Coordinating a Team of Robots for Urban Reconnaisance

Pradeep Ranganathan, Ryan Morton, Andrew Richardson,
Johannes Strom, Robert Goeddel, Mihai Bulic and Edwin Olson
University of Michigan – Ann Arbor

ABSTRACT

Gathering intelligence about a potentially hostile environment is a critical capability for war fighters. Using a team of robots for this task is an attractive option because it allows data to be gathered quickly while simultaneously removing humans from danger.

However, fielding a team of robots requires three key problems to be solved: first, the human commander must be able to efficiently interact with the robots and issue orders that they can understand; second, the system must be able to decompose orders into tasks for individual robots; and third, the robots must have the perceptual capabilities needed to operate for extended periods of time without operator assistance.

In this paper, we present our solutions to these problems, and demonstrate our approaches using our test-bed which allows a single operator to control fifteen robots.

1. Introduction

Multi-robot teams provide the ability to complete a task much more efficiently than an individual robot. However the task of controlling this team should not overwhelm the human operator.

The perceptual capabilities of the individual robotic vehicles allow unsupervised operation to a certain extent. Even in such an autonomous system, the human operator should be able to specify preferences or even take manual control of a robot. This is required to counter situations that the system was not originally designed.

In order to enable this, the system should provide the human operator with a set of options without inundating him with micro-management.

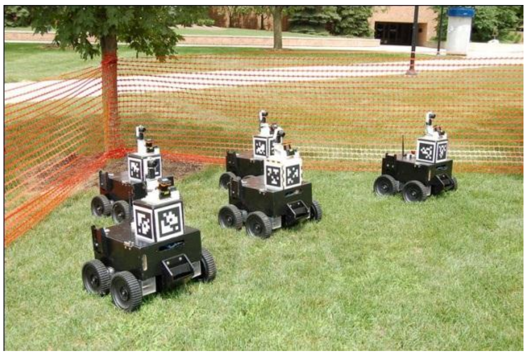

Figure 1: A coordinating team of robots. Our test bed consists of a team of fifteen such robots.

Behavorial autonomy is also critical for the human operator to productively interact with the team of robots. For example, robots should automatically stay away from dangerous objects detected by one member of the team, and slow down when going over ramps. Thus a dynamic set of behaviors enables more productive interaction.

A key challenge in coordinating a multi-robot system is building a coherent model of the world. This is non-trivial since each robot operates in a separate coordinate frame. Establishing a unified coordinate frame is critical for sharing data between robots. It also enables the human operator to make better sense of the information from different robots.

The salient contributions of this work in addressing the above challenges are:

1. Algorithms for autonomous path planning and localization, even in GPS denied environments.
2. Methods that employ machine learning for robust, real-time terrain classification, object recognition and tracking from sensor data.
3. A ground control system for efficiently coordinating a multi-robot team.
4. Algorithms for robust multi-robot localization.
5. A rewards-based modular planning framework that performs high-level mission planning.

The capabilities of our multi-robot team are primarily motivated by the MAGIC 2010 robotics challenge (1). This challenge requires competitors to demonstrate the use of multi-vehicle robotic teams that can execute an intelligence, surveillance and reconnaissance mission in a dynamic urban environment.

In the next section we will review previous work. In sections 3, 4, 5 and 6 we will describe our system. In section 7 we present an evaluation of the system.

2. Previous Work

Existing work on using robotic teams for reconnaissance include Konolige et. al. (2). This work describes a large scale system that coordinates one hundred robots to survey indoor environments and detect objects of interest. The Swarm-Bots project (3) studies coordination amongst small self-assembling, self-organizing metamorphic robot systems.

There has also been work on coordinating heterogeneous teams of robots to complete a given task. Chaimowicz and Kumar (4) describe the coordination of unmanned aerial vehicles with unmanned ground vehicles.

There has also been theoretical work complementing systems research into multi-robot systems. The work of Kolling and Carpin (5) address the theoretical aspects of multi-robot coordination. Klavins (6) provides a theoretical analysis of the complexity of several multi-robot communication schemes.

