Advances in Cognitive Systems X (2016) 1-6 Submitted 4/2016; published 6/2016

Goal Reasoning with Information Measures

Benjamin Johnson BENJAMIN.JOHNSON.CTR@NRL.NAVY.MIL
Mark Roberts MARK.ROBERTS.CTR@NRL.NAVY.MIL
NRC Research Associate at the Naval Research Laboratory, Washington, DC

Thomas Apker THOMAS.APKER@NRL.NAVY.MIL
David W. Aha DAVID.AHA@NRL.NAVY.MIL
Navy Center for Applied Research in AI, Naval Research Laboratory (Code 5514), Washington, DC

Abstract

In complex and dynamic scenarios, autonomous vehicles often need to intelligently adapt their behavior to unexpected events. Goal Reasoning provides a methodology for autonomous agents to deliberate on and adapt their goals to react to changing conditions. This paper extends and implements a Goal Reasoning system based on the Goal Lifecycle, and grounds the implementation in the information measures and expectations used by the vehicles to asses their performance. The implemented system, termed Goal Reasoning with Information Measures (GRIM), is demonstrated using a disaster relief scenario in which a small team of vehicles is tasked with surveying a pre-defined set of geographical regions. Additionally, a preliminary study shows that the inclusion of resolution strategies increases the likelihood that GRIM successfully finishes its goals.

1. Introduction

Complex applications of robotics often require autonomous vehicles to react intelligently to changes in the operational scenario. A change in the observed state of the environment (e.g., the robot senses an unexpected event) or the internal state of the vehicle (e.g., the vehicle is consuming fuel faster than expected) may necessitate a change in the robot's behavior. This change can manifest as a change to the vehicle's plans or tasks, or the underlying goals that it is trying to achieve. Intelligent adaptation to unexpected changes is vital to the design of autonomous systems for complex applications.

Goal Reasoning (GR) research aims to develop autonomous agents that can deliberate on, and change, their own goals (Vattam, Klenk, Molineaux, & Aha, 2013). Such agents are able to adapt to new and unexpected observations about their environment by creating new goals to pursue, or by modifying their existing goals. Such capabilities become even more valuable in applications where multiple robots must act collaboratively to accomplish their mission; GR allows the robots to adjust their goals to align with those of the other agents, or to leverage the assistance of other robots.

An example application that would benefit from GR is the control of autonomous vehicles in the Foreign Disaster Relief (FDR) domain (U.S. Department of Defense, 2011). The FDR domain focuses on providing humanitarian aid in the wake of natural disasters, and is an area that could greatly benefit from the deployment of autonomous vehicles. In such situations, autonomous vehicles

could be used to provide rapid surveys of the disaster area, identifying important locations and traversable routes for the responders. Additionally, the vehicles could be used to enhance the reliability and range of communications, by serving as mobile communication relay points.

For applications like FDR operations, it is important for GR techniques to be grounded in the information and capabilities that regulate the behavior of the vehicles. That is, deliberation about the goals of an autonomous agent should be performed based on the measures that define and govern those goals. This paper investigates that grounding by extending the system described in Roberts, Apker, Johnson, Auslander, Wellman, & Aha (2015a)[1] and Apker, Johnson, & Humphrey (2016). This extension frames the goal refinement and resolution processes in terms of information gathered during the execution of the goal, and is implemented in a system called Goal Reasoning with Information Measures (GRIM). The work here presents early steps in creating a full GR system for a team of robots assisting with FDR operations, and demonstrates a multi-vehicle system performing GR with respect to a set of area-survey goals. This paper is an extension of earlier, non-archival work (Johnson, Roberts, Apker, & Aha, 2016), and presents a preliminary study that shows that the use of resolution strategies increases the likelihood that GRIM successfully finishes its goals.

The paper is structured as follows. Section 2 describes a motivating example. Section 3 provides a more in-depth description of GR and the instantiation that is extended in this work. Section 4 demonstrates the GRIM system with respect to the motivating example. Section 5 evaluates the system through an initial, ablative study. Section 6 concludes the paper with a discussion of this work in the scope of other, related work, as well as the future expansion of the system.

2. Motivating Example

Consider, as a motivating example, an FDR mission where a team of Unmanned Air Vehicles (UAVs) must survey several pre-defined regions to identify and locate an important official. The team of UAVs, which start in a Base region, must survey three different regions, each with different characteristics: an Airport, and two Office Buildings.

The system's first task is to search the regions to locate the official. Once the location of the official is known, the system can then establish and maintain a communications relay for that official. While the Airport is larger than the other two regions, the establishment of a relay is made more difficult in the Office Buildings, due to the more complicated and cluttered terrain. These goals are further complicated by other restrictions and factors involved in the mission, such as:

- The need for the UAVs to refuel at the nearby base station.
- Changes to the set of resources (e.g., vehicles) that are available to the system.
- The presence of uncontrolled or adversarial environmental factors (e.g., wind).
- Additional goals, with varying or dynamic priorities/importance (e.g., the discovery of a medical emergency that must be immediately addressed).

