Proceedings Paper:

Bharadwaj, S., Dimitrova, R. and Topcu, U. (2019) Synthesis of surveillance strategies via belief abstraction. In: 2018 IEEE Conference on Decision and Control (CDC). 2018 IEEE Conference on Decision and Control (CDC), 17-19 Dec 2018, Miami Beach, FL, USA. IEEE , pp. 4159-4166. ISBN 9781538613962

https://doi.org/10.1109/cdc.2018.8619353

Synthesis of Surveillance Strategies via Belief Abstraction

Suda Bharadwaj[1] and Rayna Dimitrova[2] and Ufuk Topcu[1]

***Abstract*— We provide a novel framework for the synthesis of a controller for a robot with a *surveillance objective*, that is, the robot is required to maintain knowledge of the location of a moving, possibly adversarial target. We formulate this problem as a one-sided partial-information game in which the winning condition for the agent is specified as a temporal logic formula. The specification formalizes the surveillance requirement given by the user by quantifying and reasoning over the agent's beliefs about a target's location. We also incorporate additional non-surveillance tasks. In order to synthesize a surveillance strategy that meets the specification, we transform the partial-information game into a perfect-information one, using abstraction to mitigate the exponential blow-up typically incurred by such transformations. This transformation enables the use of off-the-shelf tools for reactive synthesis. We evaluate the proposed method on two case-studies, demonstrating its applicability to diverse surveillance requirements.**

I. INTRODUCTION

Performing surveillance, that is, tracking the location of a target, has many applications. If the target is adversarial, these applications include patrolling and defense, especially in combination with other objectives, such as providing certain services or accomplishing a mission. Techniques for tracking non-adversarial but unpredictable targets have been proposed in settings like surgery to control cameras to keep a patient's organs under observation despite unpredictable motion of occluding obstacles [1]. Mobile robots in airports have also been proposed to carry luggage for passengers, requiring the robots to follow the human despite unpredictable motion and possibly sporadically losing sight of the target [2].

When dealing with a possibly adversarial target, a strategy for the surveying agent for achieving its objective can be seen as a strategy in a two-player game between the agent and the target. Since the agent may not always observe, or even know, the exact location of the target, surveillance is, by its very nature, a partial-information problem. It is thus natural to reduce surveillance strategy synthesis to computing a winning strategy for the agent in a two-player partial-information game. Game-based models for related problems have been extensively studied in the literature. Notable examples include pursuit-evasion games [3], patrolling games [4], and graph-searching games [5], where the problem is formulated as enforcing eventual detection, which is, in its essence a search problem – once the target is detected, the game ends. For many applications, this formulation is too restrictive. Often, the goal is not to detect or capture the target, but to maintain certain level of information about its location over an unbounded (or infinite) time duration, or, alternatively, be able to obtain sufficiently precise information over and over again. In other cases, the agent has an additional objective, such as performing certain task, which might prevent him from capturing the target, but allow for satisfying a more relaxed surveillance objective.

In this paper, we study the problem of synthesizing strategies for enforcing *temporal surveillance objectives*, such as the requirement to never let the agent's uncertainty about the target's location exceed a given threshold, or recapturing the target every time it escapes. To this end, we consider surveillance objectives specified in linear temporal logic (LTL), equipped with basic surveillance predicates. This formulation also allows for a seamless combination with other task specifications. Our computational model is that of a two-player game played on a finite graph, whose nodes represent the possible locations of the agent and the target, and whose edges model the possible (deterministic) moves between locations. The agent plays the game with partial information, as it can only observe the target when it is in its area of sight. The target, on the other hand, always has full information about the agent's location, even when the agent is not in sight. In that way, we consider a model with one-sided partial information, making the computed strategy for the agent robust against a potentially more powerful adversary.

We formulate surveillance strategy synthesis as the problem of computing a winning strategy for the agent in a partial-information game with a surveillance objective. There is a rich theory on partial-information games with LTL objectives [6], [7], and it is well known that even for very simple objectives the synthesis problem is EXPTIME-hard [8], [9]. Moreover, all the standard algorithmic solutions to the problem are based on some form of *belief set construction*, which transforms the imperfect-information game into a perfect-information game and of exponentially larger size, since the new set of states is the powerset of the original one. Thus, such approaches scale poorly in general, and are not applicable in most practical situations.

We address this problem by using *abstraction*. We introduce an *abstract belief set construction*, which underapproximates the information-tracking abilities of the agent (or, alternatively, overapproximates its belief, i.e., the set of positions it knows the target could be in). We leverage this construction by *reasoning* over the agent's belief in the target location, and this allows us to specify surveillance objectives in LTL over these belief states. Thus, we provide a framework to treat surveillance synthesis as a two-player perfect-information game with an LTL objective, which we

[1]Suda Bharadwaj and Ufuk Topcu are with the University of Texas at Austin

[2]Rayna Dimitrova is with the University of Leicester, UK. Most of this work was done while Rayna Dimitrova was a postdoctoral researcher at UT Austin.

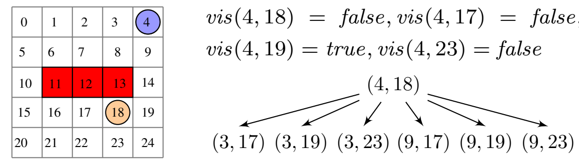
then solve using off-the shelf reactive synthesis tools [10]. Our construction guarantees that the abstraction is sound, that is, if a surveillance strategy is found in the abstract game, it corresponds to a surveillance strategy for the original game. If such a strategy is not found because the abstraction is too coarse, then techniques such as counterexample guided abstraction refinement (CEGAR) [11] can be used to automatically refine the abstraction. CEGAR has successfully demonstrated its potential in formal verification and reactive synthesis. CEGAR can be applied in our framework to automatically refine abstract belief states based on counterexamples of the agent being unable to satisfy its surveillance specification. For the full details of the CEGAR procedure we refer the reader to [12], where we describe the automated refinement of belief abstractions.