Previous approaches to multi-robot localization include Huang et. al. (7). This work describes a framework based on the extended Kalman filter (EKF) for building terrain maps from multi-robot teams. Fox et. al. describe a probabilistic approach for multi-robot localization in (8). These works present an alternate approach to the constraint based map-alignment method used in our work, which is based on (9).

Much of our sensor processing is based on lessons and technology from the MIT entry into the DARPA Urban Challenge (10). Montemerlo et. al. (11) describes Stanford’s entry into the same challenge.

User interface modelling for multi-robot systems is described in a previous work by Wagner et. al (12).

3. High-level planning

In many of today's systems, a human operator controls a single robot, micro-managing every action. This micro-management becomes impossible with more robots: in order to deploy a team of robots, the robots must be largely autonomous, with the human operator intervening only when necessary.

Our high-level planning system has two goals:

1. Explore the map efficiently.
2. Send specialized "disruptor" robots to neutralize dangerous targets found by other robots.

The system must achieve these goals efficiently, taking the minimum amount of time possible and while minimizing the operator workload.

To accomplish this, our autonomous planning system consists of several specialized planners. For example, the exploration planner is responsible for sending robots out to build a map of the world. The neutralization planner assigns a robot to neutralize hostile targets that the human or exploration planner has found. Additional planners are possible: a communications planner might be responsible for positioning communications-relay robots to maintain a stable communications network.

In our system, the human commander grants control authority for specific robots to various planners, allowing him to allocate resources according to mission priorities.

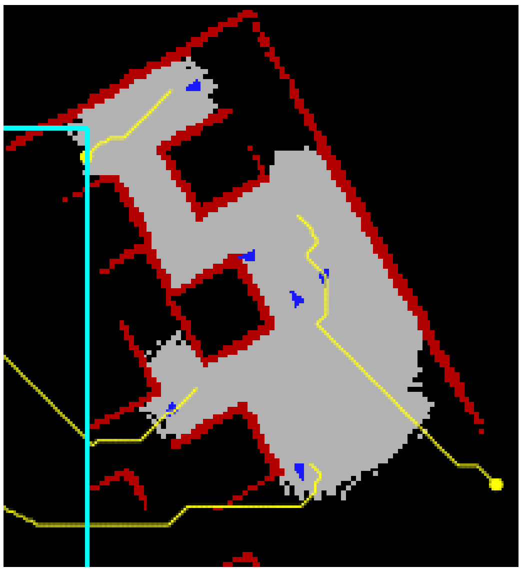
Figure 2: Exploration planning. The robots are spread across the map to avoid duplicating effort.

Exploration planner

The exploration planner is used to expand the map of the world. LIDAR data provides knowledge about the open space, obstacles and unknown regions around each robot. The exploration planner marks the regions on the map as explored and unexplored based on the sensor range of each robot. It marks the cells separating explored and unexplored regions as belonging to the *frontier*.

Our system employs a greedy rewards-based scheme to assign robots an area to explore on the frontier. Factors such as distance to the frontier and likelihood of crossing another robot's path are taken into account to ensure that goal assignments spread the robots evenly throughout the map (see Figure 2).

The human operator may further specify a region for the robots to focus their efforts on, confining their search space to the frontier contained within that region. Upon completely clearing a region, the planner notifies the operator, who may then proceed to assign a new region to explore.

Neutralization Planner

The neutralization planner is responsible for assigning disruptor robots to neutralize hostile targets discovered by the exploration planner. The planner seeks to space disruptors evenly throughout the explored portions of the map so that response time to disruption requests is minimized. When a request is issued, the neutralization planner guides a disruptor robot through explored territory, to the hostile target, and alerts the human operator that a robot is ready to neutralize a target. This ensures that no neutralization occurs without human confirmation.

4. Ground control station

The ground control station (GCS) is the center of operations for controlling the team of robots. Because our system requires a mixture of autonomous and human directed-actions, the GCS is crucial for coordinating and validating the actions of the robots with the wishes of the human operators.