1. ActorSim, an implementation of the Goal Lifecycle, is available online at `http://makro.ink/actorsim/`

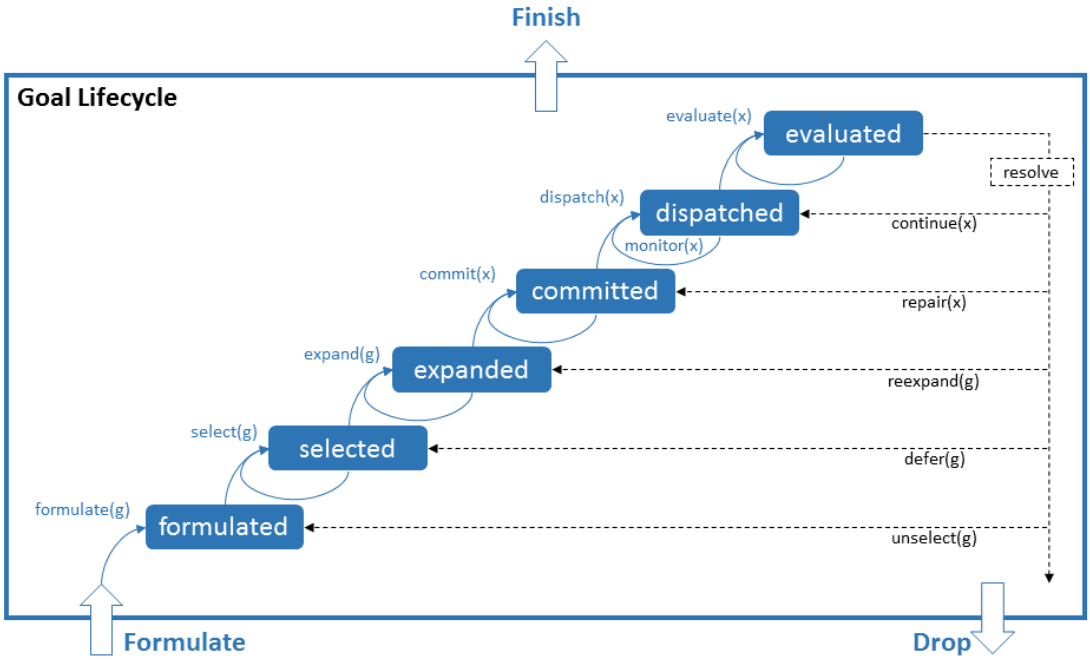


Figure 1: An adapted version of the Goal Lifecycle, from Roberts et al. (2015a). Goals transition through the modes (boxes) via strategies (arcs), adding detail to to goal's definition.

The system controls a team of two UAVs and can assign them to any of the search areas. Furthermore, due to uncontrolled factors, it is assumed that the vehicles will under-perform their expectations during execution, resulting in slower-than-expected searches of the areas.

3. Goal Reasoning and the Goal Lifecycle

GR focuses on developing agents that can deliberate on and modify their goals during execution within a dynamic environment. The work presented here leverages and adapts the *Goal Lifecycle* of Roberts et al. (2015a); Roberts, Vattam, Alford, Auslander, Apker, Johnson, & Aha (2015b), an adaptation of which is shown in Figure 1. This provides a framework for the refinement of goals and the resolution of problems that arise during execution.

GR is performed by transitioning each goal $g \in G$ through the *modes* (represented by boxes) of the Goal Lifecycle via *strategies* (the arcs). Progression through the modes represents increasing refinement in the goal detail. The remainder of this section briefly summarizes the key strategies of the Goal Lifecycle; specific details are given in Section 4, as the strategies relate to the goals for the motivating scenario described in Section 2. For the remainder of the paper, transition *strategies* are denoted with small caps (e.g., FORMULATE) and the resulting goal *modes* are denoted by monospace small caps (e.g., `FORMULATED`).

The FORMULATE strategy determines when a new goal g is created and enters the Goal Lifecycle from an external source (e.g., user input, or a triggering event). This strategy takes an abstract

goal as an input, and transitions it to a FORMULATED mode, by defining the initial constraints, the measures defining success or failure, and its prerequisites. The result of FORMULATE is the creation of a new goal with the information (e.g., constraints, metrics, and prerequisites) required for further refinement.

The SELECT strategy takes a FORMULATED goal g, and determines whether the system activates it. A goal transitions to SELECTED (i.e., it is actively pursued by the system) only if its prerequisites are satisfied and the system has the available resources (including computation) to pursue it, both of which are defined by the FORMULATE strategy. As such, some goals may not be SELECTED and will instead remain in the FORMULATED mode until their prerequisites are met and the required resources become available.

The EXPAND strategy takes a SELECTED goal g and generates one or more expansions (i.e., plans) $x \in X$ to achieve the goal. An EXPANDED goal defines how the system can satisfy the constraints that were created during the FORMULATE strategy, and generates expectations for how each expansion will perform when executed. If one or more feasible plans are created, the goal transitions to an EXPANDED mode. Otherwise, the goal remains in the SELECTED mode.

