The 2016 AAAI Fall Symposium Series:
Cross-Disciplinary Challenges for Autonomous Systems
Technical Report FS-16-03

Courses of Action Display for Multi-Unmanned Vehicle Control: A Multi-Disciplinary Approach

Michael Hansen, Gloria Calhoun, Scott Douglass
{michael.hansen.24,gloria.calhoun,scott.douglass.1}@us.af.mil
Air Force Research Laboratory
711 HPW/RHCI
Dayton, OH 45433 USA

Dakota Evans
dakota.evans@udri.udayton.edu
University of Dayton Research Institute
300 College Park
Dayton, OH 45469 USA

Abstract

Operational concepts in which a single operator teams with multiple autonomous vehicles are now considered feasible due to advances in automation technology. This will require that an operator be able to express a high-level intent, or goal, to the vehicle team rather than direct the actions of individual assets. Successful operator-autonomy collaboration must quickly capture the operator's intent and then portray the autonomy's trade-offs between different courses of action in an intuitive interface. This paper describes how a multi-disciplinary effort was employed in the design of a display that highlights the trade-off of autonomy-generated plans and supports the efficient allocation of assets to surveillance tasks. Our novel control station approach combines domain modeling and multi-objective optimization with innovative interfaces to enable a single operator to effectively command a team of unmanned vehicles.

Agility in tactical decision-making, mission management, and control is a key requirement for human and heterogeneous unmanned vehicle (UxV) teams to manage the "fog of war" with its inherent complex, ambiguous, and time-pressured conditions. A recent effort focused on command and control (C2) autonomy, which includes a human operator in the real-time mission decision loop. This involved developing a decision-aiding agent technology as well as a human-autonomy interface paradigm by which an operator commands multiple heterogeneous unmanned vehicles. These technologies enable operators to monitor and instruct the autonomy in response to dynamic environments and missions as well as allow the autonomy to make suggestions to the operator and provide rationale for generated plans. Facilitating operator-autonomy interaction is a key challenge for achieving trusted, bi-directional collaboration.

To enable operator-autonomy teamwork in control of multiple unmanned vehicles, a novel adaptable automation approach has been developed through which an operator makes inputs that determines the degree to which tasks are automated and how they are performed. Specifically, the operator communicates to a decision-aiding agent by calling a "play" (a high-level command) that is translated into a series of automation tasks for one or more unmanned vehicles to perform. For instance, a '*Monitor Target Alpha*' verbal or manual command from the operator prompts the agent to reason about which unmanned vehicle(s) should optimally be assigned to the play (e.g., those with appropriate sensors, tracking capabilities, etc.). Plays are represented formally (domain modeling) and, when combined with operator constraints and situational factors, allow the agent to enumerate all possible courses of action (COAs) that would fulfill the operator's intent. The agent then ranks these COAs according to multiple optimization criteria and presents them, along with their trade-offs, in a specialized display to the operator. The operator's approval results in the automated completion of that play.

This approach to operator-autonomy teaming was developed and implemented in a high-fidelity ground control station for evaluation using a simulated security force operations scenario in which a single operator manages 12 unmanned air, ground, and water surface (4 each) vehicles. Data collection is currently underway to confirm that operator workload is improved by the use of high-level plays and agent-aided decision making.

The objectives of the present paper are to provide details on how this approach was implemented and illustrate the cross-disciplinary contributions to the resulting capability. First, an introduction to the domain modeling formalism employed by the decision-aiding agent will be provided. Contributions from computational computer scientists are further shown by an explanation of multi-objective optimization, which forms the basis of ranking candidate courses of action (possible ways of fulfilling a play). Next, we illustrate how a specific play is instantiated in the domain modeling formalism, and then used to produce and rank candidate courses of action for the human operator to consider. Lastly, the contribution of human interface developers is illustrated with explanations of how the operator's intent is captured and how the agent's solutions are communicated to the human-operator team member.

