Proceedings of the Ninth Annual Conference on Advances in Cognitive Systems Submitted 9/2021; published 11/2021

The Rational Selection of Goal Operations and the Integration of Search Strategies with Goal-Driven Autonomy

Sravya Kondrakunta KONDRAKUNTA.2@WRIGHT.EDU
Venkatsampath Raja Gogineni GOGINENI.4@WRIGHT.EDU
Michael T. Cox MICHAEL.COX@WRIGHT.EDU
Department of Computer Science & Engineering, Wright State University, Dayton, OH 45431 USA

Demetris Coleman COLEM404@EGR.MSU.EDU
Xiaobo Tan XBTAN@EGR.MSU.EDU
Dept. of Electrical & Computer Engr., Michigan State University, East Lansing, MI 48824 USA

Tony Lin TLIN@GATECH.EDU
Mengxue Hou HOUMENGXUE@GATECH.EDU
Fumin Zhang FUMIN@GATECH.EDU
Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA

Frank McQuarrie FRANKMAC@UGA.EDU
Catherine R. Edwards CATHERINE.EDWARDS@SKIO.UGA.EDU
Skidaway Institute of Oceanography, Department of Marine Sciences, University of Georgia, Savannah, GA 31411 USA

Abstract

Intelligent physical systems as embodied cognitive systems must perform high-level reasoning while concurrently managing an underlying control architecture. The link between cognition and control must manage the problem of converting continuous values from the real world to symbolic representations (and back). To generate effective behaviors, reasoning must include a capacity to replan, acquire and update new information, detect and respond to anomalies, and perform various operations on system goals. But, these processes are not independent and need further exploration. This paper examines an agent's choices when multiple goal operations co-occur and interact, and it establishes a method of choosing between them. We demonstrate the benefits and discuss the trade offs involved with this and show positive results in a dynamic marine search task.

1. Introduction

Intelligent Physical Systems (IPS) are cognitive systems that combine perception, actuation, cognition and communication to operate in the physical world. Examples include physical robots, self-driving cars, intelligent homes, and smart grid networks. In general, most IPS need to be capable of robust, long-term autonomy in the presence of uncertain situations, unexpected events, and

dynamically changing environments with minimal human intervention. However, achieving such autonomy in a domain independent manner has long been elusive and fraught with difficulty (Sidhik et al., 2020; Mota & Sridharan, 2018; Huntsberger & Woodward, 2011; Wilson et al., 2013a,b). The research presented here proposes an approach that begins to integrate control theory techniques at the lower levels of an IPS with high-level reasoning about system goals. We show the benefit of this approach through a set of marine exploration tasks in complex, underwater environments.

The cognitive systems community has long acknowledged the importance of agents reasoning about themselves, their behaviors, and their goals[1]. Several research groups have addressed the issues regarding such full-spectrum reasoning. Approaches that address this include automated planning, belief-desire-intention frameworks, cognitive architecture frameworks, and *goal-driven autonomy (GDA)*. Among these areas, GDA explicitly focuses on the reasoning about goals by implementing various goal operations such as formulation, delegation, and goal change. Goal operations play a vital role in focusing reasoning on the most crucial information, and they also guide the behavior of the IPS when problems arise.

Here, we extend the idea that goal reasoning comprises a series of primary goal operations that can make an IPS robust and combine these principles with techniques from control theory to provide the foundation for implementing IPS in marine environments. We call this approach goal-driven marine autonomy. However, since the world is dynamic, instances exist in which the IPS must choose between multiple goal operations that are relevant. The main contribution of this paper lies in developing a procedure for an IPS to make such choices and integrating it into sophisticated problem-solving strategies performed during search applications. We implement the decision processes in a publicly available cognitive architecture named MIDCA.

The remainder of the paper is as follows. Section 2 describes our multilayer approach to IPS robustness, a framework for building IPS by integrating high-level and low-level autonomy mechanisms. It also introduces the marine problem domain. Section 3 describes search strategies used by agents to survey the marine domain and presents the results comparing the strategies with and without goal operations. Next, section 4 describes an evaluation of a method to improve performance by selecting among relevant goal operations during these strategies. Section 5 discusses related research. Finally, Section 6 presents the closing remarks and future research directions.

2. A Multilayer Approach to Intelligent Physical Systems

The objective of our research is to integrate the high-level abstract reasoning of cognitive systems with the preciseness of control theory for complex continuous domains. Such an integration enables the building of robust IPS for applications in dynamic and unpredictable marine environments. We exploit results from a research approach called *goal-driven autonomy (GDA)* that focuses on managing the goals of cognitive systems and merge them with specifics of robotic control theory in what we term *control-driven autonomy (CDA)* (see Figure 1).

With respect to actions, our GDA approach involves goal management and strategies using goal predicate structures and hierarchical task network plans. CDA involves task net optimization and

1. In this paper, we focus on attainment goals or states to achieve in the external world. Currently, we side-step the issue of applying our approach to performance goals, maintenance goals, or query goals (Van Riemsdijk et al., 2008).

path planning using waypoints and motion commands. With respect to perception, our approach involves problem recognition and state interpretation using problem schemas and explanation patterns.[2] CDA involves sensor fusion and communication generation using vector field models and grid maps. Currently, we have developed the top and bottom layers shown in this figure and await future research to provide the mechanisms of managing uncertainty with belief spaces (but see Lin et al. (2020) for preliminary work toward this goal). Furthermore, we have yet to deploy platforms with GDA in field trials. Instead, our AUVs employ CDA. Thus, when we use the term "agent," we are referring to GDA and CDA in simulation. When eventually validated and fielded, the resulting agent aboard an AUV will instantiate our initial IPS.

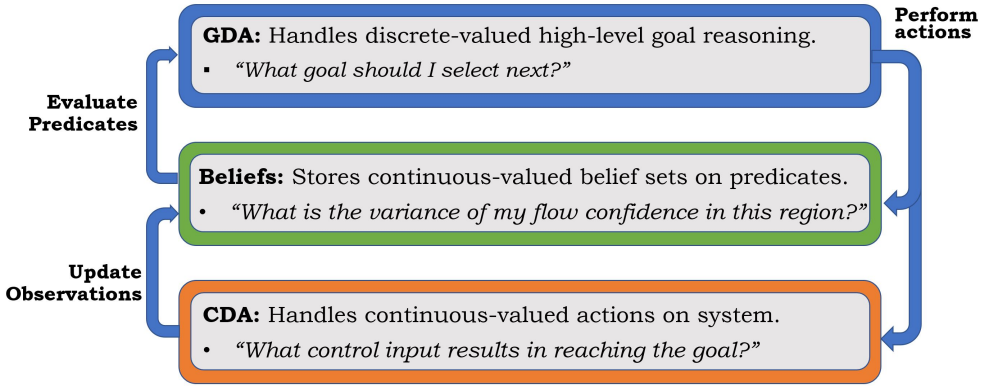

Figure 1. A multilayer approach to intelligent physical systems. At the highest level, GDA performs goal operations given beliefs about the environment. At the lowest level, control manages the actuators and updates continuous sensor values. Mediating between the two, sets of possible states are mapped to discrete predicates and actions are translated to control parameters.