Contributions. Our contributions are as follows:
(1) We propose a *formalization of surveillance objectives* as temporal logic specifications, and frame surveillance strategy synthesis as a reactive synthesis problem in a partial-information two player game.
(2) We develop an *abstraction method that soundly approximates* surveillance strategy synthesis, thus mitigating the state space explosion enabling the application of efficient techniques for reactive synthesis.
(3) We demonstrate the use of our framework in practice by evaluating our approach on different surveillance objectives (e.g, safety, and liveness) combined with task specifications, and discuss the qualitatively different behaviour of the synthesized strategies for the different kinds of specifications.

Related work. While closely related to the surveillance problem we consider, pursuit-evasion games with partial information [3], [13], [14] formulate the problem as eventual detection, and do not consider combinations with other mission specifications. Other work, such as [15] and [16], additionally incorporates map building during pursuit in an unknown environment, but again solely for target detection.

Synthesis from LTL specifications [17], especially from formulae in the efficient GR(1) fragment [18], has been extensively used in robotic planning (e.g. [19], [20]), but surveillance-type objectives, such as the ones we study here, have not been considered so far. Epistemic logic specifications [21] can refer to the knowledge of the agent about the truth-value of logical formulas, but, contrary to our surveillance specifications, are not capable of expressing requirements on the size of the agent's uncertainty.

CEGAR has been developed for verification [11], and later for control [22], of LTL specifications. It has also been extended to infinite-state partial-information games [23], and used for sensor design [24], both in the context of safety specifications. In addition to being focused on safety objectives, the refinement method in [23] is designed to provide the agent with just enough information to achieve safety, and is thus not applicable to surveillance properties whose satisfaction depends on the size of the belief sets.



(a) Surveillance arena (b) Transitions from the initial state

Fig. 1: A simple surveillance game on a grid arena. Obstacles are shown in red, the agent (at location 4) and the target (at location 18) are coloured in blue and orange respectively.

II. GAMES WITH SURVEILLANCE OBJECTIVES

We begin by defining a formal model for describing surveillance strategy synthesis problems, in the form of a two-player game between an agent and a target, in which the agent has only partial information about the target's location.

A. Surveillance Game Structures

We define a *surveillance game structure* to be a tuple $G = (S, s^{\mathsf{init}}, T, vis)$, with the following components:

- $S = L_a \times L_t$ is the set of states, with L_a the set of locations of the agent, and L_t the locations of the target;
- $s^{\mathsf{init}} = (l_a^{\mathsf{init}}, l_t^{\mathsf{init}})$ is the initial state;
- $T \subseteq S \times S$ 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 the area of sight of* l_a.

The transition relation T encodes the one-step 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 successor locations of the target:
$succ_t(l_a, l_t) = \{l_t' \in L_t \mid \exists l_a'. \ ((l_a, l_t), (l_a', l_t')) \in T\}$.

We extend $succ_t$ to sets of locations of the target by stipulating that the set $succ_t(l_a, L)$ consists of all possible successor locations 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':
$succ_a(l_a, l_t, l_t') = \{l_a' \in L_a \mid ((l_a, l_t), (l_a', l_t')) \in T\}$.

We assume that, for every $s \in S$, there exists $s' \in S$ such that $(s, s') \in T$, that is, from every state there is at least one move possible (this might be staying in the same state). We also assume that when the target moves to an invisible location, its position does not influence the possible one-step moves of the agent. Formally, we require that if $vis(l_a, l_t') = vis(l_a, \widehat{l}_t') = \mathit{false}$, then $succ_a(l_a, l_t, l_t') = succ_a(l_a, \widehat{l}_t, \widehat{l}_t')$ for all target locations $l_t, l_t', \widehat{l}_t, \widehat{l}_t' \in L_t$. This assumption is natural in the setting when the agent can move in one step only to locations that are in its sight.

Example 1: Figure 1 shows an example of a surveillance game on a grid. The sets of possible locations L_a and L_t for the agent and the target consist of the squares of the grid. The transition relation T encodes the possible one-step moves of both the agent and the target on the grid, and incorporates

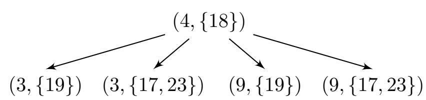


Fig. 2: Transitions from the initial state in the belief-set game from Example 2 where $vis(4, 17) = vis(4, 23) = false$.

all desired constraints. For example, moving to an occupied location, or an obstacle, is not allowed. Figure 1b shows the possible transitions from the initial state $(4, 18)$.

The function vis encodes straight-line visibility: a location l_t is visible from a location l_a if there is no obstacle on the straight line between them. Initially the target is not in the area of sight of the agent, but the agent knows the initial position of the target. However, once the target moves to one of the locations reachable in one step, in this case, locations $\{17, 19, 23\}$, this might no longer be the case. More precisely, if the target moves to location 19, then the agent observes its location, but if it moves to one of the others, then the agent no longer knows its exact location. ■

B. Belief-Set Game Structures

In surveillance strategy synthesis we need to state properties of, and reason about, the information which the agent has, i.e. its *belief* about the location of the target. To this end, we can employ a powerset construction which is commonly used to transform a partial-information game into a perfect-information one, by explicitly tracking the knowledge one player has as a set of possible states of the other player.

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 structure $G = (S, s^{\mathsf{init}}, T, vis)$ we define the corresponding *belief-set game structure* $G_{\mathsf{belief}} = (S_{\mathsf{belief}}, s^{\mathsf{init}}_{\mathsf{belief}}, T_{\mathsf{belief}})$ with the following components:

- $S_{\mathsf{belief}} = L_a \times \mathcal{P}(L_t)$ 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;
- $s^{\mathsf{init}}_{\mathsf{belief}} = (l^{\mathsf{init}}_a, \{l^{\mathsf{init}}_t\})$ is the initial state;
- $T_{\mathsf{belief}} \subseteq S_{\mathsf{belief}} \times S_{\mathsf{belief}}$ is the transition relation where $((l_a, B_t), (l'_a, B'_t)) \in T_{\mathsf{belief}}$ 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) $B'_t = \{l'_t \in succ_t(l_a, B_t) \mid vis(l_a, l'_t) = false\}$.

Condition (1) captures the successor locations of the target that can be observed from the agent's current position l_a. Condition (2) corresponds to the belief set consisting of *all possible successor locations of the target not visible from* l_a.

