Robust Scheduling and Routing for Collaborative Human-UAV Surveillance Missions

Jeffrey R. Peters[1] and Amit Surana[2]
United Technologies Research Center, East Hartford, CT, 06118-1127

Francesco Bullo[3]
University of California, Santa Barbara, Santa Barbara, CA, 93106-5070

A supervisory mission is considered in which a team of unmanned vehicles visits a set of targets and collects sensory data to be analyzed in real-time by a remotely-located human operator. A framework is proposed to simultaneously construct the operator's task-processing schedule and each vehicle's target visitation route, with the dual goal of moderating the operator's task load and preventing unnecessary vehicle loitering. The joint scheduling/routing problem is posed as a mixed-integer (non-linear) program which can be equivalently represented as a mixed-integer linear program through expansion of the solution space. In single vehicle missions, it is shown that an alternative linearization exists that does not increase the problem size. Next, a dynamic solution strategy is introduced that incrementally constructs suboptimal schedules and routes by solving a comparatively small, mixed-integer linear program whenever the operator finishes a task. Using a scenario-based extension, this dynamic framework is then modified to provide robustness to uncertainty in operator processing times. The flexibility and utility of these algorithms are explored in simulated missions.

---

[1] Senior Research Engineer; Systems Department; petersjr@utrc.utc.com
[2] Associate Director, Research; Systems Department; SuranaA@utrc.utc.com
[3] Professor and Chair; Department of Mechanical Engineering; bullo@engineering.ucsb.edu

I. Introduction

Modern autonomous sensor systems frequently rely on operator feedback to function effectively within complex scenarios [1–4]. In theory, the presence of both human and autonomous elements within a single system can be very beneficial, since the interplay between these two components can create a symbiotic relationship that emphasizes the strengths and mitigates the deficiencies of each. However, this type of interaction is not guaranteed, even if each component is optimized independently. To realize the full benefit of this setup, coordination schemes must jointly optimize the *entire* system, accounting for the inherent coupling between the human and autonomous elements.

One class of human-centered system that is prevalent in modern application consists of *supervisory systems* in which a remotely located human operator oversees a team of autonomous mobile sensors [1]. Here, the operator interacts with the sensors via a central control interface, which also serves as a platform for sensory data management (Fig. 1). In developing coordination strategies for these systems, several factors must be considered. From a human factors perspective, coordination schemes should create a high-performance work environment for the operator. In particular, for human processing of sensory data, smart strategies should ensure that (i) the required tasks are completed, (ii) the operator's cognitive state remains in a high-performance regime, and (iii) performance is robust to behavioral uncertainty. From a robotics perspective, mobile sensors must be able to operate effectively within a potentially large and dynamic environment. Mobile sensor coordination strategies should also ensure that (i) tasks requiring operator attention are generated at a rate that does not create bottlenecks, and (ii) uncertainty does not cause undesirable configurations.

This article considers a particular application in which a human operator analyzes real-time target imagery, e.g., a set of video streams, that is collected by Unmanned Aerial Vehicles (UAVs) as they visit a set of discrete, geographically spaced targets. We consider this setup, as opposed to a setup where imagery is analyzed offline, to reflect the real-time analysis requirement that is present in many real surveillance operations. As an example, the necessity for real-time analysis arises when the operator is required to adjust the surveillance parameters (zoom, focal point, etc.) of UAV payloads and/or adjust mission strategies in response to image content, e.g., if a high-value mobile target is seen in an image [5]. As such, in the setup herein, operator "tasks" correspond to

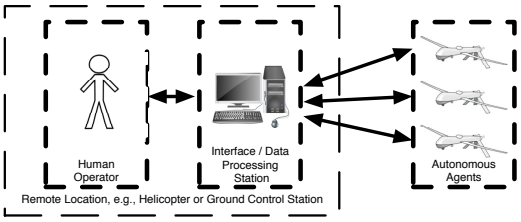

Fig. 1 Typical architecture of a human supervisory control system involving mobile sensors.

the analysis of the generated target imagery. Since real-time imagery is only available while a UAV is loitering at a target, both human and UAV resources are simultaneously required to complete each task. Our work develops a framework that coordinates these resources by jointly optimizing over the UAV routes and the operator task-processing schedule with the dual goal of (i) maintaining the operator's task load within an acceptable regime, and (ii) minimizing unnecessary UAV loiter time. This approach differs from typical approaches in that (i) it considers the scheduling and coordination of robotic tasks to support an operator by utilizing a human-factors inspired objective function, and (ii) it simultaneously coordinates both the human and autonomous system components within a single mathematical framework that explicitly considers their inherent coupling. Indeed, research efforts with respect to human-UAV supervisory systems typically study *reactive* policies to coordinate a single component (human or autonomous), while allowing the other to operate either uncontrolled or under an assumed behavioral process. For example, the work in [6] does not explicitly specify vehicle behavior, but determines optimal operator task processing times under the assumption that tasks arrive in a queue via a Poisson process. Our setup does not assume any fixed behavioral process and instead coordinates both components simultaneously.

The specific contributions of this article are as follows. First, we formulate a multi-vehicle, scheduling/routing problem as a Mixed-Integer Non-linear Program (MINLP), whose objective function accounts for both the operator's task load and unnecessary UAV loiter time. We show that the general MINLP can be re-formulated as a Mixed-Integer Linear Program (MILP), at the expense of significant increases in problem size. For single vehicle missions, we provide an alternative linearization that does not increase the problem size. Next, to ease computation, we introduce a dynamic framework for constructing suboptimal solutions to the full problem. Here, whenever the operator completes a task, a re-planning step is performed that chooses each UAV's next desti-

nation and the impending portion of the operator's schedule. We pose this re-planning operation as a comparatively small MILP, whose solutions are constructed with existing solvers. We show how this dynamic framework can incorporate robustness to uncertain task processing times via a scenario-based extension of the re-planning MILP. We then demonstrate the utility and flexibility of our framework in select simulated missions. In addition to illustrating performance and robustness properties, these examples also show how the dynamic framework readily extends to incorporate more general setups, e.g., those containing fixed-wing UAVs. We conclude with a discussion of key problem extensions, limitations, and future research directions.

II. Related Literature

Human operators play a key role in current and futuristic applications involving autonomous sensors, including military reconnaissance [7], search and rescue [8], and automated manufacturing [9]. Specifically, many applications rely on human processing of sensory data. For example, the wide-area surveillance technology Gorgon Stare uses camera arrays attached to UAVs to collect imagery which is processed by remotely located analysts [10]. In response, significant research efforts have focused on coordination schemes to improve the performance of supervisory control systems.

Human-centric methods strive to develop an understanding of operator behavior, which is used as the basis for control or design strategies. Some methods, e.g., those focusing on interface design [11, 12], are performed *offline*. Many offline methods also study human-centric phenomena to assess control strategies [13] or to suggest system architectures, e.g., the number of vehicles an operator can supervise [14]. Conversely, *online* approaches typically adapt mission strategies based on real-time data or incremental realization of uncertain parameters. For example, eye-tracking is becoming a popular tool for real-time behavioral assessment in visual tasks [15, 16]. These assessments are used to moderate operator resources through, e.g., adaptive automation [17, 18] or decision-support systems [15]. Other online approaches rely on fixed processing time or task generation models [19, 20]. The scheduling approach herein is similar to [20], which uses a MILP framework considering both processing time uncertainty and operator task load. Our work differs, however, in that we propose a coupled framework that simultaneously optimizes vehicle routes.

Autonomous agent-centric methods focus primarily on optimizing sensor behavior. Of particular relevance is research relating to route planning for discrete surveillance or target visitation, e.g., [21–26]. The UAV route planning portion of the problem considered herein is loosely derived from [27–29], which use sampling-based procedures to pose a continuous path planning problem as a *Generalized TSP* (GTSP) [30]. Indeed, the vehicle routing portion of the joint scheduling/routing problem in the present paper is essentially a multi-agent GTSP, which can result from a discretization procedure similar to that of [29]. Thus, our work is, in some sense, an extension of [29] to incorporate the simultaneous optimization of operator schedules.

Despite extensive research devoted to improving each component individually, there have been relatively few attempts to jointly optimize over operator and sensor behavior. Existing work typically assumes a "loose" coupling between human and autonomous agents, resulting in *reactive* policies. For example, the authors of [6, 19] assume that tasks arrive in a processing queue according to a fixed stochastic process, which drives optimal operator behavior. In contrast, the authors of [31] use an adaptive surveillance policy based on operator responses in a target detection task; however, operator behavior is uncontrolled. The authors of [32] consider general architectures for human-robot collaboration, but consider the human as uncontrollable. Others, such as [33], develop discrete-event simulation models, which are used for testing and optimization. However, these abstractions utilize parametric process models without explicitly considering the cause of task generation, e.g., vehicle trajectories. Some work, such as [34], considers the sequencing of robotic tasks for the purpose of working alongside humans, but do not explicitly consider human factors-inspired objectives.

Very limited work has considered tightly coupled optimization across components (human/autonomous agent). Those most closely related to our work are [35, 36], which consider optimization schemes for simultaneous routing and scheduling under operator workload constraints. However, these works have several limitations. In particular, both [35] and [36] rely on the availability of a suitable abstraction to the vehicle routing problem that may be unavailable in complex scenarios. For example, [35] requires an accurate predictor of the complete mission cost associated with a given task-agent pairing, which is often unavailable *a priori*. Additionally, both approaches consider deterministic setups; as such, solution quality may become poor or even infeasible during

mission execution when uncertainty is introduced. In [36], all planning operations are performed offline, and may not be easily applied in dynamic schemes due to computational issues. Indeed, in [36], the time continuum is discretized to obtain a pure integer program which can become intractable for even modest time horizons. We note that the ability for online re-planning is often necessary for scalability and to account for changing mission conditions. The distributed framework in [35] is designed for online implementation; however, it does not consider tasks requiring both a human and a robotic agent attention simultaneously. Our work seeks to overcome these limitations by: (i) combining vehicle routing and operator scheduling into a single, mixed-integer mathematical program; (ii) providing a scalable online/incremental solution methodology, and (iii) demonstrating a natural extension that incorporates robustness to uncertainty in operator processing times.

In an abstract sense, the problem herein can be thought of, in part, as a multi-robot task allocation problem, which seeks to optimally pair a set of tasks (targets) with available embodied mobile agents (UAVs or human). Through this lens, the optimal operator scheduling and routing problem can be classified, according to the taxonomy of Korsah, et. al. [37], as a single task, multi-robot, time-extended scheduling problem with complex dependencies, i.e., CD [ST-MR-TS]. Complex dependencies arise from the coupling between human and vehicle behavior, combined with inter-task dependencies arising from the spatial distribution of tasks. Through a different lens, the problem herein can also be interpreted as a generalized *Vehicle Routing Problem (VRP)* with multiple synchronization constraints [22], which treats the operator's visual attention as a "mobile vehicle" that operates within the target environment and moves virtually instantaneously between targets. Here, the requirement of simultaneous operator and vehicle resources for task completion is modeled as a set of *operation synchronization constraints*, since it introduces temporal inter-dependencies on "vehicle" routes. Although VRP formulations appear frequently in operations research literature, the wide-varying nature of possible problem goals and constraints make existing modeling frameworks heavily application-dependent [22, 38]. Our work studies an optimization-based framework that is tailored to human-supervisory surveillance, considering typical issues that arise in this domain, such as processing time uncertainty, and operator task-load constraints.

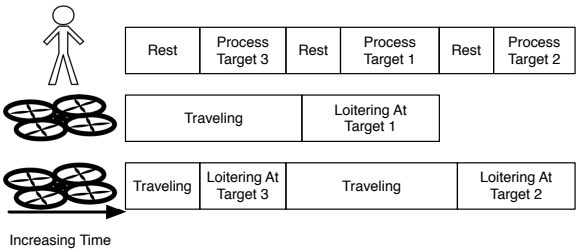

Fig. 2 Illustration of the relation between UAV and operator behavior

III. Problem Formulation

A. Mission Overview and Solution Approach

A team of UAVs, each equipped with a gimbaled, on-board camera, is tasked with collecting surveillance imagery of a set of static targets with known locations. Targets are distributed over a large planar area, so UAVs must move within the environment to collect complete sensory data. There is no restriction on which UAV images any particular target, and no single UAV can simultaneously image multiple targets. When a UAV reaches a target, it loiters in place (see Section III C) while transmitting its camera feed to a remotely located operator, who takes some amount of time to process the imagery. The mission ends when the operator has processed each target exactly once.

We seek a framework to simultaneously generate (i) UAV routes to visit/image the targets, and (ii) the operator's task-processing schedule, i.e., the time at which the operator processes each task (target image). Since imagery is transmitted in real-time, the availability of operator tasks is determined by the UAVs arrival time at each target. Conversely, the required UAV dwell-time at each target is determined by operator processing times. An illustration of this coupling is shown in Fig. 2. This relationship evokes a set of constraints to govern the synchronization of human and robotic resources. In addition to satisfying these constraints, the ideal routes/schedule is such that (i) the operator's *task load* stays within a regime that is amenable to high performance, and (ii) the time that UAVs spend loitering unnecessarily, i.e., when the operator is not analyzing their camera feed, is minimized. The following subsections expand the mission setup mathematically.

