Strategy Synthesis for Surveillance-Evasion Games with Learning-Enabled Visibility Optimization

Suda Bharadwaj[1], Louis Ly[2], Bo Wu[1], Richard Tsai[2], and Ufuk Topcu[1]

Abstract—This paper studies a two-player game with a quantitative surveillance requirement on an adversarial target moving in a discrete state space and a secondary objective to maximize short-term visibility of the environment. We impose the surveillance requirement as a temporal logic constraint. We then use a greedy approach to determine vantage points that optimize a notion of information gain, namely, the number of newly-seen states. By using a convolutional neural network trained on a class of environments, we can efficiently approximate the information gain at each potential vantage point. Subsequent vantage points are chosen such that moving to that location will not jeopardize the surveillance requirement, regardless of any future action chosen by the target. Our method combines guarantees of correctness from formal methods with the scalability of machine learning to provide an efficient approach for surveillance-constrained visibility optimization.

I. INTRODUCTION

Over the last decade, the use of autonomous agents, such as unmanned aerial vehicles, for patrolling and surveillance in adversarial environments has increased tremendously [1], [2]. In such settings, it is often necessary to not only perform routine patrolling of the area but also to maintain visibility of potentially hostile targets until an appropriate response can be formed. Thus, an autonomous agent has two objectives. Throughout the operation, it must maintain line-of-sight visibility of, or otherwise track, a moving target. Second, the agent may also need to maximize its visibility of the environment, prioritizing areas which it has not recently observed. Simply, the agent needs to *optimally patrol* an environment subject to the constraint of a surveillance requirement on a moving target.

There is significant amount of existing work on designing strategies for autonomous patrolling agents in known environments [3]–[7] and unknown environments [8]–[15]. However, in these cases, the agents are merely *passive observers*. When a hostile target is detected, the agents simply inform a human operator (for example via a live camera feed) who will take over from then on. However, in practice, targets could be lost after detection and thus need to be actively tracked by the autonomous agent. In this paper, we treat the patrol problem as finding and visiting a set of discrete locations that provides visibility of the whole map. Such an optimization problem can be seen as an instance of the *art gallery problem*, which has been shown to be NP-hard [16].

The surveillance of adversarial targets is naturally formulated as a two-player game. There have been several variants of such games studied in [17], [18], including the case in which the environment is not known apriori [19], [20]. These settings, however, only handle a simple surveillance requirement, namely to never lose sight of the target. For example, in [21] the authors formulate the problem as a two-player game in which both players choose their controls at initial time and proceed until the target is able to hide. If this requirement is relaxed, the agent may not always observe, or even know, the exact location of the target. In this case, surveillance is, by its very nature, captured by a partial-information two-player game.

While there has been a lot of work on both surveillance and patrolling individually, there has been very little on the combination of the two problems. To bridge this gap, this paper proposes a method for maximizing the visibility of the environment for the patrol objective while *actively maintaining knowledge* of the location of a hostile target for the surveillance objective. We provide a *quantitative guarantee* on surveillance performance.

Over an infinite time horizon, the agent must be able to prevent its uncertainty of the target's location from exceeding a user-defined threshold. The approach taken in this paper is closely related to the framework established in [22] and [23] where a surveillance game is formulated as a GR(1) synthesis problem. However, no patrolling objective is considered in [22] nor [23]. Another salient difference is that we do not want to synthesize a single strategy, but instead generate a *winning region*. Within the winning region, the agent is guaranteed to not violate the surveillance specification. We use the winning region as a constraint for the patrol task to guarantee correctness of the surveillance requirement.

For maximizing visibility in large-scale environments, we leverage the efficiency of data-driven techniques to determine approximately-optimal locations for the agent to move towards. We adopt the approach from [24], where a neural network is used to approximate a gain function that quantifies map visibility. However, the approach in [24] does not consider any surveillance requirements and focuses purely on visibility optimization. We constrain the generation of these locations at runtime to guarantee the correctness with respect to a surveillance specification. The concept of guaranteeing correctness on systems with different goals is studied in the field of runtime verification [25], [26] where a winning set is computed offline and actions that take the system out of the set are overwritten at runtime. We employ this in principle as new locations are computed greedily at runtime, such that those new locations must stay in the winning region.

Our contribution is summarized as follows:

The University of Texas at Austin
[1] {sudab,bwu3,utopcu}@utexas.edu
[2] {louis,ytsai}@oden.utexas.edu

1) We propose a *scalable* algorithm to approximately-optimally explore an environment while guaranteeing a quantitative surveillance specification.
2) We employ machine learning to generate locations that maximize visibility in large-scale discrete environments. Formal verification techniques are often not able to handle similar state space sizes [27]. We develop an abstraction method to reduce the state space of a discrete environment. We show that the abstraction is sound which guarantees that if a strategy is correct on an abstract environment, it is also correct on the underlying concrete environment. We are thus able to leverage the scalability of machine learning with the guarantees of formal methods.
3) We present a simulation on a case study and benchmark on various sizes of state spaces to demonstrate the scalability of the proposed approach.

