GRAMMPS represents a milestone in demonstrated cooperative robotic systems, as the first fully-integrated autonomous multi-robot system operated in natural terrain. The mission grammar supports a wide variety of missions, providing optimal execution of those missions by teams of robots, with graceful degradation of optimality for computationally intractable missions. Furthermore, GRAMMPS supports heterogeneous robots as specified via the mission statement as well as by incorporation robot speed estimates into the planning process. Finally, GRAMMPS is general, capable of functioning with a general class of local navigation systems.

It is the intent of future work to increase the capabilities of GRAMMPS by improving the robustness of the system against partial failures, and by extending the class of missions which can be executed. An alternative world representation for dynamic planning (a Framed-Quadtree) which has performed well in initial testing is likely to be fully integrated, providing significant memory savings. The mission planning component of GRAMMPS is centralized in nature, running only on one of the robots involved in the mission. It would be advantageous to enable decentralization of this component to improve system reliability and parallelize computational requirements. The mission grammar could be extended to support cooperative foraging, the meeting of robots at a central location, and the dependency of later parts of the mission on earlier choices. Finally, a randomized search algorithm could be utilized on the entire mission statement, rather than just on components thereof, permitting near-optimal execution of increasingly complex tasks.

9. Acknowledgments

The authors would like to thank the CMU UGV group for their support of live-vehicle demonstrations and testing. In particular, Jim Frazier's tireless efforts at maintaining the vehicles and Dr. Martial Hebert's ongoing research in perception and local navigation have been critical to the success of this work.

This research was sponsored by DARPA, under contract number DAAE07-96-C-X075, entitled "Technology Enhancements for Unmanned Ground Vehicles", monitored by U.S. TACOM.

10. References

[1]Alami, R, et al. "A General Framework for Multi-Robot Cooperation and its Implementation on a Set of Three Hilare Robots." Experimental Robotics IV, 1995.

[2]Allen, J., et al. *Readings in Planning*, Morgan Kauffman Publishers, San Mateo, California, 1990.

[3]Arora, S. "Polynomial-time Approximation Schemes for Euclidean TSP and other Geometric Problems," Proc. IEEE FOCS, 1996, pp. 2-13.

[4]Azarm, K., Schmidt, G. "A Decentralized Approach for the Conflict-Free Motion of Multiple Mobile Robots," Proc. IROS 1996, pp. 1667-1675.

[5]Brooks, R.A. "A Robust, Layered Control System for a Mobile Robot," IEEE Journal of Robotics and Automation, April, 1986, pp. 14-23.

[6]Brumitt, B.L., Hebert, M., et al. "Experiments in Autonomous Driving with Concurrent Goals and Multiple Vehicles," Submitted, Proc. ICRA, 1998.

[7]Brumitt, B.L. Stentz, A. "Dynamic Mission Planning for Multiple Mobile Robots," Proc. ICRA, 1996.

[8]Cao, U.Y., et al. "Cooperative Mobile Robotics: Antecedents and Directions," Autonomous Robots 4,1997.

[9]Causse, O., Pampagnin, L.H. "Management of a Mutli-robot System in a Public Environment," Proc. IROS, 1995.

[10]Erdmann, M., Lozano-Perez, T. "On Multiple Moving Objects. Algorithmica," No. 2, 1987, pp. 477-521.

[11]Feldman, J.A., Sproull, R.F. "Decision Theory and Artificial Intelligence II: The Hungry Monkey," *Cognitive Science 1*, pp.158-192, 1977.

[12]Fikes, R.E., Nilsson, N.J. "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving," *Artificial Intelligence*, Vol. 2., pp.199-208, 1971.

[13]Kirkpatrick, C.D., et al "Optimization by Simulated Annealing," Science, Vol. 220, Num 4598, May 1983, pp. 671-220.

[14]Latombe, J.C. *Robot Motion Planning*, Kluwer Academic Publishers, 1991.

[15]Lawler, E.L., et al. *The Travelling Salesman Problem*, John Wiley & Sons, 1985.

[16]Le Pape, C. "A Combination of Centralized and Distributed Methods for Multi-Agent Planning and Scheduling," Proc. ICRA 1990, pp. 488-493.

[17]Li, S. "Coordinating Multiple Mobile Robots Through Local Inter-Robot Communication," *Distributed Autonomous Robotic Systems*, Springer-Verlag, 1994.

[18]Lin S. Bell Systems Technical Journal, 1965, vol 33, p. 2245.

[19]Mackenzie, D.C., Arkin, R.A, Cameron, J.M. "Mutliagent Mission Specification and Execution," Autonomous Robots 4, 1997, pp. 29-52.

[20]Mataric, M. J. "Reinforcement Learning in the Multi-Robot Domain," Autonomous Robots 4, 1997, pp.73-83.

[21]Mataric, M.J. "Minimizing Complexity in Controlling a Mobile Robot Population," Proc. ICRA, 1992, pp. 830.

[22]Parker, L. "Task-Oriented Multi-Robot Learning in Behavior Based Systems," Proc. IROS 1996, pp. 1478.

[23]Parker, L. *Heterogeneous Multi-Robot Cooperation*, Ph.D. Thesis, MIT, Feb. 1994.