B. Human Operator Specifications

A single human operator sequentially processes imagery collected by the UAVs. Assume that all image analysis tasks are of equal importance, and that no two tasks can be executed simultaneously.

Further assume that, once a task is started, it must be completed before another task is initiated.

We are interested in moderating the operator's task load. *Task load* can be defined as "a measurement of human performance that broadly refers to the levels of difficulty an individual encounters when executing a task" [39]. Within the interpretation of [40], task load is operationally considered as a function of three factors: (i) the time taken to perform a task, (ii) the required level of information processing, and (iii) the number of task switches that occur in the context of task performance. Our chosen model focuses on capturing qualitative task load trends induced by (i) and (ii), while assuming that the effect of required task switches (iii) is small during mission execution. In particular, we introduce a *utilization-based* task load measure which evolves based on operator behavior, and attempt to maintain this dynamic measure within a set of pre-defined bounds. Note that this type of utilization-based measure has been used as a reasonable approximation to pilot workload [41–43]. We enforce task load bounds since, in many tasks, operator performance is optimal when their workload lies within a certain high-performance regime [33, 44]. We focus on task load (as a surrogate for workload), rather than other human factors phenomena, e.g., fatigue, situational awareness, among others, since it has well-established correlations with operator performance that can be feasibly exploited by automated mission planners [45]. We emphasize once again, however, that our primary goal is to illustrate a particular modeling philosophy, in which system-wide and human-factors inspired metrics are used to optimize overall performance. We do not provide formal verification of our chosen task load model, and thus practitioners should not treat this model as a general purpose solution for all problem instances. Further comments are included in Section VIII.

We represent the operator's task load as a scalar variable, which, ideally, should be maintained within the finite interval $[\underline{w}, \overline{w}]$. The bounds $\underline{w}, \overline{w}$ are chosen *a priori*; however, they are treated as "soft" constraints, and thus high precision is not generally required. The operator's task load evolves according to trend-based dynamics (similar to [20, 36]): when the operator is busy, i.e., working on a task, their task load level increases by some (task-dependent) amount, and when the operator is idle, their task load level decreases by some amount. For our purposes, we assume that task load decrements linearly during idle time at a fixed rate $\delta^- \in \mathbb{R}_{>0}$, e.g., if the operator is idle for time t, then the task load decrement is $\delta^- t$. We adopt this simple model since we are primarily concerned

with capturing qualitative trends. Task load increments are discussed further in Section III D

C. UAV Specifications

Suppose there are $N \in \mathbb{N}$ UAVs, each responsible for visiting a subset of the targets and transmitting real-time video data to the operator. A UAV must loiter at the target location while the operator processes the associated task. We do not assume a particular dynamic model, although we do assume that the time required for traversal of the optimal path between any two UAV configurations can be quantified using a known, time-invariant function, i.e., the travel time between any two given configurations is fixed and known. We collect each UAV's initial configuration in a set V_0 (having size N: create copies if two UAVs share a common initial configuration).

We develop our framework under the simplifying assumptions that (i) the UAVs are homogenous, i.e., have identical dynamic and sensing capability, and (ii) each UAV is able to "hover" in place. However, some comments are in order: First, heterogenous vehicles can be accommodated via straightforward extensions of the presented methods (see Section VIII). Second, the "hovering" assumption is not necessary when using the dynamic strategy of Sections V and VI. As such, the strategies of these sections can also be applied to fixed-wing UAVs, as we demonstrate in Section VII.

D. Target Specifications

A set of $M \in \mathbb{N}$ targets must be imaged and analyzed. Associate each target j with a finite set V_j of configurations from which the UAV is able to provide the required imagery. That is, to image target j, the UAV must travel to one of the configurations in V_j and hover until operator processing is complete (see Fig. 3). Each target j can equivalently be considered as an image analysis task, which takes the operator time $\tau_j \in \mathbb{R}_{\geq 0}$ to complete. Initially, assume that τ_j is fixed and known *a priori* for each target (this is relaxed in Section VI). Let $\Delta W_j \in \mathbb{R}_{\geq 0}$, represent the (fixed) amount that the operator's task load level increments as a result of working on task j for time τ_j.

Remark 1 (Configuration Clusters)**.** *Discrete configuration "clusters," such as $V_1, V_2, \ldots, V_M$, arise naturally in sampling-based approximations to continuous planning problems. For example, in visibility constrained routing problems where targets must be imaged from a particular range of angles,*

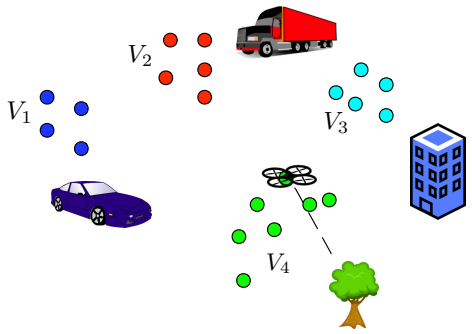

Fig. 3 Illustration of the sets V_j associated with each target j

UAVs must visit a non-discrete visibility region associated with each target. These regions can be approximated via sampling, resulting in configuration "clusters" similar to $V_1, V_2, \ldots, V_M$ [27].

E. Objectives and Performance Metrics

The goal is to develop an algorithmic framework that simultaneously manages both human and autonomous resources. We consider two performance metrics: (i) the maximum amount (absolute value) by which the operator's task load bounds are violated (upper and lower) during the mission, and (ii) the aggregate time that UAVs spend loitering unnecessarily, i.e., loitering at a target before the operator begins processing the camera feed. The first metric emerges from the relation between task load and performance mentioned in Section III B. The second metric is considered since it is often undesirable for UAVs to spend excessive time loitering near a target, e.g., in military reconnaissance operations involving hostile targets, unnecessary loitering increases target awareness of UAV presence through increased noise signature, etc [46]. We wish to minimize these metrics by jointly optimizing over (i) the operator schedule, and (ii) the UAV routes.

The joint optimization problem of interest is summarized as follows.

Problem 1 (Joint Human/UAV Optimization)**.** *Determine both an operator schedule, which dictates when the operator should process each task (target image), and a set of UAV target visitation*

routes, that together minimize the metric

$$\begin{aligned} \textit{Mission Cost} = p_{\underline{\alpha}}(&\textit{Max lower task load bound violation}) \\ + p_{\overline{\alpha}}(&\textit{Max upper task load bound violation}) \\ + p_{\beta}(&\textit{Total unnecessary loiter time}) \end{aligned}$$

where $p_{\underline{\alpha}}$, $p_{\overline{\alpha}}$, $p_{\beta} > 0$ *are fixed parameters.*

Notice that we do not require $p_{\underline{\alpha}} + p_{\overline{\alpha}} + p_{\beta} = 1$. The remainder of our analysis is focused constructing practical solutions to Problem 1.

IV. Mixed-Integer Programming Formulation

This section develops a Mixed-Integer Program (MIP) representation of Problem 1. The formulation assumes: (i) each UAV departs its initial location immediately, and departs each successive target viewpoint along its route immediately after the operator completes the associated task, and (ii) the time that any UAV takes to travel between each possible pair of starting and ending configurations is fixed and known. Let $V_0 \coloneqq \{i_0^1, i_0^2, \ldots, i_0^N\}$ be the set of initial UAV configurations, where i_0^ℓ is the configuration of the ℓ-th UAV. Define a parameter K to represent the number tasks in the operator schedule. In this section, we construct the complete schedule and thus set $K \coloneqq M$.

A. Graph Construction

Under the given assumptions, the problem of finding appropriate UAV target visitation routes reduces to a path-finding problem over a graph. Define a complete, weighted, directed graph $G \coloneqq (V, E, W)$, where $V \coloneqq V_0 \cup V_1 \cup \cdots \cup V_M$ is the union of the clusters $V_0, V_1, \ldots, V_M$ defined in Sections III C and III D, and the weight $W(i, j)$ of any $(i, j) \in E$ captures the travel time from node i to node j. Assume G also contains zero-weight self-loops, i.e., $(i, i) \in E$ and $W(i, i) \coloneqq 0$ for all $i \in V$. For each UAV ℓ, a valid target visitation path starts at v_0^ℓ, and visits at most one node from any of the clusters $V_1, \ldots, V_M$. Aggregating individual routes, exactly one node from each cluster should be visited by some UAV.

Remark 2 (GTSP)**.** *The UAV route-finding problem is similar to a* multi-vehicle, open, GTSP*.*

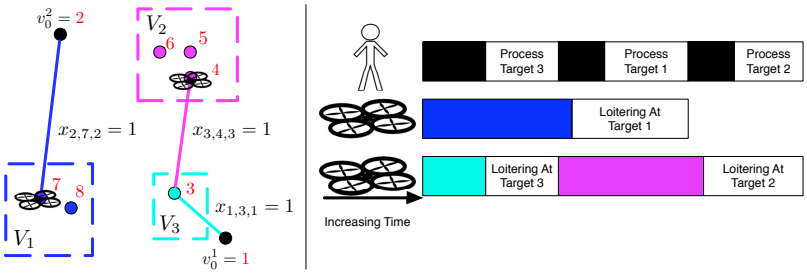

Fig. 4 Relation between binary decision variables and resulting solution

Indeed, we seek paths on G so that exactly one configuration associated to each target is visited by some UAV, and the UAVs need not return to their initial depot. However, the problem of interest herein differs from typical GTSP instances due to operator considerations: the performance metrics considered depend jointly on vehicle behavior and operator behavior, which is not fixed a priori.

B. Decision Variables

Let $x_{i,j,k} \in \{0,1\}$ be a binary decision variable, which is equal to 1 if and only if $(i,j) \in E$ appears in some UAV's route, and the target associated with node j represents the k-th task in the operator schedule (Fig. 4). The remaining decision variables are continuous: Let $A_k \in \mathbb{R}_{\geq 0}$ be the time that some UAV arrives at the target representing the k-th task in the operator schedule, and let $B_k, C_k \in \mathbb{R}_{\geq 0}$ be the time that the operator begins and completes the k-th task, resp. Next, let $\underline{W}_k, \overline{W}_k \in \mathbb{R}$ represent the operator's task load level immediately before and after processing the k-th task. Note that the subscript k indicates the order in which the operator processes tasks, and not the order in which any single vehicle visits its assigned targets. Finally, define $\underline{\alpha}, \overline{\alpha}, \beta \in \mathbb{R}_{\geq 0}$ to act as surrogates to each performance metric: max lower and upper task load bound violation (absolute value), and total unnecessary loiter time. Table 1 summarizes the decision variables.

Table 1 Decision Variables

Variable	Index Set	Description
$x_{i,j,k} \in \{0,1\}$	$(i,j) \in E$ $k \in \{1,\dots,K\}$	Indicates if: edge (i,j) appears in some vehicle's tour and the target associated with j is the k-th operator task
$A_k \in \mathbb{R}_{\geq 0}$	$k \in \{1,\dots,K\}$	The time that a UAV arrives at the target representing the k-th operator task
$B_k, C_k \in \mathbb{R}_{\geq 0}$	$k \in \{1,\dots,K\}$	The time the operator begins and completes the k-th scheduled task, resp.
$\underline{W}_k, \overline{W}_k \in \mathbb{R}_{\geq 0}$	$k \in \{1,\dots,K\}$	The operator's task load level immediately before and after competing the k-th scheduled task, resp.
$\underline{\alpha}, \overline{\alpha}, \beta \in \mathbb{R}_{\geq 0}$	-	Variables quantifying performance metrics

C. Constraints

1. *UAV Path Constraints*

The first constraints ensure that a valid UAV tour can be extracted from the decision vector.

$$\sum_{i\in V}\sum_{j\in V_m}\sum_{k=1}^{K} x_{i,j,k} = 1 \qquad \forall m \in \{1,\dots,M\} \tag{1}$$

$$\sum_{i\in V_m}\sum_{j\in V}\sum_{k=1}^{K} x_{i,j,k} \leq 1 \qquad \forall m \in \{1,\dots,M\} \tag{2}$$

$$\sum_{i\in V}\sum_{j_0\in V_0}\sum_{k=1}^{K} x_{i,j_0,k} = 0 \tag{3}$$

$$\sum_{i\in V}\left(x_{j,i,k} - \sum_{\overline{k}=1}^{k-1} x_{i,j,\overline{k}}\right) \leq 0 \qquad j \notin V_0, k \in \{1,\dots,K\} \tag{4}$$

$$\sum_{k=1}^{K}\sum_{j\in V} x_{i_0,j,k} \leq 1 \qquad \forall i_0 \in V_0 \tag{5}$$

$$\sum_{(i,j)\in E} x_{i,j,k} \leq 1 \qquad \forall k \in \{1,\dots,K\} \tag{6}$$

Eq. (1) ensures that exactly one node from each cluster is visited, and Eq. (2) ensures that at most one edge leaves any cluster. Eq. (3) ensures that the vehicles do not return to V_0 once they leave (we seek *open* paths). Eq. (4) says that any vehicle which leaves a node (excluding the initial node) must

enter the same node at an earlier time (define $\sum_{\overline{k}=1}^{0} x_{j,i,\overline{k}} := 0$). Eq. (4) also ensures that (i) the first task in the operator schedule coincides with the first target that some UAV visits, and (ii) each UAV ℓ's route begins at i_0^ℓ. Eq. (5) ensures that at most one edge leaves each initial configuration, and Eq. (6) ensures that a single task is chosen for each "slot" in the operator schedule.

