Security-Aware Synthesis of Human-UAV Protocols*

Mahmoud Elfar, Haibei Zhu, M. L. Cummings, and Miroslav Pajic

Abstract—In this work, we synthesize collaboration protocols for human-unmanned aerial vehicle (H-UAV) command and control systems, where the human operator aids in securing the UAV by intermittently performing geolocation tasks to confirm its reported location. We first present a stochastic game-based model for the system that accounts for both the operator and an adversary capable of launching stealthy false-data injection attacks, causing the UAV to deviate from its path. We also describe a synthesis challenge due to the UAV’s hidden-information constraint. Next, we perform human experiments using a developed RESCHU-SA testbed to recognize the geolocation strategies that operators adopt. Furthermore, we deploy machine learning techniques on the collected experimental data to predict the correctness of a geolocation task at a given location based on its geographical features. By representing the model as a delayed-action game and formalizing the system objectives, we utilize off-the-shelf model checkers to synthesize protocols for the human-UAV coalition that satisfy these objectives. Finally, we demonstrate the usefulness of the H-UAV protocol synthesis through a case study where the protocols are experimentally analyzed and further evaluated by human operators.

I. INTRODUCTION

Contrary to what the terminology may suggest, autonomous systems mostly involve human presence: from actively engaging with the system to merely monitoring the system status or intervening whenever necessary [1]. A typical example is the human-unmanned aerial vehicle (H-UAV) command and control system, where various deployed applications depend on having human operators responsible for supervising a fleet of UAVs during a mission. The operator performs various supervisory tasks including, for example, updating mission goals, monitoring agent status, and adjusting flight plans [2]. The operator can also be assigned primary tasks that are mission relevant, such as imagery tasks.

With the human presence, considering human factors becomes an essential part of the modeling and design of those systems, for which several attempts have been introduced in literature. The reliance on experimental data has been proposed to model human-autonomy interactions in various applications such as autonomous cars [3], [4], industrial [5], [6] and social robotics [7]. Due to their ability to model reactive systems, Markovian formalisms, e.g., Markov Decisions Processes (MDPs) and stochastic games, were exploited for the theoretical exploration of human-robot interactions [8].

On the other hand, UAV navigational systems have been recently proven to be vulnerable to cyber and physical attacks, such as false-data injection attacks that target GPS receivers [9], [10], raising security concerns in this domain. A number of studies have focused on attack detection via sensor redundancy (e.g., [11–17]). Yet a class of these attacks can remain stealthy by introducing non-aggressive and incremental deviations [17–20]. Although humans are likely to surpass autonomy in such situations of high uncertainty [21], no research has addressed the human role in ensuring the security of H-UAV systems; such as, whether we can improve the overall security guarantees by harnessing the human power of inductive reasoning and the ability to provide context and additional information to the system in real-time.

Hence, in this paper, we focus on synthesis of protocols for H-UAV systems where the operator can intermittently perform geolocation tasks to aid in detection of possible attacks. First, the system dynamics, operator geolocation task, and the adversarial behavior are modeled using stochastic games. By developing RESCHU-SA testbed, experiments were conducted to understand operator strategies during geolocation tasks. Next, we use machine learning to predict correctness of a geolocation task at a given location. Note that in security problems, the system (i.e., UAV) is not aware of the information related to attacker’s actions, which presents a significant synthesis challenge. Thus, we construct the model as a delayed-action game, which allows for the use of off-the-shelf tools (PRISM-games) to synthesize security-aware H-UAV protocols; such protocols provide UAV path plans that increase chances of attack detection. Moreover, the protocols specify time instances at which the operator is advised to perform a geolocation task, maximizing its correctness. The formal synthesis of the advisory system guarantees a limit to the workload level, in order to avoid performance deterioration without compromising system security. Finally, we present a case study where the synthesized protocols are analyzed and subjectively evaluated by human operators.

