer measurements are to be relied on. Our test platform designed for this competition is depicted in Figure 22. This section highlights the considerations required for applying our autonomous exploration methodology as an integrated solution to solving the sample return problem. We first consider the interaction between mapping and coverage planning, and second the effect of the object detection sensor footprint on planning efficiency.

5.1 Considerations for Mapping and Dynamic Path Planning

The performance of the planning and mapping algorithms is heavily coupled, as the planned path directly affects the frequency of loop closures and natural features used for mapping, and the consistency of the map directly affects the replanning behavior of the planning algorithm. To determine the effect of loop navigation versus sweep path navigation on mapping consistency, an experiment is performed in which the vehicle traverses a loop path and a sweep path using the sparse field dataset. Figure 23 provides a comparison of the map uncertainty score over time as the vehicle performs the two test cases. For the loop path, it can be seen that the map uncertainty begins at approximately 0.2 and remains above 0.2 until the vehicle reaches point A along the path, as seen in Figure 23(c). At point A, the vehicle has revisited most of the area, thus the map uncertainty returns to the value seen at the start of the test, approximately 0.2. From point A, the vehicle continues along the loop until it reaches the end point of the path. The traversal through an unexplored area causes the map uncertainty to increase to a maximum of 0.7 and finally return to the start value of 0.2 when the vehicle reaches the end point of the loop. In contrast, for the sweep path the map uncertainty is initially large and then remains relatively constant, as presented in Figure 23(d). As the vehicle traverses from the start point to point B on the path, the map uncertainty grows as a result of exploration. From point B to the end of the path, the vehicle navigates a sweep path, resulting in smaller peaks in the map uncertainty as the vehicle adds new information to the map according to the ϵ_n parameter. It is clear that the traversal of the sweep path results in a relatively constant map uncertainly, resulting in a higher map confidence over the duration of the run. From the perspective of path planning, a relatively constant level of map uncertainty with minimal adjustments at each loop closure is preferred in order to maintain sufficient confidence in the planned paths at each global replan.

The results showing the dynamic replanning technique are presented on two drivability maps generated by the mapping module at different time steps in the coverage process, as shown in Figure 24. In each map, we assume a known governing boundary for coverage planning. The first map in Figure 24(a) is the view of the environment in the initial position of the robot. It shows the tentative path generated to cover the entire environment given the current information. In Figure 24(b), the path is replanned from the current robot pose using new map information that is available once the robot is within the line of sight of a previously occluded hedge of shrubs. At each replanning step, the robot retains a memory of visited sectors as denoted by the red polygons.

One of the challenges with this method is that if the map update causes a significant shift in the features of the drivability map, the memory of visited regions is rendered invalid because there is little knowledge to quantify the resulting shifts. Updates of the map may cause small, previously traversed regions to be marked as unvisited, causing the robot to occasionally retrace its steps, which necessitates a conservative approach to coverage for object detection. At the expense of imperfect coverage, it is possible to avoid smaller segments of the map that are more susceptible to inconsistencies. An area of future work is to better couple the mapping and planning modules



(a) Sparse field dataset - loop path

(b) Sparse field dataset - sweep path

(c) Map uncertainty for loop path

(d) Map uncertainty for sweep path

Figure 23. Test results for the Waterloo Park sparse field datasets. Red denotes graph SLAM vertices, blue lines denote graph edges, and the green line denotes the instantaneous vehicle path. (a) Resulting map for the sparse field dataset with a loop path. (b) Resulting map for the sparse field dataset with a sweep path. (c) Map uncertainty for the loop path. (d) Map uncertainty for the sweep path.