The GCS encompasses both the interface through which the operators interact with the system, as well as the technology infrastructure which enables the recombination of sensor data from the entire team of robots. Our system addresses fundamental challenges in Human Computer Interfaces (HCI), multi-robot mapping, large-scale dynamic communication systems, and machine learning.

Robot-Operator Interface

The human operator requires the ability to control the multi-robot system at various levels of granularity. For example, the

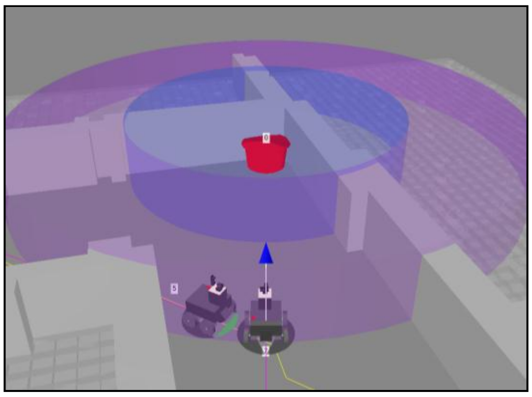
Figure 3: Notification of a dangerous object detection on the ground control interface. The activation region of the dangerous object is also clearly marked for better context.

operator may wish to modify the behavior of the robot team at a strategic level, an individual robot (e.g. assign a specific robot to explore a new area), or at sub-component level (e.g use vision algorithm X instead of Y because of harsh lighting conditions). Again, at each of these levels, there are many possible ways in which an operator might modify the desired behaviour. Because of this versatility of the robot team, there are a large number of possible ways to interact with the system. Explicitly presenting all these options to the operator at once would be overwhelming and inefficient. Instead, the robot-operator interface (ROI) needs to understand the context of a given situation, and provide the user with an easy method to select among the “best” actions for that situation.

In our system, each robot is aware of the command context in which it is operating. By exploiting this information on the ground station, our control interface presents the human with the appropriate palette of instructions which can be used to modify the robot's behavior. For example, if a robot is exploring autonomously, the interface allows the user to quickly modify the robot's destination. However, more involved tasks, such as tracking a hostile target first requires a switch of context.

To help the operator maintain situational awareness, our system provides notifications when unusual or important events such as object detections, depleted batteries or other faults occur. This monitoring and notification mechanism (see Figure 3) removes the need for frequent context switches and thus increases the number of robots an operator can manage at the same time.

The interface also provides full situational awareness to the human operator. For example, when a particular robot is selected, the human operator can see telemetry from the robot, its position on the estimated map and information about its internal state (See Figure 4). This helps the operator resolve situations that require multiple sources of information.

Our control interface also doubles as an visualization application for quickly interpreting the sensor data from several robots. A three-dimensional rending of the surrounding of each relevant robot is displayed to the operator, enabling the operator to see a robots-perspective panorama (see Figure 5). An overhead real-time structural map generated from the robots 3D laser scanner is also shown.

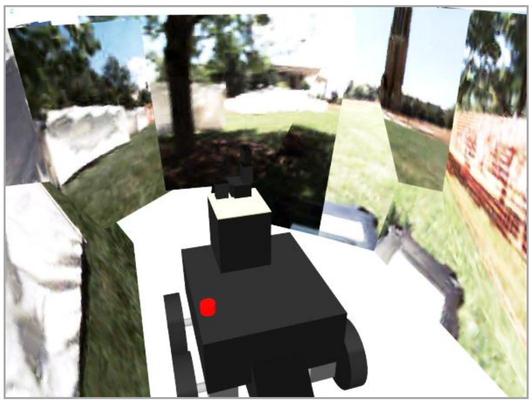
Figure 5: 3D Visualization of telemetry data. Images from various directions around the robot are assembled as a panorama to provide better context about the location of the robot to the operator.