The rest of this paper is organized as follows. Section II provides a background on stochastic games and H-UAV control systems before formulating the problem statement. Section III presents the system modeling using stochastic and delayed-action games, while Section IV describes the experiments used to obtain model parameters. The protocol synthesis framework is provided in Section V. Section VI presents a case study where the synthesized protocols are analyzed and evaluated. Finally, Section VII concludes the paper and provides avenues for future work.

II. BACKGROUND AND PROBLEM STATEMENT

We start with the related background on stochastic games and strategy synthesis, followed by the problem statement.

A. Stochastic Games

Stochastic multiplayer games (SMGs) can model reactive systems with both stochastic and nondeterministic transitions, where the latter are resolved by more than one player. Stochasticity arises when the system evolution cannot be precisely predicted, yet a probabilistic profile can be assumed.
Conversely, nondeterminism abstracts players’ decisions, either to incorporate a family of behaviors or to reason about a winning strategy. A turn-based game is played such that, at each state, only one player at most can make decisions.

Turn-based SMGs have proven to be useful for modeling reactive systems [22].

**Definition 1** (Turn-Based SMGs). A turn-based SMG over players \( \Gamma = \{I, II, O\} \) is a tuple \( \mathcal{G} = (S, (S_I, S_{II}, S_O), A, \varsigma, \delta) \), where \( S \) is a finite set of states, partitioned into players’ and stochastic states \( S_I, S_{II} \) and \( S_O \); \( A \) is a finite set of actions; \( \varsigma \in \text{Dist}(S_{II}) \) is an initial distribution over \( S_{II} \); \( \delta : S \times S \rightarrow [0,1] \) is a transition function s.t. \( \delta(s, s') \in \{0, 1\} \), \( \forall s \in S_I \cup S_{II} \) and \( s' \in S \), and \( \delta(s, s') \in [0,1] \); \( \forall s \in S_O \) and \( s' \in S_I \cup S_{II} \), where \( \sum_{s' \in S} \delta(s, s') = 1 \) holds.

In contrast to SMGs where the game state is visible to all players, in [23] we introduced delayed-action games (DAGs) that partially obscure the game state from one player by substituting the hidden information (i.e., the truth) regarding some states with the player’s belief for these hidden states.

**Definition 2** (Delayed Action Game). A delayed-action game (DAG) is a tuple \( \mathcal{G} = (\mathcal{G}, V, \Gamma) \) where \( \mathcal{G} \) is a turn-based SMG over a set of game variables \( V = V_I \cup V_{II} \) and players \( \Gamma = \{I, II, O\} \), such that \( S \subseteq \text{Eval}(V) \times \mathcal{P}(\text{Eval}(V_I)) \times \Gamma \). In addition, \( V_I \) is the set of variables holding the true values of the game, known to player \( I \); and \( V_{II} \) is the set of variables holding player \( II \) beliefs about the values of the game.

In Definition 2, for any variable \( \text{var} \) from a set \( V \), \( \text{Eval}(\text{var}) \) denotes the set of evaluations that assign values to \( \text{var} \). Also, for a set \( A \), \( \mathcal{P}(A) \) refers to the power set \( 2^A \).

Intuitively, a player’s strategy is how she resolves nondeterminism throughout the game, while a protocol is a set of strategies adopted by a coalition of players. The synthesis problem seeks a winning strategy (protocol); that is, it resolves choices for a player (coalition) such that some objectives are satisfied. Synthesis objectives can be specified using temporal logic such as ATL and rPATL [24].

**B. Human-UAV Supervisory Systems**

This work is motivated by H-UAV supervisory systems, where the UAV is supervised by a human operator [1]. A typical mission is for the UAVs to reach a number of targets to perform imagery tasks. While an autonomous planner automatically assigns UAVs to target locations, the operator can override these assignments or the path plan if necessary. Once a target is reached, the operator is notified to assist with the imagery task by analyzing the live camera feed [2].