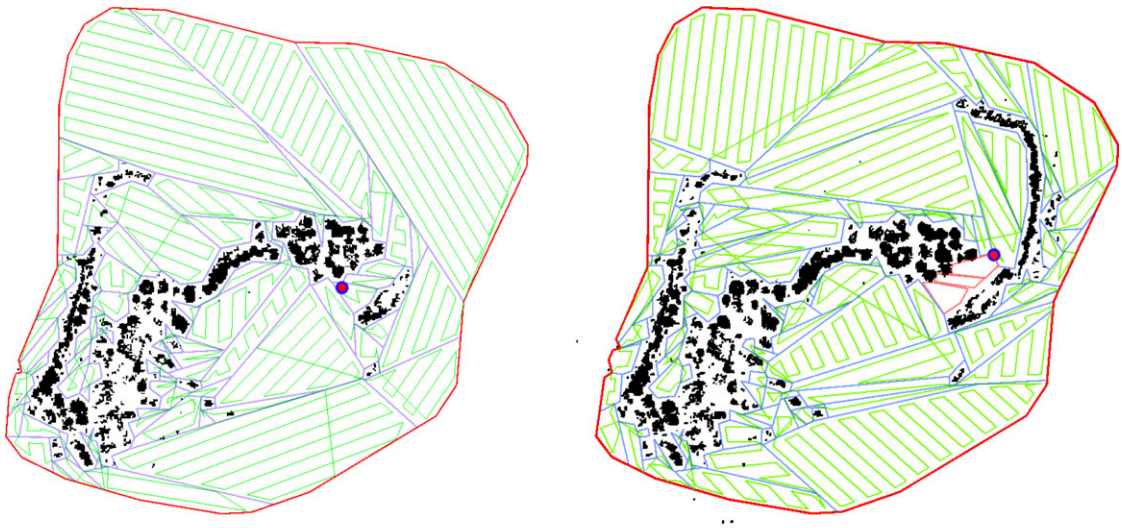
by using an active approach to planning to increase map information gain and enable more accurate coverage.

Another challenge with navigation using a graph-based search on a polygonal map is that robot pose uncertainties can present a significant hurdle to efficient path planning. Degradation of the pose estimate creates challenges in local collision avoidance as well as path planning, and a more robust approach to local path planning is desirable to help mitigate the effects of pose estimation uncertainty.

5.2 Considerations for Object Detection and Path Planning

While the mapping and planning modules are heavily coupled in their application to navigating the robot, the object detection module is coupled to the planning module through the sensor footprint, which sets the width of the sweep pattern. In Section 4.8, it is observed that the radius of the sensor footprint has a significant impact on the quality of the coverage path generated. As the sensor footprint grows, the path not only gets shorter by virtue of a larger footprint, but it also progressively performs better than the alternatives because of the cost savings with inter-sector transitions. A larger sensor radius also implies that the robot can cover the environment at a safer distance from obstacles and reduce the probability of unexpected collisions. Table IX illustrates the effect of varying the width of the sweep pattern on the quality of the computed coverage path for the drivability map in Figure 24.

Given these factors, a major consideration for the sample detection algorithms is to enlarge the camera sensor footprint as much as possible without compromising detection quality and speed. A good tradeoff between sensor footprint and detection fidelity must be achieved to ensure good coverage performance. To find a good tradeoff between detection accuracy and sensor footprint, an experiment is performed in which the vehicle passes the objects in predefined distances for detecting objects. Based on the detections recorded, we computed the detection performance based on Precision, Recall, and F1-measure to objectively evaluate different sensor footprints (radii). Table X shows the Precision, Recall, and F1-measure scores for different test radii, together with the number of sensor pixels that objects occupied when seen at certain distances. While increasing the sensor radius reduces the number of false-positive detections and results in a good Precision rate, the false-negative detection rate increases dramatically and results in low Recall rates. Hence, as seen in Table X, a sensor footprint with a radius of 3 m is a good tradeoff



(a) Path plan at initial time step (b) Re-plan at second time step

Figure 24. Demonstration of replanning at two time steps based on an updated map. The robot pose is denoted by a red dot. The maps show the polygonal decomposition and path plan at each instance.

Table IX. Path generation results for varying sensor footprints.

			Greedy Cut and Path Generation	
Sweep Pitch (m)	Lower Bound (m)	Arkin (m)	Path length (m)	Improvement (%)
1	16,992	19,286	17,614	8.67
2	8,496	10,790	9,175	14.96
3	5,664	7,957	6,332	20.42
4	4,248	6,541	4,860	25.71
5	3,398	5,692	4,218	25.89

Table X. Sweep pitch versus the number of available pixels for object detection and classification, and the overall detection performance averaged over all tested objects. A sweep pitch of 3 m is a good tradeoff between false-positive (Precision) and false-negative detections (Recall) and achieves the best F1 score.