2.1 Goal-Driven Autonomy (GDA)

Goal driven autonomy or GDA (Aha et al., 2010; Klenk et al., 2013; Munoz-Avila, 2018) is a kind of *goal reasoning* (Aha, 2018; Roberts et al., 2018) applied to issues of robust autonomy in agent-based systems. Unlike standard autonomous systems that generate behaviors given externally provided goals or tasks, the GDA approach is to independently recognize problems that arise, explain what causes the problem, and use the explanation to formulate a goal. Off the shelf planners can then generate sequences of actions that if executed achieve the goal and hence solve the problem. We use the *Metacognitive Integrated Dual-Cycle Architecture (MIDCA)* (Cox et al., 2016) to integrate GDA into the multilayer framework.

In addition to goal formulation, numerous other *goal operations* exist to manage the goals of the agent. They include goal selection, goal change, goal delegation, and goal monitoring (Cox et al., 2017). Goal selection selects the goals for the agents to execute. Goal change modifies the agent's goal to a more abstract or specific goal based on resource availability and real-world

2. Note that we are referring to a kind of internal explanation or self-diagnosis process rather than an external explanation to another agent.

changes. Goal delegation is helpful in multiagent scenarios to pass goals to other agents when the agent is constrained. Finally, goal monitors surveil the validity of goals based on state changes. In this paper, we focus on goal selection and goal formulation.

2.2 Control-Driven Autonomy (CDA)

Control theory is the study of dynamical systems control (Antsaklis & Michel, 2006; Khalil, 2002; Athans & Falb, 2013). Its aim is to develop models and algorithms that drive the system to a desired state through system inputs. Control systems vary and are found in systems such as the temperature control in buildings, control of chemical processes in industrial plants, electrical circuits, cruise control in automobiles, and autonomy of robotic platforms and autonomous vehicles. We term control theory applied to autonomous platforms control-driven autonomy or CDA.

Broadly, there are two categories of controllers: open-loop and closed-loop systems. Open-loop systems control a system purely based on models, whereas closed-loop systems include measurements as feedback to help control the system, usually through correcting behavior. It it is possible to include multiple models for a dynamical system and thus represent multiple modes of operation. Feedback can then be used to select the most appropriate model to accomplish tasks such as fault detection. This mechanism is important in the context of an IPS because of potential damage to the system. For example a drone losing a motor, an underwater glider losing a wing, or a car driving on ice may all require separate models to accurately represent behavior in those situations.

2.3 The Marine Survey Domain

Consider the problem of time-limited surveys of marine environments with autonomous underwater vehicles (AUVs). Typical missions measure salinity, temperature, and pressure throughout the water column and can incorporate acoustic receivers to investigate key aspects of marine life. An key feature within a marine ecosystem is the presence of *hot spots* or regions of high fish density. These areas and the aquatic pathways between them that fish transit represent areas of ecological sensitivity. Thus, discovering the location of major hot spots, especially for endangered species, is an important task. However, many barriers exist in such environments that make mission success difficult. Sea creatures may attach themselves to platforms and slow progress. Tides and currents exist that also impede progress and obstacles may appear requiring course change. Finally, conditions may change, limiting the detection range of acoustic receivers Edwards et al. (2020); McQuarrie et al. (2021).

Our research team regularly deploys AUVs such as Slocum gliders and custom robotic fish as part of coastal observing systems, for science-driven experiments and the testing and evaluation of new platforms. During missions, the platforms surface to communicate on regular schedules or in response to forced interrupts. AUV surveys make a valuable contribution to management efforts in Gray's Reef National Marine Sanctuary, located on the inner shelf of the South Atlantic Bight off the coast of Savannah, GA (see Figure 2). Gray's Reef contains fish tagged with transmitters that send an acoustic signal or 'ping' at a pre-determined frequency (5 minutes for short experiments, 30-180 minutes for long-term tracking) containing identifiers unique to that instrument, allowing researchers to classify detections by source.

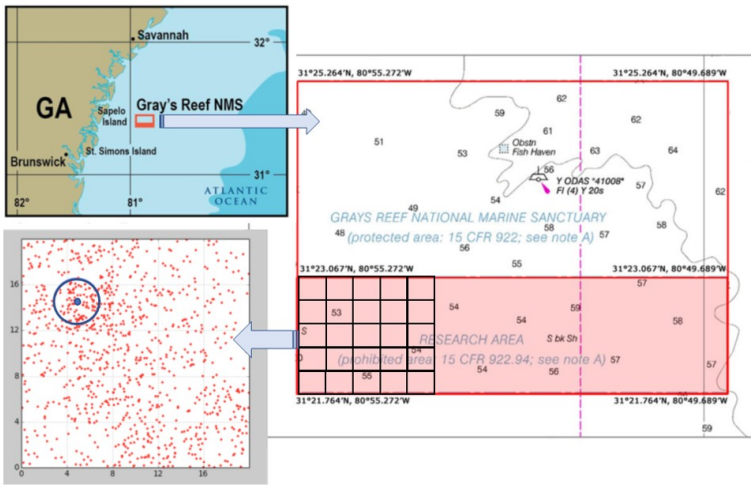

Figure 2. Gray's Reef National Marine Sanctuary is located off the coast of Georgia and contains a research area shown in the insert shaded in pink. Within this, we represent a 5x5 subsection. This grid contains fish hot-spots that are of interest to marine scientists, here within the cell at location (6,14). The agent (indicated by the blue dot) is also in this cell. The circle around the agent indicates the sensor range for detecting acoustic fish tags (the red dots).

We implemented a simulator for this domain to test search techniques prior to actual deployment and to empirically evaluate the mechanisms discussed in this paper. The lower right of Figure 2 shows the portion of the research area modeled by the simulator and split into 25 cells. One cell is shown in the lower left of the figure. The red dots depict 1000 fish (currently assumed to be static) that emit a ping every 17 time steps (time steps are determined to emulate the real-world behavior). In this cell, a hot-spot is located near the co-ordinate (6,14). The blue dot represents a simulated AUV controlled by an agent, and the blue circle represents the receiver detection radius. At Gray's Reef, the detection radius varies with environmental conditions, but currently the simulator assumes it to be 2 co-ordinate units. As mentioned, an agent can identify hot-spots based on the number of pings.

3. Search Strategies for Aquatic Surveys

An agent's initial goals are to survey the entire region and collect information within a deadline of 600 cycles. If the agent identifies a hot-spot, it needs to report the hot-spot by generating an interrupt, or it can hold the information until its next surface event. The distribution of fish tags is varied to generate successive problems having multiple hot-spots. The agent must find the hot spots and handle all unexpected situations it encounters.

One unexpected event is biofouling caused by remora attachment. In actual vehicle deployments, remora attachments cause an unexpected change in the vehicle's buoyancy and thus forward speed. The simulator models these effects and injects such anomalies at random. The simulator also

allows multiple attachments, thus posing incremental effects. For example, a remora attack reduces the speed by 50%, and if the agent ignores it, another will further reduce it by 50%.

