Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

DATA MODEL AND SIMULATION FOR PERSISTENT MISSION PLANNING WITH ENERGY-SHARING AUTONOMOUS GROUND AND AIR VEHICLES

James Humann[1], Steven Carlos Ortega[2], James Glenn[3], Jack L. Folsom[4], Subramanian Ramasamy[5], Md. Safwan Mondal[5], Pranav Bhounsule[5], Jean-Paul Reddinger[1], and James Dotterweich[1]

[1]DEVCOM Army Research Laboratory, Aberdeen Proving Ground, MD, USA
[2]Air Force ROTC, University of Texas, Austin, TX, USA
[3]Air Force ROTC, Texas A&M University, College Station, TX, USA
[4]Air Force ROTC, Texas Tech University, Lubbock, TX, USA
[5]Robotics in Motion Laboratory, University of Illinois-Chicago, Chicago, IL, USA

ABSTRACT

We introduce the Energy-Aware Mission Planning (EAMP) problem in the context of long-endurance robotic deployments using a mixture of ground and air vehicles. Unmanned ground and air vehicles have complementary strengths, namely long battery life in the ground vehicles and maneuverability in the air vehicles. To facilitate coordinated mission planning that allows the air vehicles to dock and recharge on the ground vehicles, we introduce a routing problem and specification of solver inputs and outputs. We then incorporate a solver from the literature to create plans for a realistic scenario, and show the results in a simulated 72 h deployment.

1 INTRODUCTION

Unmanned ground and air vehicles (UGVs and UAVs) can be teamed to fulfill complex missions, but their differing battery capacities cause difficulty in coordination. We can potentially take advantage of their complementary strengths, the long battery life of the UGVs and the speed and maneuverability of the UAVs, to perform tasks that neither robot type could perform alone. This is especially advantageous in long-endurance missions, where it is assumed that a UAV would have to be recharged many times. We deepen the interaction between the two agent types by allowing power sharing: the UAVs may dock on the UGVs and draw power to recharge their own batteries. This allows much longer intervals of robotic deployment without human interaction for recharging, and helps alleviate the challenges of the mismatch between ground and air vehicle endurance.

Such systems are important for persistent monitoring. Other potential applications of heterogeneous agent teams include delivery (Carlsson and Song 2018) and disaster relief (Luo et al. 2022). Path planning for coordinated air and ground robots is a very difficult problem to solve because it is a generalization of the NP-hard Traveling Salesman Problem (TSP) (Carlsson and Song 2018; Mondal et al. 2023a).

1.1 Motivation

The ultimate goal of EAMP solvers is to output valid plans that can be loaded onto UGVs and UAVs. A simple API for EAMP solvers ensures that any plans can be readily translated to the format needed for simulation or deployment. This is especially important in the case of EAMP solver research, which is multidisciplinary and includes many independent organizations. We seek to create a data model so that diverse researchers' solvers can be analyzed, compared, and ported to hardware with little to no modification.

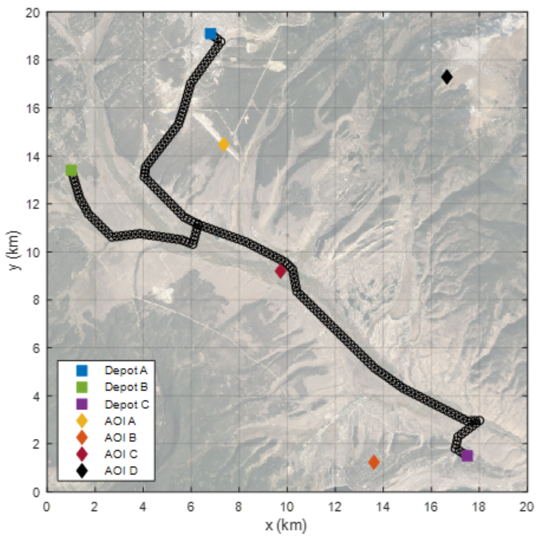

Figure 1: Example mission field. All points along the road network and the remote AOIs need to be monitored as personnel traverse this hilly terrain.

1.2 Guiding Scenario

Consider a field with a number of sites that must be continually visited by robotic agents over a long period of time (a duration that far exceeds the typical battery life of the agents). This could describe site monitoring, foraging, or infrastructure inspections.

In this work, we assume that points of interest (POIs) throughout a field must be visited repeatedly for 72 h using a mixture of small UAVs and UGVs to ensure the safety of personnel passing through the field. The UGVs are slow but have large batteries, so they can outlast the UAVs on a single charge and even recharge the UAVs through energy sharing. The UAVs are fast and agile, but constrained by their relatively short battery life. Through intelligent, energy-aware mission planning, we attempt to take advantage of the agents' complementary strengths to plan routes that continually visit all POIs without any agent's running out of battery. An example of a typical field monitoring scenario is given in Figure 1.