Once a goal g has been successfully EXPANDED, the COMMIT strategy chooses one of the feasible expansions x for execution. Such a choice involves assessing the costs of each of the expansions, as well as the likelihood that they will successfully execute the goal (per the FORMULATED constraints). A COMMITTED expansion defines how the system will satisfy g, and provides the set of expectations for the performance of the plan's execution.

The DISPATCH strategy sends the COMMITTED expansion x to an executive to run. This process amounts to allocating resources and generating metrics for plan execution. A successfully DISPATCHED expansion defines the criteria by which a goal is evaluated to ensure that it detects and reacts to discrepancies in the expected performance of the expansion.

During execution, two strategies manage updates that impact the mode of the goal g. First, EVALUATE is a passive strategy that is called (externally, or by the goal itself) whenever new information impacts the goal. In contrast, the MONITOR strategy, when enabled, proactively tracks the execution of the DISPATCHED expansion x to ensure that its expected performance will still result in successful completion of the goal. Additionally, MONITOR ensures that the prerequisite conditions for the goal remain met and that the allocated resources remain available. If MONITOR detects a problem, it triggers EVALUATE and x progresses to an EVALUATED mode, indicating that there is a discrepancy in the performance of the expansion that should be addressed.

When a goal transitions to EVALUATED, the RESOLVE strategy assesses any discrepancies that were detected, and determines how the system should resolve the discrepancy (i.e., which mode the goal should transition to). If the EVALUATED goal g is determined to have met all of the constraints and success-conditions that were generated during the goal formulation, it is resolved with FINISH and marked as completed. If g violates the formulated constraints, DROP marks it as unsuccessful (it can then be reformulated with new constraints).

Otherwise, if g still meets its formulated constraints but does not meet its success conditions, the RESOLVE strategy transitions the goal to one of the earlier modes in the Goal Lifecycle. If EVALUATE determines that the DISPATCHED plan x is still feasible, the goal is resolved back to the DISPATCHED mode (referred to as CONTINUE). If the COMMITTED plan can be fixed without

major changes by reallocating system resources, it resolves back to the COMMITTED mode (REPAIR). If the committed plan is infeasible, but another feasible plan $\bar{x} \in X$ exists, the goal g is resolved back to the EXPANDED mode and commits to a different, feasible plan $\bar{x}$ (REEXPAND). If no feasible expansion exists given the currently available resources, g is resolved back to the SELECTED mode, where it can be expanded once the necessary resources become available (DEFER). Finally, if the goal no longer meets its prerequisites for selection, but it still satisfies its constraints, it is resolved back to a FORMULATED mode until the prerequisites for selection are met once again (UNSELECT).

4. Goal Refinement with Information Measures

Goal Refinement for unmanned FDR missions can be framed in terms of refining a set of constraints and expectations for a set of measurable information metrics. In this sense, a goal is initially an abstract description of what the system is supposed to achieve (i.e., survey an area). As the goal is progressively refined, more details are added such that the meaning of the abstract description is clarified (i.e., what it means to complete the "survey"). After enough refinement, the goal will define not only what the abstract description means, but how and when it is expected to be achieved (i.e., by creating a plan). Grounding the refinement of the goal in a consistent set of information measures establishes a clear set of criteria (e.g., constraints and expectations) that can be used by the system to evaluate performance and inform changes to the systems goals.

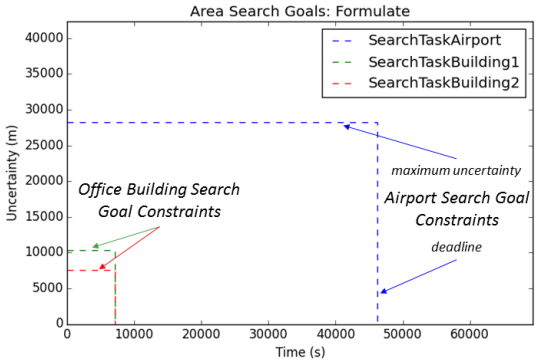
This section describes a GR system, called Goal Reasoning with Information Measures (GRIM), and demonstrates this system via simulation. GRIM instantiates the Goal Lifecycle (discussed in Section 3), and provides centralized control for a small team of 2 UAVs. While the demonstrated system uses centralized control, the structure of the Goal Lifecycle also allows for distributed control such that individual vehicles could adapt their own goals during execution.

Returning to the motivating example, described in Section 2, the goal refinement strategies of the Goal Lifecycle are defined here for the area search goals, while the relay goal (which will be the focus of future research) is only included as an abstract goal (i.e., it is not specified below). The information measures for an area search goal, defined in this section, describe the degree to which the defined region has been "searched". The metric used to evaluate the uncertainty in an area search will differ based on the sensors and algorithms used to conduct the search. For simplicity, the vehicles conduct searches in this example by following a lawnmower waypoint pattern, and the metric used is the length of that search pattern that has yet to be traversed. This measure was chosen as a simple approximation for the information gathered during the survey task, and future work will explore more accurate measures of the uncertainty in an area survey.