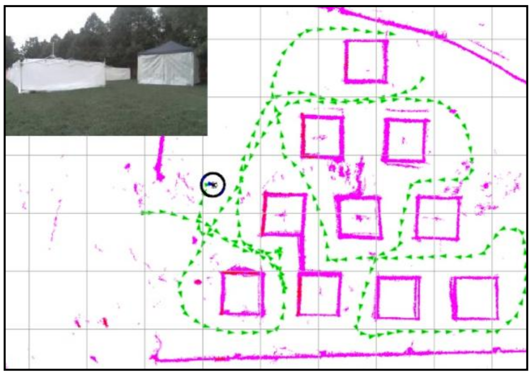
Figure 4: Situational awareness in the UI. The current location of each robot is shown on a map built from the robot's sensor data.

Multi-robot mapping

In order to facilitate a coherent situational awareness of the robot team's surroundings, sensor data from all the robots must be combined into a unified reference frame before it can be used effectively by either our autonomous task allocation system, or the human operators. Failure to maintain a common reference frame means robots are not able to share information about areas that have already been explored, and reduces the efficiency of the human operator who must switch contexts often.

However, due to wheel slippage and sensor imperfections, the robots are unable to know exactly where they are and how far they have moved. As a result, their estimated positions with respect to one another are likely to drift over time causing their common reference frame to deteriorate.

Global Positioning System (GPS) is not a solution: not only is the accuracy (3-5m) insufficient for these missions, GPS is also unreliable indoors and can be disrupted outdoors by tall buildings or even jammed.

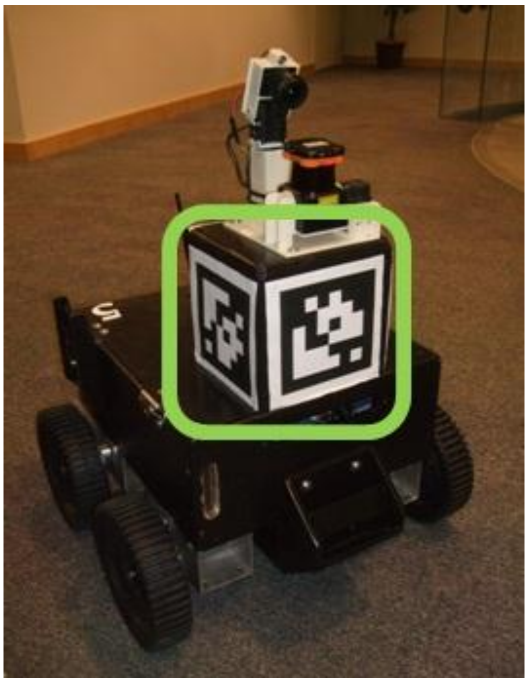

Figure 7: Robot tags. Visual fiducial markers are placed on each robot so that one robot can identify another uniquely. The ground station can build a global map from individual robot maps, when robots can place themselves relative to each other.

Because of this inherent position uncertainty, a successful implementation of a coherent multi-robot map requires the use of Simultaneous Localization and Mapping (SLAM) to robustly compute the relative positions of each robot.

Our robots rely on their laser range finders to map the world and determine their positions relative to one another. Using sensor data, we can build a consistent map by aligning every robot's internal maps with the internal maps of others. By correctly aligning these "neighborhood" maps with respect to one another, we can construct a global map which encompasses the entire area traversed by the team of robots.

However, the fusion process is non-trivial because each robot does not know its own position accurately. To ensure correct alignment, we use two separate mechanisms to match the internal maps: inter-robot "tag" observations, and map-to-map alignments.

The first alignment mechanism computes positional constraints when a robot autonomously detects a teammate. Each robot is equipped with a custom-designed fiducial marker (see Figure 7). These markers allow other robots to reconstruct the full 3D position of the robot when visually observed. These events constitute position constraints in the global mapping solution, enabling the operators to get a relative "fix" on the location of two robots. Since the robots can be observed often, the location of all the robots can quickly be determined by computing the maximum likelihood position of all the robots based on the given constraints, as described in (9).