2 OVERVIEW AND CONTRIBUTIONS

We proceed with related work in Section 3, including our own work on developing solvers for EAMP. The core of this paper, in Section 4, gives our data model for states and plans. We then introduce the simulation in Section 5 and example solvers in Section 6. We present the results in Section 7, and discuss the results and future work in Section 8.

Our main contribution in this paper is a general representation of states and plans for EAMP. These can be used by any solver to ensure compatibility among the diverse approaches to this multidisciplinary planning problem. They bridge the solvers from our prior work with the newly developed simulation. We also present results from a 72 h mission that has not been reported in the prior literature. Our final contribution is improvements to the planners described in (Mondal et al. 2023a; Ramasamy et al. 2023).

3 RELATED WORK

The two-echelon vehicle routing problem has been studied in several papers. When a drone is paired with a ground vehicle, it is sometimes referred to as the "flying sidekick" problem.

Wen and Wu (2022) considered a conceptually similar problem with a "mother drone" carrying and deploying smaller drones, which in turn deliver packages to dispersed clients. They used fuzzy C-means clustering to set the mother drone's waypoints, and dynamic planning to set the smaller delivery drones' routes. Another similar approach to paired truck-drone deliveries is given in (Luo et al. 2022), where iterated local search was used to optimize routes for a single truck paired with a single drone in a food delivery application with multiple clients.

Long et al. (2023) consider an emergency response scenario where multiple trucks are each paired with two drones to visit task sites. They used tabu search with tailored local search heuristics to solve problems with 15–100 tasks. Carlsson and Song (2018) investigated the efficiency gains from pairing a UAV with a truck for package delivery. Through numerical simulation, they approximated that the improvement in efficiency is proportional to the square root of truck:UAV speed ratio.

Liu et al. (2019) solved the two-echelon cooperative routing problem without energy sharing. They used Mixed Integer Linear Programming, and heuristics for medium-to-large problems. The first heuristic was to solve large routes using classical TSP approaches and no endurance constraints, and then to split these into smaller routes respecting the endurance constraints. The second heuristic was to cluster routes and then check to see if the clusters could be solved more quickly as Hamiltonian path problems.

Our prior work has approached the problem from several perspectives. Mondal et al. (2023a) used bilevel optimization to solve the fuel-constrained Vehicle Routing Problem in two steps. They first used a Minimum Set Cover (MSC) algorithm to find a set of UAV-UGV rendezvous points that were within UAV flight range of every site to be visited. Using this to set the UGV's route, the UAV was then tasked to visit as many nodes as possible between rendezvous. Chour et al. (2023) presented a formal definition of the rendezvous problem, an agent-based model, and synthesis of behavior trees with hybrid finite state machines, all verified in simulation. Shi et al. (2022) focused on a subproblem of the planning problem, assuming stochastic energy consumption. They assumed a priori UAV and UGV routes that were aggressive with respect to UAV battery constraints and planned for early land and recharge to keep the probability of battery depletion under a user-defined risk acceptance threshold. Thelasingha et al. (2024) developed a simplified approximation for the graph of task sites and defined conditions of reciprocal feasibility among inputs and outputs of multiple solvers to iteratively plan and improve feasible paths under time constraints.

4 REPRESENTATION

The complex problem requires a diverse set of approaches for optimization. For this reason, we lay out a general state description here, which any solver must translate into a proper formulation for its specific approach. We describe it as a dictionary of key-value pairs (in practice, we define inputs and outputs with YAML). All keys are strings, and the values may be one of several datatypes, including nested dictionaries.

4.1 State

The state describes all of the nodes and agents. Details are given in Tables 1–4.

Table 1: Top-level state keys.

key	value type	note
ID	string	
time	float	timestamp of this state
description	string	human-readable, no semantics
agents	list of <agent>	defined below
scenario	scenario	defined below

Table 2: Agent keys.

key	value type	note
ID	string	unique among UAVs and UGVs
type	string	one of [UAV, UGV]
subtype	string	for UAVs, only “standard”. For UGVs, one of [standard, road_only]
location	location	keys are x and y (float) in Cartesian frame
battery_state	battery_state	keys are max_battery_energy and current_battery_energy (Joules)
stratum	string	**UAV only.** current mode of UAV, one of [flying, docked, taking_off, landing, on_ground, return_home]
charging_pads	list of <charging_pad>	**UGV only.** Defined below.
charging_pad_ID	string	**UAV only.** ID of charging pad that the UAV is currently paired with, or *null*

Table 3: Landing pad keys.

key	value type	note
ID	string	unique among all charging pads
mode	string	one of [open, occupied, allowing_takeoff, allowing_landing]
UAV_ID	string	UAV that the pad is paired with, or *null*
is_charging	Boolean	when *True*, paired, docked UAVs can draw energy from the host UGV

4.2 Plan

A plan can be executed by agents. It consists of a sequence of actions. The top level keys of a plan are given in Table 5, and general keys for actions are given in Table 6. Certain tasks require a set of parameters beyond those of Table 6 to be fully defined. These are explained in Tables 7–9.