Sweep Pitch (m)	No. Pixels	Precision	Recall	F1-measure
1	10,176	0.873	**0.812**	0.841
2	6,598	0.893	0.805	0.846
3	3,790	0.910	0.797	**0.849**
4	1,753	0.921	0.771	0.839
5	486	**0.934**	0.727	0.818

between false-positive and false-negative detections, and yields the largest F1 score for our application. In addition, increasing the sensor radius exponentially decreased the number of pixels that were available for detecting objects being far away from the robot. The latter is challenging for reliable object detection classification due to the reduced ability to extract unique features and descriptors for classification.

The system integration results demonstrate the collective execution of navigation and object detection algorithms on the robot in their application to autonomous coverage and search. They identify the challenges involved and the tradeoffs made to ensure that each of the mapping, planning, and object detecting algorithms works to maximize the effectiveness of the autonomous rover system.

6 CONCLUSION

This work presents a comprehensive approach to solving the main challenges of mapping, object detection, and coverage for autonomous exploration. The proposed approach demonstrates globally consistent map generation over long distances, reliable long-range object detection using small image patches, and efficient global coverage planning with unknown obstacles and contours. The integrated approach is shown to be successful in a sample return application, and is able to achieve real-time performance running onboard our test vehicle.

The SPC-SLAM approach provides a real-time solution for performing 2D SLAM in unstructured environments using sparse laser point cloud data. To achieve real-time performance on a platform with limited computation, the SPC-SLAM algorithm simplifies the MLS mapping

approach to operate on a 2.5D representation of the environment, as opposed to full 3D. To robustify localization of the vehicle in the global map, we apply a class constrained ICP registration technique, operating on point classifications based on ground adjacency. We present a computationally efficient method to compute map uncertainty as well as an efficient technique to generate globally consistent maps within a graph SLAM framework. Experiments demonstrate that the proposed mapping system is able to generate sufficiently detailed and globally consistent maps that are required for global coverage path planning. Future work will involve validating the approach over a large range of environments and detecting specific failure cases for the approach.

The object detection approach presented several contributions in a unified architecture to address three essential challenges for autonomous, real-time object detection. First, the introduction of visual saliency-based presence detection enables a large portion of the image frames without a salient object to be ignored. In experiments consisting of 50% background images and 50% sample object images, a 2× improvement of the detection time with a classification accuracy of 94.4% was observed. Second, a localization and segmentation image pipeline that is able to segment candidate samples of approximately 10×15 pixels using inexpensive features was presented. Such an image pipeline allows for the use of lower-cost sensors and increases the minimum required separation between path sweeps. Although state-of-the-art methods such as those targeting the VOC dataset can likely better handle more complex and cluttered environments, they do not operate in real time, and such capability and complexity is not required for our vehicle as it is typically deployed in natural, outdoor environments. Finally, a scheme is employed where a specialized presence detection classifier is responsible for the detection of background-only images. Such a scheme results in an improvement of precision and recall for the proposed three-level classifier cascade, and it also enables the classifier cascade to robustly detect interesting objects in previously unknown environments. Future work will include extending the visual saliency and presence detection to focus on specific discriminative criteria suited for monochromatic lunar or Mars exploration, such as albedo, texture, depth, hyperspectral response, or motion. To improve real-time performance, deployment of the saliency algorithms on parallel architectures such as GPUs or FPGA will be investigated.

The path-planning approach in this work presents two major approximation schemes to generate a minimum length path that guarantees search coverage over a bounded environment. The map decomposition algorithm presents a novel application of the greedy cut approach on a 2D drivability map of the environment, which is demonstrated to consistently produce solutions that outperforms other well known decomposition methods in terms of minimizing the number of convex regions produced. The path generation algorithm computes search paths that are, on average, 25% shorter than algorithms presented in the literature that do not use decomposition methods. The planning approach presented provides a complete global path that can also be used to calculate the total required mission time. Knowledge of the mission provides a significant advantage over random search approaches such as the Frontier Exploration methods, for which no reliable predictions on path length can be made. Experiments illustrated that the greedy algorithm was capable of planning test areas between 10,000 and 20,000 m^2 in under 3 s. Given the speed of the vehicle and the desire to only replan for significant map updates, this update rate is sufficient for online coverage planning applications. Future work involves developing a solution to minimize changes of direction in paths generated with every replan.