5. Robot construction

Our team consists of fifteen robots whose design was specifically tailored for the MAGIC 2010 competition. Each robot needs to be capable of navigating in an indoor/outdoor environment, climb ramps and go over 10cm curbs.

Each robot is laser-cut out of wood and

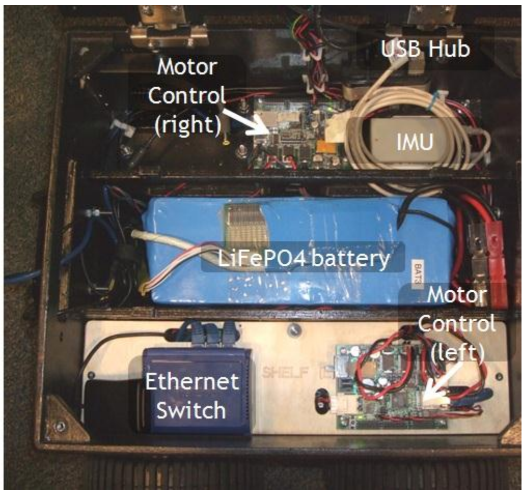

Figure 6: Internal components of our robotic ground vehicle

then glued. It has a laptop, two microcontrollers, a power distribution center, various communications buses, and sensors.

Electrical system

The electrical system on each robots centers around a 24V LiFePO4 battery. This battery provides about 4 hours of runtime. After routing through a fuse box, DC-DC converters produce the additional voltages for the various components within the robot.

Mechanical System

Chassis

Each of the 15 robot chassis consist of 9mm Baltic Birch plywood that has been lasercut, glued, sanded, and painted. We chose wood due to its ability to be easily and accurately cut using a CAD-driven laser-cutter. This manufacturing method also allows multiple design iterations at low cost.

Drive Train

Our robots employ a four wheel skid-steer drive system. Thus, like a tank, wheels must skid on the ground in order to allow the robot to turn.

Our wheels are driven along a gearway built into the inner rim of the tyre, with the weight of the robot itself being supported on a passive steel axle (see *Figure 9*). We built a custom shock isolation system with torsion bars to allow our robots to traverse on semi-rugged terrain while minimizing vibrations that can interfere with our sensors.

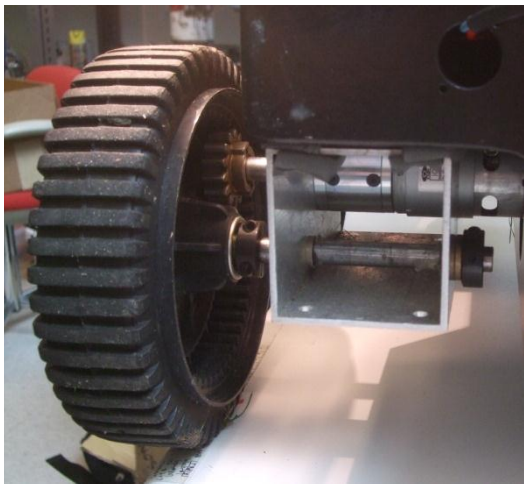

Figure 8: The drive train of our robotic ground vehicle

Sensors

The primary sensor on our robot is a Hokuyo UTM 30LX laser range-finder. This planar sensor returns range measurements at 0.25 degree increments over a 270 degree field-of-view. We actuate the sensor with a Dynamixel AX-12 servo in order to produce a 3D point-cloud. This 3D data allows the robot to determine safe traverseable regions.

Additionally, each robot has a PointGray FireflyMV USB camera with a 2.8mm fixed focal length lens giving us about a 90^o field of view. The camera is actuated via two additional AX-12 servos, to allow images to be taken in any direction relative to the robot.

The extrinsic parameters of the the laser range-finder and camera are well-calibrated due being mounted together on a unified sensor mount. This mount was custom made in ABS plastic using a uPrint rapid prototyping 3D printer.