The paper is structured as follows. Section II presents the problem informally in the context of a case study. Section III provides notations and definitions for the surveillance game structures. The patrol problem constrained with a surveillance specification is formally constructed in Section IV and the solution approach is given in Section V. We present a method to generate a sound abstraction of a large discretized state space in Section VI and provide numerical evaluations in Section VII. Finally, we conclude in Section VIII and discuss avenues of future research.

II. GAME STRUCTURE

A. Environment

We represent the environment, an example of which is shown in Figure 1a, as a set of states $\mathcal{L}$. We define $L \subseteq \mathcal{L}$ as the set of states representing free space and $L^c = \mathcal{L} \setminus L$ representing the obstacles. For example, in Figure 1b, L corresponds to all the white cells and L^c corresponds to the obstacles in red.

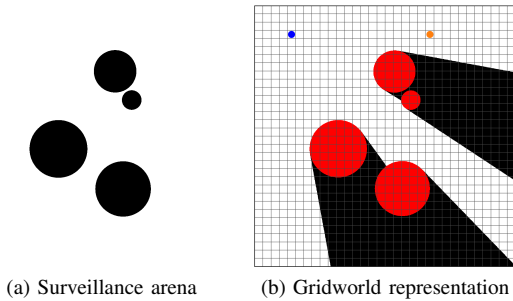(a) Surveillance arena   (b) Gridworld representation

Fig. 1: Sample environment and corresponding gridworld representation. The blue circle corresponds to the controlled agent and the orange circle corresponds to the hostile target. Red cells are obstacles that obscure vision. Black cells in (b) correspond to occlusions.

B. Surveillance games

We define a *surveillance game* to be a tuple $G = (\mathcal{G}, vis)$ with $\mathcal{G} = (S, s_0, \delta)$, where:

- $S = L_a \times L_t$ is the set of states, with L_a the set of locations of the agent, and L_t the locations of the target;
- $s^{\text{init}} = (l_a^{\text{init}}, l_t^{\text{init}})$ is the initial state;
- $\delta \subseteq S \times S$ is the transition relation describing the possible moves of the agent and the target; and
- $vis : S \to \mathbb{B}$ is a function that maps a state (l_a, l_t) to *true* iff *position l_t is in sight of l_a*.

The relation δ captures the next allowed move of both the target and the agent, where the target moves first and the agent moves second. For a state (l_a, l_t) we define $succ_t(l_a, l_t)$ as the set of possible successor states of the target. We extend $succ_t$ to sets of states of the target by stipulating that the set $succ_t(l_a, L)$ consists of all possible successor states of the target for states in $\{l_a\} \times L$. Formally, let $succ_t(l_a, L) = \bigcup_{l_t \in L} succ_t(l_a, l_t)$.

For a state (l_a, l_t) and a successor location of the target l'_t, we denote with $succ_a(l_a, l_t, l'_t)$ the set of successor locations of the agent, given that the target moves to l'_t:
$$succ_a(l_a, l_t, l'_t) = \{l'_a \in L_a \mid \delta(l_a, l_t) = (l'_a, l'_t)\}.$$

C. Belief-set game structures

For surveillance games, when the agent cannot see the target, we need to reason about the *belief* the agent has in the location of the target. To this end, we employ a powerset construction which is commonly used to transform a partial-information game into a perfect-information one, by explicitly tracking the knowledge the agent has as a set of possible locations of the target.

Given a set B, we denote with $\mathcal{P}(B) = \{B' \mid B' \subseteq B\}$ the powerset (set of all subsets) of B.

For a surveillance game $G = (\mathcal{G}, vis)$ we define the corresponding *belief-set game structure* $G_{\text{belief}} = (\mathcal{G}_{\text{belief}}, vis)$ where $\mathcal{G}_{\text{belief}} = (S_{\text{belief}}, s_{\text{belief}}^{\text{init}}, \delta_{\text{belief}})$ where:

- $S_{\text{belief}} = L_a \times \mathcal{P}(L_t)$ is the set of states, with L_a the set of locations of the agent, and $\mathcal{P}(L_t)$ the set of *belief sets* describing information about the location of the target;
- $s_{\text{belief}}^{\text{init}} = (l_{a_{init}}, B_{t_{init}})$ is the initial state;
- $\delta_{\text{belief}} : S_{\text{belief}} \times S_{\text{belief}}$ is the transition relation where $\delta_{\text{belief}}(l_a, B_t) = (l'_a, B'_t)$ iff $l'_a \in succ_a(l_a, l_t, l'_t)$ for some $l_t \in B_t$ and $l'_t \in B'_t$ and one of these holds:
  (1) $B'_t = \{l'_t\}$, $l'_t \in succ_t(l_a, B_t)$, $vis(l_a, l'_t) = true$;
  (2) $B'_t = \{l'_t \in succ_t(l_a, B_t) \mid vis(l_a, l'_t) = false\}$.

A *run* ρ in the game G_{belief} is an infinite sequence $\rho = (l_{a_0}, B_{t_0}), (l_{a_1}, B_{t_1})\ldots$ of states in S_{belief}, where $(l_{a_0}, B_{t_0}) = (l_{a_{init}}, B_{t_{init}}) = s_{\text{belief}}^{\text{init}}$, $\delta_{\text{belief}}((l_{a_i}, B_{t_i}), \sigma_i) = (l_{a_{i+1}}, B_{t_{i+1}})$ for all i. A *strategy for the agent in G_{belief}* is a function $f_a : S_{\text{belief}} \times \mathcal{P}(L_t) \to S_{\text{belief}}$ that maps the history of the play and the current belief of the target location to the agent's next action.