UAVs are prone to adversarial attacks, such as GPS spoofing, that can drive the UAV away from its planned path [9], [10], [25]. Several techniques have been proposed to aid with detecting such attack by relying on redundant sensors (e.g., [11–17]). Nevertheless, when a sufficient number of sensors is compromised, a smart attacker can remain stealthy by injecting non-aggressive and incremental deviations in sensor measurements, which will still force the UAV into any undesired state through the actions of controller [18–20].

On the other hand, a geolocation task ultimately aims at localizing the UAV through side channel information. For example, we showed that landmarks and other geo-features from a UAV’s camera feed can be used to estimate the UAV’s location in real time by a human-operator [26]. Thus, in this work, we address the problem of synthesizing protocols for the Human-UAV supervisory system that also employs the human operator for attack detection. The synthesized protocols shall provide the UAV path plan, as well as the time instances at which the operator shall be advised to perform a geolocation task, while ensuring that a given set of performance objectives (e.g., time, safety, workload) is satisfied.

**III. System Modeling**

In this section, we describe a system model, before showing how it can form a DAG used for the protocol synthesis.

**A. SMG-Based Model**

Due to the stealthy nature of cyber attacks considered in this study, one should differentiate between the UAV belief about its location, which may not be accurate, and the ground truth, which is assumed to be known to the adversary. Fig. 1(a) shows a standard UAV model \( \mathcal{M}_{\text{uav}} \) (e.g., as in [27]) where the UAV movement is discretized into action set \( A_{\text{uav}} = \{N, S, E, W, NE, NW, SE, SW\} \). Note that the UAV’s actions affect its belief \( x_B \). Conversely, \( A_{\text{adv}} \) shows how the adversary’s stealthy actions are constrained by the UAV movements, influencing the ground truth \( x_T \) to avoid detection, only slowly increasing deviations from the planned trajectory can be achieved by the attacker; e.g., through GPS spoofing and actions of the low-level controller, small errors are added to the desired commands in each step. If, for example, the UAV is heading \( N \), then the adversary available actions are \( A_{\text{adv}}(N) = \{NE, NW, SW\} \). Note that more aggressive attacks can be detected as in e.g., [11], [12].

As in Fig. 1(b), the human geolocation model \( \mathcal{M}_{\text{hgl}} \) can initiate a geolocation task via the action \( \text{start} \). The outcome of the task, however, can be successful or not with probabilities \( p(x_T, x_B) \) and \( 1 - p(x_T, x_B) \), respectively, depending on both \( x_B \) and \( x_T \). Detailed probability modeling is provided in Section IV. While the presented model is a standard SMG, since \( x_T \) is unknown to the UAV — and thus so is \( p(x_T, x_B) \) — the model cannot be used to reason about strategies. Moreover, if SMG semantics are used (i.e., the truth is implicitly known to the UAV), a synthesized strategy becomes a function of the adversarial specific actions — which are unknown — thus, rendering the strategy useless.

---

1 Other factors affecting \( p \), such as operator skills and the current workload, are not considered in this study. Also, the model assumes that task repetition has no impact on \( p \). See the discussion in Section VII.
**Fig. 2.** DAG-based model with the UAV (\(\mathcal{M}_{\text{uav}}\)); adversary (\(\mathcal{M}_{\text{adv}}\)); human geolocation (\(\mathcal{M}_{\text{hgl}}\)); and DAG memory read (\(\mathcal{M}_{\text{mrd}}\)); write (\(\mathcal{M}_{\text{mwr}}\)).