[24]Parsons, D., Canny, J. "A Motion Planner for Multiple Mobile Robots," Proc. ICRA, 1990, pp. 8-13.

[25]Press, W.H., et al. *Numerical Recipes in C.* Cambridge University Press, 1990, pp. 343-352.

[26]Reinelt, G. *The Travelling Salesman, Computational Solutions for TSP Applications*. Springer-Verlag, 1991.

[27]Stentz, A. "Optimal and Efficient Path Planning for Partially-Known Environments," Proc. ICRA, 1994.

[28]Stentz, A. "The Focussed D* Algorithm for Real-Time Replanning," Proc. IJCAI, August 1995.

[29]Stentz, A., Hebert, M. "A Complete Navigation System for Goal Acquisition in Unknown Environments," Autonomous Robots, 2(2), 1995.

[30]Yoshida, E. et al "Evaluating the Efficiency of Local and Global Communication in Distributed Mobile Robotic Systems." Proc. IROS 1996, pp. 1661-1666.

ule of GRAMMPS is responsible for the inter-vehicle aspects of execution, excluding the actual mission planning. Under its purview are map information compression and exchange, position sharing, steering vote-passing, and inter-robot collision avoidance. The intercom modules can be connected to each other either in a ring (each only talks to its neighbors) or in a fully-connected configuration. If the communication infrastructure supports multi-cast, then the latter is the preferable choice. If, however, all communications are point-to-point, then a ring structure may be more effective. For two robots, as used in these live tests, the choices are equivalent.

Each local navigator generates updates to the map within its field of view. This raw obstacle/non-obstacle information is passed to Intercom where it is first reduced in resolution to reduce the computational complexity of path planning. Given the local navigator's capability to maintain vehicle safety, and the assumption of a relatively open workspace, this does not significantly effect the completeness of the planning system. After resolution reduction, the obstacles are grown by $\frac{1}{2}$ the vehicle width vehicle, and a safety-zone is added around these obstacle regions. Finally, the latest map updates are compared to the currently known state of the world and a list of changes is generated. This list is passed to other Intercom modules, as well as to the dynamic path planners. Map changes received from other robot are likewise passed on to the path planners, maintaining a consistent knowledge-set across all robots.

Position sharing and steering-vote-passing happen similarly. Each robot knows its position via the local navigator, and publishes this information to other robots. The path planners know every robot's current goal assignment (via the Mission Planner), and generate vote vectors as needed. Intercom passes these votes either over the appropriate robot, or to the local navigator if the generated votes apply to the robot on which the Intercom is running.

Local inter-vehicle collision avoidance has been substantially investigated in other works[4][17][30]. GRAMMPS uses a simple fixed priority "stop-and-wait" algorithm for avoiding inter-vehicle collisions. If a higher-priority robot gets too close to a lower-priority one, the latter comes to a stop. While stopped, the Intercom module ignores map information from the local navigator to avoid putting the moving image of the other robot into its map. Once the other robot moves far enough away, the area under the stopped robot is cleared of obstacles (as the moving robot may have seen and avoided the stopped robot). The stopped robot begins moving, accepting perception information as before. While this mechanism can certainly cause deadlocks, for relatively open areas it provides sufficient performance. It is left to future work to implement a more complete local avoidance scheme.

7. Live results

A companion paper[6] describes in detail the component technologies which support a multi-vehicle demonstration, and provides a step-by-step interpretation of a live run. The demonstration runs utilized two HMMWVs. For terrain perception, one used trinocular vision, the other a 2-axis 360-degree field-of-view scanning laser range finder. DGPS was the primary source of positioning information. Radio modems & PPP were used for inter-robot communication. Average vehicle speed was approximately 1m/s, limited primarily by the range and fidelity of the perception systems. Test runs were performed at a nearby outdoor test site.

Figure 8 and Figure 9 offer views of two different missions-in-progress. During these two runs, a total combined distance 2.5km was autonomously driven in about 30 minutes; also, nearly 800 map cells were perceived and marked as obstacles.

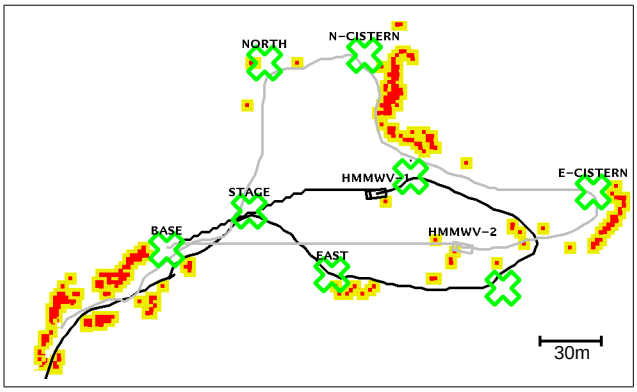

Figure 8: Example live run

In the second run (Figure 9), the failure of HMMWV-1 was simulated when it reached goal C1; HMMWV-2 was approximately at goal PIPE when this occurred. HMMWV-2's plan was altered to include that goal C2, which was previously allocated to HMMWV-1. This demonstrates the behavior of the system given a partial failure.