The robot also contains sensors for measuring inertial frame changes. The rear wheels of the robot contain encoders that report the angular distance that each motor has rotated. Additionally, we have a custom on-board Inertial Measurement Unit (IMU) that measures angular changes. This 6-DOF IMU contains 3-axis gyros and 3-axis accelerometers.

Low-level Control

The low-level control of our robots is managed by a pair of ORC boards (13). These open-source embedded microcontroller boards allow the laptop to offload the low-level control, so the laptops can focus computation on the sensing and planning. The ORC boards control all four motors, encoders and the AX-12 servos. The ORC boards also control the disruptor laser that is used to designate neutralization targets.

Computation

The primary computer aboard each robot is a dual-core 2.54 Ghz Lenovo Arrandale laptop with 4GB RAM and solid-state drives. The laptops run Ubuntu Linux 10.4. With the exception of a few device drivers, all the software has been written in Java.

Communications

The major components of the robot, such as the laptop, the micro-controllers and the mesh-networking device (Open-Mesh OM1P) are connected via ethernet. Sensors are connected via USB.

Each robot has a 900 Mhz radio that allows for command and control data between the robot and the ground station. It is capable of providing a bandwidth of 115.2 Kbps.

When robots are within range of each other they can transmit data over the 802.11 mesh network at much higher bitrates. This opportunistic network offers no critical transmissions, but allows for improvements when such communications are made possible.

6. Robot Software

Architecture

Our robot software is internally decomposed into self-contained modules

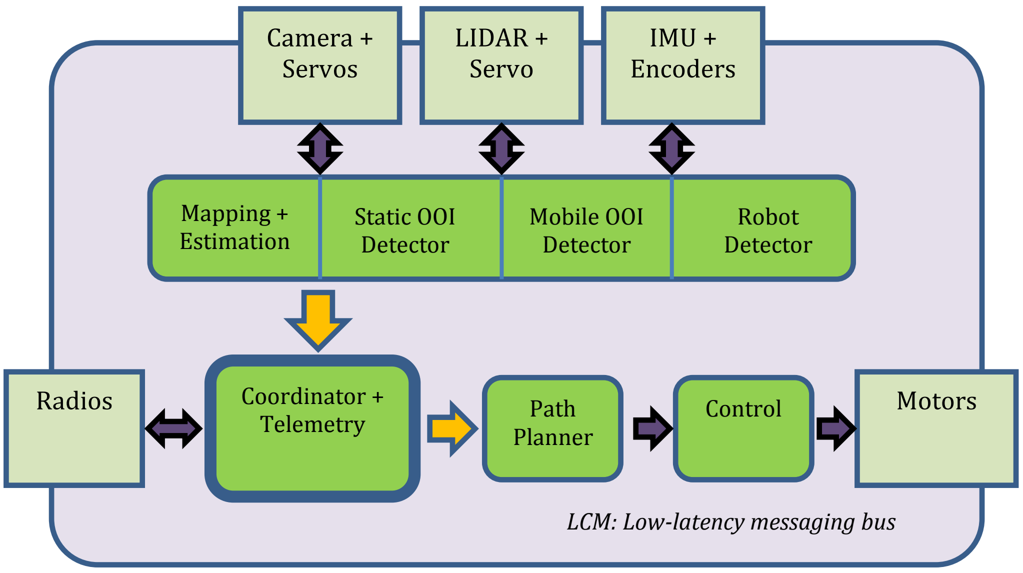

Figure 9: High level architecture of software modules on robotic ground vehicle

that communicate by exchanging messages over the Light-weight Communications and Marshalling (LCM) system. LCM (14) is a high-bandwidth, low-overhead, type-safe, multi-cast based message passing system. It was designed specifically for robotics applications and has successfully been used in other previous robotic systems (10).

One non-trivial aspect of data acquisition is time synchronization between different sensors and the processing unit. Each sensor has a different internal clock. Hence the timestamps on sensor data can vary over time because the internal sensor clocks can drift. To address this problem, our system uses a passive synchronization method described in (15).

Our software system also contains a *ProcMan*, a process that is responsible for managing other processes. The *ProcMan* process monitors other processes, restarts processes that terminated and provides a unified logging framework for other processes.

