Low-Cost Multi-Robot Exploration and Mapping

Christopher M. Gifford, Russell Webb, James Bley, Daniel Leung, Mark Calnon, Joe Makarewicz, Bryan Banz, Arvin Agah
Electrical Engineering and Computer Science Department, University of Kansas, Lawrence, KS
(785) 864-7842, cgiффord@eecs.ku.edu

Abstract—Mobile robots can perform some of the more dangerous and laborious human tasks on Earth and throughout the solar system, many times with greater efficiency and accuracy, saving both time and resources. As we explore further away from Earth, higher levels of autonomy are also becoming more desired in such applications, one of them being distributed mapping. Smaller, less expensive mobile robots are becoming more prevalent, which introduces unique challenges in terms of limited sensing accuracy and onboard computing resources. This paper presents a low-cost approach to autonomous multi-robot mapping and exploration for unstructured environments. Platform design and implementation details are discussed, along with results from a planetary style environment. Results demonstrate that mobile robots capable of SLAM can be constructed for less than $1250, and similar concepts could be used for planetary missions.

1. INTRODUCTION

Deployment in unstructured environments introduce unique challenges for mobile robot design, navigation, and actuation for science objectives. Structured environments, such as buildings and classrooms, are easier to simulate and model. Surfaces are consistent with low levels of uncertainty and typically exhibit minimal slippage. These environments are also relatively static in nature.

Unstructured environments, such as the deserts on Earth and the rocky landscapes on Mars and other planets, compound the issues encountered in structured environments, and add several others, including high levels of danger and weather extremes. Higher levels of surface detail are typically not known, and this is especially true for planets further from Earth. Surface inconsistencies are abundant, higher levels of uncertainty are present, and variable slippage is unavoidable. Mobile robots in these environments require more sophisticated intelligence and higher levels of autonomy due to substantial communication delays. Reliable navigation, advanced sensor suites, and data/imagery feedback for scientific analysis are necessary to be successful. In addition, localization, path-planning, and mapping abilities are paramount for autonomous exploration and mapping of such environments.

We have designed and implemented a system for autonomous mapping and exploration for unstructured environments. The system is comprised of four simple and low-cost exploration and mapping robots which communicate and coordinate their efforts with a base station (e.g., a lander). The resulting global map of the region can then be utilized to safely and efficiently path-plan and traverse the environment to locations of interest, as determined by the exploration team. Although designed for a planetary surface, the system could also be used for mapping of structured environments and search and rescue missions.

2. LOW-COST LOCALIZATION AND MAPPING

Simultaneous localization and mapping (SLAM) is an important problem in the field of robotics, and is well-studied from the single robot standpoint [1]. To successfully accomplish this, a robot must know where it is at all times and have highly accurate ranging sensors to accurately map an environment. The majority of SLAM systems utilize expensive laser range-finders as their means to autonomously and accurately map an environment, as well as a significant amount of onboard computing resources to perform the task online. Statistical methods such as Kalman and particle filters are used to improve odometry, sensor readings, and their integration. Many methods also rely on knowledge of landmarks and do not scale well.

Distributed SLAM uses multiple robots to map an environment. This eliminates the single point of failure inherent in single robot systems, as well as increases mapping speed, robustness, and overall task efficiency. The ability to perform distributed science experiments becomes possible (e.g., multi-robot seismic [2] and radar remote sensing). These disadvantages result in additional challenges of communication and coordination of the robots, data aggregation (e.g., merging individual maps), and managing overall behavior of the system. As SLAM and exploration extend to multiple robots, this impacts the overall system’s cost, size, and complexity. Smaller, low-cost robots are therefore desirable, and require low-cost mapping methods.