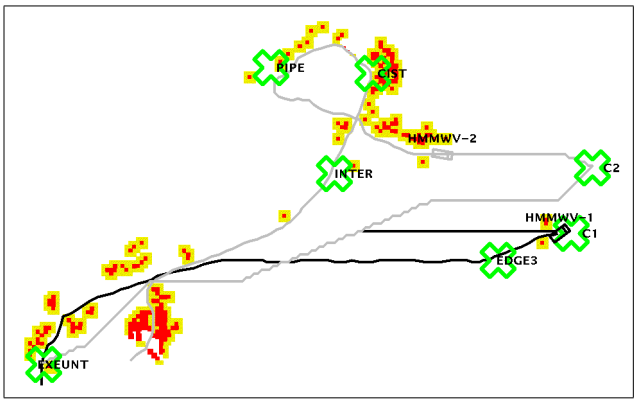

Figure 9: Live run with a simulated robot failure

8. Conclusions and Future Work

This paper has discussed the details of the central planner as well as the additional technology needed for live runs of a multi-robot system for outdoor environments. Results have been presented demonstrating the optimization and run-time modification of mission plans for single and multi- robot missions. Example live runs on two outdoor autonomous mobile robots have also been given.

Figure 6: 4 robot, 18 goal MTSP simulation run

planning capability, several other supporting technologies are needed to support distributed operation on actual hardware. Additionally, the constraints of real perception and control play a large part in the architecture of the system. An overview of the demonstration system architecture is given in Figure 7. It illustrates the breakdown of GRAMMPS into processing components, the distribution of computation between the robots, and the relationship to the local navigation components.

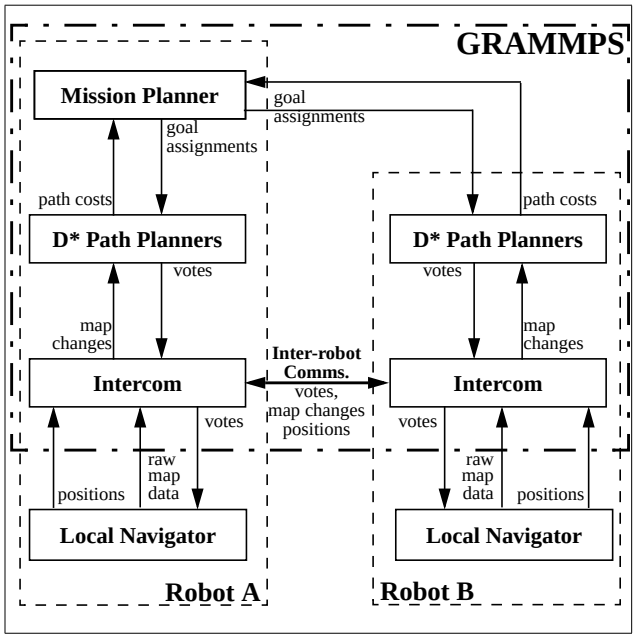

Figure 7: GRAMMPS System Overview

6.1. Architecture Issues

Every perception system introduces unavoidable noise into the system, "surprises" the planner with unexpected terrain, limits the motion of the robot due to a narrow field of view, and/or "changes its mind" about terrain when viewed from a different position. These problems imply that estimated costs of paths change frequently and the robot may not follow a previously recommended path to the goal. This implies the potential need to change mission plans (i.e. allocation and ordering of goals to robots) with some regularity to ensure optimal mission completion.

To ensure generality, GRAMMPS is designed to work with any local navigator provided the constraints detailed in Section 4.1. are met. It is assumed that the local navigator possesses all the skills to keep the robot safe while moving in the environment, and can report map information and vehicle positions. The local navigator should accept advice on how to reach a goal. It is not constrained however, to always following this advice. Instead, the dynamic path planner always provides appropriate advice for the current optimal path, regardless of how its previous plans were executed, leaving the local navigator free to make rapid responses to avoid immediate dangers.

These facts (perception noise, imprecise plan-following) coupled with the dynamic nature of an unstructured environment imply a need for an asynchronous architecture, where each component operates relatively independently. Intuition suggests that the local navigator should cycle fastest given typical perception limitations and the need to immediately respond to perceived hazards. The path planners can cycle somewhat more slowly, as they reason about larger scale issues, rather than the moment-to-moment needs of maintaining vehicle safety. The mission planner can cycle slowest of all, as it reasons about optimization at the largest relevant time-scale to the mission.

6.2. D*: Dynamic Path Planning

Dynamic path planning has been shown to be substantially faster than brute force replanning in dynamic environments[27]. In addition, D* has been demonstrated for single robot/single goal tasks utilizing a similar architecture[29]. In this application, multiple instantiations of the D* planning structure are used, one per goal in the mission statement, distributed based on the available computational capacity on each vehicle. This implies that the path to a goal being driven to by one robot may be planned on a processor located on a different robot.

The dynamic path planner accepts terrain-traversal cost information from the local navigator, and generates a vote-vector indicating which steering angle is currently preferred to reach a given goal. Vote vectors are computed by averaging the cost-to-goal of cells along each steering arc starting at the current robot location, weighted by the length of the segment of the arc contained in each cell.