For an agent to survey a region, it must follow a search strategy and handle anomalies that arise concurrently. We integrate three different search strategies. Many search strategies exist in the literature, so implementing, optimizing, and developing the best search strategy is not the focus of this paper. Rather, the strategies below are implementations of existing procedures or minor modifications of them.

- *Structured Search (SS)*: Greedy search or modified hill-climbing search;
- *Ergodic Search (ES)*: Control strategy that uses an information metric based on a Gaussian prediction model to guide exploration;
- *Ergodic Search Combined with Structured Search (ESCSS)*: Combination of the above two strategies.

SS and ESCSS search each cell in the grid, while ES considers the entire region to be one space and ignores discretization. Subsections 3.1 – 3.3 provide details on search strategies.

3.1 Structured Search (SS)

Structured search is a modified stochastic hill-climbing algorithm (O'Reilly & Oppacher, 1994). To overcome the hill-climbing inability to continue when it encounters a local maximum, the agent backtracks and searches further. Table 1 presents the procedure for structured search.

The agent starts in an arbitrary cell (line 1), adds it to the visited stack (line 3), searches the current cell for unique pings (line 4) and records all unique pings heard from each of the four boundary edges in a cell. The agent then moves to a neighboring cell in the direction of the highest unique ping density (line 6). Later, it continuously visits the cells with an increasing number of unique pings (lines 9-11). However, when the unique pings in the current cell are less than that of the previously visited cell (line 12), it backtracks and visits the unexplored best neighbor of the last visited cell (line 14). The agent does this iteratively until it visits all cells (lines 7-8) or meets the mission deadline.

CSearch(searchCell, visited) is a support function that takes as input a searched cell and the visited cell stack and returns the searched cell's best expected non-visited neighbor (lines 18-20). However, if all the input cell neighbors are visited (line 21), it recursively checks for the best non-visited neighbor from the visited cells and returns it (lines 25-26).

3.2 Ergodic Search (ES)

The Ergodic Exploration for Distributed Information (Silverman et al., 2013; Miller et al., 2015) procedure is a closed-loop solution to the problem of exploration of an unknown space. The search method uses a receding horizon control strategy to optimize the ergodicity metric in Mathew & Mezić (2011).

The ergodicity metric relates the time averaged state trajectory $x(\cdot)$ of a dynamic system to a spatial distribution $\phi(x)$. A trajectory is optimally ergodic when the amount of time spent in any

Table 1. Method for structured search (SS)

1. $currentCell \leftarrow startCell$ *// Initial location of the agent*
2. $visited \leftarrow initStack()$ *// Keep track of the surveyed cells for backtracking*
3. $visited.push(currentCell)$
4. $currentCellPings \leftarrow$ ***UniquePings***$(currentCell)$ *// Survey current cell and get the no:of fish tags*
5. *loop do*
6. $\quad currentCell, visited \leftarrow$ ***CSearch***$(currentCell, visited)$ *// Best neighbor of the current cell*
7. $\quad if\ visited.empty()$
8. $\quad\quad break$ *// Stop the search as all the cells have been surveyed*
9. $\quad currentCellPings \leftarrow$ ***UniquePings***$(currentCell)$
10. $\quad previousCell \leftarrow visited.top()$ *// Get previously surveyed cell*
11. $\quad visited.push(currentCell)$ *// Add the current cell to the visited list*
12. $\quad if\ currentCellPings <$ ***UniquePings***$(previousCell)$
13. $\quad\quad visited \leftarrow SortByMaxUniqueTags(visited)$
14. $\quad\quad currentCell \leftarrow previousCell$ *// Agent is at the top of the hill, perform backtracking*

CSearch(searchCell, visited)

15. $N, S, E, W =$ ***GetExpectedPings***$(searchCell)$ *// get north, south, east, west expected fish tags*
16. $SearchCell =$ ***UnvstedMaxCell***$(N, S, E, W, visited))$ *// Unvisited cell with max projected fish tags*
17. $if\ searchCell\ is\ None$ *// When the neighbors of the current cell are visited*
18. $\quad if\ visited.empty()$
19. $\quad\quad return\ searchCell, visited$ *// Agent has surveyed all the cells*
20. $\quad else$
21. $\quad\quad newSearchCell \leftarrow visited.pop()$ *// Backtrack to the neighbors of the previously visited cells*
22. $\quad\quad return$ ***CSearch***$(newSearchCell, visited)$ *// Recursively find the best neighbor*
23. $else$
24. $\quad return\ searchCell, visited$ *// Best neighboring cell with max projected fish tags*

given area of a search domain is proportional to the integral of $\phi(x)$ over that same domain. The metric in Equation (1) quantifies the difference between the spatial statistics of $\phi(x)$ and $x(\cdot)$.

$$\epsilon(x(\cdot)) = \sum_{k=0}^{K} \Delta_k |c_k(x(\cdot)) - \phi_k|^2 \tag{1}$$

ϕ_k and c_k are the Fourier coefficients of the spatial distribution $\phi(x)$ and of the time averaged basis functions along the trajectory $x(\cdot)$. They can be calculated as $\phi_k = \int_x \phi(x) F_k(x) dx$ and $c_k = \frac{1}{T}\int_{t_0}^{t_0+T} F_k(x(t))dt$, where $F_k(x) = \frac{1}{h_k}\Pi_{i=1}^{n} \cos(\frac{k\pi}{L_i}x_i)$ are the Fourier basis functions used to approximate the distributions over n dimensions. h_k is a normalizing factor and x_i is the i-th component of x. K determines the number of coefficients used to measure the distance from ergodicity along each of the n dimensions. $\Delta_k = \frac{1}{(1+||k||^2)^{\frac{n+1}{2}}}$ is used to place larger weight on lower frequency information.

In this work, the spatial information distribution $\phi(x)$ is a function of the mean and variance of a gaussian process generated using acoustic tag ping rates over short periods of time. The ergodic metric in (1) is then optimized using the algorithm in Miller et al. (2015). The algorithm works by

simulating the control system forward in time over a finite horizon, producing a trajectory and the low level control signal that causes the agent to follow this trajectory. The first value of the control signal is applied and the optimization process is repeated based on the current state of the agent and spatial distribution $\phi(x)$.

3.3 Ergodic Search Combined with Structured Search (ESCSS)

As its name suggests, ergodic search combined with structured search combines the two search strategies above. This strategy implements the ES in a single cell, extrapolates the readings obtained at the edges(four directions) in a cell, and performs an SS on the top of ES. In this method, the agent might not always read along all four cell edges because ES is not guided by cell boundaries. The next subsection defines how we build upon the communication established and integrate all three search strategies with goal operations in a cognitive architecture.

3.4 Goal Operations during Search Strategies

As problems arise, the GDA component performs goal operations to solve them. Currently, the system performs two goal operations, although others (e.g., goal change and goal delegation) will be added to the complement in the future.