### B. DAG-Based Model

To overcome these challenges, a DAG variation of the SMG model can be used (Fig. 2). The basic idea is that, during a mission, the UAV is unaware of differences between its belief and the ground truth until a geolocation task is correctly done. Thus, although we model missions as the players taking turns (SMG model), the same behavior can be captured if the UAV makes its moves, updating only the belief state, until a geolocation occurs; followed by the adversary's corresponding actions, updating the ground truth. With such a DAG model, the UAV takes a number of actions ahead of the adversary; that is, the adversary's actions are hidden from the UAV as they have not occurred yet in the model. Since the UAV decisions are made without knowing the adversary’s specific actions, a UAV synthesized strategy becomes independent of those actions. Hence, the DAG model provides a different representation of the system without altering its behavior, as we show in [23], and thus can be used to synthesize strategies for the original SMG model. This mechanism is realized by the auxiliary components \(\mathcal{M}_{\text{mwr}}\) and \(\mathcal{M}_{\text{mrd}}\) that model write and read operations, respectively, on a FIFO memory stack \((m_i)^{i=0}_{i=\infty} \in A_{\text{uav}}\) as shown in Fig. 2(c). By synchronizing with \(\text{write} \), \(\mathcal{M}_{\text{mwr}}\) saves the last UAV action in memory location \(m_i\), executing \(s_i \xrightarrow{\text{write}} s_{i+1}, i \in \{0, ..., n-1\}\). Similarly, \(\mathcal{M}_{\text{mrd}}\) synchronizes with the action \(\text{read}\) to read the saved UAV action from memory location \(m_j\), executing \(s_j \xrightarrow{\text{read}} s_{j+1}, j \in \{0, ..., n-1\}\).

The DAG is formed by composing both the model and auxiliary components using the procedure summarized in Algorithm 1. The composed DAG allows the UAV (\(\mathcal{M}_{\text{uav}}\)) to play a finite number of actions before attempting a geolocation task, updating its belief \(x_B\). Based on these actions, the adversary (\(\mathcal{M}_{\text{adv}}\)) plays her delayed actions, updating the ground truth \(x_T\). Hence, the probability of performing a correct geolocation task is given by some function \(f(x_T, x_B)\) — the realization of the function \(f(x_T, x_B)\) is discussed in Section IV. If the geolocation task is unsuccessful, the adversarial actions are discarded, and the UAV can continue to move or attempt another task. Otherwise, the UAV belief is updated to match the ground truth, at which point the game is repeated with a new starting location. In the rest of the paper, we will refer to each of these repetitions as a subgame \(\mathcal{G}_i\), where \(i\) is the subgame index, and to the set of all subgames of interest as the supergame \(\mathcal{G}\). Intuitively, each subgame explores the possible adversarial effect for each UAV sequence of actions, while the supergame examines the collective behavior of those subgames.

### IV. Human Geolocation Model

As the presented DAG-based model is characterized by the probabilities \(f(x_T, x_B)\), in this section we present the experimental platform and evaluations used to obtain this function.

#### A. Experimental Platform and Design

The aim of this experiment was twofold. First, we wanted to validate the hypothesis that UAV operators can successfully perform geolocation tasks. Second, we wanted to understand what strategies the operators adopt to perform the geolocation tasks and their relevant factors.

To test our hypothesis, we developed Security-Aware RESCHU extension (RESCHU-SA), a virtual platform for studying the security aspects of human-UAV supervisory systems prone to cyber attacks. Fig. 3 shows RESCHU-SA operator interface (OI) with the map area displaying flight plans, target locations, and threat zones. While the map shows a UAV’s reported location, which may be compromised, the camera feed displays the live video stream from the selected UAV, from the UAV’s true location. Through the OI, the operator can supervise a fleet of UAVs on a timed mission where a number of target locations should be visited by the UAVs, and a visual task should be completed by the operator through the UAV camera feed once each location is reached.

Each experiment consisted of two missions with high and low workloads. During each mission, a number of geolocation requests are randomly introduced, where the operator can respond by activating the camera feed to perform a geolocation task and further report whether the reported location is correct. After the two missions are done, the operator is interviewed and a retrospective verbal protocol analysis is performed to elicit more insights on their behavior and decisions made during the experiment.