In summary, the dynamic path planner acts in an advisory role to the local navigator directing it along the optimal path to the goal. Furthermore, it provides cost estimates to the CMP, which allow the CMP to generate current goal assignments as required to optimize the mission statement.

6.3. Intercom

For multiple mobile robot systems, issues of map sharing and inter-robot avoidance are relevant. The Intercom mod-

problem being optimized.

The decision to use exhaustive or randomized search occurs at the point at which they take the same amount of computation time. Once it becomes cheaper computationally to utilize a randomized search, this algorithm is preferred. It is of little utility to balance the quality of the solutions relative to the computation time, because in cases where exhaustive search remains tractable, simulated annealing usually provides the optimal path as well.

5. Simulation Results

For the first two example simulations, D*[27] is utilized as the dynamic path planner and a simulated local navigator moves the non-holonomic vehicle using D*'s advice. The perception system can see about 2 vehicle lengths in front of the robot, with a 120 degree field of view. Obstacles are "grown" by $\frac{1}{2}$ the width of the vehicle to approximate configuration space and have an infinite traversal cost. Obstacles are additionally bordered by a "caution zone" (also $\frac{1}{2}$ the robot width) to bias planning towards paths which maintain a safe distance from obstacle cells. All robots share position and map information. Optimal missions are planned by the CMP, using the time-like performance metric. Obstacle regions shown are those discovered so far during mission execution. The path behind each robot is the path driven so far, while that in front of the robot is the current optimal plan.

The first example involves only a single robot given the mission statement: $M(R_1, G_1 \wedge G_2 \wedge G_3 \wedge G_4)$. The initial plan assumes an empty world, and orders the goals given that knowledge(Figure 2). However, as the vehicle drives, it

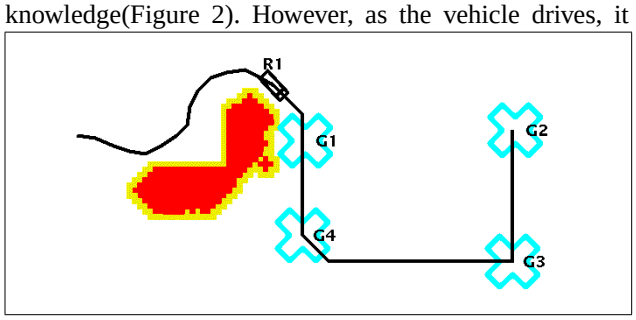

Figure 2: Beginning of a 1 robot simulation run

discovers obstacles and replans paths and the mission accordingly. This example demonstrates the robot maintaining the shortest path to each goal, as well as optimizing the order of goals to be visited.(Figure 3).

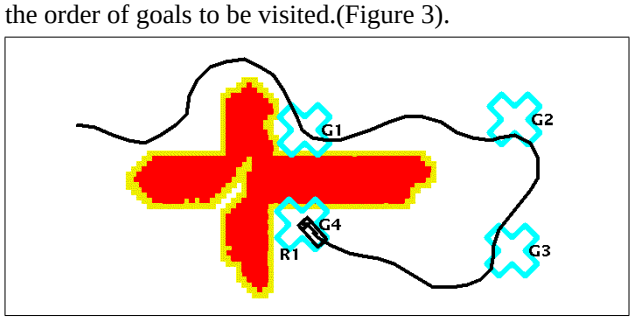

Figure 3: Completed run with goals reordered

The second example involves two robots, given the mission statement: $M(R_1 \vee R_2, G_1 \wedge G_2 \wedge G_3 \wedge G_4) \Rightarrow M(R_1 \wedge R_2, G_5)$ Though the light robot(R0) is initially to visit the left goals, and the dark robot(R1) to visit the right goals(Figure 4), the light robot ends up only visiting one of the

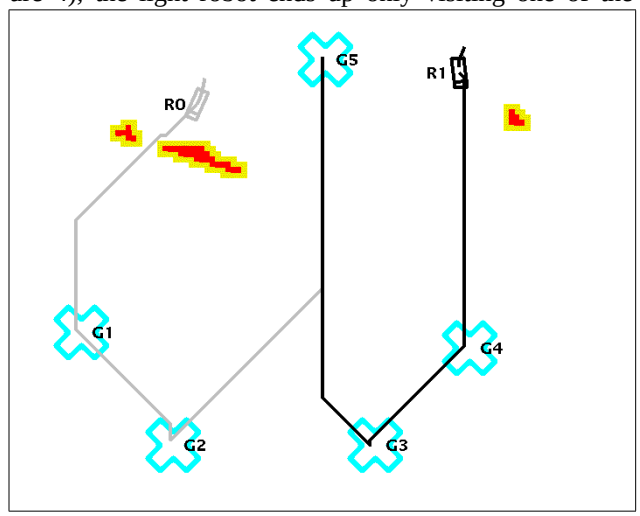

Figure 4: Beginning of a 2 robot simulation run

goals, and not one it was originally assigned(Figure 5). This demonstrates the utility of map information exchange between robots, as well as the ability for goals to be traded-off between robots.