2. UAV Arrival Time Constraint

The following constraint governs the UAV arrival times.

$$\sum_{(i,j)\in E} x_{i,j,k} \left(W(i,j) + \sum_{\overline{j}\in V} \sum_{\overline{k}=1}^{k-1} x_{\overline{j},i,\overline{k}} C_{\overline{k}} \right) = A_k \qquad \forall k \in \{1,\ldots,K\} \tag{7}$$

Eq. (7) says that a UAV's arrival time at a target must equal the time that the operator completes the previous task associated with that particular UAV plus the required travel time.

3. Task Processing Constraints

The following constraints govern the operator schedule induced by the decision vector.

$$B_k + \sum_{(i,j)\in E} x_{i,j,k} \tau_{m_j} = C_k \qquad \forall k \in \{1,\ldots,K\} \tag{8}$$

$$A_k \leq B_k \qquad \forall k \in \{1,\ldots,K\} \tag{9}$$

$$C_{k-1} \leq B_k \qquad \forall k \in \{2,\ldots,K\} \tag{10}$$

Eq. (8) relates the beginning and completion times of each task. Here, m_j denotes the index of the target associated with node j. Eqs. (9) and (10) ensure the operator cannot start a task until (i) a UAV arrives at the target, and (ii) the previous task completes.

4. Task Load Evolution Constraints

The following constraints govern the evolution of the operator's task load during the mission.

$$\underline{W}_k + \sum_{(i,j)\in E} x_{i,j,k}\Delta W_{m_j} = \overline{W}_k \qquad \forall k \in \{1,\dots,K\} \tag{11}$$

$$W_0 - \delta^- B_1 = \underline{W}_1 \tag{12}$$

$$\overline{W}_{k-1} - \delta^-(B_k - C_{k-1}) = \underline{W}_k \qquad \forall k \in \{2,\dots,K\} \tag{13}$$

Eq. (11) ensures that workload increments are allocated properly while the operator is busy, while Eq. (12) and (13) ensure that workload decrements are defined appropriately. As before, m_j is understood as the index of the target associated with node j.

5. Performance Constraints

The final constraints quantify relevant performance metrics.

$$\underline{w} - \underline{W}_k \leq \underline{\alpha} \qquad \forall k \in \{1,\dots,K\} \tag{14}$$

$$\overline{W}_k - \overline{w} \leq \overline{\alpha} \qquad \forall k \in \{1,\dots,K\} \tag{15}$$

$$\sum_{k=1}^{K} B_k - A_k = \beta. \tag{16}$$

D. MIP Formulation

The MINLP representation of Problem 1 is defined in Problem 2.

Problem 2 (Scheduling/Routing MINLP)**.** *Determine values for each of the decision variables (Table 1) that are optimal with respect to the following problem:*

$$\begin{aligned} &\textit{Minimize:} && p_{\underline{\alpha}}\underline{\alpha} + p_{\overline{\alpha}}\overline{\alpha} + p_\beta\beta \\ &\textit{Subject To:} && \textit{Constraints } (1) - \ (16), \end{aligned} \tag{17}$$

where $K \coloneqq M$ and $p_{\underline{\alpha}}$, $p_{\overline{\alpha}}$, $p_\beta > 0$ are fixed parameters.

In its raw form, the MIP (17) has $M|V|^2$ binary decision variables, $5M+3$ continuous decision

variables, and $(|V|-N+11)M+N+1$ algebraic constraints (excluding domain constraints, e.g., $\beta \in \mathbb{R}_{\geq 0}$). The only nonlinearity in (17) lies in (7). This non-linearity is eliminated by introducing a set of auxiliary variables and constraints, although this procedure results in significantly larger problems in general. For single vehicle missions, additional structure emerges that allows for elimination of the problem non-linearity without increasing the problem size. These results are formalized here.

Theorem 1 (Linearization)**.** *The non-linear program* (17) *is equivalent to a MILP. Moreover, in the single vehicle case ($N = 1$), an equivalent MILP exists with the same number of binary decision variables, continuous decision variables, and algebraic constraints as its non-linear counterpart* (17)*.*

Proof. Let $I \coloneqq \{(i,j,k,\overline{j},\overline{k}) \mid i,j,\overline{j} \in V, k \in \{2,\dots,M\}, \overline{k} \in \{1,\dots,k-1\}\}$, and define decision variables $y_{i,j,k,\overline{j},\overline{k}} \in \{0,1\}$, $D_{i,j,k,\overline{j},\overline{k}} \in \mathbb{R}_{\geq 0}$ for each $(i,j,k,\overline{j},\overline{k}) \in I$. Notice $|I| = |V|^3 \sum_{k=2}^{M}(k-1) = \frac{1}{2}(M(M-1))|V|^3$. Let $\overline{C} > 0$ be a very large constant and define constraints:

$$\sum_{(i,j)\in E} x_{i,j,k} W(i,j) + \sum_{(i,j)\in E} \sum_{\overline{j}\in V} \sum_{\overline{k}=1}^{k-1} D_{i,j,k,\overline{j},\overline{k}} = A_k \qquad \forall k \in \{2,\dots,K\} \tag{18}$$

$$y_{i,j,k,\overline{j},\overline{k}} \leq x_{i,j,k} \qquad \forall (i,j,k,\overline{j},\overline{k}) \in I \tag{19}$$

$$y_{i,j,k,\overline{j},\overline{k}} \leq x_{\overline{j},i,\overline{k}} \qquad \forall (i,j,k,\overline{j},\overline{k}) \in I \tag{20}$$

$$x_{\overline{j},i,\overline{k}} + x_{i,j,k} - 1 \leq y_{i,j,k,\overline{j},\overline{k}} \qquad \forall (i,j,k,\overline{j},\overline{k}) \in I \tag{21}$$

$$D_{i,j,k,\overline{j},\overline{k}} \leq \overline{C} y_{i,j,k,\overline{j},\overline{k}} \qquad \forall (i,j,k,\overline{j},\overline{k}) \in I \tag{22}$$

$$D_{i,j,k,\overline{j},\overline{k}} \leq C_{\overline{k}} \qquad \forall (i,j,k,\overline{j},\overline{k}) \in I \tag{23}$$

$$C_{\overline{k}} - \overline{C}(1 - y_{i,j,k,\overline{j},\overline{k}}) \leq D_{i,j,k,\overline{j},\overline{k}} \qquad \forall (i,j,k,\overline{j},\overline{k}) \in I. \tag{24}$$

This linear constraint set is equivalent to (7) for $\overline{C}$ sufficiently large: under (19) - (24), we have $D_{i,j,k,\overline{j},\overline{k}} = y_{i,j,k,\overline{j},\overline{k}} C_{\overline{k}} = x_{i,j,k} x_{\overline{j},i,\overline{k}} C_{\overline{k}}$, making (18) is equivalent to (7). An equivalent MILP results from replacing (7) with (18) - (24) in (17).

If $N = 1$, the operator's task processing order is identical to the UAV's target visitation order. As such, the arrival time A_k equals the time that the operator finishes the $k-1$-st task, plus the

required UAV travel time to the next target. Formally, when $N = 1$, Eq. (7) is equivalent to

$$\sum_{(i,j)\in E} x_{i,j,1} W(i,j) = A_1$$

$$C_{k-1} + \sum_{(i,j)\in E} x_{i,j,k} W(i,j) = A_k \qquad \forall k \in \{2, \ldots, K\}.$$

Replacing (7) with the above linear constraints in (17), results in a MILP. □

Despite the structure that emerges when $N = 1$, notice that the routing and scheduling problem are still implicitly coupled, i.e., the optimal route depends on the operator schedule and vice versa.

The proof of Theorem 1 also provides insight into the size of the equivalent MILPs. For instance, since the number of vehicles N is typically small, the problem size is usually dominated by the number of targets M and the size of the node set $|V|$ (which primarily depends on M and the number of viewpoints associated with each target). If each target has $P \in \mathbb{N}$ associated viewpoints, i.e., $|V_j| = P$ for all $j \in \{1, \ldots, M\}$, and the number of vehicles N is fixed, then, as M and P jointly tend to infinity, the number of binary decision variables, continuous decision variables, and algebraic constraints in the MINLP (17) are $O(M^3P^2)$, $O(M)$, and $O(M^2P)$. When $N = 1$, an equivalent MILP of the same size exists; however, from the proof of Theorem 1, we see that for general, multi-vehicle missions, the number of binary decision variables, continuous decision variables, and algebraic constraints required in an equivalent MILP are each $O(M^5P^3)$.

With the availability of high-quality MILP solvers, such as GLPK [47], CPLEX [48], and MATLAB's `INTLINPROG` solver [49], Theorem 1 may evoke a practical solution strategy for some missions. However, even when $N = 1$, problems may be large and computationally complex. In response, the following section develops a dynamic, heuristic strategy for constructing solutions to general instances of Problem 2.

Remark 3 (Parameter Selection)**.** *The parameters $p_{\underline{\alpha}}$, $p_{\overline{\alpha}}$, p_β serve to balance the impact of the three performance metrics: upper/lower task load bound violations, and unnecessary loiter time. In general, the appropriate parameter values depend on the mission setup, e.g., the number of targets, task processing times, workload modeling parameters, etc., as well as the mission objectives.*

Normalization of the optimization variables, particularly the loiter time measure β, may allow for more straightforward choice of $p_{\underline{\alpha}}$, $p_{\overline{\alpha}}$, p_{β}. However, since the time-horizon considered is infinite, i.e., there is no maximum mission length, the appropriate normalization scheme is not obvious. For example, normalization of β by the output mission length introduces additional non-linearity. Of course, if additional structure exists (or is artificially imposed), e.g., a maximum allowed mission time or a fixed maximum loiter time, then reasonable normalization schemes may emerge. A thorough study of the effect of normalization is left as a topic of future work.

V. Dynamic Solution Strategy

This section proposes a dynamic framework to construct solutions to Problem 2. Here, each time the operator finishes a task, a comparatively small-scale MILP is solved to select the next UAV destinations and the impending portion of the operator schedule. In particular, each re-planning operation results in a partial operator schedule containing at most N tasks, the first of which is the next task to be executed. In this section, we let $K \coloneqq \min\{N, M\}$, where M is understood here as the number of targets that have not yet been processed when the re-plan operation is initiated.

The MILP governing each re-plan operation depends on the current status of each UAV. Consider three status classifications: (i) `AVAILABLE`: the UAV has not been assigned any targets or the operator has just finished processing the UAV's imagery, (ii) `ARRIVED`: the UAV has reached its destination and is awaiting operator attention, and (iii) `TRANSIT`: the UAV is en route to its next destination. Assume that the status of each UAV is available to the optimizer during any re-planning operation. Re-planning is performed under the following assumptions on UAV behavior: (i) if its status is `ARRIVED`, then the UAV should remain loitering until the operator processes its task, and (ii) if its status is `TRANSIT`, then the UAV should continue on to its destination, i.e., the UAV should not be re-routed. These are natural assumptions that serve the dual purpose of both reducing computation and preventing excessive changes to UAV routes that may be undesirable from an operator or mission-planning standpoint. Note, however, that these assumptions can be relaxed through straightforward manipulations (Remark 4).

Broadly, the proposed dynamic solution strategy uses the following procedure:

1. Initialize each UAV status as `AVAILABLE`,

2. Formulate and solve the re-planning MILP,

3. Direct the UAVs for to their first destination, instruct the operator to execute the first task,

4. When the first task is complete, reformulate and solve the re-planning MILP, and

5. Repeat steps 3 and 4 until all tasks are complete.

The remainder of this section details the formulation and solution of the MILP in steps 2 and 4.

A. Graph Modifications

Consider some instant at which the re-planning operation is to be executed, and assume, without loss of generality, that $V_1, V_2, \ldots, V_M$ represent the node clusters associated with the targets that have not yet been processed (we retain this assumption for the remainder of this section). We reconstruct the graph G as follows: First, re-define $V_0 \coloneqq \{i^1, i^2, \ldots, i^N\}$, where i^ℓ represents UAV ℓ's current configuration. Second, for each $\ell \in \{1, \ldots, N\}$, define a set V^ℓ consisting of nodes to which UAV ℓ can travel prior to another re-assignment. That is, define each set V^ℓ as follows:

1. if UAV ℓ has status `AVAILABLE`, then $V^\ell \coloneqq V_1 \cup \cdots \cup V_{\overline{M}}$, where we have assumed, without loss of generality, that $\{V_1, V_2, \cdots, V_{\overline{M}}\}$ collects all node clusters that have not yet been visited by a UAV and are not the destination of any UAV with status `TRANSIT`, and

2. if UAV ℓ has status `ARRIVED`, then $V^\ell \coloneqq \{i^\ell\}$, where i^ℓ is UAV ℓ's current configuration, and

3. if UAV ℓ has status `TRANSIT`, then $V^\ell \coloneqq \{j^\ell\}$, where $j^\ell \in V$ is UAV ℓ's current destination.

