Attacking the V: On the Resiliency of Adaptive-Horizon MPC

Ashish Tiwari[1], Scott A. Smolka[2], Lukas Esterle[3], Anna Lukina[3], Junxing Yang[2], and Radu Grosu[2,3]

[1] SRI International, USA
[2] Department of Computer Science, Stony Brook University, USA
[3] Cyber-Physical Systems Group, Technische Universität Wien, Austria

Abstract. Inspired by the emerging problem of CPS security, we introduce the concept of *controller-attacker games*. A controller-attacker game is a two-player stochastic game, where the two players, a controller and an attacker, have antagonistic objectives. A controller-attacker game is formulated in terms of a Markov Decision Process (MDP), with the controller and the attacker jointly determining the MDP's transition probabilities. We also introduce the class of controller-attacker games we call *V-formation games*, where the goal of the controller is to maneuver the plant (a simple model of flocking dynamics) into a V-formation, and the goal of the attacker is to prevent the controller from doing so. Controllers in V-formation games utilize a new formulation of model-predictive control we have developed called *Adaptive-Horizon MPC* (AMPC), giving them extraordinary power: we prove that under certain controllability conditions, an AMPC controller can attain V-formation with probability 1. We evaluate AMPC's performance on V-formation games using statistical model checking. Our experiments demonstrate that (a) as we increase the power of the attacker, the AMPC controller adapts by suitably increasing its horizon, and thus demonstrates resiliency to a variety of attacks; and (b) an intelligent attacker can significantly outperform its naive counterpart.

1 Introduction

Many Cyber-Physical Systems (CPSs) are highly distributed in nature, comprising a multitude of computing agents that can collectively exhibit *emergent behavior*. A compelling example of such a distributed CPS is the *drone swarm*, which are beginning to see increasing application in battlefield surveillance and reconnaisance [3]. The emergent behavior they exhibit is that of *flight formation*.

A particularly interesting form of flight formation is *V-formation*, especially for long-range missions where energy conservation is key. V-formation is emblematic of migratory birds such as Canada geese, where a bird flying in the *upwash region* of the bird in front of it can enjoy significant energy savings. The V-formation also offers a *clear view* benefit, as no bird's field of vision is obstructed by another bird in the formation. Because of the V-formation's intrinsic appeal, it is important to quantify the resiliency of the control algorithms

underlying this class of multi-agent CPSs to various kinds of cyber-attacks. This question provides the motivation for the investigation put forth in this paper.

Problem Statement and Summary of Results. Inspired by the emerging problem of CPS security, we introduce the concept of *controller-attacker games*. A controller-attacker game is a two-player stochastic game, where the two players, a controller and an attacker, have antagonistic objectives. A controller-attacker game is formulated in terms of a Markov Decision Process (MDP), with the controller and the attacker jointly determining the MDP's transition probabilities.

We also introduce a class of controller-attacker games we call V-formation games, where the goal of the controller is to maneuver the plant (a simple model of flocking dynamics) into a V-formation, and the goal of the attacker is to prevent the controller from doing so. Controllers in V-formation games utilize a new formulation of model-predictive control we have developed called *Adaptive-Horizon MPC* (AMPC), giving them extraordinary power: we prove that under certain controllability conditions, an AMPC controller can attain V-formation with probability 1.

We define several classes of attackers, including those that in one move can remove a small number R of birds from the flock, or introduce random displacement (perturbation) into the flock dynamics, again by selecting a small number of victim agents. We consider both *naive attackers*, whose strategies are purely probabilistic, and *AMPC-enabled attackers*, putting them on par strategically with the controller. The architecture of a V-formation game with an AMPC-enabled attacker is shown in Figure 1.

While an AMPC-enabled controller is expected to win every game with probability 1, in practice, it is *resource-constrained*: its maximum prediction horizon and the maximum number of execution steps are fixed in advance. Under these conditions, an attacker has a much better chance of winning a V-formation game.