Certain actions require start_progress and end_progress keys because they take finite time but cause only discrete changes at the beginning and/or end of the action. For example, landing a UAV takes 1 min and changes the state of the UAV from flying to landing at the beginning, and from landing to docked at the end. Most planners would output actions that are 100% complete at the end_time, but the progress keys are required in case the action is broken up by the simulator or planner. For example, if the action crosses an important simulation timestamp, it would be executed up to that timestamp with its progress recorded, and a new action would then be executed from that point until it is finished. Other actions’ progress can be inferred by linear interpolation between the start and end states and do not require these keys.

Table 4: Scenario top level keys.

key	value type	note
description	string	human readable, no semantic meaning
type	string	always “persistent_surveillance” for now
subtype	string	always “standard” for now
nodes	list of <node>	defined below
connections	list of <connection>	*null* if UGVs are not restricted to a road network. Otherwise, each connection has keys end1 and end2 with values that match a node ID. These connections are bidirectional.

Table 5: Plan top level keys.

key	value type	note
ID	string	unique among plans
state_ID	string	ID of the state that is the initial condition for this plan
description	string	human readable, no semantic meaning
start_time	float	
end_time	float	
individual_plans	list of <individual_plan>	has keys agent_ID and actions, which is a list of <action> (defined below).

Table 6: General action keys.

key	value type	note
type	string	for any agent, may be one of [start, move_to_location, service_node, end]. **UAV only:** may be one of [perch_on_UGV, takeoff_from_UGV, land_on_UGV]. **UGV only:** may be one of [allow_takeoff_by_UAV, allow_landing_by_UAV]
start_time	float	
end_time	float	
task_parameters	type-specific dictionary	each task-specific dictionary is defined below

5 SIMULATION

The plan validators and simulation are written in Java. The inputs are a paired state and plan. The output is a new state, which reflects the changes that were effected by executing the plan. Because the inputs and outputs are YAML files, the simulator can be connected to solvers written in diverse languages. Since the primary purposes of the simulation are algorithm analysis, algorithm comparison, reinforcement learning training, and model-predictive control, we have not developed built-in visualization capabilities. Nonetheless, it is trivial to output states at any desired time resolution, and as these contain all agent locations, they can be fed into many visualization programs, as we do below.

5.1 Validation

The first phase of the simulation is static validation of the plans. This phase ensures that certain constraints are met. These constraint checks do not require a simulated execution of plan; they can be checked by direct inspection of the plan. Passing these validation constraints does not ensure a safe or successful mission, as that must be determined by simulation. The following constraints are checked during validation:

- Paired with state. The plan's key state_ID must match a state in memory.

Table 7: Specific action keys.

action type	key(s)	value type(s)	note(s)
start	location	location	
end	location	location	
move_to_location	origin	location	
	destination	location	
service_node	node_ID	string	
	location	location	

Table 8: Specific action keys for UAVs.

action type	key(s)	value type(s)	note(s)
perch_on_UGV	pad_ID	string	
	origin	location	
	destination	location	
takeoff_from_UGV	pad_ID	string	
	start_progress	float	[0–1] fraction of task complete at start_time
	end_progress	float	[0–1] fraction of task complete at end_time
	location	location	2D location, assuming vertical takeoff
land_on_UGV	pad_ID	string	
	start_progress	float	[0–1] fraction of task complete at start_time
	end_progress	float	[0–1] fraction of task complete at end_time
	location	location	2D location, assuming vertical takeoff

Table 9: Specific action keys for UGVs.

action type	key(s)	value type(s)	note(s)
allow_takeoff_by_UAV	UAV_ID	string	
	pad_ID	string	
	start_progress	float	[0–1] fraction of task complete at start_time
	end_progress	float	[0–1] fraction of task complete at end_time
	location	location	2D location, assuming vertical takeoff
allow_landing_by_UAV	UAV_ID	string	
	pad_ID	string	
	start_progress	float	[0–1] fraction of task complete at start_time
	end_progress	float	[0–1] fraction of task complete at end_time
	location	location	2D location, assuming vertical takeoff
swap_battery	start_progress	float	[0–1] fraction of task complete at start_time
	end_progress	float	[0–1] fraction of task complete at end_time

- No time gaps. Every action in sequence must start at the end_time of the prior action, leaving no gaps in the plan.
- No space gaps. Every action must start where the prior action ends, with no instantaneous motion.
- Servicing nodes at their location. The location defined in a service_node action must match the location of the node.
- Speed limits followed. Every move_to_location, takeoff, and land action must occur slower than a predefined speed limit.
- Takeoffs consistent. Every takeoff_from_UAV action must have a complementary allow_takeoff_by_UAV action occuring at the same time and location, with consistent agent IDs.
- Landings consistent. Every land_on_UAV action must have a complementary allow_landing_by_UAV action occuring at the same time and location, with consistent agent IDs.