Redefine $V \coloneqq V_0 \cup \left(\bigcup_{\ell=1}^N V^\ell\right)$, and redefine the edge set $E \coloneqq \{(i^\ell, j) \mid i^\ell \in V_0, j \in V^\ell\}$. Finally, define the weight operator W to be consistent with the new edges.

The remaining subsections formulate the re-planning MILP using the modified graph G.

Remark 4 (Re-Routing)**.** *The set V^ℓ contains a single element whenever UAV ℓ has status `ARRIVED` or `TRANSIT`. As such, UAV ℓ will not be directed to a new destination by the re-planning operation (see* (27)*). Alternatively, one could allow re-routing by instead defining each V^ℓ to contain all nodes that are associated with targets that have not yet been processed, regardless of UAV status.*

B. Decision Variables

The decision variables are defined nearly identically to those of Section IV B. However, there are two subtle differences: First, the value of the index k refers to the post-re-plan processing order, rather than the global processing order as before. Second, for re-planning, we only require a binary variable $x_{i,j,k} \in \{0,1\}$ for each index triplet in the set $\{(i,j,k) \,|\, i = i^\ell, j \in V^\ell, k \in \{1,\ldots,K\}, \ell \in \{1,\ldots,N\}\}$. This restriction typically produces a significantly smaller problem in comparison to (17). For instance, when N is fixed and $|V_j| = P$ for all $j \in \{1,\ldots,M\}$, then, as M and P jointly tend to infinity, the number of binary decision variables, continuous decision variables, and algebraic constraints are $O(MP)$, $O(M)$, and $O(M)$, resp.

C. Constraints

1. *UAV Path Constraints*

Define $\mathcal{L}$ as the set of UAV indices with status `AVAILABLE`, and consider the following constraints.

$$\sum_{\ell\in\mathcal{L}} \sum_{j\in V_m} \sum_{k=1}^{K} x_{i^\ell,j,k} \leq 1 \qquad \forall m \in \{1,\ldots,\overline{M}\} \tag{25}$$

$$\sum_{j\in V^\ell} \sum_{k=1}^{K} x_{i^\ell,j,k} \leq 1 \qquad \forall \ell \in \mathcal{L} \tag{26}$$

$$\sum_{j\in V^\ell} \sum_{k=1}^{K} x_{i^\ell,j,k} = 1 \qquad \forall \ell \notin \mathcal{L} \tag{27}$$

$$\sum_{(i,j)\in E} x_{i,j,k} = 1 \qquad \forall k \in \{1,\ldots,K\} \tag{28}$$

Eq. (25) says that at most one node from any cluster can be visited. Eq. (26) says that any `AVAILABLE` UAV is assigned at most one new destination, while Eq. (27) says that UAVs with status `WAITING` or `TRANSIT` do not get re-routed. Eq. (28) ensures that the maximum number of UAVs are assigned a destination, i.e., if at least N targets have not been processed, then N UAVs are assigned a destination; otherwise, all remaining targets are assigned a UAV.

Remark 5 (Running Cost)**.** *Eq.* (28) *ensures that a meaningful solution is produced by the re-planning MILP: Without* (28)*, UAVs may remain unassigned when there are still unprocessed targets, in which case the running cost is not a meaningful predictor of the overall mission cost. For*

example, if all UAVs are `AVAILABLE`*, then, without* (28)*, the zero vector is an optimal choice for the binary variables, making the running cost predicted by the re-planning MILP equal to zero.*

2. UAV Arrival Time Constraints

The following constraints govern the UAV arrival times:

$$\sum_{\ell=1}^{N} \sum_{j \in V^\ell} x_{i^\ell,j,k} W(i^\ell, j) = A_k \qquad \forall k \in \{1, \ldots, K\}. \tag{29}$$

If UAV ℓ has status `ARRIVED`, then $V^\ell \coloneqq \{i^\ell\}$. Since $W(i^\ell, i^\ell) = 0$, Eq. (29) implies that the arrival time associated with a UAV having status `ARRIVED` is zero (relative to the time re-plan is initiated). As such, the operator is permitted to begin any task associated with such a UAV immediately. In contrast, if a UAV has status `TRANSIT` or `AVAILABLE`, then the weight $W(i^\ell, j)$ captures the time required for the UAV to travel from its current location, i^ℓ, to its new destination in the set V^ℓ. As such, the operator cannot start any such task until the corresponding UAV arrives at its destination.

Remark 6 (Replan Computation Time)**.** *Our formulation implicitly assumes that re-planning is instantaneous, i.e., UAVs do not move during re-planning computation. If necessary, a computational "buffer" can be built into the re-plan operation, which uses projected, rather than current, UAV locations to ensure validity of the resultant route under non-negligible computation time.*

3. Task Processing, Task Load Evolution, and Performance Constraints

As before, we enforce the constraints (8)- (16), where W_0 is understood as the operator task load level at the re-plan onset. Define $\overline{\alpha}_0, \underline{\alpha}_0 \in \mathbb{R}_{\geq 0}$ as the maximum upper and lower task load bound violations that have occurred prior to the current re-plan, and introduce two final constraints:

$$\overline{\alpha} \geq \overline{\alpha}_0 \tag{30}$$

$$\underline{\alpha} \geq \underline{\alpha}_0. \tag{31}$$

These constraints ensure that the running cost correlates with the global mission cost. Note that there is no need to add an additional parameter to reflect loiter times occurring prior to the current

re-plan, since past loiter times would enter into the global objective function as a constant that is independent of the re-plan decision variables. That is, the optimal choice for the UAV destinations and operator schedule over the impending time horizon is independent of the past loiter times.

4. MILP Formulation

The MILP governing the re-plan operation is formally expressed in Problem 3.

Problem 3 (Re-planing operation as a MILP)**.** *Determine values for each of the decision variables in Table 1 that are optimal with respect to the following problem:*

$$\begin{aligned} &\textit{Minimize:} && p_{\underline{\alpha}}\underline{\alpha} + p_{\overline{\alpha}}\overline{\alpha} + p_{\beta}\beta \\ &\textit{Subject To:} && \textit{Constraints } (8) - (16) \textit{ and } (25) - (31), \end{aligned} \tag{32}$$

where $K \coloneqq \min\{N, M\}$*,* W_0 *is the operator's current task load, and* $p_{\underline{\alpha}}$*,* $p_{\overline{\alpha}}$*,* $p_{\beta} > 0$ *are constants.*

Remark 7 (Task Processing)**.** *The optimization problem* (33) *predicts the incremental cost over the impending portion of the operator's schedule under the implicit assumption that the operator processes one task generated by each UAV before processing any other tasks (see* (28)*). This assumption does not place an explicit constraint on the global structure of the operator schedule. That is, assuming Problem 3 is solved each time a task is completed, then it is still possible for the operator process two tasks in a row that are generated by the same UAV.*

VI. Uncertain Processing Times

The analysis presented thus far has assumed that the processing time associated with each target is known *a priori*. Indeed, strictly speaking, known processing times are required to ensure that the optimization framework correctly relates task beginning times and completion times (see (8)), which are used to predict subsequent UAV arrival times. This assumption does not hold in most realistic systems, since operator behavior is subject to various types of uncertainty.

The most common approach to addressing uncertainty is to use a single realization of each uncertain parameter, e.g., expected values, within the optimization framework. Sophisticated robust optimization schemes are sometimes useful, although these methods generally require bounded

uncertainty sets and very particular problem structure [50]. In contrast, *scenario-based* robust optimization schemes use discrete samples to approximate uncertainty sets, requiring optimization constraints be satisfied for *all* of the sampled conditions, rather than just one [51, 52]. As a result, solutions produced by scenario-based schemes are less likely to exhibit poor performance when uncertain parameters are realized. Although sampling-based schemes are simplistic in some sense, they produce reasonable solutions in a straightforward and intuitive manner, making them attractive to both theoreticians and practitioners alike.

This section introduces a straightforward, scenario-based extension to the framework of Section V, with the goal of providing robustness to uncertainty in operator processing times. That is, the optimization problem developed here is a robust alternative to (33). We use a sampling-based method since (i) the complex, coupled nature of the joint routing/scheduling problem makes it difficult to predict worst-case parameter values directly, (ii) typical processing time distributions used to model perceptual decision-making are skewed and have unbounded support [53, 54]; as such, expected values may be inaccurate processing time predictors and methods requiring bounded support would require additional constraints, (iii) sampling parameters allow a straightforward means of tuning the "degree of robustness" provided in order to strike a balance between performance and computational complexity, and (iv) scenario-based schemes are simple, intuitive, and use a straightforward procedure that does not require any particular uncertainty distribution.

Formally, let X be the set containing all tuples $(\{x_{i,j,k}\}, \{A_k\}, B_1)$ whose components satisfy (25)- (29). Let T be the set of all tuples $(\tau_1, \tau_2, \ldots, \tau_M)$, where τ_j is a possible processing time of target j. Finally, let $Y(\boldsymbol{x}, \boldsymbol{A}, B, \boldsymbol{\tau})$ be the set containing all tuples $(\{\overline{W}_k\}, \{\underline{W}_k\}, \{B_k\}, \{C_k\}, \underline{\alpha}, \overline{\alpha}, \beta)$ whose components are optimal with respect to Problem 3 under the additional constraint that $\{x_{i,j,k}\} \coloneqq \boldsymbol{x}$, $\{A_k\} \coloneqq \boldsymbol{A}$, $B_1 = B$, and when the processing times are defined $(\tau_1, \tau_2, \ldots, \tau_M) \coloneqq \boldsymbol{\tau}$. With these definitions, the scenario-based formulation is an approximation to Problem 4.

Problem 4 (Robust Re-Planning)**.** *Determine values for each binary decision variable $x_{i,j,k}$, each arrival time variable A_k, and the time B_1 that the operator begins the first post-re-plan task, that*

are optimal with respect to the following problem:

$$\underset{(\boldsymbol{x},\boldsymbol{A},B_1)\in X}{\textit{Minimize}:} \quad \sup_{\substack{\boldsymbol{\tau}\in T \\ Y(\boldsymbol{x},\boldsymbol{A},B_1,\boldsymbol{\tau})}} \left\{p_{\underline{\alpha}}\underline{\alpha}\right\} + \sup_{\substack{\boldsymbol{\tau}\in T \\ Y(\boldsymbol{x},\boldsymbol{A},B_1,\boldsymbol{\tau})}} \left\{p_{\overline{\alpha}}\overline{\alpha}\right\} + \sup_{\substack{\boldsymbol{\tau}\in T \\ Y(\boldsymbol{x},\boldsymbol{A},B_1,\boldsymbol{\tau})}} \left\{p_{\beta}\beta\right\}. \tag{33}$$

Problem 4 is intuitively interpreted as follows: When initiating each re-plan operation, we seek a single, well-defined choice for each UAV's next destination and associated arrival times, along with a single choice of when the operator should start the first post-re-plan task, such that the worst-case incremental cost over the immediate horizon is minimized. Notice that this optimization considers *all* possible choices of target processing times, rather than just one. Problem 4 is, in essence, a conservative min-max robust formulation that attempts to minimize the *worst-case* objective value that can be achieved. If processing times are not subject to an upper bound, i.e., the underlying distribution of processing times has semi-infinite support, then the objective function of Problem 4 will typically have no maximum over the feasible solution set. In this case, practical solution approaches seek to find a solution that performs as well or better than the predicted bound with high probability. The scenario-based approach accomplishes this using samples drawn from the underlying processing time distributions as approximations to the uncertainty sets.

A. Constructing Scenarios

Suppose the operator's processing time for each target (task) j is realized according to a probability density function $f_j : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$. For each j, generate a set of $Q \in \mathbb{N}$ possible processing times $\{\tau_j^1, \tau_j^2, \ldots, \tau_j^Q\}$, where $\tau_j^q \sim f_j$ for $q \in \{1, \ldots, Q\}$. For each $q \in Q$, the set $\{\tau_1^q, \tau_2^q, \ldots, \tau_M^q\}$ defines a *scenario*. Note that this sampling process associates each target j with a set of processing times, rather than a single realization.

B. Decision Variables

The scenario-based extension of the MILP (33) contains only slightly modified decision variables to accommodate the generated processing time sets. In particular, the binary variables $\{x_{i,j,k} \in \{0,1\}\}$, the arrival time variables $\{A_k\}$, and the performance variables $\overline{\alpha}$, $\underline{\alpha}$, β are identical to those

of (33). For the remaining decision variables, we introduce duplicates, indexed by the superscript q, that are each associated with one scenario. That is, for each $q \in \{1, \ldots, Q\}$, we define unique decision variables $B_k^q, C_k^q, \overline{W}_k^q, \underline{W}_k^q$ that are associated with the q-th scenario.