- *Goal Selection*: selects a current goal set among all the goals in the goal agenda
- *Goal Formulation*: generates new goal(s) when an agent identifies an opportunity or a problem

The functioning of the two goal operations is presented formally in detail in Tables 2 and 3. Table 2 shows the formalism for goal selection (Kondrakunta & Cox, 2021). The goal agenda ($\hat{G}$) represents all the goals of the agent. Each goal in the goal agenda follows first-order predicate-argument ($p^1(obj1, obj2)$) representation. When given $\hat{G}$, the goal selection operation checks for three preconditions ($pre(\delta^{se})$) and results in a single goal or multiple goals for immediate achievement. The first two preconditions ($pre_1(\delta^{se})$, $pre_2(\delta^{se})$) check the validity all goals in $\hat{G}$. The third precondition ($pre_3(\delta^{se})$) checks if the resources are sufficient for all goals. Specifically, the $pre_1(\delta^{se})$ checks if the predicates of each goal belong to the class hierarchy tree (CL), which is present in the domain knowledge of the agent. Bergmann (2003) presents the notation of the CL, where the are leaf classes that have a root class, and the root class has a superclass. The $pre_2(\delta^{se})$, checks if all the objects of the goal belong to the objects in the domain. Every agent needs to keep track of its own resources to ensure achievement of a maximum number of goals. Hence, the $pre_3(\delta^{se})$, evaluates if the estimated resources from the resource estimator function, $ResourcesForGoals(s, \hat{G})$ are sufficient for all goals in $\hat{G}$. The resource estimator function returns continuous values. These values are later converted to qualitative values using expert set thresholds for decision making purposes. Finally, After satisfying all the conditions in $pre(\delta^{se})$ the agent can now output a selected goal(s) ($res(\delta^{se})$) by using an appropriate algorithm. For example the paper presents three different algorithms in Sections 3.1-3.3.

Table 2. Formal representation of goal selection

$\delta^{se}(\hat{G} : G) : G$
$head(\delta^{se}) = selection$
$parameter(\delta^{se}) = \hat{G} = \{p^1(obj1, obj2), p^2(obj3, obj4), ...\}$
$pre_1(\delta^{se}) = p^1 \in CL \wedge p^2 \in CL \wedge ...$
$pre_2(\delta^{se}) = obj1 \in objs \wedge obj2 \in objs \wedge obj3 \in objs \wedge obj4 \in objs \wedge ...$
$pre_3(\delta^{se}) = ResourcesForGoals(s, \hat{G})$
$pre(\delta^{se}) = \{pre_1(\delta^{se}), pre_2(\delta^{se}), pre_3(\delta^{se})\}$
$res(\delta^{se}) = g_c = p^x(objy, objz)$

Table 3 represents the formalism for goal formulation (Kondrakunta & Cox, 2021). Goal formulation also takes in the goal agenda $\hat{G}$, and checks for two preconditions, $pre(\delta^*)$. The first precondition ($pre_1(\delta^*)$) checks if the agent really needs to formulate a goal and the second precondition checks ($pre_2(\delta^*)$) if the resources of the agent are sufficient to include the new goal. Specifically, the $pre_1(\delta^*)$ checks if the agent observed an anomaly ω (i.e., an expectation s_e does not match the observation s_c) and whether there exist a reasonable explanation χ for the anomaly. Next, $pre_2(\delta^*)$ generates an estimation for resources of the new goal $ResourceEstimatesFor(g_n)$ and checks if the agent has sufficient resources for the probable goal, g_n. If both preconditions are satisfied, the result is a new goal g_n to remove the cause $\neg\omega$ of the anomaly, adding it to the agenda. The resulting goal g_n is generated through explanation patterns as outlined in Gogineni et al. (2019).

Table 3. Formal representation of goal formulation

$\delta^*(\hat{G} : G) : G$
$head(\delta^*) = formulation$
$parameter(\delta^*) = \hat{G}$
$pre_1(\delta^*) = \exists\, \chi : \omega \rightarrow (s_e \neq s_c)$
$pre_2(\delta^*) = ResourceEstimatesFor(g_n)$
$pre(\delta^*) = \{pre_1(\delta^*), pre_2(\delta^*))\}$
$res(\delta^*) = g_n = \neg\omega$
$\hat{G} = \hat{G} \cup g_n$

In the Marine Survey domain, an agent uses goal selection to select survey goals and goal formulation to formulate goals in the event of a hot-spot or a remora attack. In our implementation, the ES strategy uses only goal formulation, as ES considers the entire grid to be its survey space, and there is only one goal for the agent. This in turn eliminates the possibility for goal selection. In contrast, SS and ESCSS use both selection and formulation. Goal selection chooses the next cell to survey based on the hill-climbing method, and goal formulation generates a goal when the agent perceives a hot-spot or a remora attack.

3.5 Experimental Design and Results

We implemented the strategies as three different planning agents (SS, ES and ESCSS). The agents search for hot-spots until reaching a deadline of 600 time units. On their own, these agents do not have any mechanism to cope with anomalies. But the agents can handle anomalies and significantly increase performance when paired with goal operations in MIDCA. MIDCA improves the agents' robustness by detecting anomalies during plan execution and dynamically formulating new goals to handle them at run-time. For example, remora attachments prompt goals to clear the remora from the vehicle. Control behavior that achieves such goals include flying backward.

We created three hot-spot patterns or scenarios in the 5x5 survey region: the first contains four hot-spots at the outer corners of the region, the second contains four hot-spots at the inner corners of the region, and the third contains five hot-spots, where four are at the inner corners, and one is at the center of the region. A trial requires an agent to traverse the 25 cells of a scenario starting from a given initial location and find all hot-spots before the deadline. With 100 initial locations for each scenario (each initial location is a specific real-world latitude and longitude co-ordinate, and the co-ordinate could be in any part in each cell). Since three scenarios, an agent performs 300 trials. The output of one trial in this search will yield 25 cells to classified as either hot-spot or not. Therefore, 300 trials will yield 300x25=7500 data points to derive F1 scores[3] for each search strategy.

Table 4 presents results for the strategies with and without anomaly responses, i.e., with just goal selection and with both goal selection and goal formulation. When agents ignore anomalies and perform only goal selection, the ES strategy performs best. That is, a raw control theory technique without goal reasoning proves most successful when ignoring remora attacks. However, all strategies improve when combined with high-level reasoning that handles such situations. But out of the three, ES performed the worst on average, while ESCSS and SS performed equally well with a marginal edge to the former.

Table 4. Search strategies with anomaly response

Goal Operations	**Parameter**	**SS**	**ES**	**ESCSS**
With only selection	Accuracy	0.830	0.839	0.834
	Recall	0.140	0.192	0.152
	Precision	0.538	0.613	0.577
	F1 score	**0.223**	**0.293**	**0.241**
With selection and formulation	Accuracy	0.910	0.875	0.916
	Recall	0.527	0.373	0.566
	Precision	0.923	0.795	0.925
	F1 score	**0.671**	**0.508**	**0.702**

Despite these averages, each strategy has its advantages under certain conditions. For example, SS performs better in the cells located at corners than ES or ESCSS, whereas ES is better in the grid center cells. We claim that such insights can benefit the agent as a heuristic method, and we will

3. F1 is the harmonic mean of precision and recall. Here precision is the fraction of true positives among the instances classified as hot-spots, while recall is the fraction of classified instances among all actual hot-spots.

demonstrate this in the next section. Yet as mentioned previously, optimizing search is not the focus of this paper. Instead, we provide such heuristics as domain-dependent rules and demonstrate the trade offs involved.