5.2 Execution

The current simulation is purely linear and deterministic. It assumes that all tasks are executed exactly as planned. In future iterations, we plan to add stochastic effects as well, such as adding randomness to the amount of energy drawn by maneuvers, or the speed of the agents. As discussed in Section 4, a simulation may be cut off or have to produce outputs during the execution of an action. In this case, the action is executed up to the cutoff point, a state is output, and the remainder of the action is split off to form a new action, which the simulator can immediately execute given the state output at the intermediate time.

Agents must expend energy to perform any action. The power consumption of air agents, corresponding to small UAVs, is 245 W, and the energy consumption of the ground agents is 200 W at rest and

$$P = 356 + 465v \tag{1}$$

when moving, where P is in W and v is in $\mathrm{m\,s^{-1}}$.

5.3 Animation

We use the GAMA platform (Taillandier et al. 2019) for visualization. It is capable of complex multiagent simulation (e.g. (Humann et al. 2023)), but here we use it solely for visualization. We generate one animation frame for every simulated second, and show an example frame in Figure 3.

6 SOLVER

Here we describe the two algorithms that we used in the present work. Note that many other solver approaches are possible, with a unifying API ensuring compatibility. Multilevel frameworks popular for cooperative routing problems, as they break the problem down into manageable subproblems, and we follow this approach. Our algorithms use the heuristic of setting the UGV route first, and then constructing the UAV route based on the UGV's path. So the algorithms have an outer level for UGV routing, and an inner level for UAV routing (Maini and Sujit 2015; Ropero et al. 2019). We describe our Multilevel Optimization Framework in the following subsections, and then describe a similar approach, A-Teams.

6.1 Outer-level UGV Routing

To plan UGV paths, we first find route endpoints by solving a minimum set cover (MSC) problem. This generates a set of points such that each task location is within the UAVs' fuel-limited range of an MSC point. This ensures viability of the route, as AOIs can be reached if the UGV at least stops at these points. The particular solver we use for this is a constraint programming solver, SP-SAT, from Google OR Tools (Google 2021). See our previous paper (Mondal et al. 2023b) for more details. Once we obtain this covering set of points, we connect them by solving a traveling salesman problem to set the UGV route.

6.2 Inner-level UAV Routing

At the inner level, we task the UAVs with visiting task locations and rendezvousing with the UGV to recharge as needed. The UGV's expected arrival times at MSC points serve as time windows for UAV recharging. This leads us to formulate the problem as an Open-ended Energy-constrained Vehicle Routing Problem with Time Windows (O-EVRPTW). The goal is to plan routes that prioritize "stale" task locations, which have not been visited recently, while never running out of energy. We again use Google OR Tools and constraint programming to solve the problem for UAV routes. A more comprehensive discussion of the inner solver and O-EVRPTW formulation can be found in (Mondal et al. 2023a; Mondal et al. 2023b)

6.3 A-Teams Architecture Framework

Similarly, the UAV-UGV cooperative routing problem can be approached as a parameter tuning problem and solved with an Asynchronous Multi-Agent Team Architecture (A-Teams). At the outer level, the problem is modeled as a function of certain free parameters (Ramasamy et al. 2022). At the inner level, the free parameters are optimized by multiple specialized, autonomous agents to specify a route. In addition to the MSC algorithm for finding UGV endpoints, we also select mid-route stops where the UAV can land on the UGV for recharge. The UGV endpoints and mid-route stops serve as the free parameters. Modeling the overall routing problem as an E-VRPTW, the resulting solved route is a function of the free parameters.