Remark 8 (Arrival Times)**.** *Scenario-dependent arrival times A_k are not required, since each re-plan only selects each UAV's next destination (its current location if its status is* `ARRIVED`*). Since UAVs begin moving immediately, these arrival times are independent of operator processing times.*

C. Constraints

The decision variables are subject to constraints (14), (15), and (25) - (29). The scenario-based analogs of the remaining constraints are as follows:

$$B_k^q + \sum_{(i,j)\in E} x_{i,j,k}\tau_{m_j}^q = C_k^q \qquad \forall k \in \{1,\ldots,K\}, q \in \{1,\ldots,Q\} \tag{34}$$

$$A_k \leq B_k^q \qquad \forall k \in \{1,\ldots,K\}, q \in \{1,\ldots,Q\} \tag{35}$$

$$C_{k-1}^q \leq B_k^q \qquad \forall k \in \{2,\ldots,K\}, q \in \{1,\ldots,Q\} \tag{36}$$

$$\underline{W}_k^q + \sum_{(i,j)\in E} x_{i,j,k}\Delta W_{m_j} = \overline{W}_k^q \qquad \forall k \in \{1,\ldots,K\}, q \in \{1,\ldots,Q\} \tag{37}$$

$$\underline{w} - \underline{W}_k^q \leq \underline{\alpha} \qquad \forall k \in \{1,\ldots,K\}, q \in \{1,\ldots,Q\} \tag{38}$$

$$\overline{W}_k^q - \overline{w} \leq \overline{\alpha} \qquad \forall k \in \{1,\ldots,K\}, q \in \{1,\ldots,Q\} \tag{39}$$

$$\sum_{k=1}^{K} B_k^q - A_k \leq \beta \qquad \forall q \in \{1,\ldots,Q\} \tag{40}$$

$$B_1^q - B_1^{q-1} = 0 \qquad \forall q \in \{2,\ldots,Q\} \tag{41}$$

Constraints (34) - (40) are analogous to those in Section V. Constraint (41) ensures that a unique start time is produced for the first task in the operator's new schedule (remaining start times should remain scenario-dependent). Since a re-plan operation is performed whenever a task is completed, an unambiguous operator schedule can always be extracted from the MILP solution.

Remark 9 (Task Load Increments)**.** *The scenario-based scheme also allows the possibility of scenario (time)-dependent increments ΔW_j. For example, if $g : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ captures the relationship*

between processing time and the resultant task load increment, then for each q, one can define $\{\Delta W_1^q, \Delta W_2^q, \ldots, \Delta W_M^q\}$, *where* $\Delta W_j^q \coloneqq g(\tau_j^q)$ *for each* $j \in \{1, \ldots, M\}$. *The analogous robust MILP is formulated by replacing each instance of* ΔW_j *with* ΔW_j^q *in the optimization constraints.*

D. Scenario-based, Robust MILP

The scenario-based extension to Problem 3 is stated as follows.

Problem 5 (Robust Re-planning as an MILP)**.** *Determine values for the decision variables described in Section VI B that are optimal with respect to the following problem:*

$$\begin{aligned} &\textit{Minimize:} && p_{\underline{\alpha}}\underline{\alpha} + p_{\overline{\alpha}}\overline{\alpha} + p_\beta \beta \\ &\textit{Subject To:} && \textit{Constraints } (14), (15),\ (25) -\ (29),\ (34) -\ (40) \end{aligned} \tag{42}$$

where $K \coloneqq \min\{N, M\}$, W_0 *is the operator's current task load, and* $p_{\underline{\alpha}}$, $p_{\overline{\alpha}}$, $p_\beta > 0$ *are constants.*

VII. Simulation Studies and Discussion

This section contains simulations and discussion to illustrate the advantages and limitations of the proposed solution strategies. In all studies, optimal MILP solutions were estimated using the stand-alone `glpsol` solver, which is included in GLPK v. 4.60 [47].

A. Performance of the Dynamic Solution Strategy

The first study compares the performance of the receding-horizon strategy of Section V to that of an *a priori* planning method, which constructs a complete solution to the full problem (17) at the mission onset. To allow direct computation of solutions to (17), we consider a small-scale mission with 2 UAVs (with "hovering" capability) and 4 targets, each with a single viewpoint ($|V_j| = 1\ \forall j$). The UAVs move with speed 39 m/s, and travel times are defined as the minimal straight-line travel time between configurations. The initial task load is $W_0 = 0.5$, with an increment/decrement rate of 0.001 during busy/idle times, and the operator takes 483.32 s to complete each task ($\tau_j = 483.32$ s $\forall j$). This particular dwell-time value is used simply for consistency with operational scenarios involving dynamically constrained UAV models, as it corresponds to the time required for a fixed-

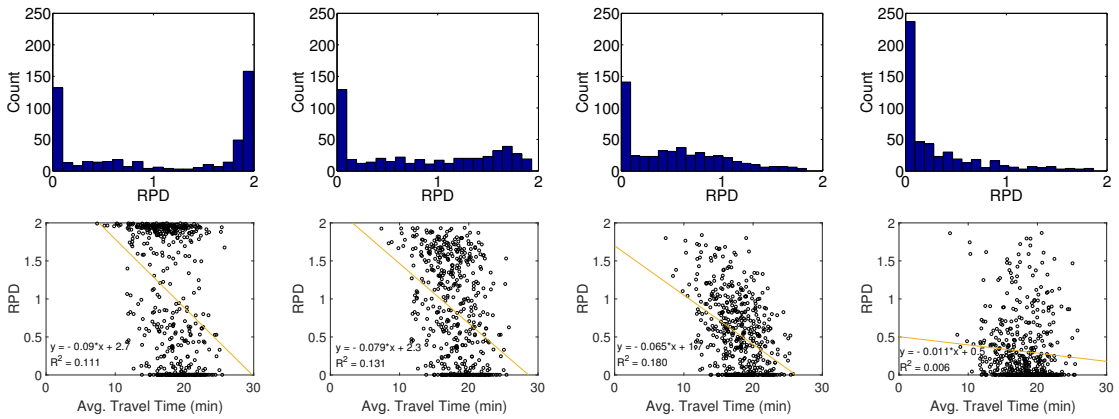

Fig. 5 RPD between dynamic and *a priori* schemes for (left to right) $p_\beta = 0.1, 0.01, 0.001, 0.0001$**.**

wing UAV to complete 4 dwell-time loops at a 750 m minimum turning radius while traveling with speed 39 m/s. The desired task-load range is $[\underline{w}, \overline{w}] = [0.2, 0.8]$ and $p_{\overline{\alpha}} = p_{\underline{\alpha}} = 1$. In each simulation run, UAV initial locations and target locations are selected randomly (uniform distribution) within an $80,000 \times 80,000$ m region, and an overall mission cost was calculated under both dynamic re-planning (Section V) and *a priori* planning for each of four p_β values: $0.1, 0.01, 0.001, 0.0001$. Direct solutions to (17) were constructed by solving the equivalent MILP (Theorem 1) using `glpsol`.

Fig. 5 shows the error generated, for each of 500 simulated missions, between the dynamic and *a priori* strategies in the form of a histogram and as a scatter plot (plotted against the average travel times, i.e., edge weight in the complete graph G), for each p_β condition. The figures also show linear regression lines and R^2 values for reference. Error is reported as the *relative percent difference* (RPD), where RPD $\coloneqq 2$ (dynamic cost $-$ *a priori* cost)/(dynamic cost $+$ *a priori* cost).

Fig. 5 suggests that the solutions provided by the dynamic heuristic are closest to the optimal solution to (17) when: (i) UAV travel times between destinations are large (resulting in the negative slopes of the corresponding regression lines), and (ii) p_β is small in comparison to $p_{\underline{\alpha}}, p_{\overline{\alpha}}$, i.e., workload considerations are more prominent than unnecessary loitering considerations (as illustrated by the decreasing magnitude of the regression slopes). Outside of these regimes, the dynamic heuristic performs significantly worse than the *a priori* method. However, a few comments are in order: First, the very small size of this example allows for accurate estimation of optimal solutions to Problem 2 through transformation to and subsequent solution of an equivalent MILP. In general, the computational burden involved with this and other similar methods will not allow for the direct

Table 2 Target Locations

Index, j	Location, t_j
1	$(10000, 0)$ m
2	$(5000, 500)$ m
3	$(1200, 5000)$ m
4	$(-400, -15000)$ m
5	$(3000, -5000)$ m
6	$(-3000, -4000)$ m
7	$(1500, 10000)$ m
8	$(-200, -10000)$ m
9	$(4000, -7000)$ m
10	$(0, -7500)$ m

Table 3 Comp Times and Costs; 2 UAV Example Mission

	A Priori (MILP)		*A Priori* (GA)		Dynamic (MILP)	
M	Cost	Comp. Time (s)	Cost	Comp. Time (s)	Cost	Comp. Time (s)
2	1.289	0.015	NA	>30000	1.403	0.011
3	1.289	0.069	NA	>30000	1.493	0.005
4	1.319	1.478	NA	>30000	1.393	0.006
5	1.289	5.506	NA	>30000	1.840	0.006
6	1.282	22.611	NA	>30000	2.201	0.007
7	1.282	161.538	NA	>30000	2.389	0.007
8	1.282	170.561	NA	>30000	2.605	0.008
9	1.282	4091.000	NA	>30000	3.107	0.008
10	NA	> 30000	NA	>30000	3.709	0.007

construction of a high-quality, complete solution to (17) in reasonable time. Thus, *a priori* methods are usually impractical for general use.

This fact is illustrated by Table 3 and Table 4, which provide plots of performance and computation time as a function of both the number of targets M and the number of vehicles N, respectively, using various solution methods. Here, computation time results were averaged over 5 runs performed on a 1.8 GHz Intel Core i7 processor. For the dynamic scheme, the figure only reports the computation time required to construct the initial plan, since the computation time for each

Table 4 Comp Times and Costs; 6 Target Example Mission

	A Priori (MILP)		*A Priori* (GA)		Dynamic (MILP)	
N	Cost	Comp. Time (s)	Cost	Comp. Time (s)	Cost	Comp. Time (s)
1	1.282	5.937	NA	> 30000	1.282	0.005
2	1.282	21.142	NA	> 30000	2.201	0.010
3	1.282	12.188	NA	> 30000	3.616	0.010
4	1.282	657.382	NA	> 30000	4.811	0.012
5	1.282	389.816	NA	> 30000	5.461	0.015
6	1.282	178.726	NA	> 30000	5.937	0.021

Table 5 Effect of Increasing Target Spacing in 9 Target Case

Mean Inter-Target Travel Time (min)	*A Priori* (MILP) Cost	Dynamic (MILP) Cost	RPD
4.15	1.282	3.107	0.832
8.57	2.564	2.574	0.004

successive re-plan was negligible in comparison to the initial planning time in all tested cases. To generate Table 3, a similar setup was used as in the previous study (setting $p_{\underline{\alpha}} = p_{\overline{\alpha}} = 10$, $p_\beta = 0.01$, $W_0 = 0.2$, $\tau_j = 241.66$ for all j, and keeping other parameters as before); however, target locations were fixed. Table 2 shows the particular target locations used in this study (Targets 1-2 were used for the 2 target scenario, Targets 1-3 in the 3 target scenario, etc.). Recall that the observed cost is a function of operator workload bound violations and unnecessary UAV loiter time, but is not a function of the overall mission time directly. As a result, the cost may remain unchanged or even decrease if additional targets are added, provided that the additions do not force the operator into a higher-workload regime, and does not introduce excessive idle time. This behavior is seen in Table 3, as the cost is unchanged for the $M = 6, 7, 8, 9$ case.

For Table 4, a 6 target scenario (Targets 1-6; Table 3) was used, and all vehicles were initially placed at a location of $(0, 0)$. Notice that the computation time for the dynamic heuristic stays relatively constant across all conditions, while the computation time for the *a priori* method (using the MILP solver) quickly diverges to infeasible levels as complexity increases. For the case shown, the relative error between the dynamic heuristic and the a priori method also increases with problem complexity. However, we note once again that, in general, the performance of the heuristic improves in situations where inter-target travel times are large. For example, in the 9 target case in Table 3, the average inter-target travel time is 4.15 minutes. If the setup is modified by shifting the target locations radially from the origin by a factor of 2 (making the average inter-target travel time 8.57 min), the relative error between the two methods becomes negligible (see Table 5). This trend is reflective of the fact that missions with larger travel times typically benefit from the inclusion of more UAV assets, since this prevents excessive operator idle time.

For comparison, an evolutionary heuristic solver was used in attempt to find solutions to the full, non-linear problem (17). In particular, the NSGA-II genetic algorithm, which is one of the

most widely used evolutionary algorithms for finding solutions to nonlinear, constrained, mixed-integer (and multi-objective) problems [55], was implemented on the test problems described in the previous paragraph. However, due to the highly constrained nature of the mixed-integer program, NSGA-II failed to find feasible solutions for any of the tested problem instances in reasonable time (40000 generations and population size of 10000; requiring > 8 hrs to complete). This result is reflective of the well-known difficulties associated with solving highly constrained optimization problems (particularly those with equality constraints and very sparse feasible regions) using genetic algorithms or other similar heuristics [56]. Regardless, the results of this exercise are included in Tables 3 and 4 for compleness (labeled "GA"). The development and/or assessment of a sophisticated constraint handling technique to allow specialized implementation of these types of metaheuristics on the joint scheduling/routing problem is an interesting avenue of future research.