Even with such a capability, the performance improvement obtained would be specific to only one type of anomaly (remora). More realistically, multiple anomalies types arise in real-world settings, and sometimes responding to some anomalies is just a strain on resources. For example at the end of a mission, it is more productive to perform goal selection (i.e., survey another cell) than to respond to a remora attack given the limitation of time. Although slower, the platform may find another hotspot; whereas, by spending the effort to remove the remora, the deadline may expire. Therefore, we provide some domain-independent rules to handle such situations and further improve the agents' robustness in the case of unknown anomalies. Furthermore, these general rules also form the basis for choosing between its goal operations. The following section provides the technical details and an evaluation of the result.

4. Agents that Select Goal Operations (ASGO)

To further improve the performance of the agent, we provide the agent with both domain-specific and domain-independent generic rules. The domain-specific rules help the agent improve each of the individual search strategies; the domain-independent rules help the agent be flexible in choosing the right goal operation at the right time. To test the performance of these new rules, we introduce two different kinds of anomalies that exist in the real world. One is the flow in water and blockades; flow displaces the agent from its current location, and blockades hinder the agent's movement from one location to another. Whenever an unknown anomaly occurs, the agent might not be in a position to pursue its current goal. In that case, the agent has two options: Select another goal from the remaining set of goals or formulate a goal to respond to the anomaly. These responses include a knowledge goal to investigate the new anomaly or make the agent enter a safe control mode to protect itself from the unknown anomaly. *Agents that Select Goal Operations (ASGO)* can choose between these two operations, goal re-selection (which is essentially goal selection) and goal formulation, smartly.

Table 5 represents ASGO's method for selecting a goal operation among multiple competing ones. Following the notation in Ghallab et al. (2004), the procedure takes the following inputs: Environment model (Σ), the current observations of the world (s_c), the expected observations of the world (s_e), current goal (g_c) and a goal agenda $(\hat{G} = \{g_1, g_2, ...g_c, ...g_m\})$. ASGO creates a plan $(\pi = \langle \alpha_1, \alpha_2, ...\alpha_n \rangle)$ of n actions to achieve its initial goal (g_c) (line 1) and starts executing each action in the plan. While doing so, it perceives observations (s_c) from the world (line 3) and obtains expected observations (s_e) from the results of the action that is currently being executed (line 5). Later, when it executes all the actions in the plan, ASGO removes the current goal from its goal agenda $(\hat{G})$ (line 17), applies goal selection operation to select a goal from its goal agenda (line 18), and plans to pursue the goal. ASGO follows the process mentioned above until it runs out of resources $(R(s_c))$ (line 2) or when an anomaly occurs. An anomaly is a condition where the observed state significantly differs from the expected state (line 6) (Dannenhauer et al., 2016). When ASGO runs out of resources, it halts. However, when an anomaly occurs and hinders its ability to

Table 5. Method for selecting goal operations. Parameter Σ is the domain knowledge of the agent, s_c is the observed state of the agent, s_e is the expectations of the agent, g_c is its current goal, and $\hat{G}$ is its goal agenda.

ExecGoalOperations **($\Sigma, s_c, s_e, g_c, \hat{G}$)**

1. $\pi \leftarrow \Pi(\Sigma, s_c, g_c)$ *// Plan to achieve the current goal*
2. $while\ R(s_c) > 0\ do$ *// Loop until there are enough resources*
3. $s_c \leftarrow \gamma(s_c, \pi[1])$ *// Execute the first action and add its results to the agent's current state*
4. $s_e \leftarrow s_e \cup pre(\pi[1]) \cup \pi[1]^+ - \pi[1]^-$ *// Agent's expectations from the results of the first action*
5. $\pi \leftarrow \langle \alpha_2, \alpha_3, ...\alpha_n \rangle$ *// Remaining Plan*
6. $if\ s_c \not\models s_e\ then$ *// When Agent's expectations donot match its observations (anomaly)*
7. $g_f \leftarrow \beta(s_c, g_c)$ *// Agent formulates a new goal*
8. $g_s \leftarrow \delta^{s_e}(s_c, \hat{G})$ *// Agent selects a new goal*
9. $g_{aff} \leftarrow$ *AllGoalsAffected*$(s_c, s_e, \hat{G})$ *// Agent estimates goals effected by the anomaly*
10. $if\ g_s\ in\ g_{aff}\ then$ *// When the agent's selected goal is effected*
11. $g_c \leftarrow g_f$ *// Agent prioritizes the formulated goal*
12. $\hat{G} \leftarrow \hat{G} \cup g_c$ *// Agent updates its goal agenda*
13. $else$
14. $g_c \leftarrow g_s$ *// Agent prioritizes the selected goal*
15. $\pi \leftarrow \Pi(\Sigma, s_c, g_c)$ *// Plan to achieve the prioritized goal*
16. $if\ \pi = \langle\rangle$ then *// When the plan is empty*
17. $\hat{G} \leftarrow \hat{G} - g_c$ *// Agent removes the current goal from its goal agenda*
18. $g_c \leftarrow \delta^{s_e}(s_c, \hat{G})$ *// Performs goal selection*
19. $\pi \leftarrow \Pi(\Sigma, s_c, g_c)$ *// Plans to achieve the selected goal*

achieve its current goal, it reasons about the anomaly to formulate a goal (g_f) as a response (line 7). Similarly, since the current goal is no longer viable to pursue, ASGO performs goal selection to pursue a different goal (g_s) from its goal agenda (line 8).

To determine which goal to pursue between the formulated goal g_f and selected goal g_s, ASGO creates an estimate of all the goals affected (g_{aff}) by the anomaly (line 9) and pursues the formulated goal g_f when the selected goal g_s is in the affected goals (lines 10-11). Otherwise, it chooses the selected goal g_s to pursue (line 14). Some limitations of the ASGO agent include: as we add more goal operations, the rules will increase, hence taking a longer retrieval duration.

4.1 Empirical Results with Domain-Specific Rules

Before implementing the ASGO agent, Let us look at an agent that improves each of the individual goal operations mentioned in subsection 3.5 with domain specific rules from the experiments on individual search strategies. The rules are tailored to the domain and are provided by a human expert with the data observations from the previous experiment. The rules are generally tied to the location of the agent in the domain and the resource availability. The major rules include:

- If the agent is in the corner cell and has sufficient resources to achieve all its goals, it should perform goal selection using SS;
- If the agent is in the center cell and has sufficient resources to achieve all its goals, it should perform goal selection using ESCSS;

- If the agent is in any cell and does not have sufficient resources to achieve its goals, it should abandon all the remaining goals and selects ES on the whole search area;

These rules aid both goal operations. Let the new agent using these rules be called *Agent with Improved Goal Operations (AIGO)*.