---

**Algorithm 1: DAG construction procedure**

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Initial SMG model:</strong> (\mathcal{G}<em>0, \mathcal{M}</em>{\text{adv}}, \mathcal{M}<em>{\text{uav}}, \mathcal{M}</em>{\text{hgl}}, \mathcal{M}<em>{\text{mwr}}, \mathcal{M}</em>{\text{mrd}})</td>
</tr>
<tr>
<td>2</td>
<td><strong>Result:</strong> (\mathcal{G}) DAG construct</td>
</tr>
<tr>
<td>3</td>
<td><strong>while end criterion not met do</strong></td>
</tr>
<tr>
<td>4</td>
<td><strong>while</strong> (\mathcal{M}_{\text{uav}} \neq \text{geolocation task} ) do</td>
</tr>
<tr>
<td>5</td>
<td>(\mathcal{M}<em>{\text{uav}}, x_B \leftarrow \text{effect}(\mathcal{M}</em>{\text{uav}}, x_B)) UAV moves, updating belief</td>
</tr>
<tr>
<td>6</td>
<td>(\mathcal{M}<em>{\text{mwr}}, \text{write}(\mathcal{M}</em>{\text{uav}}, +w)) write action to memory</td>
</tr>
<tr>
<td>7</td>
<td><strong>while</strong> (w \leq \text{rd} ) do</td>
</tr>
<tr>
<td>8</td>
<td>(\mathcal{M}<em>{\text{mrd}}, \text{read}(\mathcal{M}</em>{\text{uav}}, +\text{rd})) read UAV action from memory</td>
</tr>
<tr>
<td>9</td>
<td>(\mathcal{M}<em>{\text{madv}}, x_T \leftarrow \text{effect}(\beta(\mathcal{M}</em>{\text{uav}}), x_T))</td>
</tr>
<tr>
<td>10</td>
<td>if (\text{draw} x \sim \text{Bern}(f(x_T, x_B))) then</td>
</tr>
<tr>
<td>11</td>
<td><strong>Stochastic turn</strong></td>
</tr>
<tr>
<td>12</td>
<td><strong>else</strong> (\text{reset(rd)}) reset memory</td>
</tr>
<tr>
<td>13</td>
<td><strong>else</strong> (\text{reset(rd)}) forget (hide) ADV actions</td>
</tr>
</tbody>
</table>

---

2Research Environment for Supervisory Control of Heterogeneous Unmanned Vehicles (RESCHU) was originally developed to evaluate Human-UAV supervisory systems [28], [29]. The initial platform does not support attack modeling or real-time video feed from any location from a map, based on UAVs’ trajectories that the user can change during experiments.

2See the attached video, also available at: https://cpsl.pratt.duke.edu/research/security-aware-human-loop-cps.
C. Human Geolocation Task (GT) Predictor

The abstract goal of a geolocation task is to determine whether the UAV’s reported belief $x_B = x_{\text{map}}$ (see Fig. 1(b)) matches the ground truth $x_T = x_{\text{cam}}$ observed via the camera feed.\footnote{In this work, as we focus on attack-detection, we only address the determination of whether two locations are the same, rather than how the actual coordinates can be found.} The operator’s ability to correctly perform the task is affected by the GIS features of both $x_{\text{cam}}$ and $x_{\text{map}}$. For instance, if $x_{\text{cam}} = x_{\text{map}}$ (i.e., they refer to the same location), the presence of distinctive features is likely to increase the chances of a correct decision. Conversely, if $x_{\text{cam}} \neq x_{\text{map}}$, then the probability of a correct decision relies heavily on the relative distinctiveness between both locations. Therefore, we define a GT predictor as follows.

**Definition 3.** GT predictor is a tuple $\mathcal{K} = (\Sigma, \mathcal{H}, f)$ where $\Sigma$ is the set of GIS layers, $\mathcal{H} = \{r_i \mid r_i : X^2 \to \mathbb{R}, i = 1, \ldots, n\}$ is a set of $n$ numerical measures of similarity between two locations in $X$, and $f : \mathcal{P}(\Sigma)^2 \times \mathbb{R}^n \to [0, 1]$ is a prediction function of the GT correctness.