The NASA Sample Return Robot Challenge provides a complex autonomous exploration problem, with many distinct components that need to be addressed simultaneously on a rover of finite resources and capabilities. By identifying the key bottlenecks and limitations of existing methods for SLAM, object detection, and path planning, this work presents contributions to each of these components and demonstrates a strong application of our autonomous exploration methodology to the sample return problem. Although the experimental results and algorithm specifics are applied to the outdoor, grassy, and loosely forested environment of the NSRRC, they can readily be extended to other terrain types, and are a viable approach for fully autonomous exploration in a wide range of environments.

REFERENCES

Achanta, R., Hemami, S., Estrada, F., & Susstrunk, S. (2009). Frequency-tuned salient region detection. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 1597–1604), Miami, FL.

Arkin, E., Held, M., & Smith, C. (2000). Optimization problems related to zigzag pocket machining. Algorithmica, 26(2), 197–236.

Bachrach, A., Prentice, S., He, R., & Roy, N. (2011). RANGE Robust autonomous navigation in GPS-denied environments. Journal of Field Robotics, 28(5), 644–666.

Balakirsky, S., Carpin, S., Kleiner, A., Lewis, M., Visser, A., Wang, J., & Ziparo, V. A. (2007). Towards heterogeneous robot teams for disaster mitigation: Results and performance metrics from robocup rescue: Field reports. Journal of Field Robotics, 24(11-12), 943–967.

Barraquand, J., Langlois, B., & Latombe, J. (1992). Numerical potential field techniques for robot path planning. IEEE Transactions on Systems, Man and Cybernetics, 22(2), 224–241.

Bogdan Rusu, R., Sundaresan, A., Morisset, B., Hauser, K., Agrawal, M., Latombe, J., & Beetz, M. (2009). Leaving flatland: Efficient real-time three-dimensional perception and