Domain Modeling

The first step in transforming play calls (operator intent) into possible courses of action is formally capturing the domain of interest. We employ feature-oriented domain analysis, an approach which defines a domain in terms of its common aspects as well as the differences between related systems in the domain (Kang et al. 1990). Specifically, we specify domain models as a hierarchy of related entities and constraints. Using an automated code generation process,

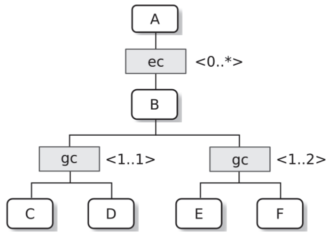


Figure 1: Simple domain with 6 events (A-F), 1 event cardinality (ec) relation, and 2 group cardinality (gc) relations.

these domain models are transformed into a dynamic constraint satisfaction problem (Mittal and Falkenhainer 1990) that can be automatically solved using an off-the-shelf constraint solver like MiniZinc (Nethercote et al. 2007). This section provides the details of our domain formalism, including the constraint language used to shape a domain by pruning potential solutions from the constraint solver.

A domain model in the IMPACT decision-aiding agent consists of *events* and two types of *relations*, alternating in a tree with an event as the root[1]. Events may contain named *attributes* whose values are read or written by constraints. The group cardinality relation allows for between n and m child events to be present in a given solution. When n and m are both 1, for example, the relation behaves like an exclusive-or (exactly 1 child in each solution) rather than a conjunction (all children) or disjunction (at least n children, at most m). The event cardinality relation always contains a single child (template) event, and requires between n and m copies or *instances* of this event be present in all solutions. Each instance may have different sub-events present in a given solution or different values for its attributes. In the space of existing feature model systems, our domain formalism is both *extended* (supports attributes) and *cardinality based* (has an event cardinality relation) (Benavides, Segura, and Ruiz-Cortés 2010). This flexibility captures rich domains, but still retains the formal rigor of feature modeling.

Figure 1 contains a simple domain with 6 events (A-F) and 3 relations (1 event cardinality, 2 group cardinality). The mins and maxes are displayed to the right of each relation ($n..m$). In this domain, any number of B's may be present (each with a copy of the sub-structure). Each B may contain *either* a C or D, and may contain E, F, or both E and F. An example solution is shown in Figure 2 with 4 B's (excluded sub-events are shown but shaded).

Without additional constraints, the domain in Figure 1 would have a total of $(2 \times 3)^4 = 1296$ solutions, assuming the event cardinality under A has 4 copies of B (i.e., $n = m = 4$). A simple constraint, like C and E cannot both be present in any B, would reduce this number to 256 (because each B now only has 4 valid configurations instead of 6). Using domain-relevant constraints, the enumerated space

[1]The term *event* reflects the decision-aiding agent's use of a complex event processing system as a working memory (Eugster et al. 2003). They are typically referred to as *features*.

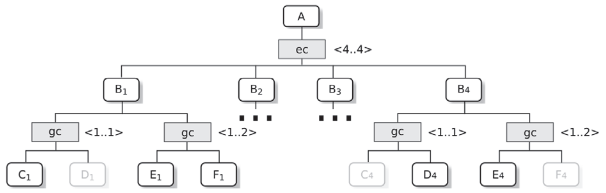


Figure 2: Example solution from the domain in Figure 1. Shaded events are not present.

of solutions can be shaped to match modeling needs. Parallel algorithms for quickly searching very large solution spaces have been developed and accelerated in hardware (Atahary, Taha, and Douglass 2016).

Non-Determinism and Constraints

By itself, a domain model represents only the structure of all possible solutions – which events may be present in a given solution. Group cardinalities whose mins (n) and maxes (m) do not equal the number of child events under them (c) capture domain non-determinism ($n = m = c$ does not hold). With non-determinism in a domain, different solutions may contain very different sub-structure under these relations, depending on the n and m of each relation and any additional constraints.

Domain and situational constraints shut down non-determinism, by either pruning impossible (domain) or irrelevant (situational) solutions. We express both kinds of constraints with a logical language that captures first-order relationships between events and attributes (Table 1) as well as second-order relationships across event cardinality instances (Table 2). For example, the sample constraint from the previous section (C and E cannot both be present in any B from the domain in Figure 1), could be expressed as $every(X, C \implies \neg E)$.