AMPC is a key contribution of the work presented in this paper. Traditional MPC uses a fixed *prediction horizon* to determine the optimal control action. The AMPC procedure chooses the prediction horizon dynamically. Thus, AMPC can adapt to the severity of the action played by its adversary by choosing its own horizon accordingly. While the concept of MPC with an adaptive horizon has been investigated before [5, 9], our approach for choosing the prediction horizon based on the progress toward a fitness goal is entirely novel, and has a more general appeal compared to previous work.

In recent work [10], we presented a procedure for synthesizing plans (sequences of actions) that take an MDP to a desired set of states (defining a V-formation). The procedure adaptively varied the settings of various parameters of an underlying optimization routine. Since we did not consider any adversary or noise in [10], there was no need for a control algorithm. Here we consider V-formation in the presence of attacks, and hence we develop a generic adaptive control procedure, AMPC, and evaluate its resilience to attacks.

Our extensive performance evaluation of V-formation games uses statistical model checking to estimate the probability that an attacker can thwart the controller. Our results show that for the bird-removal game with 1 bird being

removed, the controller almost always wins (restores the flock to a V-formation). When 2 birds are removed, the game outcome critically depends on which two birds are removed. For the displacement game, our results again demonstrate that an intelligent attacker, i.e. one that uses AMPC in this case, significantly outperforms its naive counterpart that randomly carries out its attack.

Traditional feedback control is, by design, resilient to noise, and also certain kinds of attacks; as our results show, however, it may not be resilient against smart attacks. Adaptive-horizon control helps to guard against a larger class of attacks, but it can still falter due to limited resources. Our results also demonstrate that statistical model checking represents a promising approach toward the evaluation of CPS resilience against a wide range of attacks.

2 V-Formation

We consider the problem of bringing a flock of birds from a random initial configuration to an organized V-formation. Recently, Lukina et al. [10] have modeled this problem as a deterministic Markov Decision Process (MDP) $\mathcal{M}$, where the goal was to generate actions that caused $\mathcal{M}$ to reach a desired state. In our case $\mathcal{M}$ is an MDP as actions taken lead to probability distributions over the states. The definition of $\mathcal{M}$ is given in Section 3. In this section, we present a simple model of flocking dynamics that forms the basis of this definition.

In our flocking model, each bird in the flock is modeled using 4 variables: a 2-dimensional vector $\boldsymbol{x}$ denoting the position of the bird in a 2D space, and a 2-dimensional vector $\boldsymbol{v}$ denoting the velocity of the bird. We use $s = \{\boldsymbol{x}_i, \boldsymbol{v}_i\}_{i=1}^{B}$ to denote a state of a flock with B birds. The *control actions* of each bird are 2-dimensional accelerations $\boldsymbol{a}$ and 2-dimensional position displacements $\boldsymbol{d}$ (see discussion of $\boldsymbol{a}$ and $\boldsymbol{d}$ below). Both are random variables.

Let $\boldsymbol{x}_i(t), \boldsymbol{v}_i(t), \boldsymbol{a}_i(t)$, and $\boldsymbol{d}_i(t)$ respectively denote the position, velocity, acceleration, and displacement of the i-th bird at time t, $1 \leqslant i \leqslant B$. The behavior of bird i in discrete time is modeled as follows:

$$\begin{aligned} \boldsymbol{x}_i(t+1) &= \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1) + \boldsymbol{d}_i(t) \\ \boldsymbol{v}_i(t+1) &= \boldsymbol{v}_i(t) + \boldsymbol{a}_i(t) \end{aligned} \tag{1}$$

The next state of the flock is jointly determined by the accelerations and the displacements based on the current state following Eq. 1.

The problem of whether we can go from a random flock to a V-formation can be posed as a reachability question, where the reachability goal is the set of states representing a V-formation. A key assumption in [10] was that the reachability goal can be specified as $J(s) \leqslant \varphi$, where J is a fitness function that assigns a non-negative real (fitness) value to each state s, and φ is a small positive constant.

The fitness of a state was determined by the following three terms:

– *Clear View.* A bird's visual field is a cone with angle θ that can be blocked by the wings of other birds. The clear-view metric is defined by accumulating

the percentage of a bird's visual field that is blocked by other birds. $CV(s)$ for flock state s is the sum of the clear-view metric of all birds. The minimum value of CV is $CV^* = 0$, and this value is attained in a perfect V-formation where all birds have clear view.