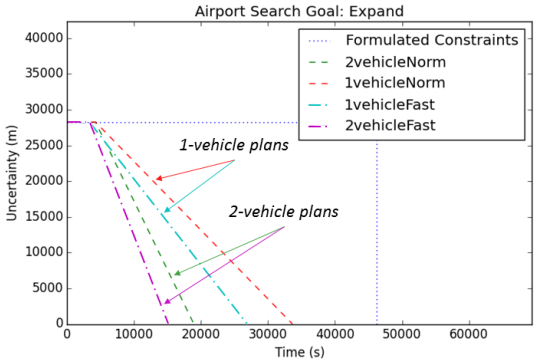
Figure 2 shows plots of the FORMULATE, EXPAND, DISPATCH, MONITOR, REPAIR, and REEXPANDstrategies from the Goal Lifecycle. The plots show each strategy as a function of the information measures used in the motivating example: uncertainty (approximated by the waypoint path length) and time. The remainder of this section explains each strategy in more depth.

4.1 Formulate

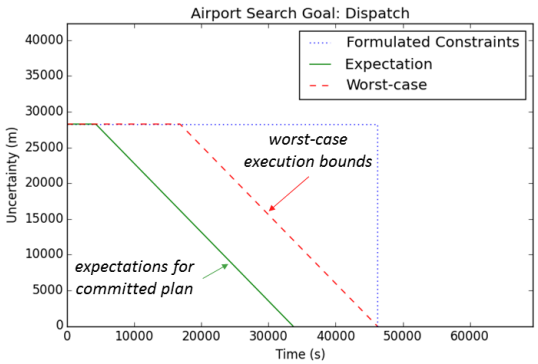
The FORMULATE strategy, in the case of the area survey, defines three parameters that describe the constraints under which each can be considered as successful or failed. These parameters are:



(a) FORMULATE: Maximum uncertainty and deadline constraints for the Airport and Office Building search goals.



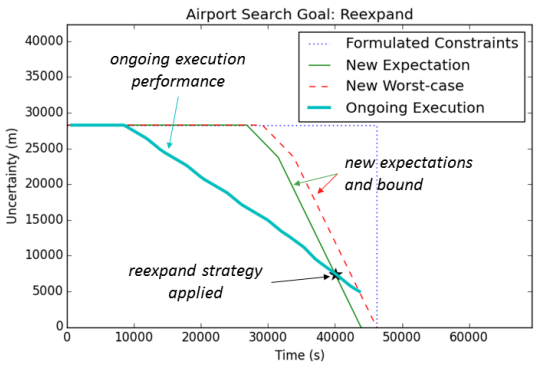
(b) EXPAND: Expected survey performance for each expansion x of the `SELECTED` search goal (Airport).



(c) DISPATCH: Expectations and bounds for the `DISPATCHED` expansion ("1vehicleNorm").

(d) MONITOR: Execution performance at time 32,824 where a violation of the bounds triggers EVALUATE.

(e) REPAIR: Monitored execution and adjusted expectations/bound, at time 39,975, after REPAIR increases the UAV speed.



(f) REEXPAND: Monitored execution and new expectations/bound, at time 43,670, after REEXPAND generates a new 2-vehicle plan.

Figure 2: Plots of selected strategies from the execution of GRIM for the motivating example.

1. *maximum uncertainty*: the upper bound on the uncertainty in the search area (i.e., the uncertainty of the area before any information has been gathered),

2. *acceptable uncertainty*: the level of uncertainty at which the goal is considered complete, and

3. *deadline*: the time by which the search must be complete.

The maximum uncertainty specifies the total amount of uncertainty in the search area prior to any search, and represents the total information that can be gathered about an area during a survey. The acceptable uncertainty parameter defines the level of uncertainty at which the area can be reliably deemed to be empty of an official (i.e., a finishing criteria for the search). Finally, the deadline defines the point in time at which a search must be finished, in order for it to be considered successful. For both (1) and (2), the constraints are defined by the measure used for uncertainty: the length of the search path (in meters) that has yet to be traversed by the vehicles. For (3), the constraint is defined in terms of mission time (seconds). No prerequisites are formulated for the survey goals, which are allowed to progress to a `SELECTED` mode as determined by the SELECT strategy.

Figure 2a displays the constraints on each of the formulated goals as a function of the area uncertainty (in meters of untraversed search path) and the execution time. Each of the dashed lines in this figure represents the allowable area of uncertainty for a survey area (i.e., the Airport and the Office Buildings) at a given time. Because of the more complicated interactions of the Office Buildings, the deadline (i.e., the time at which the area uncertainty must be within the defined acceptable level of uncertainty) for each of these is earlier than the deadline for the Airport. At all times up until the deadline, no constraint is placed on the allowable area of uncertainty, so it is set as the full length of the search path for each of the search areas (28,267 meters for the Airport, and 10,303 and 7,537 meters for the smaller Office Buildings). At the deadline, the area of uncertainty is required to not exceed the defined acceptable level of uncertainty. This acceptable level of uncertainty was defined as 500 meters for each of the search areas, but is omitted from Figure 2a for clarity.