Constraint	**Description** (C = Constraint, X/Y = Event)
$X[i, j]$	Asserts that X must be present in all solutions. Indexes i, j, etc. specify event cardinality indexes from the root.
$\neg C$	If C is true, exclude solution.
$C_1 \wedge C_2$	Both C_1 and C_2 must be true.
$C_1 \vee C_2$	Either C_1 or C_2 (or both) must be true.
$C_1 \implies C_2$	If C_1 is true, then C_2 must be true.
$C_1 \iff C_2$	C_1 must be true if, and only if, C_2 is true.
$X.a\ OP\ Y.b$	Attributes a and b of X and Y are related by operation OP (e.g., $=$, $<$).

Table 1: First-order constraint language for domain models.

Attribute values may be compared or overwritten by constraints, but they are *not treated as constraint variables*. In other words, the constraint solver will not enumerate an attribute's domain when generating solutions. This was a deliberate design decision, allowing us to isolate all non-determinism to events under group cardinality relations.

Constraint	Description (C = Constraint, E = Event Cardinality)
$every(E, C)$	C holds for all events under E.
$atLeast(n, E, C)$	C holds for at least n events under E.
$atMost(n, E, C)$	C holds for at most n events under E.
$exactly(n, E, C)$	C holds for exactly n events under E.

Table 2: Second-order constraint language for domain models.

Multi-Objective Optimization

While all solutions from a domain model will necessarily be constraint-compliant, some may be preferred more than others. Using objective functions, the "value" of each solution can be computed, allowing solutions to be *compared and ranked*. When more than one objective function is used, finding the "best" solution becomes a multi-objective optimization problem (Deb 2001) with autonomy-generated courses of action across six different conflicting objectives. By default, total minimized vehicle arrival time is the only objective for optimization. However, the objectives of minimized fuel usage, maximized stealth, maximized presence, maximized crowd effectiveness, and maximized track effectiveness can be preferred.

An important consideration in multi-objective optimization is the normalization of objective values (Yoo and Harman 2007). Because these values may be compared or combined, they must exist on the same scale (with the same units). Normalization is highly dependent on the objective function and the intended comparison/combination method. A common method is to simply scale all values relative to their observed minimums and maximums. This may artificially inflate or obscure differences between solutions (e.g., milliseconds difference appears as significant as a difference of kilometers).

Many different ways of combining objective function values exist, such as the scalarization technique or weighted sum method (Deb 2001). We use the latter, which involves normalization across each specified objectives of interest (assuming equal weight for all selected objectives) and a summing of the normalized values. Additionally, ranking is influenced by the National Imagery Interpretability Rating Scale (NIIRs) size target rating (Kim, Kim, and Kim 2008), environmental conditions, asset allocability, and other factors (see Table 3 for details). Some values, such as presence and tracking effectiveness, are based on an integral rating scale derived from each vehicle's type and payload. If a given solution has the best value across *all objective functions*, then it is said to be Pareto optimal. Multiple Pareto optimal solutions may exist in a set of solutions, but a decision maker must ultimately evaluate and accept a single course of action.

Courses of Action

A play represents an operator's high-level intent for a surveillance task, possibly involving multiple, cooperating vehicles. Figure 3 shows a domain model that captures plays

Objective	Description
Time	Estimated arrival and execution times of the play.
Fuel	Available fuel at play completion using a linear burn rate based on the vehicles' flight profile.
Stealth	Sound intensity of vehicles' engines at mission altitude.
Presence	Payload and size-based vehicle abilities to project force.
Tracking	Payload-based vehicle ability to track targets.
Crowd Control	Payload-based vehicle ability to control crowds with less-than-lethal measures.
Environment	Sensor and maneuverability-based vehicle ability to perform mission under ambient conditions (e.g., fog).
Target Size	Sensor-based vehicle ability to detect targets of an operator-provided NIIRS size using the GIQE equation (Smith et al. 1999).
Allocability	Whether or not vehicles can be immediately assigned to plays.