motion planning. Journal of Field Robotics, 26(10), 841–862.
Borrmann, D., Elseberg, J., Lingemann, K., Nüchter, A., & Hertzberg, J. (2008). Globally consistent 3D mapping with scan matching. Robotics and Autonomous Systems, 56(2), 130–142.
Broggi, A., Medici, P., Zani, P., Coati, A., & Panciroli, M. (2012). Autonomous vehicles control in the vislab intercontinental autonomous challenge. Annual Reviews in Control, 36(1), 161–171.
Castaño, R., Estlin, T., Anderson, R., Gaines, D., Castano, A., Bornstein, B., Chouinard, C., & Judd, M. (2007). Oasis: Onboard autonomous science investigation system for opportunistic rover science. Journal of Field Robotics, 24(5), 379–397.
Castaño, R., Estlin, T., Anderson, R., Gaines, D., Bornstein, B., & Judd, M. (2008). Opportunistic detection and measurement of novel rocks. In Lunar and Planetary Science Conference.
Chazelle, B. (1987). Approximation and decomposition of shapes. In Schwartz, J. T., & Yap, C. K. (eds.), Algorithmic and Geometric Aspects of Robotics (pp. 145–185). Hillsdale, NJ: Lawrence Erlbaum Associates.
Chazelle, B., & Dobkin, D. (1979). Decomposing a polygon into its convex parts. In Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing, STOC '79 (pp. 38–48). New York: ACM.
Cheng, M., Zhang, G., Mitra, N., Huang, X., & Hu, S. (2011). Global contrast based salient region detection. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 409–416). Colorado Springs, CO.
Chew, L. (1989). Constrained Delaunay triangulations. Algorithmica, 4(1), 97–108.
Coates, A., & Ng, A. (2010). Multi-camera object detection for robotics. In IEEE International Conference on Robotics and Automation (ICRA) (pp. 412–419).
Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In Conference on Computer Vision and Pattern Recognition (CVPR).
Das, A., & Waslander, S. (2012). Scan registration with multi-scale K-means normal distributions transform. In IEEE International Conference on Intelligent Robots and Systems (IROS) (pp. 2705–2710). Vilamoura, Algarve, Portugal.
Davison, A., Reid, I., Molton, N., & Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(6), 1052–1067.
de Berg, M., van Kreveld, M., Overmars, M., & Schwarzkopf, O. (2000). Computational geometry: Algorithms and applications, 2nd ed. Springer-Verlag.
Douillard, B., Quadros, A., Morton, P., Underwood, J., De Deuge, M., Hugosson, S., Hallstrom, M., & Bailey, T. (2012). Scan segments matching for pairwise 3d alignment. In 2012 IEEE International Conference on Robotics and Automation (ICRA) (pp. 3033–3040), St. Paul, MN.
Duda, R. O., Hart, P. E., & Stork, D. G. (2000). Pattern classification, 2nd ed. Wiley.
Dunbabin, M., & Marques, L. (2012). Robots for environmental monitoring: Significant advancements and applications. IEEE Robotics & Automation Magazine, 19(1), 24–39.
Ekvall, S., Kragic, D., & Jensfelt, P. (2007). Object detection and mapping for service robot tasks. Robotica, 25(02), 175–187.
Engelhard, N., Endres, F., Hess, J., Sturm, J., & Burgard, W. (2011). Real-time 3D visual SLAM with a hand-held RGB-D camera. In RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum, Vasteras, Sweden.
Estlin, T., Bornstein, B., Gaines, D., Anderson, R., Thompson, D., Burl, M., Castaño, R., & Judd, M. (2012). Aegis automated science targeting for the mer opportunity rover. ACM Transactions on Intelligent Systems and Technology (TIST), 3(3), 50.
Fong, T., Allan, M., Bouyssounouse, X., Bualat, M., Deans, M., Edwards, L., Flückiger, L., Keely, L., Lee, S., & Lees, D. (2008). Robotic site survey at Haughton Crater. In International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS) (pp. 26–29), Los Angeles.
Frintrop, S., Backer, G., & Rome, E. (2005). Goal-directed search with a topdown modulated computational attention system. In Annual Meeting of the German Association for Pattern Recognition (DAGM) (pp. 1395–1407).
Goel, A., & Gruhn, V. (2008). A general vehicle routing problem. European Journal of Operational Research, 191(3), 650–660.
Grauman, K., & Darrell, T. (2005). The pyramid match kernel: Discriminative classification with sets of image features. In IEEE International Conference on Computer Vision (ICCV) (pp. 1458–1465). IEEE.
Greene, D. H. (1983). The decomposition of polygons into convex parts. Computational Geometry, Advances in Computing Research, 1, 235–259.
Grisetti, G., Stachniss, C., & Burgard, W. (2007). Improved techniques for grid mapping with Rao-Blackwellized particle filters. IEEE Transactions on Robotics, 23(1), 34–46.
Guizzo, E. (2011). Japan earthquake: Robots help search for survivors. http://spectrum.ieee.org/automaton/robotics/industrial-robots/japan-earthquake-robots-help-search-for-survivors.
Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. (2009). The WEKA data mining software: An update. ACM SIGKDD Explorations Newsletter, 11(1), 10–18.
Hartley, R. I., & Zisserman, A. (2004). Multiple view geometry in computer vision, 2nd ed. Cambridge University Press.
Hou, X., & Zhang, L. (2007). Saliency detection: A spectral residual approach. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 1–8), Minneapolis, MN.
Howard, T., Morfopoulos, A., Morrison, J., Kuwata, Y., Villalpando, C., Matthies, L., & McHenry, M. (2012). Enabling continuous planetary rover navigation through FPGA stereo and visual odometry. In IEEE Aerospace Conference.
Im, S., & Cho, S. (2006). Context-based scene recognition using Bayesian networks with scale-invariant feature

transform. In International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS) (pp. 1080–1087), Antwerp, Belgium.

Itti, L., Koch, C., & Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(11), 1254–1259.

Jain, A., Wong, A., & Fieguth, P. (2012). Saliency detection via statistical non-redundancy. In IEEE International Conference on Image Processing (ICIP), Orlando, FL.

Kammel, S., Ziegler, J., Pitzer, B., Werling, M., Gindele, T., Jagzent, D., Schröder, J., Thuy, M., Goebl, M., Hundelshausen, F. v., Pink, O., Frese, C., & Stiller, C. (2008). Team AnnieWAY's autonomous system for the 2007 darpa urban challenge. Journal of Field Robotics, 25(9), 615–639.

Keil, J. M. (1985). Decomposing a polygon into simpler components. SIAM Journal of Computing, 14(4), 799–817.

Keil, J. M. (2000). Handbook of computational geometry, volume 2, chapter Polygon Decomposition. Amsterdam: North Holland.

Kim, W., Nesnas, I., Bajracharya, M., Madison, R., Ansar, A., Steele, R., Biesiadecki, J., & Ali, K. (2009). Targeted driving using visual tracking on Mars: From research to flight. Journal of Field Robotics, 26(3), 243–263.