Low-cost systems inherently involve limited sensing accuracy, range, and computing abilities. The system must be designed to exploit these attributes. This becomes apparent when attempting to run online SLAM algorithms, which are accompanied by assumptions and computing loads that may be incompatible with platform specifications. Increased noise and inaccuracies from sensors and localization make this a challenging task.
3. Operational Overview

The operational overview of the system from a mission perspective is illustrated using a scenario. A spacecraft containing multiple robots is to land on the surface of another planet in the solar system. The lander successfully makes the long journey, safely touches down on the surface, and communicates back to scientists and engineers on Earth for coordinates of a nearby region to explore. Based on initial images sent back, scientists decide that they want a specific rectangular region autonomously explored and a map with areas of interest provided back to them.

Upon receiving the exploration grid assignment, the lander acts as the System Commander and divides the area into sub-regions based on the number of available exploration robots. Each independent sub-region is then assigned to specific robots which are then responsible for autonomously exploring and mapping that region. The first challenge is for all robots to safely disembark the lander, descend the ramp, and navigate to their respective areas. This is accomplished by coordinating the robots to fully exit and navigate away from the ramp one at a time.

Each exploration robot proceeds to autonomously map its independent sub-region by reactively navigating between frontier waypoints it creates to efficiently explore it. While exploring, the robots process images and make note of areas or objects of interest they encounter. These could be specific to a mission (i.e., something the science team is looking for), or could be general objects such as shiny objects, objects colored differently than the normal environment color (e.g., a white patch on the surface of Mars), or another robot that has failed. Once a robot has explored its area, its map is wirelessly transmitted to the lander for integration with those from other exploration robots. If a robot finishes its mapping task away from the lander, it can use the map it built to plan a safe route back. If the mission is complete, one robot at a time will be allowed up the ramp and back onto the lander platform.

4. Exploration Robots: Specifications

Each exploration robot measures 39 x 27 x 28 cm and is capable of navigating over variable, unstructured surfaces using a four-wheel skid-steer drive system. To ensure the robots could travel over surfaces consisting entirely of sand and/or gravel, the drive train provides a 1:5 gear ratio so the robots can drive up small slopes as well as turn in-place when needed. A custom, low-cost 6-IR scanning range array is the platform’s primary range sensor. A Gumstix verdex XL6P motherboard [3] runs an online distributed particle-based SLAM algorithm (modified version of DP-SLAM [4]). It is the central hub for robot component communication. Images of the robot are shown in Figure 1. Robot platform details, including sensing system, processing capabilities, and power system are summarized in Table 1. The robot costs less than $1250 in parts, including custom circuitry for the communication node, inertial measurement unit (IMU), and power system.

Due to size, weight, and power restrictions, these systems have limited computing resources and sensing range. Their low-cost nature translates to decreased sensor accuracy and increased localization error. However, limited computing is distributed about the platform (see Table 1) to reduce the load on any one processing unit.

5. Navigation Architecture

The navigation system for the robots was implemented as a hybrid reactive-deliberative system using both the potential fields and subsumption architectures, and is based on the behavior, planning, and coordination layers described in [10]. Each behavior produces a normalized drive vector to be fused with those from other behaviors in the current state, the result of which is translated into left and right motor commands. Each state specifies which behaviors are active, and how much each is weighted in terms of its contribution. A robot can be in one of five primary states:

1. System startup and initialization
2. Ramp descent
3. Exploration and mapping
4. Ramp ascent
5. Mission complete


Table 1 - Specifications of the low-cost exploration and mapping robots.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Chassis: Custom VEX [5] metal frame, with a 3.8 cm clearance and 1:5 gear ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dimensions (LxWxH): 39 x 27 x 28 cm</td>
</tr>
<tr>
<td></td>
<td>Weight: 2.8 kg</td>
</tr>
<tr>
<td>Sensing</td>
<td>6-IR scanning range array (range up to ~1.5 m, can see objects at least 21 cm tall)</td>
</tr>
<tr>
<td></td>
<td>CMUcam3 color camera [6] (352x288, 1 fps, 60° FOV): 60 MHz ARM, 64 KB RAM, 128 KB Flash</td>
</tr>
<tr>
<td></td>
<td>Frontal bump detection using two bump sensors and spring-loaded rail</td>
</tr>
<tr>
<td>Processing</td>
<td>Gumstix verdex XL6P motherboard [3] + Robostix interface [8]: 600 MHz, 128 MB RAM, 32 MB Flash</td>
</tr>
<tr>
<td></td>
<td>Two Java Sun SPOTs [9]: 180 MHz, 512 KB RAM, 4 MB Flash, 802.15.4 integrated radio</td>
</tr>
<tr>
<td>Power</td>
<td>6 AH Li-Ion battery pack + custom power circuit</td>
</tr>
</tbody>
</table>

**Behavior Layer**

Several reactive and deliberative layer behaviors are utilized in these system states for safe navigation. Each behavior is combined using its weighted drive vector. For avoidance behaviors, the resulting drive vectors point away from the obstacles to slow down and turn the robot if necessary. A deliberative behavior, for example, may produce a drive vector that strongly points in the direction of the next exploration waypoint. These behaviors combine in situations where a combination of reactive and deliberative behaviors is required (e.g., reactive waypoint navigation). The system uses the following weighted reactive behaviors:

1. **AvoidObstacle**: Reactively navigate about obstacles sensed by the IR array. Weight decays as robot gets closer to a waypoint for increased accuracy.
2. **AttractObstacle**: Stay near obstacles sensed by the IR array. Given the robots’ limited sensing range, this provides more data points for localization.
3. **AvoidPast**: Favor making progress away from areas that have already been explored, especially the most recent. This is accomplished by maintaining a grid of the environment and visit counts for each cell [11].
4. **BumperBehavior**: Back up and turn 60° when a collision is detected. This helps avoid obstacles shorter than the 21-cm IR array viewing height.

**Planning and Coordination Layers**

The planning layer implements the deliberative side of the navigation system. It maintains a sequence of goals that can be provided to perform coordination and exploration, such as from the System Commander. For each goal, the planning layer generates a series of waypoints using a DP-SLAM occupancy grid and wavefront propagation path-planning. These waypoints are generated to enable the robot to quickly and safely navigate to goals according to a number of constraints, including total distance and terrain features. Additionally, these waypoints serve as the pseudo-sensor input into the reactive navigation system through the waypoint navigation behavior. There are two main deliberative behaviors used by the system, which can be combined with reactive behaviors for a more sophisticated navigation system:

1. **DriveToGoal**: Combined with reactive behaviors, favor driving to a goal coordinate (waypoint) for region exploration. Goal is considered reached if within 30 cm.
2. **PathPlan**: A probable path is planned between the current position and a goal point (or the lander), using the 4-way wavefront propagation method [12] on a DP-SLAM occupancy grid map.

The coordination layer implements both the potential fields and subsumption architectures for combining the reactive and deliberative navigation systems. As in [10], a finite state machine is implemented in which each state operates as a conventional potential fields navigation system, combining the behaviors according to predefined weights. The subsumption architecture is implemented through a system of monitors, responding to environmental events or the robot, that enable the transition from state to state and, therefore, for different weight-based behavior combinations.

**6. LOCALIZATION AND MAPPING ARCHITECTURE**

A primary goal for the exploration robots is to produce a coarse map for a remote science team, or potentially for a more complex and costly science platform before it descends into an unknown environment. The exploration robots are built under the assumption that they could incur damage or fail, and thus several robots are needed. Cost factor becomes a major issue in the design and deployment of these robots. A custom array of short-range infrared range sensors (constructed for $75 in parts) was chosen instead of a high-cost laser range-finder for the robots.