D. Temporal surveillance objectives

We consider a set of *surveillance predicates* $\mathcal{SP} = \{p_k \mid k \in \mathbb{N}_{>0}\}$, where for $k \in \mathbb{N}_{>0}$ we say that a state (l_a, B_t) in the belief game structure satisfies p_k (denoted $(l_a, B_t) \models p_k$) iff $|\{l_t \in B_t \mid vis(l_a, l_t) = false\}| \leq k$. Informally, a state in the belief game satisfies p_k if the size of the belief set does not exceed the threshold $k \in \mathbb{N}_{>0}$.

Of focus in this paper will be linear temporal logic (LTL) surveillance formulas of the form $\square p_k$, termed *safety surveillance objective* which is satisfied if at each point in time the size of the belief set does not exceed k.

LTL formulas of this type are interpreted over (infinite) runs. If a run ρ satisfies an LTL formula φ, we write $\rho \models \varphi$. The formal definition of LTL semantics can be found in [28].

III. PROBLEM FORMULATION

A. Patrol problem

Consider an environment as in Figure 1 represented by a set $\mathcal{L}$, with a set $L \subseteq \mathcal{L}$ of free states. We construct a surveillance game structure $G = (\mathcal{G}, vis)$ with $\mathcal{G} = (S, s_0, \Sigma, \delta)$. Recall that $S = L_a \times L_t$ is the joint state space of the target and agent. We assume without loss of generality that $L_a = L_t = L$, i.e., both target and agent share the same state space, which is natural in the setting where both the agent and target can move in the same discretized space.

Given the visibility function vis, let $\mathcal{V}_{vis}(l) \subseteq L$ be the set of points in L that are visible from $l \in L$, i.e.,

$$\mathcal{V}_{vis}(l) := \{l_t \in L \mid vis(l, l_t) = true\}.$$

We first present the patrol problem studied in [24].

Problem 1: Find the smallest set $O \subseteq L$ of *vantage points* such that each point in L is visible from at least one vantage point $l \in O$, i.e.,

$$\min_{O} \quad |O| \qquad \text{s.t.} \quad \bigcup_{l \in O} \mathcal{V}_{vis}(l) = L.$$

This problem was solved in [24], by using a greedy approach to determine vantage points sequentially; i.e., a new vantage point is determined based on the visibility of the map obtained from the previous vantage points. Each new point is chosen to maximize information gain. In this case, since we deal with discrete state spaces, the information gain is quantified as the number of previously-unseen states that become visible after moving to a potential vantage point l_i. We define this *gain function* as:

$$g^K(l_i; \Omega^i_{i-K}) = |\mathcal{V}_{vis}(l_i) \cup \Omega^i_{i-K}| - |\Omega^i_{i-K}|, \qquad (1)$$

where Ω^i_{i-K} represents the set of visible points for the last K observations:

$$\Omega^i_{i-K} = \bigcup_{j=i-K}^{i} \mathcal{V}_{vis}(l_j).$$

We enforce that $i - K = 0$ when $i \leq K$. When $K = \infty$, the agent remembers everything it sees and patrolling is complete when all the states in the map have been seen at least once. When K is finite, the agent has to try to observe states it has not seen in the last K steps. In the next section, we discuss the patrol problem with finite K.

Example 1: Figure 2 shows a visualization of the gain function for the environment shown in Figure 1a.

Unfortunately, computing $g^K(l; \Omega^i_{i-K})$ is costly. For each l, computing the gain function requires $\mathcal{O}(|L|)$ operations. Thus computing the gain for all $l \in L$ has complexity

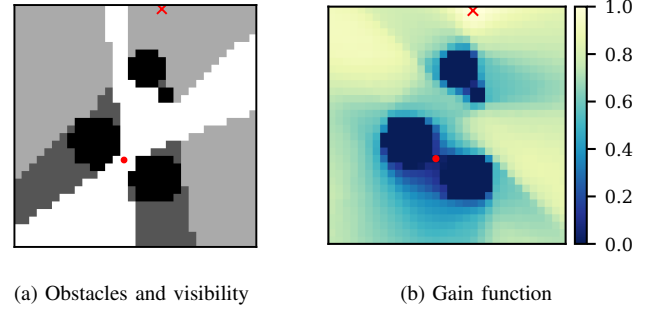(a) Obstacles and visibility (b) Gain function

Fig. 2: (a) Black regions are obstacles. White regions are visible from the red dot. Light gray regions are visible from the cross, but not from the dot. The area of light gray region corresponds to the gain at the cross. (b) The gain function (normalized for visualization), given that the dot has been visited. The cross marks the location of maximum information gain.

$\mathcal{O}(|L|^2)$ [24]. The computation becomes prohibitively expensive, especially for large state spaces. It is worth noting that this computation cannot be performed offline as it will depend on the choices of the target at run time.