Konolige, K., Agrawal, M., & Sol, J. (2011). Large-scale visual odometry for rough terrain. In Kaneko, M., & Nakamura, Y., (eds.), Robotics Research, volume 66 of Springer Tracts in Advanced Robotics (pp. 201–212). Berlin/Heidelberg: Springer.

Konolige, K., Augenbraun, J., Donaldson, N., Fiebig, C., & Shah, P. (2008). A low-cost laser distance sensor. In IEEE International Conference on Robotics and Automation (pp. 3002–3008).

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., & Burgard, W. (2011). g2o: A general framework for graph optimization. In IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China.

Kümmerle, R., Hahnel, D., Dolgov, D., Thrun, S., & Burgard, W. (2009). Autonomous driving in a multi-level parking structure. In IEEE International Conference on Robotics and Automation (ICRA) (pp. 3395–3400), Kobe, Japan.

Levinson, J. (2011). Automatic laser calibration, mapping, and localization for autonomous vehicles. PhD thesis, Stanford University.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. Operations Research, 21(2), 498–516.

Liu, Y., & Nejat, G. (2013). Robotic urban search and rescue: A survey from the control perspective. Journal of Intelligent and Robotic Systems, 72(2), 147–165.

Magnusson, M., Duckett, T., & Lilienthal, A. J. (2007). Scan registration for autonomous mining vehicles using 3D-NDT. Journal of Field Robotics, 24(10), 803–827.

Mitri, S., Frintrop, S., Pervolz, K., Surmann, H., & Nuchter, A. (2005). Robust object detection at regions of interest with an application in ball recognition. In IEEE International Conference on Robotics and Automation (ICRA) (pp. 125–130).

Moret, B. M. E., Collins, M. J., Saia, J., & Ling, Y. (1997). Ice rinks and cruise missiles: Sweeping a simple polygon. In Workshop on Algorithm Engineering (WAE), Venice, Italy.

Murphy, R. (2004). Trial by fire [rescue robots]. IEEE Robotics Automation Magazine, 11(3), 50–61.

Murphy, R. R., Tadokoro, S., Nardi, D., Jacoff, A., Fiorini, P., Choset, H., & Erkmen, A. M. (2008). Search and rescue robotics. In Siciliano, B., & Oussama, K. (eds.), Springer Handbook of Robotics (pp. 1151–1173). Springer-Verlag.

Nguyen, V., Gächter, S., Martinelli, A., Tomatis, N., & Siegwart, R. (2007). A comparison of line extraction algorithms using 2D range data for indoor mobile robotics. Autonomous Robots, 23(2), 97–111.

Ntafos, S. (1992). Watchman routes under limited visibility. Computational Geometry, 1(3), 149–170.

Nüchter, A. (2009). 3D robotic mapping—The Simultaneous Localization and Mapping Problem with six degrees of freedom, volume 52 of Springer Tracts in Advanced Robotics. Springer.

Office of the Chief Technologist, T. (2013). NASA sample return robot challenge 2013. http://www.nasa.gov/robot. Accessed: 3/12/2012.

Oliva, A., Torralba, A., Castelhano, M., & Henderson, J. (2003). Top-down control of visual attention in object detection. In IEEE International Conference on Image Processing (ICIP) (vol. 1, pp. I–253), Barcelona, Spain.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2(6), 559–572.

Pfaff, P., Kümmerle, R., Joho, D., Stachniss, C., Triebel, R., & Burgard, W. (2007). Navigation in combined outdoor and indoor environments using multi-level surface maps. In Workshop on Safe Navigation in Open and Dynamic Environments at the 2007 IEEE Int. Conf. on Intelligent Robots and Systems (IROS), San Diego, CA.

Pugh, S., Tyler, L., & Barnes, D. (2010). Automatic pointing and image capture (APIC) for exomars type mission. In International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS).

Pugh, S., Tyler, L., Barnes, D., Labrosse, F., & Neal, M. (2011). Automatic pointing and image capture: A field study. In Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA).

Rudinac, M., & Jonker, P. (2010). Saliency detection and object localization in indoor environments. In International Conference on Pattern Recognition (ICPR) (pp. 404–407).

Rutishauser, U., Walther, D., Koch, C., & Perona, P. (2004). Is bottom-up attention useful for object recognition? In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (vol. 2, pp. II–37).