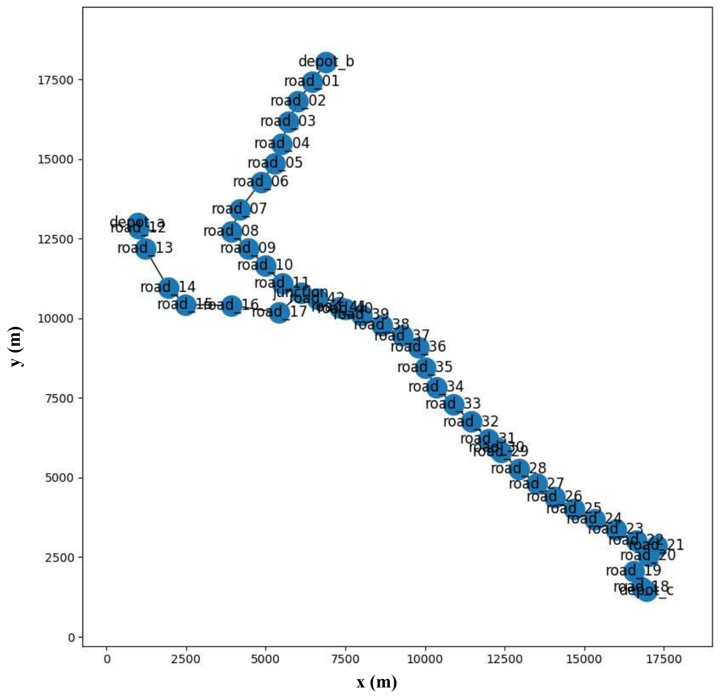

Figure 2: Placement of task sites and road network for the solvers, with each site labeled by its ID. Note that the terminal points on the network are called depots, and the central point is called a junction.

A distinctive feature of A-Teams is the use of multiple specialized agents for optimization. A-Teams tunes the free parameters iteratively to improve solution quality. Here we use four different agents:

- Constructor agent: develops an initial pool of solutions randomly or using heuristics
- Improver agent: enhances the pool of solutions using a genetic algorithm and local search
- Destroyer agent: eliminates redundant nonoptimal solutions to speed convergence
- Predictor agent: predicts whether a set of free parameters will lead to an infeasible solution without exhaustive simulation, to speed elimination of infeasible candidates

These four agents work asynchronously to enhance the overall cooperative route's solution quality. Through this iterative process, we identify the optimal parameter set. For a detailed explanation of each agent, see our previous work (Ramasamy et al. 2022).

7 RESULTS

We applied the A-Teams solver to the state shown in Figure 2, which is a simplified version of Figure 1, without the remote AOIs, and with fewer sites along the road network. The task sites, labeled by their IDs, are shown in Figure 2. It represents a field with three roads meeting at a central junction point. Depot A, farthest to the left, is where UGV batteries can be swapped.

7.1 Constraints

The UAVs' maximum flying speed is $13\,\mathrm{m\,s^{-1}}$. The ground vehicles are typically limited to $5\,\mathrm{m\,s^{-1}}$. The UGVs are restricted to the road network, but the UAVs are not. The UAVs have 360 kJ batteries and recharge in 16 min on the UGVs. The UGVs recharge at Depot A with a 5 min battery swap. Their batteries have a 30010 kJ capacity. For every 1 J of energy that is transferred to a UAV, 1.1 J are taken from the UGV's battery to account for transmission losses.

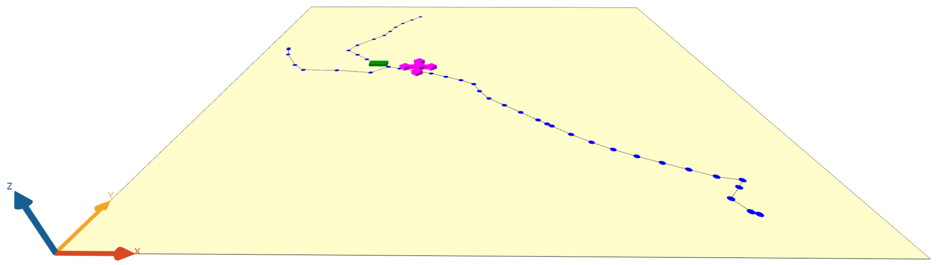

Figure 3: Animation frame at time 10:00:00. At this instant, the UGV (green), is moving to rendezvous with the UAV (magenta), which is returning after visiting the sites near Depot C.

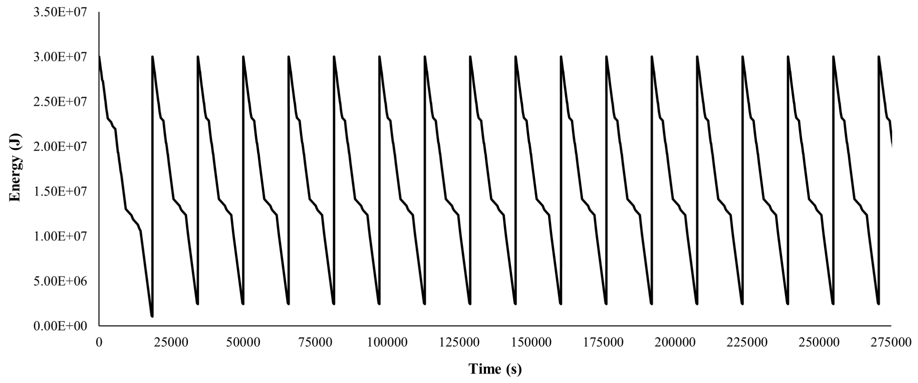

Figure 4: Simulated energy level of the UGV over a 72 h mission.

7.2 Simulated Route

The results of the solver are simulated here and are animated using agent-based modeling software. We first performed all of the static validation checks given in Section 5. Once all agents' routes passed the validation checks, we simulated execution of the plans. The most important outputs of the simulation are energy levels of the agents over time, and the objective function (mission score) at the end. An animation frame is given in Figure 3.