Software Modules

Separate modules are responsible for handling navigation, mapping, estimation, and perception on our robots.

Navigation module

Each robot in our system is responsible for individual path planning within its own sensory horizon, while ground control provides waypoints for longer traverses. Thus each robot plans locally within a distance radius of approximately 20m to find the best path to the goal specified by the high level planner. It also ensures safe obstacle avoidance while navigating this path.

The navigation module first builds a discrete terrain map from the LIDAR points and assigns costs to each terrain cell. Costs are assigned such that objects have high costs and navigable terrain has low cost. The shortest path to the goal is then computed using a wavefront approach.

To ensure smooth driving over the lowest-cost path to the goal, this path is filtered by an iterative algorithm which moves a given point on the path nearer to the line connecting the points preceding and following it. This path smoothing is cost-sensitive and will never increase the path cost when attempting to remove sharp turns.

Mapping and Estimation module

Each robot maps the area it has traversed and estimates its position within this local map. This task is performed by the mapping and estimation module using a SLAM solution. This module uses scan matching on LIDAR data to improve on the intertial frame changes.

Global maps for coordination and information sharing are constructed on the ground station, using map alignment criteria and robot-to-robot detections as described in Section 4: Multi-robot mapping.

Perception module

The perception module handles the task of identifying hostile objects and tracking them when necessary. It relies on the color and shape information from the cross-calibrated camera and LIDAR sensors for this purpose. After preliminary filtering of extracted objects based on shape characteristics, the dimensions of extracted objects are computed by bounding the object with a (non-axis aligned) bounding rectangle.

A discriminative model was learned using the shape, dimension and color information extracted. This discriminative model is then used by the perception module to identify objects of interest in real-time.

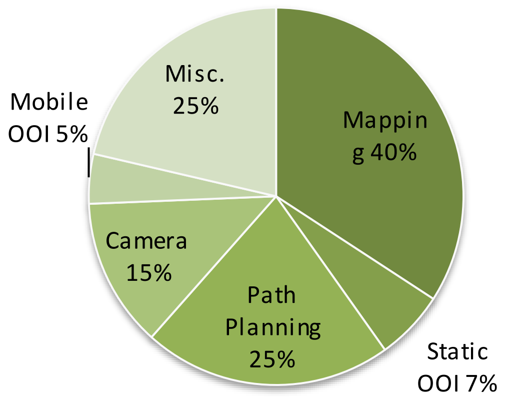

Figure 11: CPU utilization. This pie chart shows the amount of computation required by each of the software modules

7. Evaluation

Figure 12 shows the computational requirements for different software modules. Much of the computational effort goes into path planning, mapping and perception. These are the modules that contribute much of the autonomy of the system.

Figure 11 compares the commanded target traversal lengths and the traversal distances achieved by the autonomous navigation systems on the robotic ground vehicle. The key observation is that the robotic vehicles are often times able to execute long traversals on their own to a target location. This autonomy is critical for allowing a single operator to control a large team of robots.