B. Constrained patrol problem

We first provide some additional notation. We note that to help readability and avoid nested subscripts, we slightly abuse notation by dropping the subscripts a, t in the state $s_i = (l_{a_i}, B_{t_i})$. The state in the belief-set game at time i is referred to in this section as (l_i, B_i). Given a belief-set game $G_{\text{belief}} = (\mathcal{G}_{\text{belief}}, vis)$, let $\rho = s_0, s_1, \cdots = (l_0, B_0), (l_1, B_1), \ldots$ be an infinite run in the belief game $\mathcal{G}_{\text{belief}}$. We define $\rho^i_j = ((l_j, B_j), \ldots (l_i, B_i))$ with $j < i$ as the finite *word* in the infinite run from j to i. We can thus write the infinite run as $\rho = \rho_0^{j-1} \cdot \rho_j^i \cdot \rho_{i+1}^{\omega}$ and denote $\rho_0^{j-1} = (l_0, B_0), \ldots (l_{j-1}, B_{j-1})$ as a *finite prefix* and $\rho_{i+1}^{\omega} = (l_{i+1}, B_{i+1}), (l_{i+2}, B_{i+2}), \ldots$ as an *infinite suffix*. We are interested in constraining the patrol problem with the addition of a surveillance requirement. However, recall that the temporal surveillance objectives are only satisfied over infinite runs. This setting differs from the approach in [24] in that it now involves an infinite time horizon. We thus want to maximize the short-term visibility in a time interval of length K in order to incentivize the agent to gain visibility of states it has not seen from the last K vantage points.

We construct a *sequentially optimal* vantage point generation problem. Simply, given the current state of the belief game (l_i, B_i) and an integer $K > 0$, we want to choose the next vantage point that maximizes the visibility of states that have not been seen by the last K vantage points and does not violate the surveillance safety specification φ.

We now present the main problem of this paper.

Problem 2: Given a belief-set surveillance game G_{belief}, at time step $i - 1$, the run in the game is ρ_0^{i-1}. Additionally,

given $K \in \mathbb{N}$, and a surveillance requirement φ, solve

$$\arg\max_{l_i} g^K(l_i; \Omega_{i-K}^i) \qquad \text{s.t.} \qquad \rho \models \varphi$$

where $\rho = \rho_0^{i-1} \cdot (l_i, B_i) \cdot \rho_{i+i}^\omega$ for all $B_i \in succ_t(l_{i-1}, B_{i-1})$.

Informally the agent needs to choose the next vantage point that guarantees the infinite suffix will still be correct with respect to the surveillance specification regardless of the target's action choices.

In the next section we present how to translate the constraint on the infinite suffix as a feasible set for the optimization problem. The resulting optimization problem needs to be solved online for every time step. As discussed previously, evaluating the exact gain function for all feasible points is a computationally intensive task. We thus show how to approximate the gain function using a convolutional neural network which allows for the generation of the next vantage point in real time even for large maps.

IV. SOLUTION APPROACH

A. Surveillance winning region

Consider a pair (G, φ), where G is a surveillance game structure and φ is a surveillance objective. Given the relevant surveillance predicates and specification over the belief-set game structure G_{belief}, we obtain a *safety game*.

A safety game defines a set $F \subseteq L \times \mathcal{P}(L)$ of safe states. A run $\rho \models \varphi$ iff $s_i \in F$ for all $i \geq 0$, i.e., if only safe states are visited in the run. Otherwise, $\rho \nvDash \varphi$. The *winning region* $\mathcal{W} \subseteq F$ is the set of states from which a winning strategy exists. Informally a state (l_a, B_t) in the belief game is said to be in the winning region $\mathcal{W}$ iff, for all future steps the target can make from the current belief state B_t, there exists a strategy from l_a that guarantees that only states in $\mathcal{W}$ will be visited. We can use standard algorithms for safety games (e.g. [29]), to compute the winning region.

We note that to avoid the state space blow-up arising from the subset construction in the partial information game, we solve the surveillance games using the *belief-set abstraction* techniques detailed in [22].

B. Feasible set construction

Synthesizing a strategy for a surveillance specification is inherently *reactive*, the agent observes the target's action and reacts appropriately in order to satisfy its specification. However, planning only the immediate action, as is done in [22], will not result in a large gain in visibility coverage.

Informally, we want to be able to move to the best vantage point possible in a region that guarantees that no matter what the target does in the future, there will always exist a way for the agent to satisfy its surveillance requirement.

More concretely, consider the belief-set game G_{belief} along with a surveillance specification φ. Assume we have solved the resulting safety game and computed the winning region $\mathcal{W} \subseteq L \times \mathcal{P}(L)$. Recall that the state of the belief-set game structure is the joint position and belief $s = (l_a, B_t)$. We extend the definition of $succ_a(l_a, l_t, l'_t)$ to sets by stipulating that the

set $succ_a(l_a, B_t) = \bigcup_{l'_t \in succ_t(l_a, B_t)} succ_a(l_a, B_t, l'_t)$. Additionally, we extend the definition of $succ_t(l_a, B_t)$ to more than the immediate successor by recursively defining $succ_t^k(l_a, B_t) = \bigcup_{B'_t \in succ_t^{k-1}(l_a, B_t)} succ_t(l_a, B_t)$.