Table 3: Agent objective functions. Solutions are sorted by the objectives' summed, normalized values.

for point inspect and patterned search plays. Each play has a set of mission requirements (e.g., requiring a specific sensor), and an allocation of unmanned assets. The details of each vehicle are shown in Figure 4, which includes its type (air, ground, surface) and whether or not it will participate in the play. When a specific play is "called" by the operator, the agent *asserts* information into its domain model. For example, '*point inspect at alpha with IR*' would constrain the domain model with a `Sensor` mission requirement, a `Point Inspect` task type, and a `Target` with point alpha's latitude/longitude (previously briefed).

Because in-situ play and vehicle details will fully determine most group cardinality choices, the non-determinism in this play domain model will **only** capture uncertainty about *vehicle participation* in the play (the `Active`/`Not Active` choices in Figure 4). When run through the constraint solver, each solution (a constraint-compliant assignment of choices to group cardinalities) will be a possible vehicle allocation, or course of action (COA), that could be taken to achieve the operator's intent. Each COA can then be ranked according to its objective values (Table 3), and presented to the operator in a specialized display (Figure 9) for consideration and approval. The agent's domain model includes a-priori domain constraints on vehicle-task eligibility – e.g., surface vehicles cannot participate in ground surveillance tasks – as well as details of the current vehicle states (positions, fuel, sensors), so the operator can be assured that

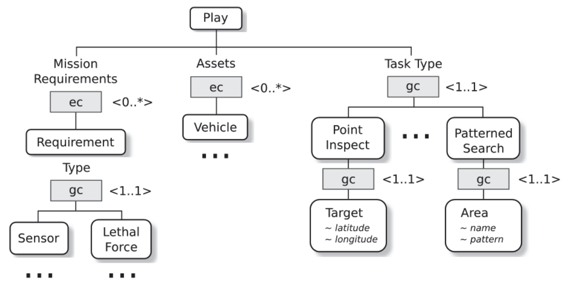


Figure 3: Example play domain with mission requirements, autonomous assets, and task details.

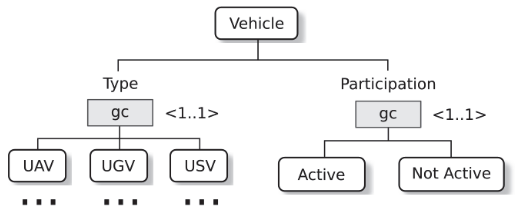


Figure 4: Vehicle sub-domain from Figure 3 with fact sheet information and participation group cardinality.

the suggested COAs comply with actual vehicle capabilities, mission restrictions, and doctrine.

Human-Autonomy Interfaces

For envisioned future multi-UxV command and control, operators will communicate intent to the agent by calling a play that defines what defense mission related task needs to be completed, as well as where the task needs to be completed. Play calling can be initiated with a speech command or selection of an icon on one of the Play Calling Interfaces (Calhoun et al. 2017). Figure 5 explains the symbology used to denote play type (center of each icon) and UxV (shape and location of symbols on exterior circle) in the interfaces. For example, the interface shown in Figure 6 includes an icon for each of the twenty-five types of plays currently supported. Here, the icons are arranged by play type (point, route, area, "target plays" (non-threat versus potential threats)). In contrast to this Play Calling Interface that is always available overlaid on a map, Play Radial menus can be called up by clicking or touching a vehicle symbol or location on a map (Figure 8). This interface approach filters plays such that only the options relevant to that vehicle or location are presented (e.g., the menu does not include ground-based plays for locations that are in the water). With the specification of task type (and location), the agent can generate COAs based on pre-established defaults for each play type (e.g., for "Air Inspect" at a point, the default settings are: environment-unobscured, optimize-time, priority-high, standoff distance-0 m, loiter radius-1000 m, length-200 m, direction-clockwise). However, to tailor the generated COAs to current conditions, the operator can communicate constraints to the agent via the Play Workbook that