Example 2: Consider the surveillance game structure from Example 1. The initial belief set is $\{18\}$, consisting of the target's initial position. After the first move of the target, there are two possible belief sets: the set $\{19\}$ resulting from the move to a location in the area of sight of the agent, and $\{17, 23\}$ consisting of the two invisible locations reachable in one step from location 18. Figure 2 shows the successor states of the initial state $(4, \{18\})$ in G_{belief}. ■

Based on T_{belief}, we can define the functions $succ_t : S_{\mathsf{belief}} \to \mathcal{P}(\mathcal{P}(L_t))$ and $succ_a : S_{\mathsf{belief}} \times \mathcal{P}(L_t) \to \mathcal{P}(L_a)$ similarly to the corresponding functions defined for G.

A *run* in G_{belief} is an infinite sequence $s_0, s_1, \ldots$ of states in S_{belief}, where $s_0 = s^{\mathsf{init}}_{\mathsf{belief}}$, $(s_i, s_{i+1}) \in T_{\mathsf{belief}}$ for all i.

A *strategy for the target in* G_{belief} is a function $f_t : S^+_{\mathsf{belief}} \to \mathcal{P}(L_t)$ such that $f_t(\pi \cdot s) = B_t$ implies $B_t \in succ_t(s)$ for every $\pi \in S^*_{\mathsf{belief}}$ and $s \in S_{\mathsf{belief}}$. That is, a strategy for the target suggests a move resulting in some belief set reachable from some location in the current belief.

A *strategy for the agent in* G_{belief} is a function $f_a : S^+_{\mathsf{belief}} \times \mathcal{P}(L_t) \to S_{\mathsf{belief}}$ such that $f_a(\pi \cdot s, B_t) = (l'_a, B'_t)$ implies $B'_t = B_t$ and $l'_a \in succ_a(s, B_t)$ for every $\pi \in S^*_{\mathsf{belief}}$, $s \in S_{\mathsf{belief}}$ and $B_t \in \mathcal{P}(L_t)$. Intuitively, a strategy for the agent suggests a move based on the observed history of the play and the current belief about the target's position.

The outcome of given strategies f_a and f_t for the agent and the target in G_{belief}, denoted $outcome(G_{\mathsf{belief}}, f_a, f_t)$, is a run $s_0, s_1, \ldots$ of G_{belief} such that for every $i \geq 0$, we have $s_{i+1} = f_a(s_0, \ldots, s_i, B^i_t)$, where $B^i_t = f_t(s_0, \ldots, s_i)$.

C. Temporal Surveillance Objectives

Since the states of a belief-set game structure track the information that the agent has, we can state and interpret surveillance objectives over its runs. We now formally define the surveillance properties in which we are interested.

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 $|\{l_t \in B_t \mid vis(l_a, l_t) = false\}| \leq k$. Intuitively, p_k is satisfied by the states in the belief game structure where the size of the belief set does not exceed the threshold $k \in \mathbb{N}_{>0}$.

We study surveillance objectives expressed by formulas of linear temporal logic (LTL) over surveillance predicates. The LTL surveillance formulas are generated by the grammar $\varphi := p \mid true \mid false \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \bigcirc \varphi \mid \varphi \mathcal{U} \varphi \mid \varphi \mathcal{R} \varphi$, where $p \in \mathcal{SP}$ is a surveillance predicate, $\bigcirc$ is the *next* operator, $\mathcal{U}$ is the *until* operator, and $\mathcal{R}$ is the *release* operator. We also define the derived operators *finally*: $\Diamond \varphi = true \, \mathcal{U} \, \varphi$ and *globally*: $\Box \varphi = false \, \mathcal{R} \, \varphi$.

LTL formulas 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 [25]. Here we informally explain the meaning of the formulas we use.

Of special interest will be surveillance formulas of the form $\Box p_k$, termed *safety surveillance objective*, and $\Box \Diamond p_k$, called *liveness surveillance objective*. Intuitively, the safety surveillance formula $\Box p_k$ is satisfied if at each point in time the size of the belief set does not exceed k. The liveness surveillance objective $\Box \Diamond p_k$, on the other hand, requires that infinitely often this size is below or equal to k.

Example 3: We can specify that the agent is required to always know with certainty the location of the target as $\Box p_1$. A more relaxed requirement is that the agent's uncertainty never grows above 5 locations, and it infinitely often reduces this uncertainty to at most 2 locations: $\Box p_5 \wedge \Box \Diamond p_2$. ■

D. Incorporating Task Specifications

We can integrate LTL objectives not related to surveillance, i.e., *task specifications*, by considering, in addition to $\mathcal{SP}$, a set $\mathcal{AP}$ of atomic predicates interpreted over states of G. In order to define the semantics of $p \in \mathcal{AP}$ over states of G_{belief}, we restrict ourselves to predicates observable by the agent. Formally, we require that for $p \in \mathcal{AP}$, and states (l_a, l_t') and (l_a, l_t'') with $vis(l_a, l_t') = vis(l_a, l_t'') = \mathit{false}$ it holds that $(l_a, l_t') \models p$ iff $(l_a, l_t'') \models p$. One class of such predicates are those that depend only on the agent's position.

Example 4: Suppose that at_goal is a predicate true exactly when the agent is at some designated goal location. We can then state that the agent visits the goal infinitely often while always maintaining belief uncertainty of at most 10 locations using the LTL formula $\Box\Diamond\, at\_goal \wedge \Box p_{10}$. ■

E. Surveillance Synthesis Problem

A *surveillance game* is a pair (G, φ), where G is a surveillance game structure and φ is a surveillance objective. A *winning strategy for the agent for* (G, φ) is a strategy f_a for the agent in the corresponding belief-set game structure G_{belief} such that for every strategy f_t for the target in G_{belief} it holds that $outcome(G_{\mathsf{belief}}, f_a, f_t) \models \varphi$. Analogously, a *winning strategy for the target for* (G, φ) is a strategy f_t such that, for every strategy f_a for the agent in G_{belief}, it holds that $outcome(G_{\mathsf{belief}}, f_a, f_t) \not\models \varphi$.