We now define the *reachable* successor set $R(l_a, B_t)$ as

$$R(l_a, B_t) = \{l'_a \in L \mid l_a \in succ_a(l_a, B_t),$$
$$\forall B'_t \in succ_t(l_a, B_t), (l'_a, B'_t) \in \mathcal{W}\}. \quad (2)$$

Informally, a successor state l'_a is a reachable successor of (l_a, B_t) if, *for all possible successor belief states B'_t*, we have that (l'_a, B'_t) does not violate the surveillance specification. A state being in the set $R(l_a, B_t)$ a strict condition – the same successor state l'_a has to be correct for all possible B'_t choices the target can make.

We can then recursively define reachable successor set over a finite horizon k, by applying Equation 2 k times:

$$R^k(l_a, B_t) = \{l'_a \in \bigcup_{l_a \in R^{k-1}} succ_a(l_a, B_t) \mid$$
$$\forall B'_t \in succ_t^k(l_a, B_t), (l'_a, B'_t) \in \mathcal{W}\}.$$

Intuitively, as k gets larger, finding a point $l \in L$ that guarantees staying in the winning region $\mathcal{W}$ gets harder. We define k^*, the largest k such that $R^{k^*+1} = R^k$ and refer to the set as the *maximum reachability set $R^*(l_a, B_t)$*.

Remark: Computing the set R^* is similar to standard fixed point set computations detailed in [30] for computing the winning region $\mathcal{W}$ in temporal logic synthesis problems. The main difference is that these algorithms focus on computing a set where *for all* actions by a player *there exists* a strategy for the other player to win. We are interested in reversing the order of the quantifiers which results in a more restrictive condition. One can think of k^* as the maximum number of actions the agent can make without "reacting" to the target's actions and still guaranteeing correctness with regards to the specification.

The reachable set computation can be performed online in $\mathcal{O}(|L \times \mathcal{P}(L)|^2)$. However, if required, the entire computation can be performed offline with complexity $\mathcal{O}(|L \times \mathcal{P}(L)|^4)$, and the results stored for all possible states – there will be a set R^* stored for up to $|L \times \mathcal{P}(L)|$ states.

We now present an *equivalent* version of Problem 2 with the constraint on the infinite run translated to a feasible set. Given that the state of the game at time i is (l_i, B_i), the next vantage point is the solution to the following problem:

Problem 3: Given a belief-set surveillance game G_{belief} at time $i-1$, $K \in \mathbb{N}$, and a surveillance requirement φ, solve

$$\arg\max_{l \in R^*(l_{i-1}, B_{i-1})} g^K(l_i; \Omega_{i-K}^i)$$

where (l_{i-1}, B_{i-1}) is the state in the game at $t = i - 1$.

C. Approximating the gain function

When the state space is large, computing the gain function is computationally expensive. In such cases, we approximate $g^K(l_i; \Omega_{i-K}^i)$ using a convolutional neural network which enables efficient computation of the gain function at run time.

Without loss of generality, we discuss the training process for $K = \infty$. The approximated gain function $g_\theta(l_i; \Omega_0^i, B_i)$ takes as input the cumulatively visible set Ω_0^i, and the associated shadow boundaries $\mathcal{B}_i = \partial\Omega_0^i \setminus L^C$. Here, $\partial\Omega_0^i$ is boundary of the set of states visible from at least one of the previous vantage points. Intuitively, the shadow boundaries (or frontiers) $\mathcal{B}_i$ are the boundaries between free space and occlusion. In order to train the neural network, we sample different environments $\mathcal{L}$ cropped from the INRIA Aerial Image Labeling Dataset [31]. For each $\mathcal{L}$, an initial position $l_0 \in L \subseteq \mathcal{L}$ for the agent is randomly sampled. We train on 100k corresponding data pairs $\{(\Omega_0^i, \mathcal{B}_i), g(l_i; \Omega_0^i)\}$.

Each subsequent l_i after the initial position is chosen as

$$l_i = \arg\max_l g(l; \Omega_0^{i-1}).$$

The parameters θ of the neural network are updated to minimize the cross entropy loss $E\Big(g(l_i; \Omega_0^i), g_\theta(l_i; \Omega_0^i, \mathcal{B}_i)\Big)$ between the normalized exact gain function g and the normalized prediction g_θ, where E is defined as

$$E(p, q) := \int p(x) \log q(x) + (1 - p(x)) \log(1 - q(x)) \ dx.$$

The architecture of g_θ is based on U-Net [32], a standard CNN for dense inference problems which aggregates information across various scales using skip connections. The network has nineteen 3x3 convolution layers, each followed by batch norm and leaky ReLU. Six of the convolutions use stride 2 for downsampling, while six are preceded by bilinear upsampling. For more details we refer the reader to [24].

At run time, we approximate the gain function using g_θ, which is measure of how useful it is to place a new vantage point at l. Since not all l are reachable due to the surveillance constraint, we restrict g_θ to be zero for l outside R^*. The next vantage point is chosen as the point within the R^* that maximizes g_θ and this process is repeated for all time.

V. SURVEILLANCE GAME ABSTRACTION

In order to scale to large state spaces, we present a conservative abstraction of the surveillance game G with visibility function vis. We first define an abstraction function.