Table 6 compares the data from four different agents: The first three agents are the same agents mentioned in the previous subsection 3.5, the fourth agent is the AIGO. The data collected is over the pattern presented in Figure 2 with a single hot-spot. The data is collected over 100 different trials by varying the agent's initial location; this experiment provides 2500 data points to calculate the F1 scores. The anomalies in this test case are once again the remora attacks.

Table 6. Search Strategies with Goal Operations

Parameter	**SS**	**ES**	**ESCSS**	**AIGO**
Accuracy	0.968	0.937	0.966	0.998
Recall	0.990	1.000	1.000	0.960
Precision	0.559	0.390	0.543	1.000
F1 score	**0.715**	**0.562**	**0.704**	**0.979**

As the data depicts, agents SS, ES, and ESCSS output a similar pattern of results as in Table 4. ES performed the worst, and both SS and ESCSS performed comparably. The method that outperforms all individual strategies is AIGO, which uses generic rules to improve individual goal operations. If the rules always hold, then the agent's performance is near-perfect. Since the world is dynamic, the rules might not always hold; even if the rules do hold, the agent might not implement them due to unforeseen reasons. In such situations, multiple goal operations co-occur, and the agent must intelligently choose a goal operation to improve performance. Next, we elaborate on responding to such problems by implementing the ASGO agent.

4.2 A Working Example of ASGO in the Marine Survey Domain

An example scenario from the Marine Survey Domain can help us understand the prioritization of goal operations by the ASGO agent. Moreover, this will also present the importance of such prioritization for an agent in the real world.

Figure 3 represents an example scenario in the Marine Survey Domain. As mentioned, there are twenty-five survey goals for the ASGO agent. To test the performance of the ASGO agent, we test the agents' performance against several types of anomalies. First, Remora attacks hinder the agent's movement; second, Blockades (represented in blue lines) hinder the agent's movement from one location to the blocked location. Finally, flow (represented in green arrows) displaces the agent in the direction of the flow.

Consider a scenario where the ASGO agent performs goal selection and surveys the location (6,14). It then encounters a Remora attack. The agent has two choices; select a new survey location (Goal Selection), or formulate a goal to glide backward to respond to the Remora attack (Goal Formulation). Since the Remora attack hinders the agent's movement, affecting all the survey goals, including the goal of selecting any new survey location. So, the ASGO agent, as per the algorithm

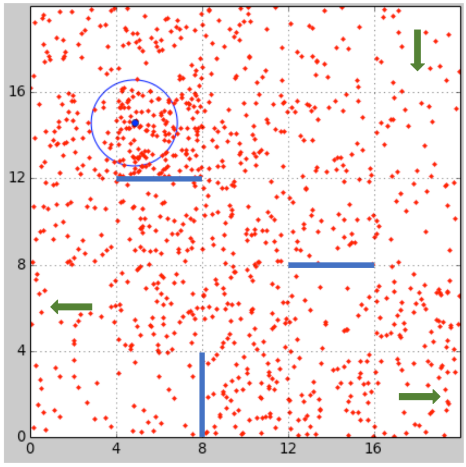

Figure 3. An example in the Marine Survey Domain with Remora attacks and Blockades (indicated by the black Lines). The agent (indicated by the blue dot) is at the location (6,14). The circle around the agent indicates the sensor range for detecting acoustic fish tags (the red dots).

5, prioritizes goal formulation and glides backward to free itself from the Remora attack; and completes the survey in (6,14). After which, it selects the survey goal (6,10). So, the agent must move from its current location (6,14) to the destination location (6,10).

The ASGO agent then encounters a blockade anomaly that hinders movement from (6,14) to (6,10). Thus, it now has two choices; select a new survey location (10,14) (Goal Selection) or formulate a new goal to inspect the entrance to understand the cause of blockade. In this scenario, the blockade does not affect the goal to survey the location (10,14). So, the agent prioritizes goal selection and surveys location (10,14).

Thus, ASGO prioritizes goal formulation and goal selection operation based on the goals affected by the anomaly. If the selected goal is affected, the agent prioritizes goal formulation or else goal selection. Such prioritization helps the agent address its goals promptly.

4.3 Empirical Results with Domain-Independent Rules

We consider two additional anomalies in this test case, namely blockade and flow. Having a single hotspot, the fish distribution scenario for these results is the same as shown in Figure 2. We use the same 100 initial locations as in Section 3.5 and thus obtain 2500 data points. As mentioned, we introduce a new agent that uses domain-specific rules to enhance its performance. The new agent utilizes domain-specific rules to select a best-fitting search strategy corresponding to its location and resources. This new agent has a performance increase compared to the agents that use a single search strategy (SS, ES, ESCSS). We have two versions of such an agent: *Select First* agent, which always gives priority to goal selection, and the *Formulate First* agent always prioritizes goal formulation. We compare their performance against that of the ASGO strategy.

The first set of results in Table 7 compares agent performance when flow anomaly occurs with the existing remora attacks. The second set includes one other kind of anomaly, blockade. The

algorithm is generalizable to any other future anomalies. As the values depict, both Select-1st and Formulate-1st agents perform similarly, but ASGO outperforms them.

Table 7. Agent performance with intelligent decision making between goal operations

Anomalies	**Parameter**	**Select-1st**	**Formulate-1st**	**ASGO**
Remora and flow	Accuracy	0.984	0.983	0.993
	Recall	0.620	0.670	0.840
	Precision	0.984	0.893	0.988
	F1 score	**0.760**	**0.766**	**0.908**
Remora, block and flow	Accuracy	0.981	0.984	0.990
	Recall	0.540	0.650	0.790
	Precision	0.981	0.928	0.975
	F1 score	**0.697**	**0.765**	**0.872**

5. Related research

Similar research work that incorporates both control and cognitive level intelligence includes, the *Control Architecture for Robotic Agent Command and Sensing (CARACaS)*, developed for autonomy in underwater platforms (Huntsberger & Woodward, 2011). CARACaS uses intelligent decision-making to make the unmanned underwater agent robust. CARACaS uses three different components to make such decisions. The first is a behavioral engine which is the core of the system. It is used in real-time for several generic behaviors of the agent. Second is a perception engine which produces maps. Agents use these maps for navigation purposes. Finally, there is a dynamic planner called Continuous Activity Scheduling Planning Execution and Re-planning which is used to handle goal-based planning and re-planning operations on-board. In this work, the burden is on the planner for goal prioritization and goal achievement. However, in our work we use planner for just goal achievement and perform goal selection operation for choosing goals.

Another research effort in underwater platforms that also uses goal reasoning is Wilson et al. (2013a,b). The work formulates goals in unexpected situations using social, opportunity and exploration motivators. It then re-prioritizes the agent's goals based on the new goals. It also presents an approach to integrating high-level intelligent behavior with motion autonomy while extending the work in multiagent scenarios. In addition more recent works from Wilson et al. (2018) presents a goal manager which chooses goals based on a priority value. The goal formulator assigns a priority value when it generates a goal and provides it to the goal manager. Although Wilson shares common interests with the current paper, they do not address the issue of interactions between multiple goal operations. Also, Nelson & Schoenecker (2018) uses goal reasoning to improve a sonar sensor's performance, but avoids the interactions. In addition, Niemueller et al. (2019) uses rule based approach to integrate execution and planning. It uses goal life cycle model (Roberts et al., 2014) to handle goal-mode transitions. However, it does not address the decision making involved when multiple goal-transitions occur.