Surveillance synthesis problem: Given a surveillance game (G, φ), compute a winning strategy for the agent for (G, φ), or determine that such a strategy does not exist.

It is well-known that two-player perfect-information games with LTL objectives over finite-state game structures are determined, that is exactly one of the players has a winning strategy [26]. This means that the agent does not have a winning strategy for a given surveillance game, if and only if the target has a winning strategy for this game. We refer to winning strategies of the target as *counterexamples*.

III. BELIEF SET ABSTRACTION

We used the belief-set game structure in order to state the surveillance objective of the agent. While in principle it is possible to solve the surveillance strategy synthesis problem on this game, it is in most cases computationally infeasible, since the size of this game is exponential in the size of the original game. To circumvent such a construction when possible, we propose an abstraction-based method, that given a surveillance game structure and a partition of the set of the target's locations, yields an approximation that is sound with respect to surveillance objectives for the agent.

An *abstraction partition* is a family $\mathcal{Q} = \{Q_i\}_{i=1}^n$ of subsets of L_t, $Q_i \subseteq L_t$ such that the following hold:

- $\bigcup_{i=1}^n Q_i = L_t$ and $Q_i \cap Q_j = \emptyset$ for all $i \neq j$.
- For each $p \in \mathcal{AP}$, $Q \in \mathcal{Q}$ and $l_a \in L_a$, it holds that $(l_a, l_t') \models p$ iff $(l_a, l_t'') \models p$ for every $l_t', l_t'' \in Q$.

Intuitively, these conditions mean that $\mathcal{Q}$ partitions the set of locations of the target, and the concrete locations in each of the sets in $\mathcal{Q}$ agree on the value of the propositions in $\mathcal{AP}$.

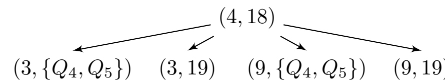


Fig. 3: Transitions from the initial state in the abstract game from Example 5 where $\alpha_{\mathcal{Q}}(17) = Q_4$ and $\alpha_{\mathcal{Q}}(23) = Q_5$.

For $\mathcal{Q} = \{Q_i\}_{i=1}^n$, we define a function $\alpha_{\mathcal{Q}} : L_t \to \mathcal{Q}$ by $\alpha_{\mathcal{Q}}(l_t) = Q$ for the unique $Q \in \mathcal{Q}$ with $l_t \in Q$. We denote also with $\alpha_{\mathcal{Q}} : \mathcal{P}(L_t) \to \mathcal{P}(\mathcal{Q})$ the *abstraction function* defined by $\alpha_{\mathcal{Q}}(L) = \{\alpha_{\mathcal{Q}}(l) \mid l \in L\}$. We define a *concretization function* $\gamma : \mathcal{P}(\mathcal{Q}) \cup L_t \to \mathcal{P}(L_t)$ such that $\gamma(l_t) = \{l_t\}$ for $l_t \in L_t$, and $\gamma(A) = \bigcup_{Q \in A} Q$ if $A \in \mathcal{P}(\mathcal{Q})$.

Intuitively, the abstraction of a set L of locations of the target is a set of elements of $\mathcal{Q}$ that cover L, and each of them has non-empty intersection with L. The concretization of a set of elements of $\mathcal{Q}$ is the set of locations of the target formed by the union of these sets. Thus, we have $\gamma(\alpha_{\mathcal{Q}}(L)) \supseteq L$, which means that the belief of the agent is overapproximated by the abstraction as desired.

Given a surveillance game structure $G = (S, s^{\mathsf{init}}, T, vis)$ and an abstraction partition $\mathcal{Q} = \{Q_i\}_{i=1}^n$ of the set L_t, we define the *abstraction of G w.r.t. $\mathcal{Q}$* to be the game structure $G_{\mathsf{abstract}} = \alpha_{\mathcal{Q}}(G) = (S_{\mathsf{abstract}}, s^{\mathsf{init}}_{\mathsf{abstract}}, T_{\mathsf{abstract}})$, where

- $S_{\mathsf{abstract}} = (L_a \times \mathcal{P}(\mathcal{Q})) \cup (L_a \times L_t)$ is the set of *abstract states*, consisting of states approximating the belief sets in the game structure G_{belief}, as well as the states S;
- $s^{\mathsf{init}}_{\mathsf{abstract}} = (l_a^{\mathsf{init}}, l_t^{\mathsf{init}})$ is the *initial abstract state*;
- $T_{\mathsf{abstract}} \subseteq S_{\mathsf{abstract}} \times S_{\mathsf{abstract}}$ is the transition relation such that $((l_a, A_t), (l_a', A_t')) \in T_{\mathsf{abstract}}$ if and only if one of the following two conditions is satisfied:
  (1) $A_t' = l_t'$, $l_t' \in succ_t(\gamma(A_t))$ and $vis(l_a, l_t') = \mathit{true}$, and $l_a' \in succ_a(l_a, l_t, l_t')$ for some $l_t \in \gamma(A_t)$.
  (2) $A_t' = \alpha_{\mathcal{Q}}(\{l_t' \in succ_t(\gamma(A_t)) | vis(l_a, l_t') = \mathit{false}\})$, and $l_a' \in succ_a(l_a, l_t, l_t')$ for some $l_t \in \gamma(A_t)$ and some $l_t' \in succ_t(\gamma(A_t))$ with $vis(l_a, l_t') = \mathit{false}$.

As for the belief-set game structure, the first condition captures the successor locations of the target, which can be observed from the agent's current location l_a. Condition (2) corresponds to the *abstract belief set* whose concretization consists of all possible successors of all positions in $\gamma(A_t)$, which are not visible from l_a. Since the belief abstraction overapproximates the agent's belief, that is, $\gamma(\alpha_{\mathcal{Q}}(B)) \supseteq B$, the next-state abstract belief $\gamma(A_t')$ may include positions in L_t that are not successors of positions in $\gamma(A_t)$.