Second, we remark that instances in which the dynamic heuristic performs poorly seem to be correlated with unnecessary UAV utilization. For example, the dynamic heuristic often assigns the same number of targets to each UAV, which, when targets are close together, forces the UAVs to loiter at target destinations for long time intervals while awaiting operator attention. In contrast, the *a priori* method will avoid these penalties by leaving a subset of the UAVs un-utilized. This is supported by Table 4, which shows an increasing trend in cost values as the number of UAVs is increased. Here, in all cases, the solution provided by the full, a priori solver only utilized one of the available UAVs, and achieved much better scores. In light of this, performance of the dynamic heuristic in some cases could be improved by omitting a subset of the available UAVs.

To illustrate this point, Fig. 6 shows the relative error produced for the $p_\beta = 0.1$ case when one of the UAVs is omitted from the mission in Fig. 5, i.e., errors are reported for 500 simulation runs for a single-vehicle mission using the same setup as before. Analogous plots for other p_β conditions are nearly identical and are thus omitted. This observation suggests that a contributing factor to poor performance of the dynamic heuristic is a poorly designed mission, i.e., too many UAVs assigned to a mission. This is an important issue that can guide structural modification of both mission plans and the algorithm to improve performance. Investigation into this issue is a topic of future work.

We finish by noting that *a priori* methods that seek to solve Problem 2 directly do not readily

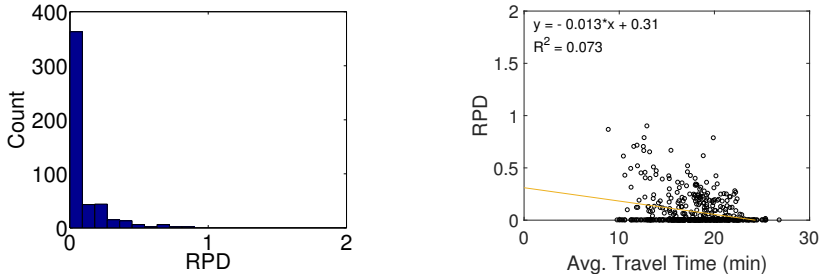

Fig. 6 RPD between dynamic and *a priori* schemes when $N = 1$ and $p_\beta = 0.1$.

extend to cases in which processing times are uncertain or in which UAVs do not have hovering capability, whereas the dynamic scheme is easily extended to incorporate these scenarios (see Section VII B). A thorough exploration of alternative formulations to improve performance with respect to direct solution methods is left as a topic of future work.

B. Scenario-Based Planning

The next study demonstrates the utility of the scenario-based scheme of Section VI. We consider a realistic example which relaxes many assumptions on UAV and operator behavior. Thus, this study not only shows robustness qualities of the scenario-based method, but also illustrates its flexibility in accommodating constraints such as (i) fixed-wing UAVs, which cannot "hover" and have a minimum turning radius, (ii) user-specified imaging constraints, and (iii) uncertain processing times.

1. Mission Setup

We consider a mission involving $N = 3$ fixed-wing UAVs and $M = 6$ static ground targets. The UAVs are each modeled as a Dubins vehicle, which has a fixed forward speed of 39 m/s, a fixed altitude of 1000 m, and a minimum turning radius of 750 m. The initial location and heading of each UAV is $(0, 0)$ and 0, resp. We neglect all other dynamic effects, e.g., drag, and we do not consider the possibility of collision. The mission requires that each target be imaged with a tilt-angle (measured below the horizontal flight plane) within the range $[\pi/8, 3\pi/8]$. The ground-plane location of each target is presented in the table on the left-side of Fig. 7.

Processing times are modeled as random variables, each realized according to a probability density function $f : \mathbb{R}_{>0} \to \mathbb{R}_{\geq 0}$. More specifically, for each $j \in \{1, \ldots, M\}$, the processing time is

Index, j	Location, t_j
1	$(10000, 0)$ m
2	$(5000, 500)$ m
3	$(1200, 5000)$ m
4	$(-400, -15000)$ m
5	$(3000, -5000)$ m
6	$(-3000, -4000)$ m

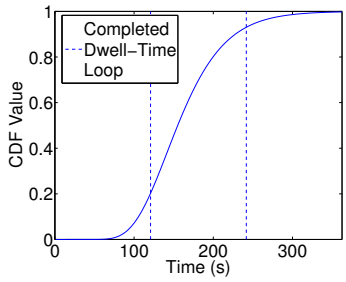

Fig. 7 Target locations (left) and processing time CDF (right)

log-normally distributed with parameters $\mu = 5.044$, $\sigma = 0.25$ (log mean and log standard deviation). Intuitively, these parameters indicate that, at any single target, the operator will usually ($\sim 75\%$ probability) require the UAV to make between 1 and 2 complete dwell-time loops in order to complete the analysis task. The cumulative distribution function (CDF) generated by f is included in Fig. 7. Let $W_0 = 0.2$, and $[\underline{w}, \overline{w}] = [0.2, 0.8]$, and assume a time-dependent, linear task load evolution model, in which task-load levels increment or decrement by $0.001t$ when the operator is busy or idle for time t, respectively. We select the scaling factors $p_{\underline{\alpha}} = p_{\overline{\alpha}} = 10$ and $p_\beta = 0.01$.

2. Solution Approach

We apply the dynamic scheduling/routing framework to the example mission using the following steps: First, we construct $V_1, \ldots, V_N$ by applying applying the discretization strategy of [29]; namely, each set V_j consists of discrete points in the configuration space ($\mathbb{R}^2 \times [0, 2\pi)$) from which the UAV is able to immediately begin an appropriate loitering pattern (a "loop" at minimum turn radius) at the target j. More precisely, each sample $(x, \theta) \in V_j \subset \mathbb{R}^2 \times [0, 2\pi)$ has a heading θ that points in a direction tangent, at the location x, to a circle of radius 750 m (the minimum possible turning radius), which lies entirely within the annulus of locations from which the tilt-angle specification is met. Edge weights in G are defined as the time required for the UAV to traverse the optimal Dubins path between configurations. After constructing G, we apply the scenario-based scheme of Section VI, using scenario-dependent task load increments according to the linear model, i.e., for each sampled processing time τ_j^q, we let $\Delta W_j^q := 0.001\tau_j^q$ (see Remark 9).

3. Performance Analysis

For comparison, we consider a baseline solution method that ignores human factors considerations, as well as the need to synchronize operator and UAV resources. In particular, the baseline method proceeds as follows. First, targets are assigned to each UAV using a greedy heuristic (similar to the well-known "nearest neighbor" heuristic for TSPs) that selects target/UAV pairings incrementally according to which unvisited target destination can be reached at the earliest time. Here, the heuristic assumes a processing time of 362.49 s at each target (ignoring operator availability), equivalent to the time required for the UAV to complete 3 dwell-time loops. Intuitively, this choice can be thought of as a type of "worst"-case processing time, since the probability of realizing a longer processing time is $< .01$. Next, given the UAV-target pairings, each UAV's route is then constructed using a TSP-based method similar to that of [29], which attempts to minimize the total route time under the assumed task processing times and an initial maneuver constraint (for consistency with [29], the baseline method used here constrains the time that the UAVs reach their first target to be minimal during the initial planning phase, and no constraint is enforced in successive re-plans). The baseline solution then assumes that the operator processes tasks as soon as possible, in a first-come first-serve basis. UAVs dwell at each successive target destination until the operator processes the task. Each time the operator finishes a task, the UAV routes are re-planned by repeating the same assignment/routing process over the remaining targets.

We compare the baseline method to the scenario-based method over multiple simulated mission executions. Here, the "actual" processing times (unknown to the planner) were sampled from the distribution f. Fig. 8 shows an example mission progression for the baseline solution (left column) and for the scenario-based strategy of Section VI (right column), using identical "actual" processing times (processing time realizations) in each case. Here, the shaded annular regions represent the locations at which the tilt angle specification can be met. Fig. 9 depicts the utilization of UAV and human resources during the same mission. Notice, in particular, how the optimal routing strategy differs when using the joint optimization as opposed to the baseline solution. Indeed, in the joint optimization case, longer UAV routes between targets may be chosen if the operator's task load level is high or to avoid unnecessary loitering times. In contrast, the baseline solution attempts to

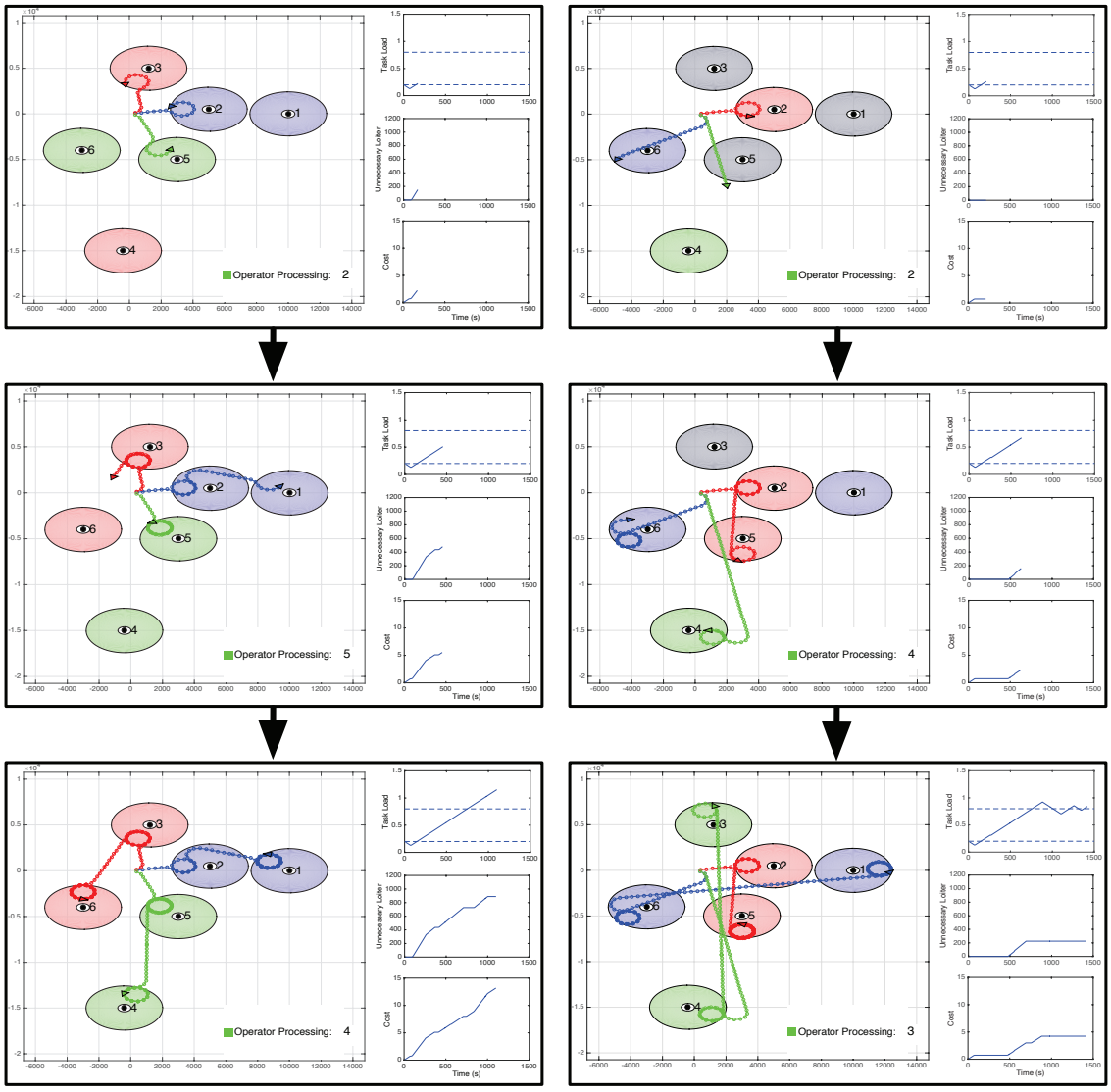

Fig. 8 An example mission progression for the baseline solution (left column) and the scenario-based solution (right column).

minimize route lengths and ignores operator behavior.