Given two sets of labels and a tuple of numeric measures of similarity between two given locations, the function $f$ predicts the correctness of a GT. In this case-study, we used $\Sigma = \bigcup_{j \in \{\text{tr, rd, lm}\}} \Sigma_j$; as a similarity measure we used the maximum normalized 2D cross-correlation coefficient $r_{rd} \in [0, 1]$ between the road patterns of two locations, $\mathcal{H} = \{r_{rd}\}$

We derive the prediction function $f$ (and hence $\mathcal{K}$) using machine learning-based techniques. Basically, a predictive model is trained on a database of image pairs, where the training inputs are the GIS labels and similarity measures of each pair; the training output is the corresponding human estimation of the similarity of each pair; and the model outcome is $f : \mathcal{P}(\Sigma)^2 \times \mathbb{R}^n \to [0, 1]$. A survey was administrated to human participants to collect their opinion of how similar two locations are. During the survey, each participant was shown a total of 100 pairs of locations. For each pair, the participant was asked to select their estimation on a scale from 1 (very similar) to 5 (very dissimilar), as shown in Fig. 6. Next, labels from $\Sigma$ were manually assigned to the selected locations. The function $f$ was realized using a bagged-trees ensemble learner (RMSE = 0.635, RS = 0.650).

Finally, for two locations $x_{\text{cam}}$ and $x_{\text{map}}$ we have that\footnote{For notational simplicity, we employ $f(x_{\text{cam}}, x_{\text{map}})$ to refer to $f(L(x_{\text{cam}}), L(x_{\text{map}}), \mathcal{H}(x_{\text{cam}}, x_{\text{map}}))$, where $L(x)$ is the suitable layer-labeling function, capturing the GIS layers at location $x$ (as in Table I).}

$$f(x_{\text{cam}}, x_{\text{map}}) = \begin{cases} \hat{f}(x_{\text{cam}}, x_{\text{map}}) & x_{\text{cam}} = x_{\text{map}} \\ 1 - \hat{f}(x_{\text{cam}}, x_{\text{map}}) & \text{otherwise}. \end{cases}$$

That is, the measure of correctness is the human tendency to consider two images similar or dissimilar if they represent the same or different locations, respectively.
V. PROTOCOL SYNTHESIS

A. Synthesis Objectives

The primary synthesis objective is to find protocols for the H-UAV coalition based on the following requirements.

(a) Reach the target location. As one subgame may not yield a feasible flight plan to directly reach the target, checkpoints can be set as intermediate targets to render the objective feasible. By assigning the label reach to the set of states with acceptable checkpoint locations, the objective can then be formalized as $Pr_{max}[\text{reach}] \geq p_{min}$ for some bound $p_{min}$, and has to hold for all encountered subgames.

(b) Avoid hazard zones. To reach the target, the UAV must avoid all known hazard zones, where the UAV is likely to endure damage. If the corresponding states are assigned the label hazard, the objective can be formalized as $Pr_{max}[\neg \text{hazard}] \geq p_{min}$ for some bound $p_{min}$. We assume here that encountering a hazard zone results in losing the asset, hence the global operator. Consequently, this objective has to hold for all encountered subgames.

These objectives are refined to elicit rPATL queries [24]. Specifically, for a subgame $\mathcal{G}_i$, the following query is used $\phi_{syn}(k) := \langle(uav) Pr_{max}[\neg \text{hazard}] \neg U \leq k (locate \land reach) \rangle$, i.e., find a strategy that maximizes the probability of not encountering a hazard until a checkpoint is reached (reach) and a geolocation task is successful (locate) within a horizon $k$.