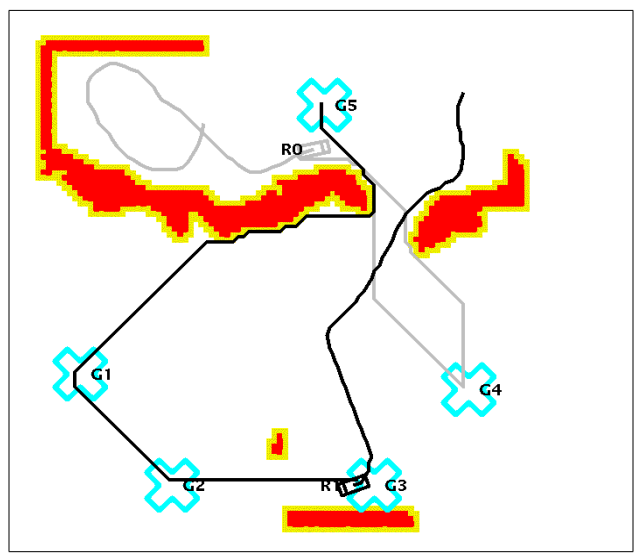

Figure 5: Later in run, after goal exchanges

The final simulation (Figure 6) involved 5 robots reaching 18 goals in a much more cluttered environment given the MTSP as a mission statement. The robots are omnidirectional, with a 360 degree field of view of limited range. This simulation demonstrates the scalability of GRAMMPS to larger problems.

6. Distributed Implementation

GRAMMPS was successfully demonstrated in natural terrain on a pair of autonomous HMMWV's. While the planning engine described above provides the mission

domized searches as necessary. Given that this system is designed for robots operating in a dynamic environment, it is acceptable to lose full optimality during the initial execution of a large mission, as the information being used to reach decisions before much exploration has occurred is likely to be inaccurate. Furthermore, as the mission progresses, the planning problem will become sufficiently small so that the planner can revert to an optimal decision making process. This allows GRAMMPS to scale naturally to missions involving larger numbers of robots and goals.

The notion of an "optimal" solution for a mission plan involving multiple robots is more complex than for a single robot. With one robot, whichever solution minimizes the cost for that robot is the optimal solution. With more than one robot, should the sum of all robots' costs be minimized? or, rather, should the maximum cost of any participating robot be minimized? These two alternatives specify different potentially useful metrics for comparing solutions. For example, consider a mission that involves three robots and that has two alternative solutions with one with a costs-per-robot $C_1 = [12, 4, 6]$ [1], and the other with a costs-per-robot $C_2 = [9, 9, 8]$. A metric which uses the sum of the costs would prefer C_1 over C_2; this is similar to minimizing the total distance crossed or energy used during the mission. A metric which seeks to minimize the most expensive component would prefer C_2 over C_1; this is similar to minimizing the time for the mission, by making the most expensive component as small as possible. Depending on the application and the interpretation of costs provided by the dynamic path planners, either metric could be appropriate for a mission. For the purposes of this work, the latter (time-like) metric is utilized, though switching to a distance metric, or even a linear combination of the two, is a straightforward modification.

The path planners provide the estimated costs to reach each goal. This cost is proportional to the time expected for a nominal robot to traverse this distance. By multiplying costs by a different constant factor for robots with different expected speeds, GRAMMPS can model an additional type of robot heterogeneity, beyond that supported by the mission grammar.

Rapid replanning is facilitated by the initial construction of a structure which represents the entirety of the search space. Replanning then consists of traversing this structure, adding costs, and calling the optimization routines on components as needed. On average, replanning with this system is approximately twice as fast as the initial planning, though this varies substantially depending on the mission statement. When a goal is reached, it is removed from the mission statement, and the planning structure pruned appropriately, leaving only the unfinished portions of the mission.

1. i.e.the cost incurred by Robot 1 during its portion of the mission is 12 units, the cost for Robot 2 is 4 units, etc.

4.4. Component Planners: TSP & MTSP

Given the enormous volume of research into solutions to the TSP and MTSP[15], it is reasonable to solve these mission components outside of the above framework. Furthermore, since the exponential nature TSP and MTSP can potentially introduce the largest number of additional solutions to the mission statement, it is profitable to offer randomized algorithms for larger instances of these problems.

It is more difficult to solve the TSP and MTSP as components of the larger missions, as this imposes additional unknowns for the optimal solution. Independently generated solutions to the TSP/MTSP components cannot be easily integrated into globally optimal solutions, as the effects of future goals on optimality are not incorporated. To resolve this, the algorithm produces a set of tours, each optimized for a particular potential next-goal for the robot(s).

The number of solutions examined by exhaustively solving the MTSP is given by: $\frac{(R-1+G)!}{(R-1)!}$, where G is the number of goals and R is the number of robots involved. For the TSP, $R = 1$, reducing the search complexity to $G!$. Obviously, the time to perform an exhaustive search grows very rapidly, necessitating an alternative method for computing a very good (if not optimal) solution.

While heuristics exist for solving the TSP and MTSP in polynomial time within a bounded amount of optimal, they either remain computationally intractable with very large exponents for even relatively large bounds[3], or do not provide a solution regularly superior[26] to that which can be found via randomized search. Simulated annealing[13] was chosen in this work for a randomized algorithm, as it has been shown to be very successful in this application[26].