4.2 Select

The next strategy in the Goal Lifecycle is for GRIM to SELECT one or more goals to pursue. The SELECT strategy requires comparing high-level estimates of expected performance and value (cost/reward) for each goal, and assessing the available resources for the system. This strategy determines which goals are operational (i.e., those that are selected). For the example presented here, GRIM selects only a single goal: the one that minimizes the ratio $\frac{max\,uncertainty}{deadline}$. Due to the significantly longer deadline, the ratio for the Airport is (despite the larger uncertainty) smaller than the ratio for the Office Buildings, and GRIM selects the Airport goal.

4.3 Expand

After selecting a particular goal g, EXPAND generates a set of plans[2] X to accomplish it. For each plan $x \in X$ that is generated, a set of expectations is also generated that describe its expected performance

2. The original Goal Lifecycle (Roberts et al., 2015a) used the term *expansion* to refer to the possible plans that could be applied to a goal; the remainder of this paper uses the term *plan* interchangeably with the term *expansion*.

with respect to the information measures used in the formulation strategy. A successful EXPAND strategy generates at least one feasible plan (i.e., it is expected to satisfy the formulated constraints). In the example used here, a feasible plan is one where the expected value of the uncertainty at the deadline is no greater than the defined acceptable uncertainty.

Figure 2b displays the expectations of four feasible plans that are generated during the EXPAND strategy, for the selected goal g. The expectations are shown as the expected change in the uncertainty of g over time. The resulting plans are for a single vehicle moving at normal speed ("1vehicleNorm"), a single vehicle moving at a faster speed ("1vehicleFast"), two vehicles moving at normal speed ("2vehicleNorm"), and two vehicles moving at a faster speed ("2vehicleFast"). It is assumed that a faster vehicle speed improves the search rate at the cost of higher fuel consumption.

4.4 Commit

Once the goal g has been `EXPANDED` into a set of feasible plans, GRIM must COMMIT to a single plan. To do so, it assesses the costs (i.e., the expended resources, including time) of each feasible plan, and commits to the least costly plan. In this example, GRIM chooses the "1vehicleNorm" plan, as the expectations lie well within the formulated constraints while conserving the most resources. By using only a single vehicle, GRIM leaves the second vehicle available in reserve, and by committing to the plan that moves the vehicle at the normal speed the system preserves fuel for other tasks, such as searching another area or providing a relay for a discovered official.

4.5 Dispatch

Once GRIM has a `COMMITTED` plan, it must then DISPATCH that plan to the appropriate vehicles. At the same time, it must also determine the performance bounds for successful plan execution. These bounds represent the worst-case scenario from which the plan can still be expected to satisfy the formulated constraints. To generate these bounds, GRIM uses the expected performance of the plan adjusted such that the expectations reach the acceptable level of uncertainty at the deadline. If the performance exceeds these bounds during execution, GRIM will need to adapt the goal g, by applying the RESOLVE strategies described in Section 3. Figure 2c displays the expectations and worst-case execution bounds of the dispatched plan.

In GRIM, a plan is dispatched by scheduling pre-defined commands for execution by vehicles. When the vehicles receive these commands, they are passed to a synthesized Finite State Automaton (FSA) that is running on the vehicle, and executed according to the rules that were used to synthesize that FSA. A more thorough description can be found in Apker et al. (2016). In the case of the dispatched "1vehicleNorm" plan, the command to search the Airport region is sent to a single vehicle, and the vehicle's speed is left at its default, more fuel-efficient value.

4.6 Monitor

During execution, GRIM will actively MONITOR the progress of the `DISPATCHED` expansion to ensure that it will satisfy the constraints of the selected goal. Figure 2d shows the system's estimate of the search area uncertainty over time, as well as the constraints, expectations, and worst-case bound. This figure shows the execution performance up until a time of 32,824 seconds. Prior to this

time, the execution was proceeding slower than expected, but still remained within the worst-case bound. At any time prior to this point, if the execution were to proceed forward at the *expected rate*, it would satisfy the formulated constraints on the goal. At the execution time of 32,824 seconds, the performance violates the worst-case bound; at this point, if it were to continue to execute at the expected rate, it would not complete the search before the deadline. As such, the MONITOR strategy triggers a subsequent EVALUATE strategy, which determines how to best RESOLVE the violation.

4.7 Evaluate and Resolve

Once MONITOR detects that an execution has violated some constraint or bound, it passes the goal to the EVALUATE strategy. If the execution had satisfied the acceptable level of uncertainty that was generated during formulation, the goal would be passed to the FINISH strategy, where it would be marked as successfully completed. If it violated the other formulated constraints, the DROP strategy would mark it as failed. In this example, the execution violates the worst-case bound generated during DISPATCH and the goal g is passed to the RESOLVE strategy, where GRIM attempts to change the execution such that it may still satisfy the constraints on g.