**Sun SPOT IMU**

A custom 6-axis IMU was developed using a free range Java Sun SPOT, containing 3-axis accelerometers, as a base. A Sparkfun 3-axis gyroscope system (300 deg/sec) was integrated onto this unit. Kalman filters are applied to accelerometer and gyroscope readings to reduce their noise. However, the accelerometers onboard the Sun SPOTs are noisy and were found to reduce localization accuracy. Thus, they were disabled for the experiments. Additionally, two 90-tick optical shaft encoders with quadrature were utilized for odometry. A technique called Gyrodometry [13] was employed to increase localization accuracy, by using the...
gyrosopes’ theta only when the difference between it and the odometry’s theta is larger than a predefined threshold. As the robots navigate and skid-steer in a high-slippage environment, this proved to be necessary. The Sun SPOT unit keeps track of localization based on the 3-axis gyroscope and odometry readings at a rate of 40 Hz. It also receives synchronizing pose updates from DP-SLAM.

**SLAM Algorithm**

The DP-SLAM 2.0 [4] algorithm was used to build maps on the exploration robots. SLAM is a technique focused on building a map of the environment and at the same time keeping track of the location of the robot. FastSLAM [14] and DP-SLAM are popular algorithms, and both were considered. These algorithms utilize Rao-Blackwellized particle filter variants. In an unstructured environment (especially on a planetary surface), landmarks of significant size are usually rare in a large and potentially sparse field. DP-SLAM, on the other hand, does not rely on landmark identification, and uses a particle filter to represent both robot poses and possible map configurations.

There are two major issues in using DP-SLAM on the exploration robots: (1) the assumption of using an extremely accurate laser range-finder, and (2) computation power of the processing unit. IR sensors are relatively slow, less accurate in measuring distance, and exhibit a much larger variance in readings [15] compared to commercially available laser range-finders. The Sharp IR sensors on the exploration robots have a stabilization plus acquisition time of approx. 32 ms, and a maximum sensing range of 1.5 m. Acquiring 1° readings in the frontal 180° FOV takes in excess of 6 seconds (including servo movement), and it would be impractical to stop the robot for every scan. To speed up acquisition time, 6 IR sensors were vertically mounted 30° apart on a servo-controlled turret. Readings are taken every 3° such that there are a total of 60 readings over the 180° FOV. Because these sensors can interfere with each other, the 30° mounting offset was experimentally determined to produce acceptable performance. One sweeping scan takes approx. 700 ms as opposed to <100 ms by some laser range-finders. Although the robot is moving while a scan is performed, the robot pose is within acceptable error since the robot moves with low speed. The IR sensors are 21 cm above the surface. Obstacles taller than this are generally observable to the sensors.

DP-SLAM (and SLAM in general) assumes a fast computer is available to keep up with scanning speed, which was another issue. The Gumstix motherboard on the robots is not able to keep up with every scan. As continuous scans are required by the navigation system for obstacle avoidance, SLAM uses the latest complete scan to update the map. The map produced by DP-SLAM is an occupancy grid with each cell representing a 5 x 5 cm square space. The cell size was chosen based on observed IR distance variance. Experimentally, running DP-SLAM alone on the Gumstix with 15 particles took on average 3 seconds per update, while using 25 particles took more than 10 seconds per update. Thus, use of 15 particles as found to provide the best time-to-result quality ratio. Lastly, the Gumstix processor does not possess a hardware floating point unit, so DP-SLAM was modified to use fixed-point math.

**Communication Architecture**

A component-based message passing system following the Joint Architecture for Unmanned Systems (JAUS) specifications [16] was implemented for the communication architecture. JAUS provides a wrapper around much of the low-level logic at the component level, including sensor input and motor control, path finding and navigation, localization and mapping, and communication. The lander represents the System Commander as it dictates sub-region assignment and coordinates robots around the lander platform. Similarly, each robot represents a complete subsystem and is identified by a unique ID. Each platform is further separated into nodes and components within each node. This allows passing of intra-robot messages (using serial and I2C lines) and inter-robot messages (via Sun SPOT 802.15.4 wireless radio) within the team and lander for varying levels of communication and coordination.