The energy levels for the UAV and UGV are given in Figures 4 and 5. We can see from the figures that the agents never ran out of battery. We also observe strongly cyclic behavior, indicating that the solver output repetitive paths for the agents. This makes sense given that there is no randomness in the current simulation, so once a good route is solved that visits all the task sites in minimum time, the solver mostly just needs to repeat this route to visit all task sites at regular intervals.

The plan figure of merit that we use is the mission score, S. Each time a node is visited, an associated timer will be reset. The total score for this mission will be related to the intervals at which each node is visited. All solvers attempt to minimize this score. The following function gives the instantaneous derivative of the score from node i:

$$\dot{S}_i = \frac{t - t_{\text{last},i}}{1800},$$

where t is measured in minutes, and $t_{\text{last},i}$ is the last time that node i was visited. The rate of score accrual increases with time (making the actual score quadratic in time), penalizing nodes with longer inter-visit intervals harshly. The equation must be integrated to find the score for a given node, and must be summed over all nodes for the total score:

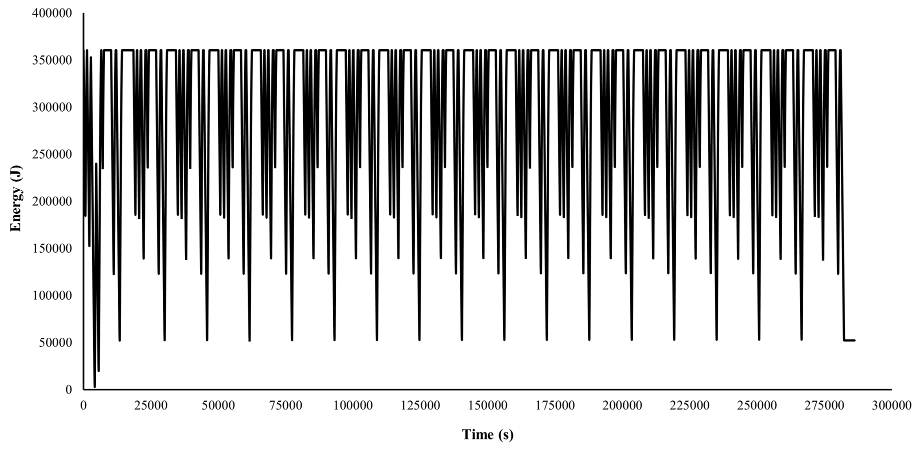

Figure 5: Simulated energy level of the UAV over a 72 h mission.

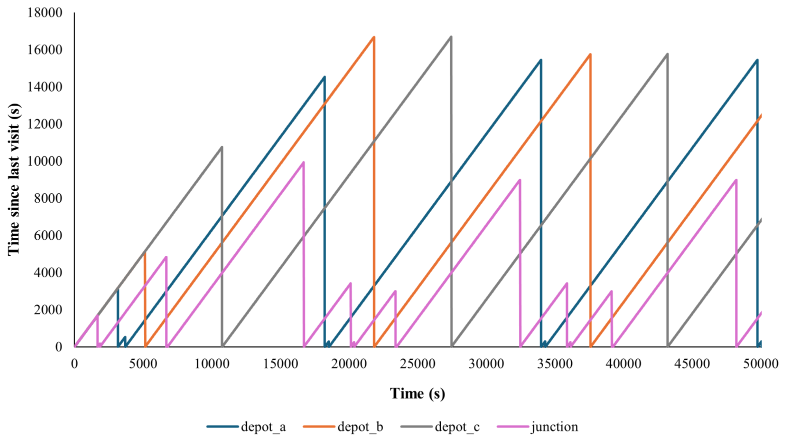

Figure 6: Latency of four task sites in the first 50 000 s of simulated mission.

$$S = \sum_{i=1}^{N} \int_{0}^{T} \frac{t - t_{\text{last},i}}{1800}\, dt,$$

where T is the total length of time, and N is the total number of task sites.

Figure 6 shows the latency (time since last visit) of four task sites. Due to the central location of the junction, it was serviced more often then the extremal depots. It was serviced at approximately 3200 s intervals, whereas Depots B and C were serviced approximately every 16 000 s. Only the first 50 000 s of the results are shown, as the simulation shows mostly periodic subsequent behavior. The other nodes are not shown in Figure 6 for readability; they show similar periodic behavior.

The repeated visits give a site score of 8602 for the junction, which is among the lowest of the sites (site road_39 is the lowest at 7852). The depots A, B, and C have scores of 19 832, 20 498, and 20 426, respectively, which are higher than most other sites (with site road_03 being the highest at 20 536). The total score summed over all sites is 702 465, for an average of 15 271 per site. The sites are sorted by decreasing score in Figure 7.