First, GRIM attempts to REPAIR the expansion by adjusting the expansion chosen in the COMMIT strategy. This involves changing the vehicle speed by committing to a different instance of the 1-vehicle plan from the original expansion: "1vehicleFast". The new instance of the expansion is `COMMITTED`, and it is dispatched and monitored as before. Figure 2e shows the expectation and worst-case bound for the newly repaired plan, which now requires the vehicle to move more quickly and expend more fuel. However, as the execution continues, the repaired plan also fails to meet expectations, and at time 39,975 the system execution crosses the new bound and triggers the EVALUATE strategy, again activating the RESOLVE strategies.

This time, when GRIM attempts to RESOLVE the failing goal, it finds that neither instance of the originally expanded plan can be expected to satisfy the formulated constraints. Thus, it cannot REPAIR the expansion, and it instead attempts to REEXPAND the goal g. Doing so allows GRIM to attempt to generate new plans that might be feasible by incorporating resources that were not used by the current expansion. In the example shown here, the second vehicle (which was not used in the originally committed expansion) is available for the newly expanded plans. As a result, the re-expansion strategy finds two new feasible plans (the single-vehicle plans are deemed infeasible): using both vehicles at their default ("2vehicleNorm") and fast ("2vehicleFast") speeds. With these new plans, the goal g returns to the `EXPANDED` mode, and progresses through the Goal Lifecycle again, committing and dispatching the "2vehicleNorm" plan.

Figure 2f shows the expectation and worst-case bounds for the newly re-expanded and `DISPATCHED` plan x, which now assigns both vehicles to search the area. Because the 2nd UAV must first traverse to the search region, it does not arrive in time to assist the search before the deadline, and the new plan also fails to meet expectations. At time 43,670 the execution violates the bounds of the new expansion, and GRIM uses the REPAIR strategy to increase the speed of both vehicles. At time 44,380 MONITOR again triggers the EVALUATE and RESOLVE strategies, but both the REPAIR and REEXPAND strategies fail. In this case, which was designed specifically to fail (in order to demonstrate the RESOLVE strategies), both of the other search goals have already passed their deadlines, and are

	Number of Goals Successfully Completed		Computation Time Spent RESOLVE Strategies (μs)	
	Mean	95% C.I.	Mean	95% C.I.
5 Vehicles, 15 Regions - 500 Random Scenarios				
REPAIR + REEXPAND	11.53	$\pm$0.005	2595	$\pm$10
REEXPAND only	11.54	$\pm$0.005	3837	$\pm$12
REPAIR only	11.41	$\pm$0.004	814	$\pm$2
No RESOLVE strategies	11.01	$\pm$0.005	39	$\pm$0
20 Vehicles, 50 Regions - 100 Random Scenarios				
REPAIR + REEXPAND	43.92	$\pm$0.221	101426	$\pm$3061
REEXPAND only	43.07	$\pm$0.215	235548	$\pm$3256
REPAIR only	43.01	$\pm$0.232	4561	$\pm$60
No RESOLVE strategies	39.16	$\pm$0.245	530	$\pm$5

Table 1: Results from an ablative study of the effects of RESOLVE strategies on successful completion of survey goals and time spent executing the RESOLVE strategies.

dropped as they are deemed to have failed. Once the execution time passes the deadline for the Airport search goal, it is also dropped.

5. Evaluation

The demonstration scenario in Section 4 was specifically designed to fail, in order to illustrate the RESOLVE strategies in GRIM. These strategies can often help recover from problems during execution. This section describes an initial simulation study to evaluate the advantage of using the REPAIR and REEXPAND strategies, to test the following hypothesis: *Including the* RESOLVE *strategies during execution will increase the likelihood that GRIM will successfully finish more of its goals.*

The evaluation was conducted for 2 sets of scenarios: the first used 5 vehicles to survey 15 randomly generated regions, while the second used 20 vehicles to survey 50 randomly generated regions. In both cases, the UAVs were affected by randomly generated wind. The survey navigation was determined via a greedy algorithm that directs an assigned vehicle to the closest unexplored cell in a discretized region; whenever a vehicle enters a cell, that cell gets marked as explored. The study focused on two metrics. First, it recorded the number of survey regions that were successfully explored (i.e., they had one or less unexplored cells) by their deadline. Second, the study tracked the amount of computation time that was spent in the RESOLVE strategies (summed over all search goals in the scenario). The experiment was performed on 500 different scenarios of 5 vehicles and 15 regions, and 100 scenarios of 20 vehicles and 50 regions. In each case, the experiment was run for each of the following configurations of GRIM: both REPAIR and REEXPAND strategies, only the REEXPAND strategy, only REPAIR, and no RESOLVE strategies.

Table 1 displays the average (mean) and 95% confidence interval for the number of successful goals and the computation time spent in RESOLVE strategies. The results (for both sets of scenarios)

indicate that the inclusion of the RESOLVE strategies increased the number survey goals that were successfully completed, thus supporting the hypothesis.