Simulated annealing requires[25] an objective function E, a set of random functions which can change one solution into another, and an annealing schedule which defining how many random solutions will be produced at a given step and how quickly the system will be "cooled."

The objective function used for the TSP is simply the cost of the tour. For the MTSP, the objective function is:

$$E(C) = \frac{1}{2}C_L(C_L - 1)(N - 1) + \sum_N C_i$$

where N is the number of robots, C_i is the cost of the ith robot's tour, and C_L is the maximum C_i. This objective function captures the same time-like metric discussed previously and ensures smooth changes in E as robots' costs change. For example, if $N = 4, C1 = [5, 0, 0, 0]$, and $C2 = [4, 4, 4, 4]$, then $E(C1) = 35$ and $E(C2) = 34$, reflecting that C2 is a slightly superior solution, as expected.

The randomized moves for the TSP are "reverse" and "transport" as suggested by Lin[18]. For the MTSP, additional moves of "swap" and "transplant" are utilized. The former exchanges portions of a pair of robot tours, while the latter removes a portion of one robot's tour and inserts it into another's. The annealing schedule is empirically determined, though the number of solutions explored at a given temperature varies proportionally to the size of the

- Should Robot 1 or Robot 2 go to Goal A?
- Should Robot 1 go to Goal A then Goal B, or the other way around?

This structure allows GRAMMPS to reason about the essentially mobile parts of the mission. Rather than being explicitly concerned about what is to be done at each goal, it allows the operator (or higher level planning system) to specify the constraints on the mobile component of the mission, i.e. which robot can/should do each task, in what order tasks should be done, and which tasks are viable alternatives for each other.

Missions in GRAMMPS are expressed in the following grammar:

$$
\begin{aligned}
MG: \quad & m \rightarrow M(r,g) | m \wedge m | m \vee m | m \Rightarrow m | (m) \\
& r \rightarrow R_i | r \wedge r | r \vee r | (r) \\
& g \rightarrow G_j | g \wedge g | g \vee g | g \Rightarrow g | (g)
\end{aligned}
$$

The symbols above can be interpreted as follows:

Expression	Semantics
$A \Rightarrow B$	A FOLLOWED BY B
$A \vee B$	A OR B
$A \wedge B$	A AND B
R_i	Robot i
G_j	Goal j
$M(r,g)$	MOVE robot r TO goal g

For example, consider an the following expression in this grammar: $M((R_1 \wedge R_2) \vee R_3, G_1 \Rightarrow (G_2 \vee G_3))$. This mission statement says either both robots R_1 and R_2 or only robot R_3 should first go to goal G_1, then to either G_2 or G_3. This expression could be used in a trash-collection application where trash is located at G_1, and trash receptacles are at G_2 and G_3; furthermore this statement might also imply that robot R_3 is larger than robots R_1 and R_2, allowing it to carry the trash in a single load, while R_1 and R_2 would need to share the task. This example also illustrates how the grammar enables the use of heterogeneous robots within GRAMMPS via constraints on their possible actions expressed in the mission statement.

A general expression in this grammar can be quite complicated, expressing many conflicts between robots and constraints on mission execution. Once the CMP is given a mission statement, the mission statement is manipulated into a *plannable form*, intended to simplify the planning and optimization problem.

An expression in plannable form must meet two requirements. First, it must be expressible in this grammar:

$$
\begin{aligned}
PG: \quad & m \rightarrow M(R_i, g_o) | M(R_i, g_s) | M(r, g_a) | m \wedge m | m \vee m | m \Rightarrow m | (m) \\
& r \rightarrow R_i | r \vee r \\
& g_o \rightarrow G_i | g_o \vee g_o \\
& g_s \rightarrow G_i | g_s \Rightarrow g_s \\
& g_a \rightarrow G_i | g_a \wedge g_a
\end{aligned}
$$

Second, any subexpression involving conjunctions of missions (i.e. $m \wedge m$) must not have the same robot R_i appearing in independent parts of the conjunction. The constraints on the form of $M(r,g)$ components of the mission imposed by grammar PG limits them to the following set of expressions which have straightforward solutions:

Name	Solution Method
$M(R_i, G_j)$	min. cost path
$M(R_i, G_1 \wedge \ldots \wedge G_M)$	TSP[a]
$M(R_i, G_1 \vee \ldots \vee G_M)$	select min. cost path
$M(R_i, G_1 \Rightarrow \ldots \Rightarrow G_M)$	add min. cost paths
$M(R_1 \vee \ldots \vee R_N, G_1)$	select min. cost path
$M(R_1 \vee \ldots \vee R_N, G_1 \wedge \ldots \wedge G_M)$	Multiple TSP

a.Travelling Salesman Problem

By constraining the statement in this fashion, complex interdependencies between sub-missions are removed. In addition, a form is ensured which naturally utilizes the minimum path cost information provided by the path planners.