Abstraction function: Given an environment with state space L, an *abstraction function* $\psi: L \rightarrow 2^L$ yields a partition $L_\psi = l_1, \ldots, l_n$ of the state space L with $\bigcup_{i=1}^n l_i = L$ and $\forall i \neq j$ we have $l_i \cap l_j = \emptyset$.

Theorem 1: Given *concrete* surveillance game $G = (\mathcal{G}, vis)$, we say the abstract surveillance game $G_\psi = (\mathcal{G}_\psi, vis_\psi)$ with $\mathcal{G}_\psi = (S_\psi, s_{0\psi}, \Sigma, \delta_\psi)$, and $s_\psi = (l_\psi^a, l_\psi^t)$ with $l_\psi^a, l_\psi^t \in L_\psi$, generated by abstraction function ψ is *sound* if the following holds

1) $\delta_\psi(s_\psi, \sigma) = s_\psi'$ with $s_\psi' = (l_\psi^{a'}, l_\psi^{t'})$ iff
   - For all $l^a \in l_\psi^a$ in the concrete game, it must be the case that $\delta((l_a, l_t), \sigma) = (l_a', l_t')$ with either $l_a' \in l_\psi^{a'}$ or $l_a' \in l_\psi^a$. Or equivalently, we require there does not exist $l^a \in l_\psi^a$ such that $l_a' \notin l_\psi^{a'}$ and $l_a' \notin l_\psi^a$.

   - *There exists* an $l^t \in l_\psi^t$ in the concrete game, such that $\delta((l_a, l_t), \sigma) = (l_a', l_t')$ with $l_t' \in l_\psi^{t'}$.

2) $vis_\psi((l_\psi^a, l_\psi^t)) = \top$ iff *for all* $(l_a, l_t) \in (l_\psi^a, l_\psi^t)$ we have $vis(l_a, l_t) = \top$.

Simply, the abstract version of the surveillance game gives more power to the target and removes power from the agent by disallowing transitions between abstract states for the agent if it is not possible *for all* underlying concrete states. Conversely transitions between abstract states are allowed for the target if *there exists* a corresponding transition in any of the underlying concrete states.

Example 2: Figure 1 shows an example of abstracting a concrete map with a large state space to a coarser representation. Each grid state in Figure 1b represents a collection of pixels from Figure 1a.

This construction of the abstraction guarantees that a strategy that is correct on the abstract game with respect to safety surveillance objective φ corresponds to a winning strategy on the concrete game. However, since the abstraction is conservative, not finding a winning strategy on the abstraction does not mean one does not exist in the concrete game. In the experiments that follow in the next section, we use these abstractions to reduce the size of the state space for the winning region computation in the safety game.

Remark: The methodology detailed in this paper introduces several sources of conservatism – state abstraction shown in this section and the belief abstraction in [22] used in generating strategies. These allow for more scalable synthesis on large problems. However, due to the added conservatism, there is no guarantee that there exists a winning strategy in the abstract model, even if one does exist in the concrete models. Techniques such as *counterexample guided abstraction refinement* (CEGAR) can be used to generate new abstractions if no winning strategy can be found.

VI. NUMERICAL EXPERIMENTS

A. Effectiveness of patrol

In order to illustrate the efficacy of including the vision-maximization for the patrol problem, we compare an agent employing the surveillance constrained patrol method with a window of $K = 5$ to an agent following a pure safety surveillance strategy. We compare the *receding gain* — the ratio of newly seen states (that have not been seen in the last $K = 5$ steps), to the total number of traversable states in the map.

Figure 4 shows the results of the comparison. We note that when no patrol optimization is enforced, the agent tends to stay still as it is sufficient to maintain surveillance of the target. This causes no new states to be seen and results in the value of the receding gain to be zero for much of the time. Meanwhile, when visibility-maximization is taken into account, the agent actively moves to the best vantage point it can safely reach. The sum of gain over the time period is 2.07 when patrol optimization is performed vs 0.52 for the pure surveillance case. This corresponds to average gain of 3.4% per time step vs 0.8%.

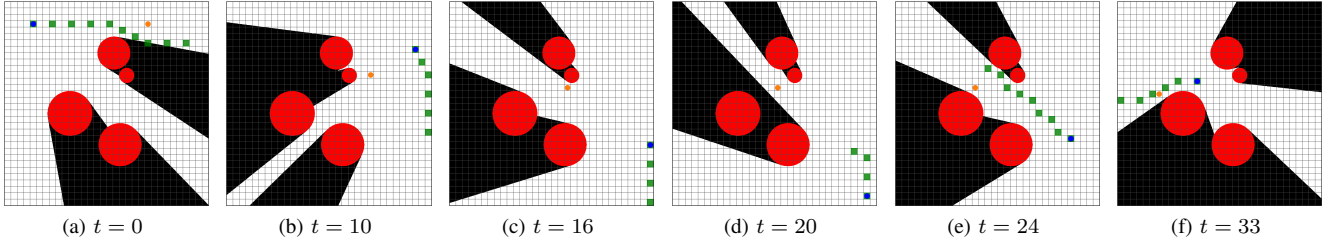| (a) $t = 0$ | (b) $t = 10$ | (c) $t = 16$ | (d) $t = 20$ | (e) $t = 24$ | (f) $t = 33$ |