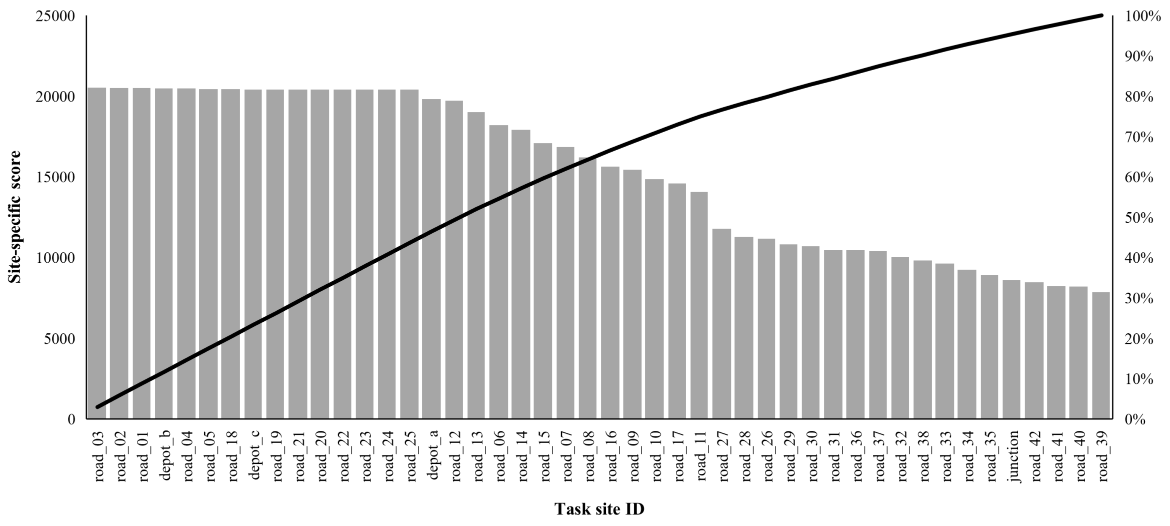

Figure 7: Sites sorted by decreasing score with cumulative percentage of total score shown on right axis.

8 CONCLUSION

8.1 Summary

In this paper, we introduced a data model and simulation for the energy-aware mission planning problem with power-sharing unmanned ground and air vehicles. The discussion of prior work showed a diverse set of approaches to this problem, without a common format for comparison, simulation, and plug-in support. We presented an exhaustive set of variables necessary to define a system state and plan, in the form of key-value pair dictionaries. We instantiated this data model using YAML. We used A-Teams to generate a plan, which we could then validate and simulate. Our contributions were the new data model, new simulation, updated solvers, and presentation of 72 h mission plans.

8.2 Limitations and Future Work

In its current state, the simulation is purely deterministic. While useful for validating plans, it cannot be used to test the robustness of plans to the randomness inherent in robotic deployments. In future versions, we will add stochastic effects such as randomness to the amount of energy drawn by maneuvers, success or failure of servicing nodes, and the speed of the agents. We have ongoing research on robust solvers using risk budgets (Asghar et al. 2023), and a stochastic simulator could support these efforts.

REFERENCES

Asghar, A. B., G. Shi, N. Karapetyan, J. Humann, J.-P. Reddinger, J. Dotterweich *et al.* 2023. "Risk-aware Recharging Rendezvous for a Collaborative Team of UAVs and UGVs". In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 5544–5550.

Carlsson, J. G. and S. Song. 2018. "Coordinated logistics with a truck and a drone". *Management Science* 64(9):4052–4069.

Chour, K., J.-P. Reddinger, J. Dotterweich, M. Childers, J. Humann, S. Rathinam *et al.* 2023. "An agent-based modeling framework for the multi-UAV rendezvous recharging problem". *Robotics and Autonomous Systems* 166:104442.

Google 2021. "Google OR-Tools". *https://developers.google.com/optimization*.

Humann, J., T. Fletcher, and J. Gerdes. 2023. "Modeling, simulation, and trade-off analysis for multirobot, multioperator surveillance". *Systems Engineering* 26(5):627–640.

Liu, Y., Z. Luo, Z. Liu, J. Shi and G. Cheng. 2019. "Cooperative routing problem for ground vehicle and unmanned aerial vehicle: The application on intelligence, surveillance, and reconnaissance missions". *IEEE Access* 7:63504–63518.

Long, Y., G. Xu, J. Zhao, B. Xie and M. Fang. 2023. "Dynamic Truck–UAV Collaboration and Integrated Route Planning for Resilient Urban Emergency Response". *IEEE Transactions on Engineering Management*.

Luo, Z., R. Gu, M. Poon, Z. Liu and A. Lim. 2022. "A last-mile drone-assisted one-to-one pickup and delivery problem with multi-visit drone trips". *Computers & Operations Research* 148:106015.