Manipulation of the mission statement is performed by applying transformation rules which do not alter the semantics to portions of the mission statement. These rules can be loosely classified as the identity, associative, commutative and distributive laws for this grammar. Rather than given an exhaustive set of laws, the following transformation of a statement in MG into plannable form gives the flavor of the algorithm and the allowable transformations. Starting from the previous example: $M((R_1 \wedge R_2) \vee R_3, G_1 \Rightarrow (G_2 \vee G_3))$. Each step below uses a different distributive rule to break down the statement:

1. $M((R_1 \wedge R_2), G_1 \Rightarrow (G_2 \vee G_3)) \vee M(R_3, G_1 \Rightarrow (G_2 \vee G_3))$
2. $(M(R_1 \wedge R_2, G_1) \Rightarrow M(R_1 \wedge R_2, G_2 \vee G_3)) \vee$
   $(M(R_3, G_1) \Rightarrow M(R_3, G_2 \vee G_3))$
3. $((M(R_1, G_1) \wedge M(R_2, G_1)) \Rightarrow (M(R_1, G_2 \vee G_3) \wedge M(R_2, G_2 \vee G_3))) \vee$
   $(M(R_3, G_1) \Rightarrow M(R_3, G_2 \vee G_3))$

The full expression-manipulation algorithm performed by the Grammar Compiler involves 3 steps. First, subexpressions which are not part of grammar PG are broken down. Next, conjunctions of $M(r,g)$ terms which contain robot conflicts are broken down and resolved. Finally, compatible parts of the mission are recombined to produce $M(r,g)$ components with as many terms as possible. These steps serve to generate a mission statement that allows rapid planning and replanning by making explicit the constraints and limitations of the mission statement, as well as locally minimizing the number of alternative missions which must be examined.

4.3. Planning, Replanning, and Pruning

The actual planning component of the CMP takes as input a mission statement in plannable form from the Grammar Compiler. The algorithm used by this component is a relatively straightforward depth-first exhaustive search. However, the nature of the mission grammar implies mission components (such as the TSP) which are NP-complete[15]. Therefore, while optimal solutions are possible for relatively small mission statements, this engine is designed to gracefully degrade optimality by using ran-

to address optimizing the cost of reaching this state[11], there is a fundamental paradigm mismatch; physically situated mobile missions are not easily linked with logic-oriented symbolic planners.

Cao[8] provides an excellent overview of the research to date in cooperative robotic systems. These system can be broken into two classes: those which are aimed at particular applications, and those designed to study the nature of cooperative robotics itself.

Virtually all demonstrated application-oriented systems are intended for use in indoor environments[16], though some planned systems are aimed at a less-controlled hospital environment[9]. The MARTHA project has supported the design of cooperative systems intended for use in dockyard applications, however, published demonstrations[1] have involved only indoor test beds. In general, these systems have not fully considered the needs of unstructured or dynamic environments where changes in knowledge about the world happen with great regularity.

Many cooperative systems are reactive[5] in nature, such as work by Parker[23], Mataric[21], and Mackenzie[19]. Reactive systems are particularly adept at operation in "noisy" environments, and these systems have also exhibited the ability to learn behaviors for cooperation[20][22]. However, they are not traditionally concerned with the efficient performance of their missions, rather only with the successful completion of the task. In addition, while behavioral systems are effective at a limited class of tasks (foraging[21][23], formation marching[23][19], box pushing[22]) it is unclear if capabilities are scalable to more general types of missions, or to those with significant operational constraints.

In summary, motion planning systems fail to address the dynamic- and mission- aspects planning of this problem. AI-planning systems are not well suited for mobile missions in unstructured environments. Existing multi-robot systems either are limited in the class of missions they can perform, have not been designed for operation in unstructured environments, or do not optimize the execution of these missions. GRAMMPS is designed to address these limitations to facilitate the application of multi-robot systems to real-world tasks in unstructured environments. Previous work in this area[7] motivated the need for generalized mission planning and presented a single, hard-coded example to illustrate its potential benefit. This work presents a general solution capable of handling an infinite variety of mission statements expressible in a useful grammar, demonstrated both in simulation and on real vehicles.

4. Mission Planning

The mission planning component of GRAMMPS called the Central Mission Planner (CMP), responsible for determining what each robot should do to best assist in completing the mission. Figure 1 shows the internals of the CMP, divided into functional blocks. This section discusses first the I/O of the CMP, and then each of the functional blocks, relating design decisions to the needs of the problem statement

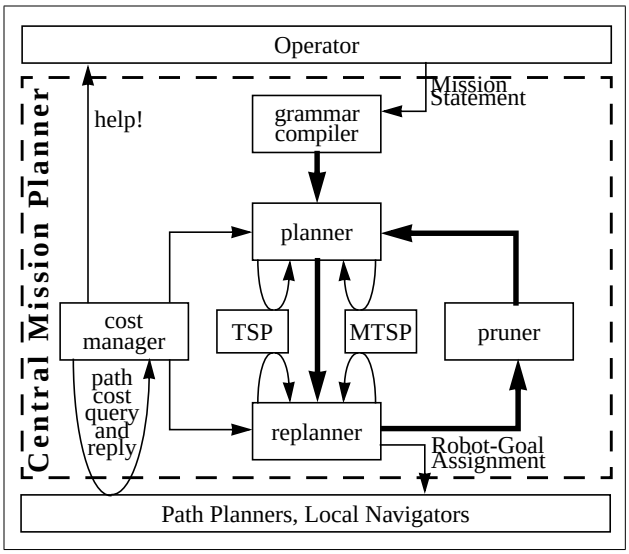

Figure 1: GRAMMPS Central Mission Planner (CMP)

4.1. CMP I/O & Cost Manager

The CMP takes as input a mission statement defining the mission to be performed. The CMP also requests estimates from a set (one per goal) of dynamic path planners for costs from each robot to each applicable goal and from goal to goal. When a plan is generated, the CMP informs paths planners which robot they should currently be guiding, if any. This process cycles regularly to ensure that each robot is assigned the correct action to further mission completion given the regularly changing estimates of component costs as reported by the path planners.