Fig. 3: Simulation of the case study presented in Section II. The green cells correspond to the path to the optimal vantage point which is the last green square on the path. Snapshots of when a new vantage point is recomputed is shown here.
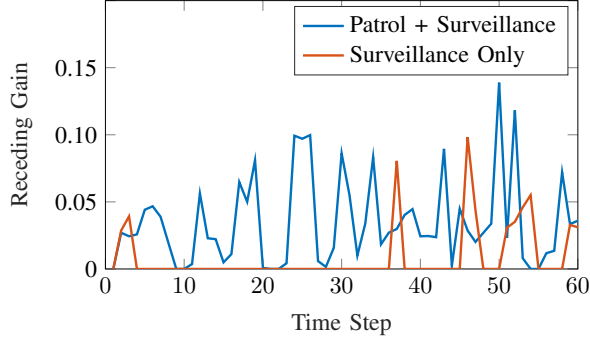

Fig. 4: Comparison of the effects of visibility optimization. Adding the patrol objective drastically increases the agent's visibility of the environment.

Finally, we remark that when the agent in the pure surveillance case stays still for longer than K steps, all states not in its current vision become unseen states. Hence, when it finally moves, the gain becomes high, causing the spikes.

B. Motivating case study

Here we present the results of solving the case study presented in Section II with surveillance requirement $\Box p \leq 1$ – which corresponds to not allowing the agent to lose sight of the target. We assume the agent can move 4 cells for every 1 cell the target can move. The target is controlled by a human moving it with the arrow keys on a keyboard.

We have implemented the simulation in Python, using the slugs reactive synthesis tool [33] to solve the safety game and find the winning region $\mathcal{W}$. The experiments were performed on an Intel i5-5300U 2.30 GHz CPU with 8 GB of RAM. We solve for the winning surveillance region on the 32×32 abstraction shown in Figure 1b. New vantage points and the paths to those points were computed at each snapshot shown in Figure 3 in green, and took approximately ≈ 1 second for each new computation. Once the agent reaches the vantage point, the next one is computed and this process is repeated. We see that the agent can leverage its superior speed to move away from the target in order to see new areas, but it is still able to keep sight of the target at all times. A video of the simulation can be seen at `https://bit.ly/2F2Vnjb`.

Table I: Comparison of gain function computation time (secs) for various map sizes. Using the exact gain function quickly becomes prohibitively expensive.

	Grid size				
	64^2	128^2	256^2	512^2	1024^2
Exact	0.499	2.565	36.234	597.797	8105.750
CNN	0.058	0.071	0.126	0.334	1.194

C. Computation benchmarks

Here we compare the process on various sized maps in order to demonstrate the scalability of using a CNN to compute the gain function. See Table I for a comparison of computation time for the gain function. The CNN computes the gain function within 1 second for maps up to 1024×1024. Meanwhile, the exact gain function computation already takes more than 1 second for 128×128 maps and more than 2 hours for 1024×1024 maps. The scalability is crucial as the vantage point computation is conducted online on the concrete environment, compared to the surveillance strategy synthesis which is performed offline on an abstraction.

We measure the effectiveness of the gain function prediction by the decay of the residual, i.e. the number of steps required to reduce the occlusions to a certain threshold. The exact computation and CNN approximation generated sequences of vantage points that converged to the same level of coverage in a comparable number of steps. For a more detailed discussion on the evaluation of the gain function approximation, we refer the reader to [24].

D. Discussion

We are able to generate approximately optimal vantage points in real time for very large scale maps. Due to restricting the search space of these vantage points through the winning region in a surveillance game, we are also able to guarantee correctness on these large-scale maps.

We note that for illustration, the experiments shown here use a simple surveillance specification $\varphi = \Box p_1$ which means the agent cannot lose sight of the target. We provide additional examples and animations with looser surveillance requirements at `http://visibility.page.link/evasion`.

VII. Conclusion

We proposed a framework for constraining patrol strategies to be formally correct with respect to a surveillance requirement defined as a temporal logic constraint. Our approach is flexible as it allows a user to tailor the surveillance requirement for their specific application. Using convolutional neural networks, we compute a measure of information gain which prioritizes observing states that have not been seen for a given time interval. We choose the next vantage point as the location within the maximal reach set R^* that optimizes the gain function. We show that employing the patrol strategy leads to better visibility of the environment, as measured by a receding gain metric.

For future work, we will extend the framework to handle the case of *exploration*, where the environment is initially unknown. We also plan to consider more sophisticated surveillance requirements, such as *surveillance liveness objectives*, where, infinitely often, the agent must keep the uncertainty of the target's location below a predefined threshold.

VIII. Acknowledgment

References

[1] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in uav communication networks," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1123–1152.

[2] K. Kanistras, G. Martins, M. J. Rutherford, and K. P. Valavanis, "Survey of unmanned aerial vehicles (uavs) for traffic monitoring," *Handbook of unmanned aerial vehicles*, pp. 2643–2666, 2015.

[3] J. Oyekan and H. Hu, "Towards autonomous patrol behaviours for uavs," in *Proceedings of UK EPSRC Workshop on Human Adaptive Mechatronics, Staffordshire University, Stafford, UK*, 2009, pp. 15–16.