Example 5: Consider again the surveillance game from Example 1, and the abstraction partition $\mathcal{Q} = \{Q_1, \ldots, Q_5\}$, where the set Q_i corresponds to the i-th row of the grid. For location 17 of the target we have $\alpha_{\mathcal{Q}}(17) = Q_4$, and for 23 we have $\alpha_{\mathcal{Q}}(23) = Q_5$. Thus, the belief set $B = \{17, 23\}$ is covered by the abstract belief set $\alpha_Q(B) = \{Q_4, Q_5\}$. Figure 3 shows the successors of the initial state $(4, 18)$ of the abstract belief-set game structure. The concretization of the abstract belief set $\{Q_4, Q_5\}$ is the set $\{15, 16, 17, 18, 19, 20, 21, 22, 23, 24\}$ of target locations. ■

An abstract state (l_a, A_t) *satisfies a surveillance predicate* p_k, denoted $(l_a, A_t) \models p_k$, iff $|\{l_t \in \gamma(A_t) \mid vis(l_a, l_t) = false\}| \leq k$. Simply, the number of states in the concretized belief set has to be less than or equal to k. Similarly, for a predicate $p \in \mathcal{AP}$, we define $(l_a, A_t) \models p$ iff for every $l_t \in \gamma(A_t)$ it holds that $(l_a, l_t) \models p$. With these definitions, we can interpret surveillance objectives over runs of G_{abstract}.

Strategies (and winning strategies) for the agent and the target in an abstract belief-set game $(\alpha_{\mathcal{Q}}(G), \varphi)$ are defined analogously to strategies (and winning strategies) in G_{belief}.

In the construction of the abstract game structure, we overapproximate the belief-set of the agent at each step. Since we consider surveillance predicates that impose upper bounds on the size of the belief, such an abstraction gives more power to the target (and dually less power to the agent). This construction guarantees, as stated in Theorem 1, that the abstraction is *sound*, meaning that an abstract strategy for the agent that achieves a surveillance objective corresponds to a winning strategy in the concrete game. Soundness follows from the fact that the abstract belief game simulates the concrete one. The following theorem formally states the soundness property of the abstraction.

Theorem 1: Let G be a surveillance game structure, $\mathcal{Q} = \{Q_i\}_{i=1}^n$ be an abstraction partition, and $G_{\mathsf{abstract}} = \alpha_{\mathcal{Q}}(G)$. For every surveillance objective φ, if there exists a winning strategy for the agent in the abstract belief-set game $(\alpha_{\mathcal{Q}}(G), \varphi)$, then there exists a winning strategy for the agent in the concrete surveillance game (G, φ).

IV. ABSTRACTION PRECISION

The ideal choice of an abstraction partition is the one that balances precision and computational burden. More precisely, the abstraction should be precise enough for the agent to satisfy its surveillance objective. On the other hand, an abstraction that is too precise, often results in an intractably large state space of the resulting game. Thus, a good abstraction is one that gives the right level of precision where it is needed, and is coarse (that is, generates fewer abstract belief states) where precision is not needed. Thus, choosing a good abstraction partition is often specific to the game environment and the surveillance specification. In section V, we present examples with user specified partitions resulting in feasible abstract games.

In the previous section, we discussed that a winning strategy for the agent in the abstract belief game corresponds to a strategy for the agent in the concrete belief game. This, fact does not hold in general for the abstract winning strategies of the target. We refer to the abstract winning strategies for the target as *abstract counterexamples*.

Given an abstract counterexample, there are two possibilities: it can either be a counterexample in the concrete belief game, meaning that the agent cannot satisfy the surveillance objective, or it may exist due to the coarseness of the abstraction partition. We now discuss in more detail counterexamples in safety and liveness surveillance games. The latter generalizes also to general surveillance objectives.

$(4, 18)$

$(3, \{Q_4, Q_5\})$ $(9, \{Q_4, Q_5\})$

(a) Abstract counterexample tree

$(4, 18)$

$(3, \{17, 23\})$ $(9, \{17, 23\})$

(b) Concrete counterexample tree

Fig. 4: Abstract and corresponding concrete counterexample trees for the surveillance game in Example 6.

A. Counterexamples for Safety Surveillance Properties

A winning strategy for the target in a game with safety surveillance objective can be represented as a tree. An *abstract counterexample tree* $\mathcal{C}_{\mathsf{abstract}}$ for $(G_{\mathsf{abstract}}, \Box p_k)$ is a finite tree, whose nodes are labelled with states in S_{abstract} such that the following conditions are satisfied:

- The root node is labelled with the initial state $s^{\mathsf{init}}_{\mathsf{abstract}}$.
- A node is labelled with an abstract state which violates p_k (that is, s_{abstract} where $s_{\mathsf{abstract}} \not\models p_k$) iff it is a leaf.
- The tree branches according to all possible transition choices of the agent. Formally, if an internal node v is labelled with (l_a, A_t), then there is unique A'_t such that: (1) $((l_a, A_t), (l'_a, A'_t)) \in T_{\mathsf{abstract}}$ for some $l'_a \in L_a$, and (2) for every $l'_a \in L_a$ such that $((l_a, A_t), (l'_a, A'_t)) \in T_{\mathsf{abstract}}$, there is a child v' of v labelled with (l'_a, A'_t).

A *concrete counterexample tree* $\mathcal{C}_{\mathsf{belief}}$ for $(G_{\mathsf{belief}}, \Box p_k)$ is a finite tree defined analogously to an abstract counterexample tree with nodes labelled with states in S_{belief}.

Due to the overapproximation of the belief sets, not every counterexample in the abstract game corresponds to a winning strategy for the target in the original game.

An abstract counterexample $\mathcal{C}_{\mathsf{abstract}}$ in $(G_{\mathsf{abstract}}, \Box p_k)$ is *concretizable* if there exists a concrete counterexample tree $\mathcal{C}_{\mathsf{belief}}$ in $(G_{\mathsf{belief}}, \Box p_k)$, that differs from $\mathcal{C}_{\mathsf{abstract}}$ only in the node labels, and each node labelled with (l_a, A_t) in $\mathcal{C}_{\mathsf{abstract}}$ has label (l_a, B_t) in $\mathcal{C}_{\mathsf{belief}}$ for which $B_t \subseteq \gamma(A_t)$.