Goal-driven autonomy and goal reasoning more generally are relatively new frameworks used in research compared to many technical areas of AI, but they are active in the cognitive systems

community. The applications of goal reasoning include its usage in the *Autonomous Response to Unexpected Events (ARTUE)* system (Klenk et al., 2013). Shivashankar et al. (2014) outlines solutions to a number of goal reasoning challenges. This work uses a hierarchical goal network structure to decompose a higher-level goal into several sub-goals to overcome particular limitations using *Motivated ARTUE (M-ARTUE)*. ARTUE and MIDCA agents both employ goal reasoning to identify and respond to unexpected situations in a dynamic environment, both use goal reasoning to perform several operations on their goals. Some of the works mentioned here might not use goal reasoning explicitly, but they share similar goal-based behaviors. Roberts et al. (2014, 2021) presents work on goal-life cycle which closely ties with the idea of goal operations. The work describes phases of a goal from its initial formulation to achievement. It does not address the interaction between multiple goal operations. However, more specifically Gogineni et al. (2019) presents goal formulation through explanation patterns and anomaly detection. Goal formulation based on domain-independent heuristics called motivators (opportunity, exploration, and social), where each motivator is weighed based on urgency and fitness, are presented in M-ARTUE (Wilson et al., 2013b). On similar terms, Karneeb et al. (2018) presents discrepancy detection in air combat agents. The paper focuses on discrepancy detection and response in the real-world. However, none of the works mentioned explicitly dive into the decision making process when multiple kinds of goal operation co-occur.

One reason for the applicability of goal reasoning in such diverse environments is due to its ability to perform various operations on its goals. For example, Rabideau et al. (2011) (inspired by Chien et al. (2005)) defines constraints and priorities to determine which goal to select among the set of all goals. Furthermore, domain-specific information metrics (e.g., distance traveled and time to perform cost estimation) can aid in goal selection (Johnson et al., 2016). This work is adapted and generalized to some domains using cost-benefit analysis (Kondrakunta, 2017; Kondrakunta & Cox, 2021). Trainable-ARTUE Powell et al. (2011) also presents goal selection with expert-based interactive learning. If the system selects a wrong goal, it accepts a penalty.

While the methods presented above take advantage of implementing various goal operations, none of them explicitly shed light on the agent's performance when there is a possibility for interaction of multiple goal operations. This paper addresses the issue by developing a method to prioritize one goal operation over another given the situation in a marine survey domain. The uncertainty in this domain arises from the agent's limited communication when underwater, the unpredictable currents, or the fish attacks. The next section concludes the paper and presents future research.

6. Conclusions and Future Research

In this paper, we explored the structure of decision process when multiple-goal operations co-occur. This paper examines such problems while establishing a communication with the control architecture of the agent. For exploration, the agent performs three search strategies. Initially, we implemented and evaluated each search strategy independently. Subsequently, we introduce goal operations to improve agent performance in the presence of unexpected events. A comparison of agent performance with and without goal operations shows that the agent with goal operations is more robust. To further improve performance, the agent must decide between two goal operations

(i.e., goal selection and goal formulation) or go into a safe control mode. The paper develops a response algorithm that aids the agent to make such decisions and maintain its performance in the dynamic world. We compare the performance of this new agent (ASGO) to two other agents (i.e., Select-1st and Formulate-1st). To support our claims of robustness and generality we introduced two new anomalies (i.e., flow, and blockade). The data support our claims that ASGO outperforms others in a dynamic environment.

We intend to extend this research to include several other goal operations (e.g., goal change and goal delegation) (Kondrakunta et al., 2021) in the future. An agent performs goal change operation when it is not able to achieve its current goal due to changes in environment. Goal delegation is useful in a multiagent scenario, for an agent to pass its goals to a different agent. Specifically, goal change is useful when there is variability in the receiver radius. As mentioned in earlier sections, the difference in the water column's stratification can result in variation of the receiver range. From this information, goal change can help the agent decide the amount of resources it needs to spend on the search, such as a sparse search with a high receiver radius or a denser search with a lower radius. Goal delegation can be particularly useful when the agent drifts off the survey region due to flow conditions, or in case of physical damage. It is smart for the agent to delegate its goals to a different agent to complete the mission. Apart from goal operations, we can also integrate the real world parameters. For example, flow prediction parameter for the path planning agent.

Acknowledgements

The National Science Foundation supported this research under grants 1849131, 1848945, S&AS-1849137, and also by the Office of Naval Research under grant N00014-18-1-2009. We thank the anonymous reviewers for their comments and suggestions.

References

Aha, D., Klenk, M., Munoz-Avila, H., Ram, A., & Shapiro, D. (2010). Goal-directed autonomy. *Notes from the AAAI workshop*. Menlo Park, CA: AAAI Press.

Aha, D. W. (2018). Goal reasoning: Foundations, emerging applications, and prospects. *AI Magazine*, *39(2)*, 3–24.

Antsaklis, P. J., & Michel, A. N. (2006). *Linear systems*. Springer Science & Business Media.

Athans, M., & Falb, P. L. (2013). *Optimal control: an introduction to the theory and its applications*. Courier Corporation.

Bergmann, R. (2003). *Experience management: foundations, development methodology, and internet-based applications*, volume 2432. Springer.

Chien, S., Sherwood, R., Tran, D., Cichy, B., Rabideau, G., Castano, R., Davis, A., Mandl, D., Trout, B., Shulman, S., & Boyer, D. (2005). Using autonomy flight software to improve science return on earth observing one. *Journal of Aerospace Computing, Information, and Communication*, *2(4)*, 196–216.

Cox, M. T., Alavi, Z., Dannenhauer, D., Eyorokon, V., Munoz-Avila, H., & Perlis, D. (2016). MIDCA: A metacognitive, integrated dual-cycle architecture for self-regulated autonomy. *Proceedings of the 30th AAAI Conference on Artificial Intelligence* (pp. 3712–3718). AAAI Press.

Cox, M. T., Dannenhauer, D., & Kondrakunta, S. (2017). Goal operations for cognitive systems. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence* (pp. 4385–4391). San Francisco, CA: AAAI Press.

Dannenhauer, D., Munoz-Avila, H., & Cox, M. T. (2016). Informed expectations to guide GDA agents in partially observable environments. *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence* (pp. 2493–2499). New York, NY: AAAI Press.

Edwards, C., Cho, S., Zhang, F., & Fangman, S. (2020). *Field and numerical studies to assess performance of acoustic telemetry collected by autonomous mobile platforms*. Technical report, National Marine Sanctuaries Conservation Series, Silver Spring, MD.

Ghallab, M., Nau, D., & Traverso, P. (2004). *Automated planning: theory and practice*. Elsevier.