Segal, A., Haehnel, D., & Thrun, S. (2009). Generalized-ICP. In Robotics: Science and Systems (RSS) (vol. 25, pp. 26–27), Seattle.

Sjoe, K., Lopez, D., Chandana, P., Jensfelt, P., & Kragic, D. (2009). Object search and localization for an indoor mobile robot. Journal of Computing and Information Technology, 17(1), 67–80.

Stoyanov, T. D., Magnusson, M., Andreasson, H., & Lilienthal, A. (2012). Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations. The International Journal of Robotics Research, 31(12), 1377–1393.

Suzuki, S., & Be, K. (1985). Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing, 30(1), 32–46.

Thompson, D. R., & Wettergreen, D. (2005). Multiple-object detection in natural scenes with multiple-view expectation maximization clustering. In International Conference on Intelligent Robots and Systems (IROS '05) (pp. 448–453).

Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic robotics (intelligent robotics and autonomous agents). Cambridge, MA: MIT Press.

Tomasi, C., & Manduchi, R. (1998). Bilateral filtering for gray and color images. In Computer Vision, 1998. Sixth International Conference (pp. 839–846). IEEE.

Tong, C., Barfoot, T., & Dupuis, É. (2012). Three-dimensional SLAM for mapping planetary work site environments. Journal of Field Robotics, 29(3), 381–412.

Tongtong, C., Bin, D., Daxue, L., Bo, Z., & Qixu, L. (2011). 3D LIDAR-based ground segmentation. In First Asian Conference on Pattern Recognition (ACPR) (pp. 446–450), Beijing, China.

Torralba, A., & Sinha, P. (2001). Statistical context priming for object detection. In International Conference on Computer Vision (ICCV) (pp. 763–770).

Triebel, R., Pfaff, P., & Burgard, W. (2006). Multi-level surface maps for outdoor terrain mapping and loop closing. In IEEE International Conference on Intelligent Robots and Systems (IROS) (pp. 2276–2282), Beijing, China.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M. N., Dolan, J., Duggins, D., Gittleman, M., Harbaugh, S., Wolkowicki, Z., Ziglar, J., Bae, H., Brown, T., Demitrish, D., Sadekar, V., Zhang, W., Struble, J., Taylor, M., Darms, M., & Ferguson, D. (2008). Autonomous driving in urban environments: Boss and the urban challenge. Journal of Field Robotics, 25(8), 425–466.

Vitus, M. P., Waslander, S. L., & Tomlin, C. J. (2008). Locally optimal decomposition for autonomous obstacle avoidance with the tunnel-MILP algorithm. In IEEE Conference on Decision and Control (CDC) (pp. 540–545), Cancun, Mexico.

Walther, D., & Koch, C. (2006). Modeling attention to salient proto-objects. In Science Direct. Neural Networks 19 (pp. 1395–1407).

Wettergreen, D., Wagner, M., Jonak, D., Baskaran, V., Deans, M., Heys, S., Pane, D., Smith, T., Teza, J., Thompson, D., Tompkins, P., & Williams, C. (2008). Long-distance autonomous survey and mapping in the robotic investigation of life in the Atacama desert. In International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS).

Xu, T., Chenkov, N., Kuhnlenz, K., & Buss, M. (2009a). Autonomous switching of top-down and bottom-up attention selection for vision guided mobile robots. In IEEE International Conference on Intelligent Robots and Systems (IROS) (pp. 4009–4014).

Xu, T., Wu, H., Zhang, T., Kuhnlenz, K., & Buss, M. (2009b). Environment adapted active multi-focal vision system for object detection. In IEEE International Conference on Robotics and Automation (ICRA) (pp. 2418–2423).

Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA) (p. 146), Washington, D.C.

Yu, Y., Mann, G., & Gosine, R. (2009). A novel robotic visual perception method using object-based attention. In Robotics and Biomimetics (ROBIO), 2009 IEEE International Conference (pp. 1467–1473).

Zhai, Y., & Shah, M. (2006). Visual attention detection in video sequences using spatiotemporal cues. In ACM International Conference on Multimedia (pp. 815–824), Santa Barbara, CA.

Zheng, X., Jain, S., Koenig, S., & Kempe, D. (2005). Multi-robot forest coverage. In IEEE International Conference on Intelligent Robots and Systems (IROS) (pp. 3852–3857), Edmonton, Canada: IEEE.