Example 6: Figure 4a shows an abstract counterexample tree $\mathcal{C}_{\mathsf{abstract}}$ for the game $(\alpha_{\mathcal{Q}}(G), \Box p_1)$, where G is the surveillance game structure from Example 1 and $\mathcal{Q}$ is the abstraction partition from Example 5. The counterexample corresponds to the choice of the target to move to one of the locations 17 or 23, which, for every possible move of the agent, results in an abstract state with abstract belief $B = \{Q_4, Q_5\}$ violating p_1. A concrete counterexample tree $\mathcal{C}_{\mathsf{belief}}$ concretizing $\mathcal{C}_{\mathsf{abstract}}$ is shown in Figure 4b. ■

B. Counterexamples for Liveness Surveillance Properties

The counterexamples for general surveillance properties are directed graphs, which may contain cycles. In particular, for a liveness surveillance property of the form $\Box\Diamond p_k$ each infinite path in the graph has a position such that, from this position on, each state on the path violates p_k. An *abstract counterexample graph* in the game $(G_{\mathsf{abstract}}, \Box\Diamond p_k)$ is a finite graph $\mathcal{C}_{\mathsf{abstract}}$ defined analogously to the abstract counterexample tree. The difference is that there are no leaves, and that for each cycle $\rho = v_1, v_2, \ldots, v_n$ with $v_1 = v_n$ in $\mathcal{C}_{\mathsf{abstract}}$ that is reachable from v_0, every node v_i in ρ is labelled with state s^i_{abstract} where $s^i_{\mathsf{abstract}} \not\models p_k$.

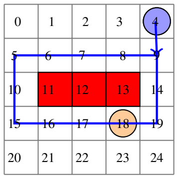


Fig. 5: Agent locations on an (infinite) path in the abstract counterexample graph from Example 7. In the graph, the first node is labelled with $(4, 18)$, the second with $(9, \{Q_2\})$, and all other nodes with some $(l_a, \{Q_1, Q_2\})$.

Example 7: We saw in Example 6 that in the safety surveillance game $(G, \Box p_1)$ the agent does not have a winning strategy. We now consider a relaxed requirement, namely, that the uncertainty drops to at most 1 infinitely often. We consider the liveness surveillance game $(G, \Box\Diamond p_2)$.

Let $\mathcal{Q} = \{Q_1, Q_2\}$ be a partition such that Q_1, corresponds to the first two columns of the grid in Figure 1a and the set Q_2 contains the locations from the other three columns of the grid. Figure 5 shows an infinite path (in lasso form) in the abstract game $(\alpha_{\mathcal{Q}}(G), \Box\Diamond p_2)$. The figure depicts only the corresponding trajectory (sequence of positions) of the agent. The initial abstract state is $(4, 18)$, the second node on the path is labeled with the abstract state $(9, \{Q_2\})$, and all other nodes on the path are labeled with abstract states of the form $(l_a, \{Q_1, Q_2\})$. As each abstract state in the cycle violates p_2, the path violates $\Box\Diamond p_2$. The same holds for all infinite paths in the existing abstract counterexample graph. ∎

A *concrete counterexample graph* $\mathcal{C}_{\mathsf{belief}}$ for the belief game $(G_{\mathsf{belief}}, \Box\Diamond p_k)$ is defined analogously.

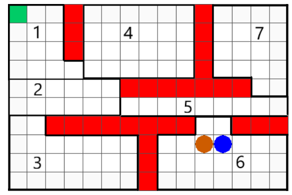
An abstract counterexample graph $\mathcal{C}_{\mathsf{abstract}}$ for the game $(G_{\mathsf{abstract}}, \Box\Diamond p_k)$ is *concretizable* if there exists a counterexample $\mathcal{C}_{\mathsf{belief}}$ in $(G_{\mathsf{belief}}, \Box\Diamond p_k)$, such that for each infinite path $\pi_{\mathsf{abstract}} = v^0_{\mathsf{abstract}}, v^1_{\mathsf{abstract}}, \ldots$ starting from the initial node of $\mathcal{C}_{\mathsf{abstract}}$ there exists an infinite path $\pi_{\mathsf{belief}} = v^0_{\mathsf{belief}}, v^1_{\mathsf{belief}}, \ldots$ in $\mathcal{C}_{\mathsf{belief}}$ starting from its initial node. Furthermore, if v^i_{abstract} is labelled with (l_a, A_t) in $\mathcal{C}_{\mathsf{abstract}}$, then the corresponding node v^i_{belief} in $\mathcal{C}_{\mathsf{belief}}$ is labelled with (l_a, B_t) for some $B_t \in \mathcal{P}(L_t)$ for which $B_t \subseteq \gamma(A_t)$.

C. Counterexample-Guided Refinement

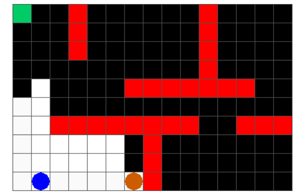
Since the game structures we consider are finite, and counterexamples have finite representation, we can effectively determine whether an abstract counterexample is concretizable or not. In the first case we report the unrealizability of the surveillance objective. In the second case, the counterexample can be used to determine which parts of the abstraction need to be refined in order to eliminate this, and possibly other, counterexamples. This analysis and refinement procedure can be automated, and we refer the reader to [12] for more details on the process. We remark that even when the refinement is automated, the choice of the initial abstraction often plays a crucial role in keeping the size of the abstract game within feasible limits.

V. EXPERIMENTAL EVALUATION

We report on the application of our approach for surveillance synthesis in two case studies. We have implemented the proposed method in `Python`, using the `slugs` reactive

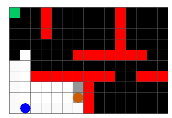


(a) Gridworld with a user provided abstraction partition with 7 sets, marked by black lines.

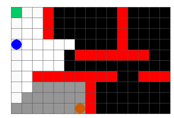


(b) Gridworld showing visibility of the agent. All locations shown in black are invisible to the agent.

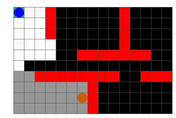
Fig. 6: 10x15 gridworld with a surveillance liveness specification. The agent is blue, and the target to be surveilled is orange. Red states are obstacles.