[4] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, "Autonomous uav surveillance in complex urban environments," in *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 02*. IEEE Computer Society, 2009, pp. 82–85.

[5] K. S. Lee, M. Ovinis, T. Nagarajan, R. Seulin, and O. Morel, "Autonomous patrol and surveillance system using unmanned aerial vehicles," in *2015 IEEE 15th International Conference on Environment and Electrical Engineering (EEEIC)*, June 2015, pp. 1291–1297.

[6] Z. Liu, J. Dai, B. Wu, and H. Lin, "Communication-aware motion planning for multi-agent systems from signal temporal logic specifications," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 2516–2521.

[7] Z. Liu, B. Wu, J. Dai, and H. Lin, "Distributed communication-aware motion planning for multi-agent systems from stl and spatel specifications," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 4452–4457.

[8] Y. Landa, R. Tsai, *et al.*, "Visibility of point clouds and exploratory path planning in unkown environments," *Communications in Mathematical Sciences*, vol. 6, no. 4, pp. 881–913, 2008.

[9] L. Valente, Y.-H. R. Tsai, and S. Soatto, "Information-seeking control under visibility-based uncertainty," *Journal of Mathematical Imaging and Vision*, vol. 48, no. 2, pp. 339–358, 2014.

[10] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on*. IEEE, 1997, pp. 146–151.

[11] S. K. Ghosh, J. W. Burdick, A. Bhattacharya, and S. Sarkar, "Online algorithms with discrete visibility - exploring unknown polygonal environments," *IEEE Robotics Automation Magazine*, vol. 15, no. 2, pp. 67–76, June 2008.

[12] H. H. González-Banos and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *The International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 829–848, 2002.

[13] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon path planning for 3d exploration and surface inspection," *Autonomous Robots*, vol. 42, no. 2, pp. 291–306, 2018.

[14] L. Heng, A. Gotovos, A. Krause, and M. Pollefeys, "Efficient visual exploration and coverage with a micro aerial vehicle in unknown environments," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 1071–1078.

[15] H. Surmann, A. Nüchter, and J. Hertzberg, "An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments," *Robotics and Autonomous Systems*, vol. 45, no. 3-4, pp. 181–198, 2003.

[16] J. Urrutia, "Art gallery and illumination problems," in *Handbook of computational geometry*. Elsevier, 2000, pp. 973–1027.

[17] N. Basilico, N. Gatti, and F. Amigoni, "Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder," *Artif. Intell.*, vol. 184-185, June 2012.

[18] T. H. Chung, G. A. Hollinger, and V. Isler, "Search and pursuit-evasion in mobile robotics," *Autonomous Robots*, vol. 31, p. 299, Jul 2011.

[19] M. Burger, Y. Landa, N. M. Tanushev, and R. Tsai, "Discovering a point source in unknown environments," in *Algorithmic Foundation of Robotics VIII*. Springer, 2009, pp. 663–678.

[20] Y. Landa, N. M. Tanushev, and R. Tsai, "Discovery of point sources in the helmholtz equation posed in unknown domains with obstacles," *Communications in Mathematical Sciences*, vol. 9, no. 3, pp. 903–928, 2011.

[21] R. Takei, R. Tsai, Z. Zhou, and Y. Landa, "An efficient algorithm for a visibility-based surveillance-evasion game," *Communications in Mathematical Sciences*, vol. 12, no. 7, pp. 1303–1327, 2014.

[22] S. Bharadwaj, R. Dimitrova, and U. Topcu, "Synthesis of surveillance strategies via belief abstraction," *2018 IEEE Conference on Decision and Control (CDC)*, Dec 2018. [Online]. Available: http://dx.doi.org/10.1109/CDC.2018.8619353

[23] ——, "Distributed synthesis of surveillance strategies for mobile sensors," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 3335–3342.

[24] L. Ly and Y.-H. R. Tsai, "Autonomous exploration, reconstruction, and surveillance of 3d environments aided by deep learning," *2019 IEEE International Conference on Robotics and Automation (ICRA). To Appear.*, 2019.

[25] S. Bharadwaj, R. Bloem, R. Dimitrova, B. Könighofer, and U. Topcu, "Synthesis of minimum-cost shields for distributed systems," in *2019 Annual American Control Conference, ACC 2019, Philadelphia, PA, USA, July 10-12, 2019*. IEEE.

[26] R. Bloem, B. Könighofer, R. Könighofer, and C. Wang, "Shield synthesis," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2015, pp. 533–548.

[27] S. Ray, *Scalable techniques for formal verification*. Springer Science & Business Media, 2010.

[28] C. Baier and J. Katoen, *Principles of model checking*. MIT Press, 2008.

[29] R. Mazala, "Infinite games," in *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, 2001, pp. 23–42.

[30] E. Gradel and W. Thomas, *Automata, logics, and infinite games: a guide to current research*. Springer Science & Business Media, 2002, vol. 2500.

[31] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark," in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2017.

[32] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[33] R. Ehlers and V. Raman, "Slugs: Extensible GR(1) synthesis," in *Proc. CAV 2016*, ser. LNCS, vol. 9780. Springer, 2016, pp. 333–339.