Gogineni, V. R., Kondrakunta, S., Brown, D., Molineaux, M., & Cox, M. T. (2019). Probabilistic selection of case-based explanations in an underwater mine clearance domain. *Proceedings of the twenty seventh ICCBR Conference* (pp. 110–124). Otzenhausen, Germany: Springer.

Huntsberger, T., & Woodward, G. (2011). Intelligent autonomy for unmanned surface and underwater vehicles. *Proceedings of the OCEANS'11 MTS/IEEE KONA* (pp. 1–10). IEEE.

Johnson, B., Roberts, M., Apker, T., & Aha, D. W. (2016). Goal reasoning with informative expectations. *Proceedings of the 4th Annual Conference on Advances in Cognitive Systems* (pp. 12: 1–14). Evanston, IL: Cognitive Systems Foundation.

Karneeb, J., Floyd, M. W., Moore, P., & Aha, D. W. (2018). Distributed discrepancy detection for a goal reasoning agent in beyond-visual-range air combat. *AI Communications*, *31(2)*, 181–195.

Khalil, H. K. (2002). *Nonlinear systems*. Prentice Hall, Upper Saddle River, NJ, 3 edition.

Klenk, M., Molineaux, M., & Aha, D. W. (2013). Goal-driven autonomy for responding to unexpected events in strategy simulations. *Computational Intelligence*, *29(2)*, 187–206.

Kondrakunta, S. (2017). *Implementation and evaluation of goal selection in a cognitive architecture*. Master's thesis, Department of Computer Science and Engineering, Wright State University.

Kondrakunta, S., & Cox, M. T. (2021). Autonomous goal selection operations for agent-based architectures. *Proceedings from Advances in Artificial Intelligence and Applied Cognitive Computing*. Las Vegas, NV: Springer.

Kondrakunta, S., Gogineni, V. R., & Cox, M. T. (2021). Agent goal management using goal operations. *Proceedings of the 9th Goal Reasoning Workshop* (p. in press).

Lin, T. X., Hou, M., Edwards, C. R., Cox, M., & Zhang, F. (2020). Bounded cost htn planning for marine autonomy. *Proceedings of Global Oceans 2020: Singapore – U.S. Gulf Coast* (pp. 1–6). Los Alamitos, CA: IEEE.

Mathew, G., & Mezić, I. (2011). Metrics for ergodicity and design of ergodic dynamics for multi-agent systems. *Physica D: Nonlinear Phenomena*, *240(4-5)*, 432–442.

McQuarrie, F., Woodson, C. B., & Edwards, C. (2021). Modeling acoustic telemetry detection ranges in a shallow coastal environment. *Proceedings of the 14th ACM International Conference on Underwater Networks and Systems (WUWNet'21)* (p. in press). Guangdong, China: ACM.

Miller, L. M., Silverman, Y., MacIver, M. A., & Murphey, T. D. (2015). Ergodic exploration of distributed information. *IEEE Transactions on Robotics*, *32(1)*, 36–52.

Mota, T., & Sridharan, M. (2018). Incrementally grounding expressions for spatial relations between objects. *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 1928–1934). Stockholm, Sweden: IJCAI.

Munoz-Avila, H. (2018). Adaptive goal driven autonomy. *Proceedings of the 26th International Conference Case-Based Reasoning* (pp. 3–12). Berlin, Germany: Springer.

Nelson, J. K., & Schoenecker, S. (2018). Goal driven autonomy as a model for reasoning in cognitive active sonar. *Proceedings of the 2018 IEEE 10th Sensor Array and Multichannel Signal Processing Workshop (SAM)* (pp. 1–5). Sheffield, United Kingdom: IEEE.

Niemueller, T., Hofmann, T., & Lakemeyer, G. (2019). Goal reasoning in the clips executive for integrated planning and execution. *Proceedings of the International Conference on Automated Planning and Scheduling* (pp. 754–763). Berkeley, CA: AAAI Press.

O'Reilly, U.-M., & Oppacher, F. (1994). Program search with a hierarchical variable length representation: Genetic programming, simulated annealing and hill climbing. *International Conference on Parallel Problem Solving from Nature* (pp. 397–406). Jerusalem, Israel: Springer.

Powell, J., Molineaux, M., & Aha, D. W. (2011). Active and interactive discovery of goal selection knowledge. *Proceedings of the Twenty-Fourth International FLAIRS Conference* (pp. 413–418). Palm Beach, FL: AAAI Press.

Rabideau, G., Chien, S., & McLaren, D. (2011). Tractable goal selection for embedded systems with oversubscribed resources. *Journal of Aerospace Computing, Information, and Communication*, *8(5)*, 151–169.

Roberts, M., Borrajo, D., Cox, M., & Yorke-Smith, N. (2018). Special issue on goal reasoning. *AI Communications*, *31(2)*, 115–116.

Roberts, M., Hiatt, L. M., Shetty, V., Brumback, B., Enochs, B., & Jampathom, P. (2021). Goal lifecycle networks for robotics. *The International FLAIRS Conference Proceedings* (p. in press).

Roberts, M., Vattam, S., Alford, R., Auslander, B., Karneeb, J., Molineaux, M., Apker, T., Wilson, M., McMahon, J., & Aha, D. W. (2014). Iterative goal refinement for robotics. *Planning and Robotics: Papers from the ICAPS Workshop*. Jerusalem, Israel: AAAI Press.

Shivashankar, V., Kaipa, K. N., Nau, D. S., & Gupta, S. K. (2014). Towards integrating hierarchical goal networks and motion planners to support planning for human-robot teams. *AAAI Fall Symposium Series* (pp. 139 – 141). Arlington, VA: AAAI Press.

Sidhik, S., Sridharan, M., & Ruiken, D. (2020). Hybrid models for control of changing-contact manipulation tasks. *Proceedings of the Eighth Annual Conference on Advances in Cognitive Systems*. Menlo Park, CA: Cognitive Systems Foundation.

Silverman, Y., Miller, L. M., MacIver, M. A., & Murphey, T. D. (2013). Optimal planning for information acquisition. *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 5974–5980). Tokyo, Japan: IEEE.

Van Riemsdijk, M. B., Dastani, M., & Winikoff, M. (2008). Goals in agent systems: A unifying framework. *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2* (pp. 713–720).

Wilson, M., Auslander, B., Johnson, B., Apker, T., McMahon, J., & Aha, D. W. (2013a). *Towards applying goal autonomy for vehicle control*. Technical report, Naval Research Lab, Navy Center for Applied Research in Artificial Intelligence, Washington, DC.

Wilson, M. A., McMahon, J., Wolek, A., Aha, D. W., & Houston, B. H. (2018). Goal reasoning for autonomous underwater vehicles: Responding to unexpected agents. *AI Communications*, *31(2)*, 151–166.

Wilson, M. A., Molineaux, M., & Aha, D. W. (2013b). Domain-independent heuristics for goal formulation. *Proceedings of the Twenty-Sixth International FLAIRS Conference* (pp. 160–165). St. Pete Beach, FL: AAAI Press.