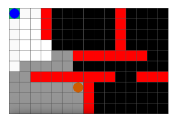
(a) t_1



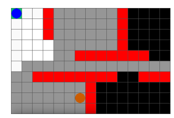
(b) t_3



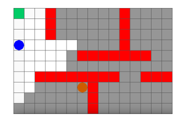
(c) t_4



(d) t_5



(e) t_6



(f) t_7

Fig. 7: Evolution of the agent's belief about the target's location as it moves to the goal and loses sight of the target. Grey cells represent the locations the agent believes the target could be in. We show the belief at different timesteps $t_1, \ldots, t_7$ (note that t_2 is excluded for simplicity).

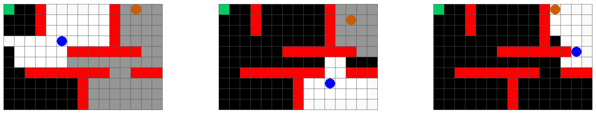
synthesis tool [10]. The experiments were performed on an Intel i5-5300U 2.30 GHz CPU with 8 GB of RAM.

A. Liveness Surveillance Specification + Task Specification

Figure 6a shows a gridworld divided into regions. The surveillance objective requires the agent to infinitely often know precisely the location of the target (either see it, or have a belief consisting of one cell). Additionally, it has to perform the task of patrolling (visiting infinitely often) the green 'goal' cell. Formally, the specification is $\Box\Diamond p_1 \wedge \Box\Diamond\, \mathit{goal}$. The agent can move up to 3 grid cells away at each step, and the target can move 1 cell at each step. The sensor mode, that is, the visibility function, used here is 'line-of-sight' with a range of 5 cells. The agent cannot see through obstacles (shown in red) and cannot see farther than 5 cells away.

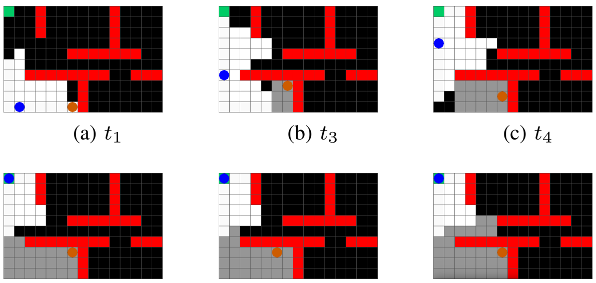
Using the abstraction partition of size 7 shown in Figure 6a, the overall number of abstract belief states is $15 \times 10 + 2^7 = 278$ states. In contrast, solving the full belief game will have in the order of 2^{150} states, which is a state-space size that state-of-the-art synthesis tools cannot handle.

Figure 7 shows how the belief of the agent (shown in grey) can grow quickly when it cannot see the target. This growth occurs due to the coarseness of the abstraction, which overapproximates the target's true position. In 7 steps, the agent believes the target can be anywhere in the grid that



(a) t_5 (b) t_7 (c) t_9

Fig. 8: The agent has to search for the target in order to lower its belief below the surveillance liveness specification.



(a) t_1 (b) t_3 (c) t_4

(d) t_5 (e) t_6 (f) t_7

Fig. 9: Evolution of the agent's belief about the target's location in a game with an abstraction partition of size 12.

is not in its vision. It has to then find the target in order to satisfy the surveillance requirement. Figure 8 illustrates the searching behaviour of the agent when it is trying to lower the belief below the threshold in order to satisfy the liveness specification. The behaviour of the agent shown here will contrast with the behaviour under safety surveillance which will we look at next.

In this example, an abstraction partition of size 7 was enough to guarantee the satisfaction surveillance specification. For the purpose of comparison, we also solve the game with an abstraction partition of size 12 to illustrate the change in belief growth. Figure 9 shows the belief states growing much more slowly as the abstract belief states are smaller, and thus they more closely approximate the true belief of the agent.

The additional abstraction partitions result in a much larger game as the state space grows exponentially in the size of the abstraction partition. Table I compares the sizes of the corresponding abstract games, and the time it takes to synthesize a surveillance controller in each case.

Size of abstraction partition	Size of abstract game	Synthesis time
7	278	237s
12	4346	810s

TABLE I: Comparison of synthesis times for the two cases

A video simulation of the synthesized surveillance strategy against a target controlled by a human can be found at `http://goo.gl/YkFuxr`.

B. *Safety Surveillance Specification + Task Specification*

Figure 10 depicts an environment created in *Gazebo* where the red blocks model buildings. The drone is given full line

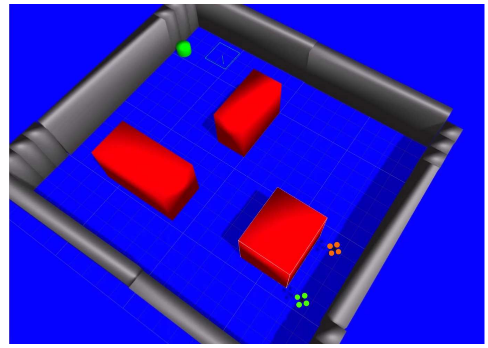


Fig. 10: A Gazebo environment where the red blocks are obstacles that the drones cannot see past. The green drone is the agent and the orange drone is the target.

of sight vision - it can detect the target if there is no obstacle in the way.

In this setting, we enforce the safety surveillance objective $\square p_{30}$ (the belief size should never exceed 30) in addition to infinitely often reaching the green cell. The formal specification is $\square p_{30} \wedge \square\diamond \mathit{goal}$. Additionally, the target itself is trying to reach the goal cell infinitely often as well, which is known to the agent.

We used an abstraction generated by a partition of size 6, which was sufficiently precise to compute a surveillance strategy in 210 s. Again, note that the precise belief-set game would have in the order of 2^{200} states.

We simulated the environment and the synthesized surveillance strategy for the agent in Gazebo and ROS. In the simulation, the target is being controlled by a human while the agent responds using the synthesized surveillance strategy. A video of the simulation can be found at `http://goo.gl/LyC1gQ`. This simulation shows a qualitative difference in behaviour compared to the previous example. There, in the case of liveness surveillance, the agent had more leeway to completely lose the target in order to reach its goal location, even though the requirement of reducing the size of the belief to 1 is quite strict. Here, on the other hand, the safety surveillance objective, even with a large threshold of 30, forces the agent to follow the target more closely, in order to prevent its belief from getting too large. The synthesis algorithm thus provides the ability to obtain qualitatively different behaviour as necessary for specific applications by combining different objectives.