Maini, P. and P. Sujit. 2015. "On cooperation between a fuel constrained UAV and a refueling UGV for large scale mapping applications". In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, 1370–1377. IEEE.

Mondal, M. S., S. Ramasamy, J. D. Humann, J.-P. F. Reddinger, J. M. Dotterweich, M. A. Childers *et al*. 2023a. "Optimizing Fuel-Constrained UAV-UGV Routes for Large Scale Coverage: Bilevel Planning in Heterogeneous Multi-Agent Systems". In *2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 114–120. IEEE.

Mondal, M. S., S. Ramasamy, J. D. Humann, J.-P. F. Reddinger, J. M. Dotterweich, M. A. Childers *et al*. 2023b. "Cooperative Multi-Agent Planning Framework for Fuel Constrained UAV-UGV Routing Problem". *arXiv e-prints*:arXiv–2309.

Ramasamy, S., M. S. Mondal, J.-P. F. Reddinger, J. M. Dotterweich, J. D. Humann, M. A. Childers *et al*. 2022. "Heterogenous vehicle routing: comparing parameter tuning using genetic algorithm and Bayesian optimization". In *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, 104–113. IEEE.

Ramasamy, S., M. S. Mondal, J.-P. F. Reddinger, J. M. Dotterweich, J. D. Humann, M. A. Childers *et al*. 2023. "Solving Vehicle Routing Problem for Unmanned Heterogeneous Vehicle Systems using Asynchronous Multi-Agent Architecture (A-teams)". In *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*, 95–102.

Ropero, F., P. Muñoz, and M. D. R-Moreno. 2019. "TERRA: A path planning algorithm for cooperative UGV–UAV exploration". *Engineering Applications of Artificial Intelligence* 78:260–272.

Shi, G., N. Karapetyan, A. B. Asghar, J.-P. Reddinger, J. Dotterweich, J. Humann *et al*. 2022. "Risk-aware UAV-UGV rendezvous with chance-constrained Markov decision process". In *Conference on Decision and Control (CDC)*, 180–187. IEEE.

Taillandier, P., B. Gaudou, A. Grignard, Q.-N. Huynh, N. Marilleau, P. Caillou, *et al*. 2019. "Building, composing and experimenting complex spatial models with the GAMA platform". *GeoInformatica* 23:299–322.

Thelasingha, N., A. A. Julius, J. Humann, J.-P. Reddinger, J. Dotterweich and M. Childers. 2024. "Iterative Planning for Multi-Agent Systems: An Application in Energy-Aware UAV-UGV Cooperative Task Site Assignments". *IEEE Transactions on Automation Science and Engineering*.

Wen, X. and G. Wu. 2022. "Heterogeneous multi-drone routing problem for parcel delivery". *Transportation Research Part C: Emerging Technologies* 141:103763.

AUTHOR BIOGRAPHIES

JAMES HUMANN is a mechanical engineer with the US DEVCOM Army Research Laboratory. He primarily works in modeling and simulation of multiagent systems. His email address is james.d.humann.civ@army.mil.

STEVEN CARLOS ORTEGA is currently pursuing an MS in Mechanical Engineering at the University of Texas at Austin. He is a Space Force cadet in the AFROTC program and member of the Human Centered Robotics Lab. His research interests are in multi-agent path planning, swarm robotics, and human robot interactions. His email address is stevencortega@austin.utexas.edu.

JAMES GLENN is an AFROTC cadet at Texas A&M University. His email address is james.glenn@eccalon.com.

JACK L. FOLSOM is an AFROTC cadet at Texas Tech University. His email address is jack.folsom@eccalon.com.

SUBRAMANIAN RAMASAMY is currently pursuing the PhD degree in the Department of Mechanical and Industrial Engineering at The University of Illinois Chicago. His research interests include Operations Research, Deep Learning, optimization and path planning of Autonomous Vehicles. His email address is sramas21@uic.edu.

MD. SAFWAN MONDAL is a PhD student in the Department of Mechanical and Industrial Engineering at the University of Illinois Chicago. His research spans robotics, multi-robot systems, optimization, and artificial intelligence, with a focus on multi-robot planning and optimization. His email address is mmonda4@uic.edu.

PRANAV BHOUNSULE is an Assistant Professor of Mechanical Engineering at the University of Illinois Chicago. He has a PhD in Mechanical Engineering from Cornell University. His email address is pranav@uic.edu.

JEAN-PAUL REDDINGER is an aerospace engineer with DEVCOM Army Research Laboratory. His research focuses on modeling and simulation for novel VTOL UAS platforms. His email address is jean-paul.f.reddinger.civ@army.mil.

JAMES DOTTERWEICH is a robotics research engineer with the DEVCOM Army Research Laboratory. He holds an MS in Mechanical Engineering from The University of Utah. His email address is james.m.dotterweich.civ@army.mil.