The results for the scenarios with 5 vehicles and 15 regions showed that the REEXPAND strategy was equally as effective as using both the REEXPAND and REPAIR strategies. This is likely because of the method by which plans are generated during the EXPAND and REEXPAND strategies: these strategies generate plans for both the nominal and increased vehicle speeds. As such, the adjustments that can be made by the REPAIR strategy can also be made by the REEXPAND strategy. While the improvement in the mean value for using both RESOLVE strategies in comparison to using none appears small, it is statistically significant (as evidenced by the confidence intervals on each).

The more complicated scenarios, with 20 vehicles and 50 regions, showed a more dramatic effect for using both RESOLVE strategies. In this case, using both RESOLVE strategies showed significant improvement over each of the REPAIR and REEXPAND strategies, individually, and even more so in comparison to using no RESOLVE strategies. The individual RESOLVE strategies, on the other hand, resulted in a statistically insignificant difference in the number of completed survey goals.

In both cases, the REPAIR strategy reduced the computation time needed to RESOLVE problems with the `DISPATCHED` plan. By using a new instance of a previously computed plan, the REPAIR strategy reduces the time that GRIM spends in the RESOLVE strategies, as shown in Table 1. The non-zero value shown for computation spent in the RESOLVE strategies for the last entry in each set, where neither REEXPAND or REPAIR were used, can be attributed to the system architecture. During execution, GRIM automatically enters the RESOLVE strategy before determining which resolution strategies are enabled.

6. Discussion and Conclusion

This paper describes initial efforts towards grounding a GR system, termed GRIM, in the information measures used by the controlled vehicles during execution. In particular, this work adapts the Goal Lifecycle introduced in Roberts et al. (2014, 2015a) and instantiates it in GRIM: a centralized GR system that provides commands to independent vehicles. Vattam et al. (2013) describe a GR agent as an autonomous agent that is "aware of its own goals and [can] deliberate upon them," and provides a useful survey of related GR research. Autonomous agents that deliberate on their goals are not an isolated concept, and substantial research has been conducted on this topic (Norman & Long, 1996; Altmann & Trafton, 2002; Cox, 2007; Molineaux, Klenk, & Aha, 2010; Thangarajah, Harland, Morley, & Yorke-Smith, 2010; Harland, Morley, Thangarajah, & Yorke-Smith, 2014; Paisner, Cox, Maynord, & Perlis, 2014).

The individual strategies used in the Goal Lifecycle are, themselves, important research topics, and each can be accomplished in a variety of ways. For example, goal formulation may occur externally to the system (i.e., a user may provide a goal), or may be conducted autonomously during execution (Talamadupula, Benton, Schermerhorn, Kambhampati, & Scheutz, 2010; Weber, Mateas, & Jhala, 2012; Jaidee, Muñoz-Avila, & Aha, 2013; Klenk, Molineaux, & Aha, 2013). Similarly, the specific method for goal selection can vary widely, from domain-specific rule-based selection (Shapiro, Sardina, Thangarajah, Cavedon, & Padgham, 2012; Thangarajah et al., 2010) to the evaluation of domain-independent heuristics (Wilson, Molineaux, & Aha, 2013), or goal priorities

(Young & Hawes, 2012). In many cases, plan generation will generate expectations for the plan's execution performance, though in some cases it may be necessary to generate expectations separately, as in Auslander, Floyd, Apker, Johnson, Roberts, & Aha (2015).

The Goal Lifecycle provides a formal structure for these strategies, such that the resulting system can deliberate on and adapt its goals to dynamic and unpredictable events. This paper extends the Goal Lifecycle within the FDR domain by grounding its implementation using the vehicle's information measures, and by implementing and demonstrating the RESOLVE strategies. This work focused specifically on area survey goals within a disaster relief scenario, though related goals exist that may also be characterized in a similar fashion (e.g., communications relay). An ablative study was used to show that the REPAIR and REEXPAND strategies can improve GRIM's ability to successfully complete its goals, and how the inclusion of the REPAIR strategy can reduce the computation time needed for the resolve strategies.

Future extensions will investigate additional goal types (e.g., communications relay for a discovered official), as well as the use of more complex algorithms and information measures. The uncertainty of an area survey was approximated, for expectations and progress of the goal, by the length of the lawnmower search pattern. A more accurate measure of the uncertainty (e.g., tracking the total area covered by the vehicle's sensors) and corresponding expectations in an area survey would allow GRIM to improve its performance estimates and react accordingly. Additionally, a planner or scheduler could be used to SELECT goals while accounting for the likelihood of discovering an official in each region, thus enabling GRIM to more intelligently choose which goals to pursue. Likewise, adapting plan expectations (e.g., recognizing that the vehicles are not completing the survey at the expected rate, and changing the expectations accordingly) would enable GRIM to more quickly identify and evaluate problems, and thus improve the likelihood that it could RESOLVE any discrepancies. These extensions will enable a more thorough evaluation of GRIM's benefits via an experiment with randomly generated scenarios.

Acknowledgments

This work was performed at the Naval Research Laboratory. The authors were funded by the Office of Strategic Defense.

References

Altmann, E. M., & Trafton, J. G. (2002). Memory for Goals: An Acctivation-Based Model. *Cognitive Science*, *26*, 39–83.