C. *Discussion*

The difference in the behaviour in the case studies highlights the different use cases of the surveillance objectives. Depending on the domain, the user can specify a combination of safety and liveness specification to tune the behaviour of the agent. In a critical surveillance situation (typical in defense or security situations), the safety specification will guarantee to the user that the belief will never grow too large. However, in less critical situations (such as luggage carrying robots in airports), the robot has more flexibility in allowing the belief to grow as long as it can guarantee its reduction in the future.

VI. CONCLUSIONS

We have presented a novel approach to solving a surveillance problem with information guarantees. We provided a framework that enables the formalization of the surveillance synthesis problem as a two-player, partial-information game. We then presented a method to reason over the belief that the agent has over the target's location, which allows for specifying and enforcing surveillance requirements. The user can tailor the behaviour to their specific application by using a combination of safety and liveness surveillance objectives.

The benefit of the proposed framework is that it allows us to employ techniques successfully used in verification and reactive synthesis to develop efficient methods for solving the surveillance problem. There are several promising avenues of future work using and extending this framework. Some of the directions currently being explored are the following:

- Synthesizing distributed strategies for multi-agent surveillance in a decentralized manner. Compositional synthesis methods can be used to avoid the blow-up of the state space that occurs in centralized synthesis procedures as the number of surveillance agents grows.
- Incorporating static sensors or alarm triggers for the mobile agent(s) to coordinate with.
- Allowing for sensor models to include uncertainty and detection errors while still providing surveillance guarantees.

Acknowledgement: This work was supported in part by grant Sandia National Lab 801KOB, grant ARO W911NF-15-1-0592, and grant DARPA W911NF-16-1-0001.

REFERENCES

[1] T. Bandyopadhyay, M. H. Ang, and D. Hsu, *Motion Planning for 3-D Target Tracking among Obstacles*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 267–279.

[2] H. H. Gonzalez-Banos, C.-Y. Lee, and J. C. Latombe, "Real-time combinatorial tracking of a target moving unpredictably among obstacles," in *Proc. ICRA 2002*, vol. 2, May 2002, pp. 1683–1690 vol.2.

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

[4] 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.

[5] S. Kreutzer, *Graph Searching Games*. Cambridge University Press, 2011, pp. 213–261.

[6] L. Doyen and J. Raskin, *Games with Imperfect Information: Theory and Algorithms*. Cambridge University Press, 2011, pp. 185–212.

[7] K. Chatterjee, L. Doyen, and T. A. Henzinger, "A survey of partial-observation stochastic parity games," *Formal Methods in System Design*, vol. 43, no. 2, pp. 268–284, Oct 2013.

[8] J. H. Reif, "The complexity of two-player games of incomplete information," *J. Comput. Syst. Sci.*, vol. 29, no. 2, pp. 274–301, 1984.

[9] D. Berwanger and L. Doyen, "On the power of imperfect information," in *Proc. FSTTCS 2008*, ser. LIPIcs, vol. 2, 2008, pp. 73–82.

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

[11] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement," in *Proc. CAV 2000*, ser. LNCS, vol. 1855. Springer, 2000, pp. 154–169.

[12] S. Bharadwaj, R. Dimitrova, and U. Topcu, "Synthesis of surveillance strategies via belief abstraction," 09 2017.

[13] J.-C. Chin, Y. Dong, W.-K. Hon, C. Y.-T. Ma, and D. K. Y. Yau, "Detection of intelligent mobile target in a mobile sensor network," *IEEE/ACM Trans. Netw.*, vol. 18, no. 1, pp. 41–52, Feb. 2010.

[14] A. Antoniades, H. J. Kim, and S. Sastry, "Pursuit-evasion strategies for teams of multiple agents with incomplete information," in *Proc. CDC 2003*, vol. 1, Dec 2003, pp. 756–761 Vol.1.

[15] R. Vidal, O. Shakernia, H. J. Kim, D. H. Shim, and S. Sastry, "Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 662–669, Oct 2002.

[16] H. J. Kim, R. Vidal, D. H. Shim, O. Shakernia, and S. Sastry, "A hierarchical approach to probabilistic pursuit-evasion games with unmanned ground and aerial vehicles," in *Proc. CDC 2001*, vol. 1, 2001, pp. 634–639 vol.1.

[17] A. Pnueli and R. Rosner, "On the synthesis of a reactive module," in *Proc. POPL'89*. ACM, 1989, pp. 179–190.

[18] N. Piterman, A. Pnueli, and Y. Sa'ar, *Synthesis of Reactive(1) Designs*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 364–380.

[19] T. Wongpiromsarn, U. Topcu, and R. M. Murray, "Receding horizon temporal logic planning," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2817–2830, Nov 2012.

[20] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "From structured english to robot motion," in *Proc. IROS 2007*, Oct 2007, pp. 2717–2722.

[21] R. van der Meyden and M. Y. Vardi, "Synthesis from knowledge-based specifications (extended abstract)," in *Proc. CONCUR'98*, ser. LNCS, vol. 1466. Springer, 1998, pp. 34–49.

[22] T. A. Henzinger, R. Jhala, and R. Majumdar, "Counterexample-guided control," in *Proc. ICALP 2003*, 2003, pp. 886–902.

[23] R. Dimitrova and B. Finkbeiner, "Abstraction refinement for games with incomplete information," in *Proc. FSTTCS 2008*, ser. LIPIcs, vol. 2, 2008, pp. 175–186.

[24] J. Fu, R. Dimitrova, and U. Topcu, "Abstractions and sensor design in partial-information, reactive controller synthesis," in *Proc. ACC 2014*. IEEE, 2014, pp. 2297–2304.

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

[26] D. A. Martin, "Borel determinacy," *Annals of Mathematics*, vol. 102, no. 2, pp. 363–371, 1975.