Fig. 10 shows the costs obtained over 100 simulated missions using various planning strategies, in both tabular and graphical form. Table 6 contains the average computation times for each method. The strategies included are: (i) the baseline method, (ii) two instances of the dynamic

Table 6 Computation Times (s)

	Baseline	Expect	"Worst"	$Q = 1$	$Q = 5$	$Q = 10$
Initial Planning Step	0.066	0.900	0.174	0.859	7.071	23.251
Successive Re-plan Steps	0.0414	0.019	0.017	0.026	0.077	0.138

scheme of Section V, one of which assuming expected processing time values (labeled "Expect") and

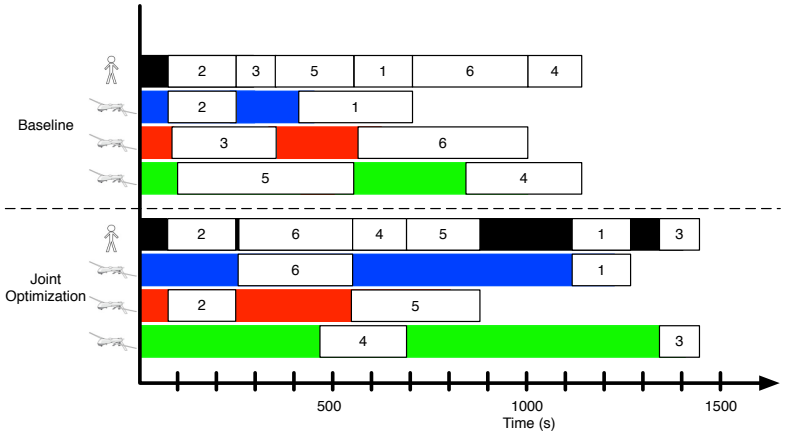

Fig. 9 Resource utilization during the mission depicted in Fig. 8

Method	Median	IQR	Whisker Span
Baseline	11.2053	3.2488	11.7385
Expect	3.3812	2.7701	7.7412
"Worst" Case	2.7247	2.1602	6.5284
$Q = 1$	2.9615	2.2192	6.7322
$Q = 5$	2.7821	2.2750	5.6693
$Q = 10$	2.9158	1.9449	5.4648

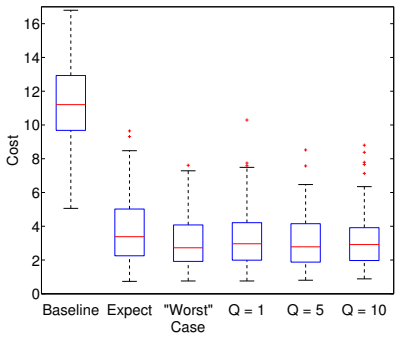

Fig. 10 Scenario-Based Method vs. the Baseline Method

one assuming "worst"-case processing times (as in the baseline case), and (iii) three instances of the scenario-based method of Section VI corresponding to $Q = 1$, 5, and 10. Different "actual" processing times were sampled for each run, but were held constant across conditions, i.e., each simulation run was performed by first sampling f to obtain "actual" processing times, and subsequently testing each condition using the realized values. Any points whose distance from the median is further than 2.5 times the inter-quartile range (IQR) are considered outliers. Means produced by the different methods were compared pairwise using two-sample t-tests, without assuming equal variance, and the

Table 7 Mean Comparison p-values

Method	Baseline	Expect	"Worst"	$Q = 1$	$Q = 5$	$Q = 10$
Baseline	–	$< 10^{-20}$	$< 10^{-20}$	$< 10^{-20}$	$< 10^{-20}$	$< 10^{-20}$
Expect	$< 10^{-20}$	-	0.039	0.125	.0461	0.073
"Worst"	$< 10^{-20}$	0.039	-	0.603	0.9358	0.7858
$Q = 1$	$< 10^{-20}$	0.125	0.603	-	0.6590	0.8033
$Q = 5$	$< 10^{-20}$	.0461	0.9358	0.6590	-	0.846
$Q = 10$	$< 10^{-20}$	0.073	0.7858	0.8033	0.846	-

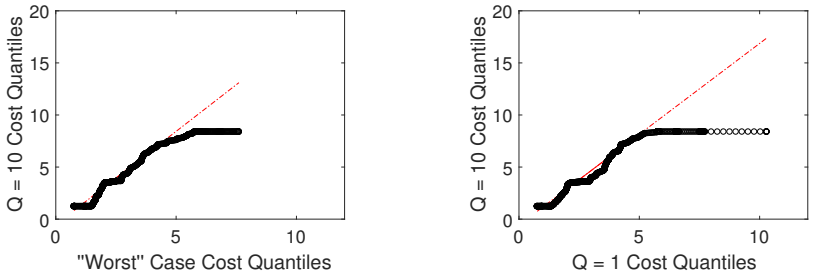

Fig. 11 Q-Q plots of "Worst" case vs $Q = 10$ case (left), and $Q = 1$ vs $Q = 10$ case (right)

result is shown Table 7. As expected, all of the joint optimization schemes performed significantly better than the baseline. Also note that the expectation case yielded a higher mean cost than the other optimization-based methods, producing statistically significant differences at 90% confidence from the "Worst Case", $Q = 5$, and $Q = 10$ cases. While there were no statistically significant differences in mean between the "Worst," $Q = 1$, $Q = 5$, and $Q = 10$ cases, qualitative trends emerge as the number of samples increases that are indicative of increased robustness. In particular, increasing the number of samples provides increased protection against worst-case performance. This is best illustrated by the Q-Q plots in Figure 11, which compare the quantiles of the "Worst" case versus the $Q = 10$ case (left) and the $Q = 1$ versus the $Q = 10$ case (right). Notice how the upper quantiles have significantly lower values for the $Q = 10$ case, indicating better worst-case performance. Note also that decreasing the number of poor-performing instances typically has the secondary effect of reducing cost variation, as shown by the emerging trend of decreasing IQR and whisker span as the number of samples increase.

Using expected values generally produced higher costs (and much higher cost variance than the scenario-based scheme, highlighting the risk involved with this type of naive sampling in the presence uncertainty. With respect to the "worst"-case approach, we remark that, due to the potentially conflicting nature of the performance metrics, longer processing times do not necessarily translate into inferior performance. As such, the 10-scenario method often out-performed the "worst"-case method. As the number of scenarios increases, one expects the appearance of poor solutions (outliers) to become less frequent, which generally results in a decrease in cost variance. This property is an inherent benefit of the scenario-based method: it protects against poor quality solutions even in complex missions where it is not straight-forward to assess which conditions will produce the

worst-case mission cost. The scenario-based method also allows the user to tune the degree of robustness, by using greater or fewer numbers of scenarios. Note, however, that increased robustness introduced by higher numbers of scenarios may be at the expense of inferior *average* performance.

Although the trends illustrated in this example are somewhat typical, it is important to note that the benefit that is gained from implementing a scenario-based approach as opposed to, e.g., choosing expected values, is sensitive to the particular problem at hand. That is, there may be missions in which choosing expected or "worst"-case processing times may actually result in very high-quality solutions, in which case the benefit gained by the scenario-based method may be offset by the added computation. We leave a thorough study of the sensitivity of the scenario-based optimization scheme to problem parameters as a topic of future work.

VIII. Other Extensions

To aid in reader understanding and to more clearly convey intuitive ideas, the optimization framework proposed herein has been presented under a number of strategically chosen assumptions, e.g., homogenous vehicles, single operator, etc. We emphasize, however, that many of these assumptions can be relaxed and a number of additional layers of complexity can be incorporated via straightforward modifications to the MIP structure. The potential for these types of additions allows system designers to tailor the framework to particular applications.

Since there are a virtually infinite number of possible problem variations, a complete discussion of all potential extensions is infeasible. However, we briefly discuss a few key extensions of interest here, and give descriptions of the appropriate manipulation required to address these factors.

A. Structural and Mission-Specific Assumptions

Multiple Operators - The presence of multiple operators can be incorporated into our framework by adding additional workload terms into the objective function, and augmenting the decision variables $x_{i,j,k}$ with an extra index representing operator assignment.

Agent-Specific Tasks - If certain tasks can only be processed by a particular vehicle or operator set, then an additional constraint set can be added which restricts the sum of the binary decision variables to be non-zero for the appropriate choices of indices.

Multi-tasking - Operator multi-tasking can be permitted by relaxing constraint (10), and making modifications to the task-load evolution model, if necessary (see Section VIII B).

Heterogenous Vehicles - Heterogenous UAVs, e.g., those with differing airspeeds or dynamic constraints, can be considered within the framework by making copies of the travel route graph G, and defining the edge weights in each copy to be consistent with travel times for one of the vehicles.

Vehicle Endurance - Endurance, e.g., battery life constraints can be placed on the vehicles by enforcing a constraint on the total route time or distance traveled by each vehicle, i.e., the sum of the edge weights on the appropriate travel graph.

Time Windows - Time windows for task processing can be incorporated by constraining the variables B_k (or C_k) to lie within pre-specified bounds. If the time windows are task-dependent, then a non-linear constraint of the form $\sum_{i \in V} \sum_{j \in V_m} \sum_{k=1}^{K} x_{i,j,k} B_k \in \text{WINDOW}_m$ for each $m \in \{1, \ldots, M\}$ would be required.

Task Priorities - "Hard" constraints on task ordering can be added into the formulation through the addition of ordinal constraints on the appropriate sums of the binary variables $x_{i,j,k}$, while "soft" constraints, i.e., ordinal preferences, can be incorporated through a number of straightforward modifications, e.g., addition of penalty terms on the task start times into the objective function.

Although these and many other extensions can be considered within similar MIP-based frameworks, these additional layers of complexity may add non-linearities and are typically at the expense of additional computation. Therefore, a natural tradeoff emerges between computational feasibility and the chosen problem assumptions, which should be assessed on a case by case basis.

B. Operator Modeling Assumptions

To illustrate the incorporation of human-factors inspired metrics into the system-level optimization framework, we have chosen a simple, utilization-based task-load measure, which serves as a proxy for operator workload. We have presented the proposed optimization framework under a number of simplifying assumptions, e.g., linear task-load decrements, etc. However, a number of extensions are possible, although they are often at the expense of added computational complexity.

Task Load Decrements - Linear decrements can be replaced with virtually any other function that

defines task load decrements as a function of rest time and system variables; however, alternative models of this type will add additional problem non-linearity.

Task Load Increments - Similar to decrements, the presented framework allows for alternative workload increment models; If the chosen model provides increments solely as a function of time, i.e., the increments do not depend on other variables, the model can be used within the presented framework without modification (by defining $\Delta W_j = f(\tau_j)$, where f is a function relating processing time and task load increments). Otherwise, the model will likely introduce additional non-linearity.

Multi-tasking - If an extension is used that allows operators to work on multiple tasks at once, the presented framework will allow for the use of any task-load model that satisfies an additive property, e.g., $\Delta W_{\text{total}} = \Delta W_{m_1} + \Delta W_{m_2}$, where $\Delta W_{m_1}, \Delta W_{m_2}$ are the task load increments associated with two individual tasks. An additional constraint will be also required to ensure that decrements will only be assessed when the operator is not performing a task. We also note that, under this type of extension, the interpretation of the variables $\underline{W}_k, \overline{W}_k$ as the task load level immediately before and after task k, resp., is lost, though these variables still capture the task load characteristics necessary to evaluate the objective. Alternative models for multi-tasking that do not possess an additive property may not be viable within the proposed framework or introduce significant non-linearity.

Although these types of extensions may be feasible in some cases, and may be useful for capturing qualitative properties within a supervisory control framework, this class of utilization-based, scalar task-load modeling fails to capture many of the nuances associated with operator modeling that may be required for many applications. Indeed, our utilization-based model focuses mainly on capturing trends related to the mental, physical, and/or temporal demand associated with a given task; however, there are other factors that can play a role such as level of frustration, overall performance, among others [57]. The incorporation and validation of higher fidelity workload, and other human-factors inspired metrics, is an important topic of future work.

IX. Limitations

It should be noted that the purpose of this work is to illustrate a straightforward, intuitive, practical, and flexible method for placing a complex routing and scheduling problem (generally

involving multiple components, heterogenous tasks and agents, uncertain parameters, and complex task dependencies) into a single optimization framework that incorporates system-wide and human-factors inspired objectives. In establishing this framework, we have included discussion to provide meaningful insight into the problem structure and complexity, as well as potential solution methods, that will both allow others to tailor this method to particular problems of interest and give guidance into future research. However, it is important to highlight a few limitations of the analysis herein: First, the presented framework is *not* intended to be a "catch-all" solution that can be applied to *any* human-supervisory task; rather, our framework should be modified to capture constraints and modeling nuances associated with a given application. The impact of this limitation, is that poor or unpredictable performance may occur if the methods herein are blindly applied without prior verification that the assumptions are appropriate for the application at hand. Examples of other approaches to coordinating human and robotic behavior while enforcing different constraints and modeling assumptions include [35, 36]. Future work should thoroughly classify the types of tasks that are well suited for MIP-based approaches to optimization of robotic and operator behavior.

Second, it is *not* our purpose to provide a low-level, high-fidelity model of human cognition, or to provide formal verification of human models; rather, we operate on simple, flexible, human-factors inspired metrics and constraints that capture qualitative trends and illustrate a particular modeling philosophy. As such, the methods herein may not provide sufficient fidelity in situations where slight errors in operator modeling can have severe effects on mission-critical tasks or effective operation of decision support systems. Examples of higher-fidelity cognitive architectures for modeling operator behavior include ACT-R [58], Soar [59], EPIC [60], and CLARION [61]. Future work should focus on the integration of these types of cognitive architectures within optimization-based planning schemes.

Finally, it is *not* our purpose to provide exhaustive statistical or sensitivity analysis for *all* problem parameters and instances; rather, we analyze select illustrative examples and provide simulation results to help the reader build intuition and insight for future studies. As a result, there may exist some problem instances in which the general trends that are illustrated herein may not hold. As noted in Section I, there are very few research studies that attempt to jointly schedule vehicle and operator behavior within a MIP framework, namely [35, 36]. While these results, provide

some statistical insight, a thorough sensitivity analysis of integer programming methods for joint optimization to problem parameters should be a topic of future work.