The procedure described in Algorithm 2 is then used to solve the synthesis problem. Starting with the initial subgame, the procedure checks for each horizon whether $\phi_{syn}$ is satisfied (hence a strategy exists) if the geolocation task is performed at the last stage. These checks are terminated when reaching the maximum search horizon or failing to satisfy the objectives.6 Next, subgames are pruned to discard strategies that fail to satisfy local bounds. The remaining strategies are used to populate the set of end locations, where each end-location represents an initial location to a reachable subgame that needs to be explored. The procedure is repeated for all subgames with initial state in the end locations set to obtain a strategy $\pi_i$ for each reachable subgame $\mathcal{G}_i$. Thus, for a $q$ number of reachable subgames, the supergame is reduced to an MDP $\mathcal{G}^{(q)}_{\pi}$ (whose states are the reachable subgames); the MDP is then checked against the query to find the minimum probability of eventually reaching the target within a maximum number of geolocation tasks $n$. $\phi_{ana}(n) := \langle(adv) Pr_{min}[F^n \text{target}] \rangle$.

Algorithm 2: Protocol synthesis procedure

\begin{algorithm}
\begin{algorithmic}[1]
\State \textbf{Input:} Initial location $x_0$, synthesis query $\phi_{syn}$, max horizon $h_{max}$
\State \textbf{Output:} H-UAV protocols $\Pi = \{(\pi_{uav}, \pi_{ana})\}$
\State $X \leftarrow \{x_0\}$ initialize set of initial locations (subgames)
\State \textbf{foreach} unexplored initial location $x_i \in X$ \textbf{do}
\State $h_0 \leftarrow (UAV, x_i, e)$ set subgame initial state
\State $\text{stop} \leftarrow \bot$, $h \leftarrow 1$ reset stopping flag and horizon
\State \textbf{while} $h \leq h_{max}$ \& $\neg \text{stop}$ \textbf{do}
\State $\langle\pi_{uav}, \phi\rangle \leftarrow \text{synth}(\mathcal{G}^{h_0}_{\pi}, \phi_{syn})$ find a winning strategy
\State \textbf{if} $\pi_{uav}$ exists \textbf{then}
\State $\Pi \leftarrow \Pi \cup \{(\pi_{uav}, \pi_{ana})\}$ add to the protocol
\State $X \leftarrow X \cup \text{reach}(\pi_{uav})$ update reachability set
\State $h \leftarrow h + 1$ explore next horizon
\State \textbf{else} \text{stop} \leftarrow \top\text{ prune}(\Pi)$
\End{algorithmic}
\end{algorithm}

B. Synthesis Procedure

The protocol synthesis procedure is summarized in Algorithm 2. Starting from the initial location $x_0$ and horizon $h = 1$, the first subgame is constructed and explored. For each horizon, the synthesizer searches for a corresponding optimal strategy and set of reachable locations, until either the maximum search horizon $h_{max}$ is reached or a no feasible strategy can be found. The same process is repeated using unexplored locations in the reachability set. Note here that the set $\Pi$ can be pruned (e.g., by discarding strategies that violates auxiliary requirements) to reduce computation time.

VI. EXPERIMENTAL RESULTS

Fig. 7(a) shows the environment setup used for evaluation. The map was discretized into a $10 \times 10$ grid, where crossing the map boundaries is penalized for both $x_B$ and $x_T$. Also, for the UAV to ever arrive to the designated target, the adversary is prohibited from launching attacks for at least the first horizon. The UAV mission is to reach the target without encountering any of the hazard zones. The model shown in Fig. 2 was implemented using the PRISM-games [24] model checker on an Intel Core i7 4.0 GHz CPU with 16GB RAM.

From the synthesis procedure (Algorithm 2), the protocols to complete the mission were obtained as shown in Fig. 7(b). For the first subgame, Fig. 8(a) shows how the horizon at which the geolocation task is performed impacts the probability of correctness that can be guaranteed, regardless of the adversarial actions, while Fig. 8(b) shows both the remaining distance to target and the increase in traveled distance, relative to the shortest path. The trends show that, as time passes...
Fig. 8. Analysis results for the synthesized protocol, where $d_{\text{inc}}$ is the remaining distance to target, and $d_{\text{ana}}$ is the increase in distance to target. Times and distances are relative to those of the shortest path.

without performing a geolocation task, the probability of its correctness decreases, while the remaining distance mostly decreases as the UAV approach the target. Interestingly, the strategy where the geolocation task is scheduled at horizon $h = 6$ provides a local minimum for the increase in traveled distance while maintaining a probability of correctness close to $h = 7$ and $h = 8$. This probability drops at $h \geq 9$ as guaranteeing that the UAV never encounter a hazard zone before the scheduled geolocation task becomes infeasible.