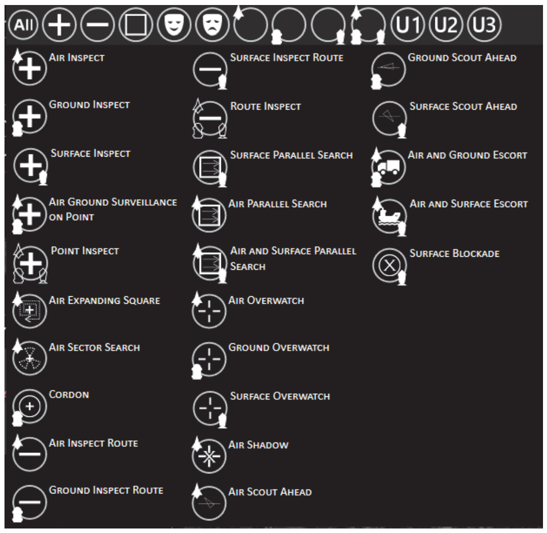


Figure 5: Icons and descriptions of all agent-supported plays.

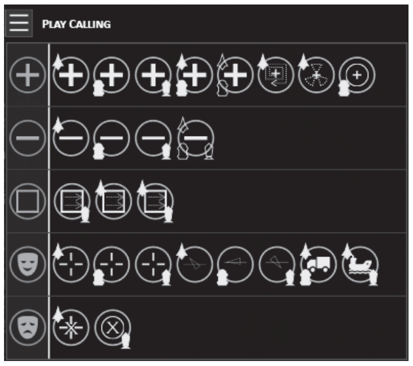


Figure 6: Play calling interface. The center icon indicates the play type. Peripheral icons indicate vehicle type.

is displayed after the play is called (Calhoun et al. 2017).

Figure 7 shows the play workbook for a surface inspect play call (asking an unmanned sea vehicle to inspect a lat/lon). This workbook view contains rows of icons that represent default mission constraints (e.g., an infra-red sensor must be used) as well as additional operator knowledge that the autonomy can use for more relevant COA suggestions (e.g., there is smoke at the target lat/lon). The highlighted icons on the right indicate that the weather is sunny (minimal clouds), that time is the primary optimization criteria, and that lethal force is necessary to complete the mission.

The details of the play call (type, location, etc.) and the operator restrictions/additional knowledge from the workbook are provided to the agent along with the current states of all vehicles. Each piece of information will either set event attributes in the domain model or reduce non-determinism by closing off possible group cardinality choices. Before recommending courses of action, the only non-determinism remaining in the domain model is regard-

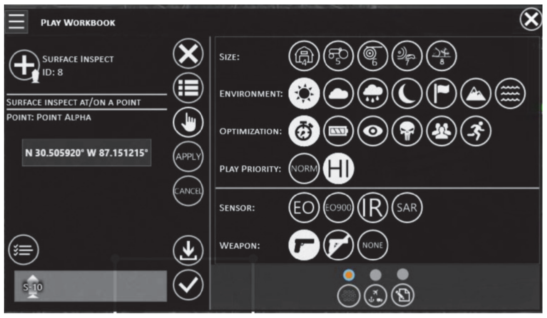


Figure 7: Play workbook example for a Surface Inspect play call. Play constraints and objectives are shown on the right.

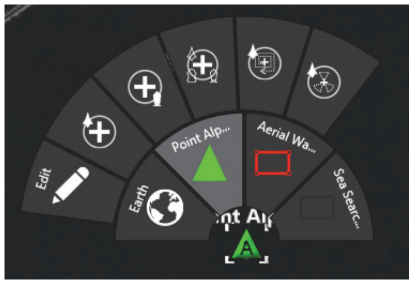


Figure 8: Play radial menu for calling plays directly on a vehicle or at a specific location.

ing *vehicle participation* in the play. Each solution, therefore, is a different constraint-compliant allocation of vehicles.

COA Display

Visualizing the trade-offs of different courses of action becomes challenging as more factors are considered by the decision-aiding agent. An approach that allows a comparison of N COAs across M parameters is needed. We've adapted a parallel coordinate plot (Edsall 2003) such that each COA is represented by a line that passes through a series of parallel axes, with each axis representing a different objective function (Behymer et al. 2014). In this manner, all the objective functions can be compared: estimated time of execution (ETE), fuel (ENERGY) needed, stealth (DETECT), presence (FORCE), effectiveness in managing a crowd (CC), and ability to track an object (TRACK).