- *Velocity Matching.* $VM(s)$ for flock state s is defined as the difference between the velocity of a given bird and all other birds, summed up over all birds in the flock. The minimum value for VM is $VM^* = 0$, and this value is attained in a perfect V-formation where all birds have the same velocity.
- *Upwash Benefit.* The trailing upwash is generated near the wingtips of a bird, while downwash is generated near the center of a bird. An upwash measure um is defined on the 2D space using a Gaussian-like model that peaks at the appropriate upwash and downwash regions. For bird i with upwash um_i, the upwash-benefit metric UB_i is defined as $1 - um_i$, and $UB(s)$ for flock state s is the sum of all UB_i for $1 \leqslant i \leqslant B$. The upwash benefit $UB(s)$ in V-formation is $UB^* = 1$, as all birds, except for the leader, have minimum upwash-benefit metric ($UB_i = 0, um_i = 1$), while the leader has upwash-benefit metric of 1 ($UB_i = 1, um_i = 0$).

Given the above metrics, the overall fitness (cost) metric J is of a sum-of-squares combination of VM, CV, and UB defined as follows:

$$J(s) = (CV(s) - CV^*)^2 + (VM(s) - VM^*)^2 + (UB(s) - UB^*)^2. \qquad (2)$$

A state s^* is considered to be a V-formation whenever $J(s^*) \leqslant \varphi$, for a small positive threshold φ.

3 Controller-Attacker Games

We are interested in games between a controller and an attacker, where the goal of the controller is to take the system to a desired set of states, and the goal of the attacker is to keep the system outside these states. We formulate our problem in terms of Markov Decision Processes for which the controller and the attacker jointly determine the transition probabilities.

Definition 1. *A **Markov Decision Process** (MDP) $\mathcal{M} = (S, A, T, J, I)$ is a 5-tuple consisting of a set S of* states*, set A of* actions*,* transition function $T : S \times A \times S \mapsto [0, 1]$*, where $T(s, a, s')$ is the probability of transitioning from state s to state s' under action a,* cost function $J : S \mapsto \mathbb{R}$*, where $J(s)$ is the cost associated with state s, and I is the* initial state distribution*.*

Our definition of an MDP differs from the traditional one in that it uses a cost function instead of a reward function. We find this definition more convenient for our purposes. Our focus is on continuous-space MDPs; i.e., the state space S is $\mathbb{R}^n$ and the action space A is in $\mathbb{R}^m$. For the bird-flocking problem, $n = m = 4B$, where B is the number of birds. We have four state variables and four action variables for each bird. The state variables represent the x- and the y-components

of the position $\boldsymbol{x}_i$ and velocity $\boldsymbol{v}_i$ of each bird i, whereas the action variables represent the (x- and y-components of the) acceleration $\boldsymbol{a}_i$ and displacement $\boldsymbol{d}_i$ of each bird i. The transition relation for the bird-flocking MDP is given by Eq. 1.

A *randomized strategy* over an MDP is a mapping taking every state s to a probability distribution $P(a \mid s)$ over the (available) actions. We formally define randomized strategies as follows.

Definition 2. *Let $\mathcal{M} = (S, A, T, J, I)$ be an MDP. A **randomized strategy** σ over $\mathcal{M}$ is a function of the form $\sigma : S \mapsto PD(A)$, where $PD(A)$ is the set of probability distributions over A. That is, σ takes a state s and returns an action consistent with the probability distribution $\sigma(s)$.*

A *controller-attacker game* is a stochastic game [18], where the transition probability from state s to state s' is controlled jointly by two players, a controller and an attacker in our case. To view an MDP as a stochastic game, we assume that the set of actions A is given as a product $C \times D$, where the controller chooses the C-component of an action $\boldsymbol{a}$ and the attacker chooses the D-component of $\boldsymbol{a}$. We assume that the game is played in parallel by the controller and the attacker; i.e., they both take the state $s(t) \in S$ of the system at time t, compute their respective actions $c(t) \in C$ and $d(t) \in D$, and then use the composed action $(c(t), d(t))$ to determine the next state $s(t+1) \in S$ of the system (based on the transition function T). We formally define a controller-attacker game as follows.