X. Conclusion

A joint optimization framework has been developed for simultaneously constructing UAV routes and operator schedules for use within a supervisory surveillance mission involving a set of static targets. The complete planning operation can be formulated as a MINLP, which can be equivalently represented as a MILP of much larger size. However, in the case of a single-vehicle, an equivalent MILP of identical size exists. For scalability, a dynamic heuristic has been developed for constructing solutions dynamically. This framework can provide robustness to uncertainty in processing times using a straightforward, scenario-based extension. Numerical examples have illustrated the potential performance advantages of this joint optimization approach over alternatives that do not explicitly consider the synchronization of human and autonomous resources.

Future work should include a thorough study of the scenario-based scheme in order to generate performance bounds and to quantify its sensitivity to various problem parameters. Such a study should include more thorough statistical analysis of the properties of the scenario-based extension. In addition, the study and development of alternative heuristics for solving the full, non-linear MIP could also be potentially very useful. Adaptations to the optimization framework in order to handle higher-fidelity task load evolution models could improve performance as well. Finally, validation of the proposed framework on a carefully designed human-test subjects is crucial to further development of these systems.

Acknowledgments

This work has been sponsored by the U.S. Army Research Office and the Regents of the University of California, through Contract Number W911NF-09-D-0001 for the Institute for Collaborative Biotechnologies, and that the content of the information does not necessarily reflect the position or the policy of the Government or the Regents of the University of California, and no official endorsement should be inferred.

References

[1] Peters, J., Srivastava, V., Taylor, G., Surana, A., Eckstein, M. P., and Bullo, F., "Mixed Human-Robot Team Surveillance: Integrating Cognitive Modeling with Engineering Design," *IEEE Control Systems*, Vol. 35, No. 6, 2015, pp. 57–80.

[2] Goodrich, M. A. and Cummings, M. L., "Human Factors Perspective on Next Generation Unmanned Aerial Systems," *Handbook of Unmanned Aerial Vehicles*, 2015, pp. 2405–2423.

[3] Jentsch, F., *Human-Robot Interactions in Future Military Operations*, CRC Press, 2016.

[4] Guizzo, E., "Obama Commanding Robot Revolution Announces Major Robotics Initiative," IEEE Spectrum, June 2011.

[5] US Office of the Secretary of Defense, "Unmanned Aircraft Systems (UAS) Roadmap, 2005-2030," 2005.

[6] Srivastava, V., Carli, R., Langbort, C., and Bullo, F., "Attention Allocation for Decision Making Queues," *Automatica*, Vol. 50, No. 2, 2014, pp. 378–388.

[7] US Air Force, "Report on technology horizons, a vision for Air Force Science And Technology during 2010–2030," Tech. rep., AF/ST-TR-10-01-PR, United States Air Force., 2010, Retrieved from http://www.defenseinnovationmarketplace.mil/resources/AF_TechnologyHorizons2010-2030.pdf on Feb. 8, 2016.

[8] Liu, Y. and Nejat, G., "Robotic urban search and rescue: A survey from the control perspective," *Journal of Intelligent & Robotic Systems*, Vol. 72, No. 2, 2013, pp. 147.

[9] Sheridan, T., "Human–robot interaction status and challenges," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, Vol. 58, No. 4, 2016, pp. 525–532.

[10] Maurer, K., "Visual power: The scopic regime of military drone operations," *Media, War & Conflict*, 2016, pp. 1–11.

[11] Kilgore, R. and Voshell, M., "Increasing the transparency of unmanned systems: Applications of ecological interface design," *International Conference on Virtual, Augmented and Mixed Reality*, Springer, 2014, pp. 378–389.

[12] Hou, M., Zhu, H., Zhou, M., and Arrabito, G. R., "Optimizing operator–agent interaction in intelligent adaptive interface design: A conceptual framework," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 41, No. 2, 2011, pp. 161–178.

[13] Cummings, M. L., Bertucelli, L. F., Macbeth, J., and Surana, A., "Task Versus Vehicle-Based Control Paradigms in Multiple Unmanned Vehicle Supervision by a Single Operator," *IEEE Transactions on Human-Machine Systems*, Vol. 44, No. 3, 2014, pp. 353–361.

[14] Cummings, M. and Mitchell, P., "Predicting controller capacity in supervisory control of multiple

UAVs," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, Vol. 38, No. 2, 2008, pp. 451–460.

[15] Deza, A., Peters, J., Taylor, G., Surana, A., and Eckstein, M., "Attention allocation aid for visual search," *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, ACM, Boulder, CO, USA, 2017, pp. 220–231.

[16] McIntire, L., McKinley, R., Goodyear, C., and McIntire, J., "Detection of vigilance performance using eye blinks," *Applied Ergonomics*, Vol. 45, No. 2, 2014, pp. 354–362.

[17] Klein, D. J., *Coordinated Control and Estimation for Multi-agent Systems: Theory and Practice*, Ph.D. thesis, University of Washington, 2008.

[18] Scerbo, M., "Adaptive automation," *Neuroergonomics: The Brain At Work*, edited by R. Parasuraman and M. Rizzo, Oxford University Press, 2001, pp. 239–252.

[19] Savla, K. and Frazzoli, E., "A Dynamical Queue Approach to Intelligent Task Management for Human Operators," *Proceedings of the IEEE*, Vol. 100, No. 3, 2012, pp. 672–686.

[20] Peters, J. and Bertuccelli, L., "Robust Task Scheduling for Multi-Operator Supervisory Control Missions," *Journal of Aerospace Information Systems*, 2016, pp. 393–406.

[21] Toth, P. and Vigo, D., editors, *The Vehicle Routing Problem*, Monographs on Discrete Mathematics and Applications, SIAM, 2001.

[22] Drexl, M., "Synchronization in vehicle routingâĂŤa survey of VRPs with multiple synchronization constraints," *Transportation Science*, Vol. 46, No. 3, 2012, pp. 297–316.

[23] Arulselvan, A., Commander, C., and Pardalos, P., "A hybrid genetic algorithm for the target visitation problem," *Naval Research Logistics*, 2007, pp. 1–20.

[24] Nigam, N., "The Multiple Unmanned Air Vehicle Persistent Surveillance Problem: A Review," *Machines*, Vol. 2, No. 1, 2014, pp. 13–72.

[25] Srivastava, K., Stipanovič, D. M., and Spong, M. W., "On a stochastic robotic surveillance problem," *IEEE Conf. on Decision and Control*, Shanghai, China, Dec. 2009, pp. 8567–8574.

[26] Pasqualetti, F., Franchi, A., and Bullo, F., "On Cooperative Patrolling: Optimal Trajectories, Complexity Analysis and Approximation Algorithms," *IEEE Transactions on Robotics*, Vol. 28, No. 3, 2012, pp. 592–606.

[27] Obermeyer, K. J., Oberlin, P., and Darbha, S., "Sampling-based path planning for a visual reconnaissance UAV," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 2, 2012, pp. 619–631.

[28] Isaacs, J. and Hespanha, J. P., "Dubins traveling salesman problem with neighborhoods: A graph-based approach," *Algorithms*, Vol. 6, No. 1, 2013, pp. 84–99.

[29] Peters, J., Surana, A., Taylor, G., Turpin, T., and Bullo, F., "UAV Surveillance under Visibility and Dwell-Time Constraints," *ASME Journal of Dynamic Systems, Measurement, and Control*, 2017, Submitted.

[30] Laporte, G., Mercure, H., and Nobert, Y., "Generalized travelling salesman problem through n sets of nodes: The asymmetrical case," *Discrete Applied Mathematics*, Vol. 18, No. 2, 1987, pp. 185–197.

[31] Srivastava, V., Pasqualetti, F., and Bullo, F., "Stochastic Surveillance Strategies for Spatial Quickest Detection," *International Journal of Robotics Research*, Vol. 32, No. 12, 2013, pp. 1438–1458.

[32] Shah, J. A., Saleh, J. H., and Hoffman, J. A., "Review and synthesis of considerations in architecting heterogeneous teams of humans and robots for optimal space exploration," *IEEE Transactions on Systems, Man & Cybernetics. Part C: Applications & Reviews*, Vol. 37, No. 5, 2007, pp. 779–793.

[33] Nehme, C. E., *Modeling Human Supervisory Control in Heterogeneous Unmanned Vehicle Systems*, Ph.D. thesis, Department of Aeronautics and Astronautics, MIT, Feb. 2009.

[34] Gombolay, M. C., Wilcox, R., and Shah, J. A., "Fast scheduling of multi-robot teams with temporospatial constraints," *Robotics: Science and Systems*, Berlin, Germany, June 2013.

[35] Ponda, S., Choi, H., and How, J., "Predictive planning for heterogeneous human-robot teams," *AIAA Infotech@ Aerospace*, Atlanta, GA, USA, April 2010, AIAA 2010-3349.

[36] Murray, C. and Park, W., "Incorporating human factor considerations in unmanned aerial vehicle routing," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 43, No. 4, 2013, pp. 860–874.

[37] Korsah, G. A., Stentz, A., and Dias, M. B., "A comprehensive taxonomy for multi-robot task allocation," *International Journal of Robotics Research*, Vol. 32, No. 12, 2013, pp. 1495–1512.

[38] Drexl, M., *On Some Generalized Routing Problems*, Ph.D. thesis, Faculty of Business and Economics, RWTH Aachen University, 2007.

[39] Zimmerman, M., "Task Load," *Encyclopedia of Clinical Neuropsychology*, edited by J. Kreutzer, J. DeLuca, and B. Kaplan, Springer, 2011, pp. 2469–2470.

[40] Neerincx, M., "Cognitive task load design: Model, methods and examples," *Handbook of cognitive task design*, 2003, pp. 283–305.

[41] Rouse, W., *Systems engineering models of human-machine interaction*, Vol. 6, North-Holland, 1980.

[42] Cummings, M. and Guerlain, S., "Developing operator capacity estimates for supervisory control of autonomous vehicles," *Human Factors*, Vol. 49, No. 1, 2007, pp. 1–15.

[43] Stimpson, A., Ryan, J., and Cummings, M., "Assessing Pilot Workload in Single-Pilot Operations with Advanced Autonomy," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*,

Vol. 60, SAGE Publications Sage CA: Los Angeles, CA, 2016, pp. 675–679.

[44] Teigen, K. H., "Yerkes-Dodson: A law for all seasons," *Theory & Psychology*, Vol. 4, No. 4, 1994, pp. 525–547.

[45] Cummings, M. and Mitchell, P., "Operator scheduling strategies in supervisory control of multiple UAVs," *Aerospace Science and Technology*, Vol. 11, No. 4, 2007, pp. 339–348.

[46] Army, U., "Attack Reconnaissance Helicopter Operations," Tech. Rep. FM 3-04.126, Department of the Army (US), Feb. 2007, Retrieved from http://usacac.army.mil/sites/default/files/misc/doctrine/CDG/fms.html on Dec. 5, 2016.

[47] Makhorin, A., "GNU Linear Programming Kit," `https://www.gnu.org/software/glpk/`, 2000–2012, Accessed 2017-02-22.

[48] ILOG, Mountain View, CA, *ILOG CPLEX User's guide*, 1999.

[49] MathWorks, "Linear Programming and Mixed-Integer Linear Programming," `http://www.mathworks.com/help/optim/linear-programming-and-mixed-integer-linear-programming.html`, Accessed: 2015-08-29.

[50] Ben-Tal, A., El-Ghaoui, L., and Nemirovski, A., *Robust Optimization*, Princeton University Press, 2009.

[51] Calafiore, G. C. and Campi, M. C., "The scenario approach to robust control design," *IEEE Transactions on Automatic Control*, Vol. 51, No. 5, 2006, pp. 742–753.

[52] Kouvelis, P., Daniels, R., and Vairaktarakis, G., "Robust scheduling of a two-machine flow shop with uncertain processing times," *IIE Transactions*, Vol. 32, No. 5, 2000, pp. 421–432.

[53] Southern, D. N., *Human-Guided Management of Collaborating Unmanned Vehicles in Degraded Communication Environments*, Master's thesis, Electrical Engineering and Computer Science, Massachusetts Institute of Technology, May 2010.

[54] Bertuccelli, L. F., Pellegrino, N., and Cummings, M. L., "Choice Modeling of Relook Tasks for UAV Search Missions," *American Control Conference*, Baltimore, MD, USA, June 2010, pp. 2410–2415.

[55] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, Vol. 6, No. 2, 2002, pp. 182–197.

[56] Coello, C., "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art," *Computer methods in applied mechanics and engineering*, Vol. 191, No. 11, 2002, pp. 1245–1287.

[57] Hart, S. G. and Staveland, L. E., "Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research," *Advances in Psychology*, Vol. 52, 1988, pp. 139–183.

[58] Anderson, J. R., "ACT: A simple theory of complex cognition," *American Psychologist*, Vol. 51, No. 4,

1996, pp. 355.

[59] Laird, J. E., *The Soar Cognitive Architecture*, MIT Press, 2012.

[60] Kieras, D. E. and Meyer, D. E., "An overview of the EPIC architecture for cognition and performance with application to human-computer interaction," *Human-Computer Interaction*, Vol. 12, No. 4, 1997, pp. 391–438.

[61] Sun, R., "The CLARION cognitive architecture: Extending cognitive modeling to social simulation," *Cognition and Multi-Agent Interaction*, edited by R. Sun, Cambridge University Press, 2006, pp. 79–99.