In the COA Comparison Interface (Figure 9), each mission related objective is assigned a column, and each candidate plan is assigned a unique color and shape for the plotted points. For this example, the COA indicated by the bottom left purple plus sign is slower than all other COAs in terms of estimated time enroute. However, this COA, as well as the yellow/square coded COA, are the only two options that rate high on three of the mission parameters. The values of each column are normalized on a scale from 0 to 100, with 100 being the ideal value for the underlying objective function. This allows an overall score for each plan ("AVG") column to be represented. Selecting one of the COAs also calls up a dashed route on the map and additional textual explanations

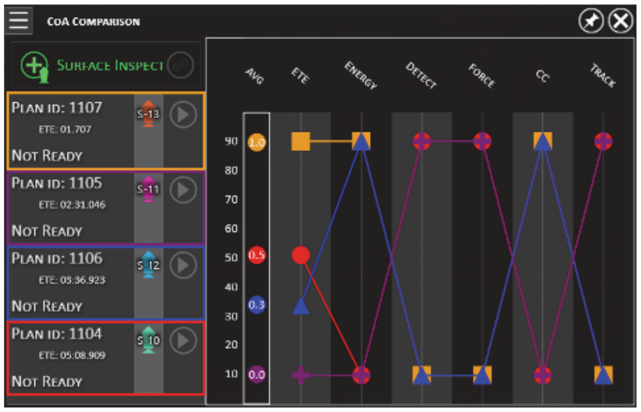
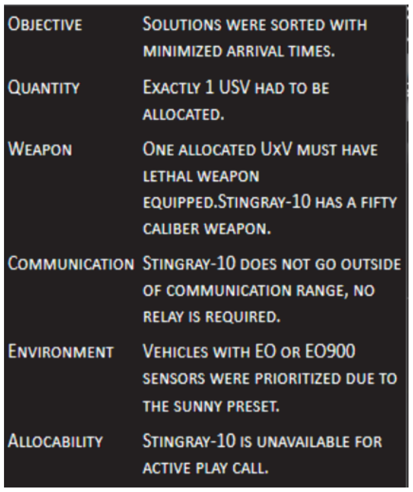


Figure 9: Course of action comparison for multiple surface inspect plans. The outlining colors of each plan match the plot colors.



Objective	Solutions were sorted with minimized arrival times.
Quantity	Exactly 1 USV had to be allocated.
Weapon	One allocated UxV must have lethal weapon equipped.Stingray-10 has a fifty caliber weapon.
Communication	Stingray-10 does not go outside of communication range, no relay is required.
Environment	Vehicles with EO or EO900 sensors were prioritized due to the sunny preset.
Allocability	Stingray-10 is unavailable for active play call.

Figure 10: Explanations for a single course of action.

from the decision-aiding agent (Figure 10).

Conclusion and Future Work

The interfaces described herein enable bi-directional communication between the operator and the decision-aiding agent. The operator informs the agent of intent with respect to vehicle control and any constraints related to the current situation. The agent, via the interfaces, provides feedback on how the play call was interpreted, enumerates the possible COAs that fulfill the operator's specified intent, ranks these options to facilitate comparison, and provides additional explanations. The subjective data collected in simulation to date employing the operator-autonomy team approach have been very positive. Personnel familiar with military base defense and/or unmanned vehicle operation have rated the interfaces as very easy to learn, easy to use, and well integrated into the station. They have also commented that the flexible control approach with the intelligent agent support would be a great aid to workload and help operators maintain situation awareness. One commented that this approach would be "spectacular" for force protection.