The mission statement expresses the task for the robot team in terms of the involved robots and goals, plus conjunctions, disjunctions and sequences thereof. The goals are assumed to be known within a graph representation of the environment. An 8-connected grid is the graph representation used by the dynamic planners for this work, though any graph structure is acceptable, provided that (within the chosen map representation) the path planners are capable of providing path-cost estimates and that the perception systems on the local navigators are capable of providing updates of state-to-state traversal costs. Within these constraint, the CMP receives path-cost estimates in a meaningful form for planning, and returns assignments in a manner useful for guiding execution.

Should problems arise (e.g. a required goal becomes unreachable), the CMP notifies the operator that it is no longer possible to complete the mission, and requests a clarification to the mission statement, Meanwhile, it continues to execute those parts of the mission which remain feasible.

4.2. Mission Grammar & Compiler

GRAMMPS is designed to reason primarily about mobile missions (i.e. moving robots between locations), making decisions based on operator-specified constraints. The type of decisions the mission planner makes are:

- Should Robot 1 go to Goal A or Goal B?

GRAMMPS: A Generalized Mission Planner for Multiple Mobile Robots In Unstructured Environments

Barry L. Brumitt, Anthony Stentz
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

For a system of cooperative mobile robots to be effective in real-world applications, it must be able to efficiently execute a wide class of complex tasks in potentially unknown and unstructured environments. Previous research in multi-robot systems has either been limited to relatively structured domains or to small classes of feasible missions. This paper describes a field-capable system called GRAMMPS which addresses this problem by coupling a general-purpose interpreted grammar for task definition with dynamic planning techniques. GRAMMPS supports a general class of local navigation systems and heterogeneous groups of robots, providing optimal execution of missions given current world knowledge. Simulation runs illustrating the capabilities of this system are provided. Results showing successful runs of this system on two autonomous off-road vehicles are also given.

1. Introduction

Work in cooperative robotics to date[8] has largely focused on indoor systems performing a relatively small class of missions. Cooperative robot systems which have been designed for outdoor environments are typically limited to specific structured environments. For a cooperative robotic system to be effective and generally applicable, it should support a wide variety of cooperative tasks, execute these tasks efficiently, and be able to operative in unknown or unstructured environments.

For many applications, there exists the need to coordinate the motion of a group of robots for multiple concurrent goals. For example, in a construction task, it may be necessary to coordinate a number of dump trucks as they move to collect dirt from a number of excavators. Alternatively, in a planetary exploration scenario, a pair of robots may have a number of target sites which need to be investigated. Both of these examples are primarily mobile missions, i.e. a task where the majority of the effort is moving the robots in their environment.

This paper discusses the implementation and demonstration of GRAMMPS, a Generalized Robotic Autonomous Mobile Mission Planning System for multiple mobile robots operating in unstructured environments. The generality of GRAMMPS is reflected in a formal grammar in which missions can be expressed, as well as by a flexible interface to local navigation systems. GRAMMPS is designed for real applications, not satisfied with only task completion, but rather with performing the task in an optimal fashion.

Section 2 of this paper more precisely states the problem addressed by GRAMMPS. Section 3 discusses the relationship of existing planning and cooperative robotic systems to the needs of this problem. Sections 4-5 define the mission grammar, describe the planning engine and show simulation results. Sections 6-7 move from simulation to operation on real robots, showing results from live runs.

2. Problem Statement

The problem addressed can be stated as follows:

To support:
- motion planning and
- complex missions,

for:
- multiple robots,
- multiple concurrent goals, and
- dynamic environments,

using mobile robots which have:
- relatively open operational workspaces,
- similar mobility characteristics, and
- effective positioning, communication and perception,

permit:
- successful, optimized task execution, and
- dynamic replanning as world knowledge increases.

3. Related Work

There are several related areas of research related to cooperative mission planning which address aspects of this problem. All fall short, for a variety of reasons, of permitting optimized mobile mission execution in unstructured environments.

A huge body of research exists in motion planning for single and multiple mobile robots; Latombe[14] has an excellent text on the space of approaches in this area. For multiple robots, these algorithms typically plan motion from a single start state to a single goal state[10][24], which implies a very limited class of applicable missions. Furthermore, focussing primarily on the constraints imposed by inter-robot collision avoidance is not appropriate in many environments, as robots' motions do not frequently conflict. These algorithms also do not perform well in dynamic environments, requiring complete replanning when a knowledge discrepancy is found.

The demands of mission planning (i.e. optimizing the selection and ordering of goals within the constraints of a stated mission) imply the need for a symbolic reasoning system, as typically found in AI-planning systems. While such systems[12] can capably reason about series of actions to reach a goal state, and have also been extended