### 7. RESULTS FROM A PLANETARY STYLE ARENA

Several local tests were performed to calibrate odometry and SLAM algorithm parameters, as well as evaluate performance. The results of the final robot’s odometry and IR scanning array are shown in Figure 2 next to the local test sandbox in which the data were gathered. Due to sensor noise and changes in robot pitch, some ranges varied around obstacles as the robot looped around the environment. The similarities between the actual and mapped environment demonstrate that the low-cost system’s gyro-corrected odometry and IR ranging arrays work quite well.

We used this system to participate in the 2008 ICRA Space Robotics Challenge held in Pasadena, CA in May 2008 [17]. The mapping and exploration team of four robots was the only entry consisting of multiple mobile robots, and also the only approach focusing on low-cost and limited computation for autonomously mapping the environment. The planetary landscape was a 6 x 6 m square covered with 6.5 mm pea gravel. It also contained rocks of varying shape, color, and size, and was made with an uneven surface to make mapping more challenging. There were a simulated lander and a ramp which the robots must safely descend and perform one or more tasks. Figure 3 shows two images of the robots in the arena. Figure 4 shows arena illustrations with each robot’s path and map beneath it. Black patches in the arena illustrations denote observable obstacles, whereas grey patches are unobservable obstacles which were mostly flat rocks. The panels bounding the arena were not high enough for the robots to detect with their scanning IR arrays. To demonstrate robustness of the multi-robot team, one of the
robots was removed to simulate failure. Thus, three robots took on the task of coordinated descent onto the surface and to distributively explore and map the unknown environment.

The robots successfully descended the ramp and were allowed three minutes to explore and map. Exploration robots #1 and #2 followed a similar path. In the lower-right corner of the arena, bump sensors were frequently triggered due to the robots not being able to perceive the arena walls. This caused them to turn in place multiple times, creating additional error in odometry-derived robot poses. Because the robots were in close proximity at one point, the beam of one IR sensor could be received by another, producing additional interference and variance. Robot #3, as shown by its path, briefly became disoriented on the right side of the ramp, which caused its odometry to become corrupted. However, as it approached the lower-right portion of the arena, the SLAM algorithm corrected part of its map. This location of the arena was where all robots ended their task and their maps retrieved for analysis. Robots #1 and #2 obtained the most accurate depictions of the environment. For these results, the robots received two awards in the Onto the Surface and Map the Environment events.

8. CONCLUSIONS

Overall, this work has shown that SLAM-capable mobile robots can be successfully constructed without expensive sensing and computing equipment. The system that has been developed is scalable and extensible, with the JAUS implementation making it easy to swap or add components and robots to the system. We have also demonstrated that such a multi-robot system is robust. Due to the use of low-cost sensors and limited computing power, there are some limitations to the current system. Noisy data from sensors represents the tradeoff of accuracy and expense that is expected of such systems. Also, the 802.15.4 protocol on the Sun SPOTs is limited in distance, forcing the robots to remain geographically close for communication. The limited distance that the current IR sensors can provide also causes the system to suffer in sparse environments, as obstacles must be seen to act as reflectors for the SLAM algorithm. To relieve this, we are developing a dual-range scanning IR array that incorporates both long- and short-range sensors to improve localization. We plan to scale the system up by building more robots, investigating more low-cost sensing systems, and introducing more sophisticated multi-robot coordination. Use of more formal science sensors and map sharing represent directions of future work for this system.

REFERENCES


Figure 2 - Test sandbox (left); the odometry, range, and robot path (middle); and its corresponding DP-SLAM map (right). The robot’s path is marked as a series of circles, starting at the bottom. Range scan data are marked with stars (*).

Figure 3 - Images from the 2008 ICRA Space Robotics Challenge: robots leaving lander (left), and mapping the arena (right).

Figure 4 - Paths and corresponding maps that each robot developed during exploration using online DP-SLAM and IR array.