Evaluations are on-going and it is anticipated that comments from the experimental participants will further inform interface design and how the agent supports single operator control of multiple unmanned vehicles. Meanwhile, agent and interface developers are already considering improvements to support operator-agent communication and establishment of a shared situation awareness. For instance, the temporal component of vehicle operations could be better supported. It is desirable that the interfaces support communication of desired temporal constraints to the agent (e.g., to have a weaponized vehicle strike a hostile at Gate Charlie, followed 10 minutes later by a vehicle with an appropriate sensor to capture imagery for damage assessment). Similarly, the agent's reasoning could be expanded to support scheduling of assets that perform periodic random antiterrorism measures (e.g, capture image of armory every 15 minutes). These advancements may involve creation of new interfaces (e.g., a temporal view of ongoing and projected actions of each vehicle and associated play) or modifications of existing interfaces (e.g, the ability for an operator, via the COA Comparison Interface (Figure 9), to adjust the weights of the different objectives for the multi-objective optimization process).

Any changes to the human-autonomy interfaces, domain modeling, or multi-objective optimization will be followed by experimental sessions in which human participants will perform tasks in the simulation with both objective and subjective data collected. In this manner, the utility and value of any developments that influence the operator-agent dialogue will be verified. Multi-disciplinary team members are involved in the evaluation cycles, and have successfully demonstrated the value of this approach in the development of COA-centric interfaces for multi-unmanned vehicle control stations.

Acknowledgments

This development supports The Office of the Assistant Secretary of Defense for Research and Engineering (ASD(R&E)) Autonomy Research Pilot Initiative (ARPI) titled "Realizing Autonomy via Intelligent Adaptive Hybrid Control" that is developing an Intelligent Multi-UxV Planner with Adaptive Collaborative/Control Technologies (IMPACT).

References

Atahary, T.; Taha, T. M.; and Douglass, S. A. 2016. Parallelized mining of domain knowledge on GPGPU and Xeon Phi clusters. *The Journal of Supercomputing* 1–25.

Behymer, J.; Mersch, E.; Ruff, H.; Calhoun, G.; and Spriggs, S. 2014. Unmanned vehicle plan comparison visualizations for effective human-autonomy teaming. In *6th International Conference on Applied Human Factors and Ergonomics (AHFE) Conference*.

Benavides, D.; Segura, S.; and Ruiz-Cortés, A. 2010. Automated analysis of feature models 20 years later: A literature review. *Information Systems* 35(6):615–636.

Calhoun, G. L.; Ruff, H. A.; Behymer, K. J.; and Mersch, E. M. 2017. Operator-autonomy teaming interfaces to support multi-unmanned vehicle missions. In Savage-Knepshield, P., and Chen, J., eds., *Advances in Human Factors in Robots and Unmanned Systems*. Springer. 113–126.

Deb, K. 2001. *Multi-objective optimization using evolutionary algorithms*, volume 16. John Wiley & Sons.

Edsall, R. M. 2003. The parallel coordinate plot in action: design and use for geographic visualization. *Computational Statistics & Data Analysis* 43(4):605–619.

Eugster, P. T.; Felber, P. A.; Guerraoui, R.; and Kermarrec, A. 2003. The many faces of publish/subscribe. *ACM Computing Surveys (CSUR)* 35(2):114–131.

Kang, K. C.; Cohen, S. G.; Hess, J. A.; Novak, W. E.; and Peterson, A. S. 1990. Feature-oriented domain analysis (foda) feasibility study. Technical report, DTIC Document.

Kim, T.; Kim, H.; and Kim, H. 2008. Image-based estimation and validation of NIIRS for high-resolution satellite images. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 37:B1.

Mittal, S., and Falkenhainer, B. 1990. Dynamic constraint satisfaction. In *Proceedings Eighth National Conference on Artificial Intelligence*, 25–32.

Nethercote, N.; Stuckey, P. J.; Becket, R.; Brand, S.; Duck, G. J.; and Tack, G. 2007. Minizinc: Towards a standard cp modelling language. In *International Conference on Principles and Practice of Constraint Programming*, 529–543. Springer.

Smith, S. L.; Mooney, J.; Tantalo, T. A.; and Fiete, R. D. 1999. Understanding image quality losses due to smear in high-resolution remote sensing imaging systems. *Optical Engineering* 38(5):821–826.

Yoo, S., and Harman, M. 2007. Pareto efficient multi-objective test case selection. In *Proceedings of the 2007 International Symposium on Software Testing and Analysis*, 140–150. ACM.