For the supergame, Fig. 8(c) shows the impact of the geolocation task budget on the probability of a successful mission. The graph conveys that more than 4 geolocation tasks are required to guarantee a non-zero minimum probability of reaching the target against the worst-case attack. Fig. 8(d) shows the total expected mission time given a budget for the geolocation tasks, relative to the shortest-path time.

For subjective evaluation, an experiment was conducted where a number of participants were shown triplets of images ($I_a, I_b, I_c$) as shown in Fig. 9(left). In each triplet, $I_a$ depicts a target location to which a shortest-path $\pi_b$ and safest-path $\pi_c$ plans are generated, where $I_b$ and $I_c$ are the sets of reachable locations if the UAV is under attack, respectively. The two images $I_b$ and $I_c$ are randomly withdrawn such that $I_b \in I_b \setminus I_c$ and $I_c \in I_c \setminus I_b$, i.e., each image is exclusive to the corresponding reachability set. Participants were asked to indicate which image from $(I_b, I_c)$ is less likely to be confused with $I_a$ (and hence safer). Fig. 9(right) shows that, while group A selections indicated no significant difference, group B indicated that locations reachable by the safest-path plan are less likely to be mistaken with $I_a$. In retrospective interviews, group A explained that they picked the upper right image most of the time as the tight time constraints gave them no time to inspect the lower right image, which may explain why $I_b$ and $I_c$ had similar chances of being selected since they were randomly ordered. On the contrary, group B found the given time to be sufficient to inspect both images. As these results highlight the effect of individual imagery skills on the geolocation task, further investigation may be needed to confirm this observation.

The performance results obtained for this case study are listed in Table II. Note that, for the same grid size, more complex maps require more time for model checking, while the state space size remains unaffected. Although the number of states is $O\left((|A_{\text{uav}}||A_{\text{adv}}|)^h\right)$, the growth rate is typically reduced by the presence of hazard zones as the game stops branching at such states. Following the construction of the subgames, exploring the supergame itself consumes significantly less resources for model construction and analysis.

VII. DISCUSSION AND CONCLUSION

In this paper, we have presented an approach to synthesize collaboration protocols for human-UAV command and control systems. The approach utilizes delayed-action games to model the system such that the ground truth is hidden from the UAV, rendering the model useful for synthesis using off-the-shelf tools. As the geolocation task is used to confirm the UAV location, the model includes a measure of geolocation task correctness. Moreover, experimental results have shown that the operators mostly adopted three geolocation strategies. By extracting the GIS features relevant to these strategies, machine learning techniques were deployed to predict the aforementioned measure of correctness. Based on the developed model and a formal representation of system objectives, the DAG synthesis procedure uses PRISM-games model checker to synthesize and analyze the collaboration protocols. Finally, experimental analyses and subjective feedback were used to evaluate the synthesized protocols.

Though DAGs exploit parallel computation to reduce total processing time, the time to explore subgame horizons exponentially grows. Nevertheless, the DAGs as a formalism can benefit from approximate model checking techniques to explore larger horizons and higher orders of map discretization, where accuracy is traded-off with speed [31].

Another improvement to this study would be to consider other factors affecting the operator’s performance, such as perceived workload, operator skill level, map complexity, and proper quantification of task correctness. Albeit challenging, this may open the door for individualized protocol synthesis. Moreover, available datasets for aerial images [32] can be examined to study the proposed approach on a wider scale. When it comes to advisory systems, however, one must carefully examine factors affecting human trust in autonomy — a quality that, if absent, may render collaborative protocols ineffective. Therefore, reinforcing trust through explanatory communication of those protocols is an avenue for future work.
REFERENCES