8. Conclusion

In this paper we have described system for

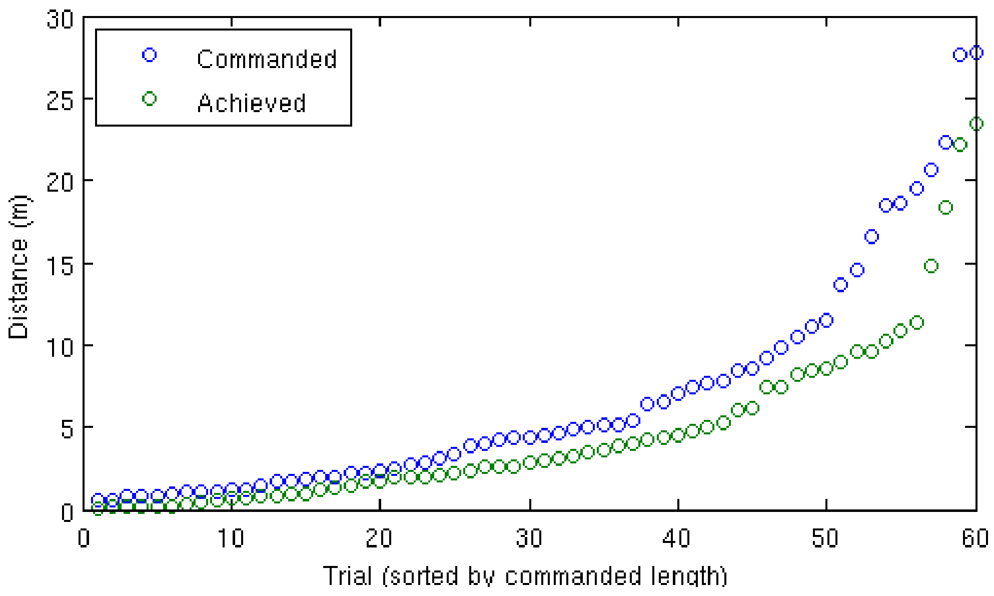

Figure 10: Commanded vs Achieved traversal lengths. The plot compares the traversal lengths.

coordinating a team of autonomous robots for autonomous reconnaissance. The system includes an autonomous high-level planner and a streamlined user interface for controlling the team of robot.

Our system is built over a custom low-cost robotics platform which can carry a flexible payload. Our robotic vehicles also exhibit extensive autonomy through the use of robust sensor technology and perception software.

9. Bibliography

1. MAGIC 2010: Super-smart robots wanted for international challenge. *Australian government, Department of Defence.* [Online] [Cited: 10 September 2010.] http://www.dsto.defence.gov.au.

2. *Centibots: Very Large Scale Distributed Robotic Teams.* Konolige, Kurt and et al. 2006, Springer Tracts in Advanced Robotics, pp. 131-140.

3. *The SWARM-BOTS project.* Dorigo, Marco and et al. 2005, Swarm Robotics, Lecture notes in Computer Science, pp. 31-44.

4. *Aerial Shepherds: Coordination among UAVs and Swarms of Robots.* Chaimowicz, Luiz and Kumar, Vijay. 2007, Distributed Autonomous Robotic Systems, pp. 243-252.

5. *Muti-robot surveillance: An improved algorithm for the GRAPH-CLEAR problem.* Kolling, A. and Carpin, S. 2008, Robotics and Automation, pp. 2360-2365.

6. *Communication Complexity of Multi-robot systems.* Klavins, Eric. s.l. : Springer, 2003, Algorithmic Foundations of Robotics, Vol. V, pp. 275-292.

7. *On the Consistency of Multi-robot Cooperative Localization.* Huang, Guoquan and et al. Robotics Science And Systems : s.n., 2009, Vol. V.

8. *A Probabilistic Approach to Collaborative Multi-Robot Localization.* Fox, Dieter and et al. Autonomous Robots, Vol. 8, pp. 325-344.

9. *Learning Globally Consistent Maps by Relaxation.* Duckett, T., S.Marsland and J.Shapiro. s.l. : Proceeding of the IEEE International Conference on Robotics and Automation (ICRA), 2000.

10. *A Perception Driven Autonomous Urban Vehicle.* Leonard, J. and et al. 2008, Journal of Field Robotics.

11. *Junior: The Stanford entry in the Urban Challenge.* Montemerlo, et al., et al. 2008, Journal of Field Robotics, pp. 567-597.

12. *Muti-robot User Interface Modeling.* Wagner, Alan and et al. s.l. : Springer, 2006, Distributed Autonomous Robotic Systems, Vol. 7, pp. 237-248.

13. OrcBoard. *OrcBoard.* [Online] www.orcboard.org.

14. *LCM: Lightweight Communications and Marshalling.* Huang, Albert, Olson, Edwin and Moore, David. s.l. : Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010.

15. *A Passive Solution to the Sensor Synchronization Problem.* Olson, Edwin. s.l. : Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010.