Apker, T. B., Johnson, B., & Humphrey, L. (2016). LTL Templates for Play-Calling Supervisory Control. *Proceedings of AIAA Science and Technology Exposition*.

Auslander, B., Floyd, M. W., Apker, T., Johnson, B., Roberts, M., & Aha, D. W. (2015). Learning to Estimate : A Case-Based Approach to Task Execution Prediction. *Proceedings of the 23rd International Conference on Case-Based Reasoning* (pp. 15–29). Frankfurt, Germany.

Cox, M. T. (2007). Perpetual Self-Aware Cognitive Agents. *AI Magazine*, *28*, 32–46.

Harland, J., Morley, D. N., Thangarajah, J., & Yorke-Smith, N. (2014). An Operational Semantics for the Goal Life-Cycle in BDI Agents. *Autonomous Agents and Multi-Agent Systems*, *28*, 682–719.

Jaidee, U., Muñoz-Avila, H., & Aha, D. W. (2013). Case-Based Goal-Driven Coordination of Multiple Learning Agents. *Proceedings of the 21st International Conference on Case-Based Reasoning* (pp. 164–178). Saratoga Springs, NY, USA.

Johnson, B., Roberts, M., Apker, T., & Aha, D. W. (2016). Goal Reasoning with Informative Expectations. *Planning and Robotics: Papers from the ICAPS Workshop*. London, UK: AAAI.

Klenk, M., Molineaux, M., & Aha, D. W. (2013). Goal-Driven Autonomy for Responding to Unexpected Events in Strategy Simulations. *Computational Intelligence*, *29*, 187–206.

Molineaux, M., Klenk, M., & Aha, D. W. (2010). Goal-Driven Autonomy in a Navy Strategy Simulation. *Proceedings of the 24th AAAI Conference on Artificial Intelligence*. Atlanta, GA.

Norman, T., & Long, D. (1996). Alarms: An implementation of motivated agency. *Intelligent Agents II Agent Theories, Architectures, and Languages*, (pp. 219–234).

Paisner, M., Cox, M. T., Maynord, M., & Perlis, D. (2014). Goal-Driven Autonomy for Cognitive Systems. *Proceedings of the 36th Annual Conference of the Cognitive Science Society* (pp. 2085–2090). Quebec City, Canada.

Roberts, M., Apker, T., Johnson, B., Auslander, B., Wellman, B., & Aha, D. W. (2015a). Coordinating Robot Teams for Disaster Relief. *Proceedings of the 28th International FLAIRS Conference*. Hollywood, Florida, USA.

Roberts, M., Vattam, S., Alford, R., Auslander, B., Apker, T., Johnson, B., & Aha, D. W. (2015b). Goal Reasoning to Coordinate Robotic Teams for Disaster Relief. *Planning and Robotics: Papers from the ICAPS Workshop*. Jerusalem, Israel: AAAI.

Roberts, M., et al. (2014). Iterative Goal Refinement for Robotics. *Working Notes of the Planning and Robotics Workshop at ICAPS*. Portsmouth, NH: AAAI.

Shapiro, S., Sardina, S., Thangarajah, J., Cavedon, L., & Padgham, L. (2012). Revising Conflicting Intention Sets in BDI Agents. *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems* (pp. 1081–1088). Valencia, Spain.

Talamadupula, K., Benton, J., Schermerhorn, P., Kambhampati, S., & Scheutz, M. (2010). Integrating a Closed World Planner with an Open World Robot : A Case Study. *In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)* (pp. 1561–1566). Atlanta, GA.

Thangarajah, J., Harland, J., Morley, D., & Yorke-Smith, N. (2010). Operational Behaviour for Executing, Suspending, and Aborting Goals in BDI Agent Systems. *Declarative Agent Languages and Technologies VIII: Papers from the AAMAS Workshop* (pp. 1–21). Toronto, CA.

U.S. Department of Defense (2011). Department of Defense Support to Foreign Disaster Relief (Handbook for JTF Commanders and Below). *GTA 90-01-030*.

Vattam, S., Klenk, M., Molineaux, M., & Aha, D. W. (2013). Breadth of Approaches to Goal Reasoning : A Research Survey. *Goal Reasoning: Papers from the ACS Workshop (Techinical Report CS-TR-5029)* (p. 111). College Park, MD: University of Maryland.

Weber, B. G., Mateas, M., & Jhala, A. (2012). Learning from Demonstration for Goal-Driven Autonomy. *Proceedings of the 26th AAAI Conference on Artificial Intelligence* (pp. 1176–1182). Toronto, Canada.

Wilson, M., Molineaux, M., & Aha, D. W. (2013). Domain-Independent Heuristics for Goal Formulation. *Proceedings of 26th Int. FLAIRS Conference* (pp. 160–165). St. Pete Beach, USA.

Young, J., & Hawes, N. (2012). Evolutionary Learning of Goal Priorities in a Real-Time Strategy Game. *Proceedings of the 8th Annual International Conference on Artificial Intelligence and Interactive Digital Entertainment* (pp. 87–92). Palo Alto, CA, USA.