Definition 3. *A **controller-attacker game** is an MDP $\mathcal{M} = (S, A, T, J, I)$ with $A = C \times D$, where C and D are action sets of the controller and the attacker, respectively. The transition probability $T(s, c \times d, s')$ is jointly determined by actions $c \in C$ and $d \in D$.*

The actions of the controller and the attacker are determined by their randomized strategies. Once we fix a randomized strategy for the controller, and the attacker, the MDP reduces to a Markov chain on the state space S. Thus, the controller and the attacker jointly fix the probability of transitioning from a state s to a state s'. We refer to the underlying Markov chain induced by σ over $\mathcal{M}$ as $\mathcal{M}_\sigma$.

We define controller-attacker games on the flocking model by considering the scenario where the accelerations are under the control of one agent (the controller), and the displacements (position perturbations) are under the control of the second malicious agent (the attacker).

Definition 4. *A **V-formation game** is a controller-attacker game $\mathcal{M} = (S, A, T, J, I)$, where $S = \{s \mid s = \{\boldsymbol{x}_i, \boldsymbol{v}_i\}_{i=1}^{B}\}$ is the set of states for a flock of B birds, $A = C \times D$ with the controller choosing accelerations $\boldsymbol{a} \in C$ and the attacker choosing displacements $\boldsymbol{d} \in D$, T and J are given in Eq. 1 and 2, respectively.*

In this paper, we consider *reachability games* only. In particular, we are given a set G of *goal states* and the goal of the controller is to reach a state in G. Let

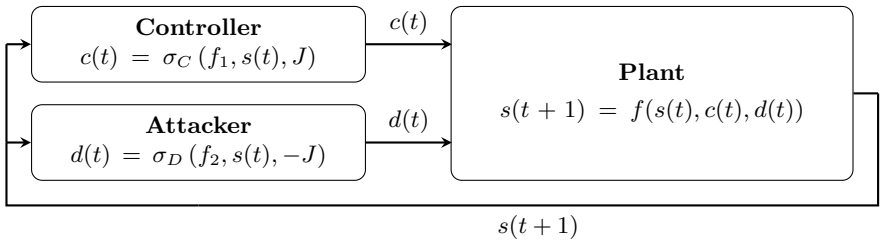

Fig. 1. Controller-Attacker Game Architecture. The controller and the attacker use randomized strategies σ_C and σ_D to choose actions $c(t)$ and $d(t)$ based on dynamics, respectively, where $s(t)$ is the state at time t, and f is the dynamics of the plant model. The controller tries to minimize the cost J, while the attacker tries to maximize it.

$s_0 \to s_1 \to s_2 \to \cdots$ be a sequence of states (a run of the system). The controller wins on this run if $\exists i : s_i \in G$, and the attacker wins otherwise.

A classical problem in the study of games pertains to determining the existence of an optimal winning strategy (e.g. a Nash equilibrium) for a player. We are *not* concerned with such problems in this paper. Due to the uncountably many states in the state- and action-space, solving such problems for our games of interest is extremely challenging. Instead, we focus on the problem of determining the likely winner of a game where the strategy of the two players is fixed. Since we consider randomized strategies, determining the likely winner is a statistical model checking problem, which allows us to evaluate the resilience of certain controllers under certain attack models. We are now ready to formally define the problem we would like to solve.

Definition 5. *Let $\mathcal{M} = (S, A, T, J, I)$ be an MDP, where $A = C \times D$, and let $\sigma_C : S \mapsto PD(C)$ and $\sigma_D : S \mapsto PD(D)$ be randomized strategies over M. Also, let $G \subseteq S$ be the set of* goal states *of $\mathcal{M}$. The* stochastic game verification problem *is to determine the probability of reaching a state in G in m steps, for a given m, starting from an initial state in $\mathcal{M}_{(\sigma_C, \sigma_D)}$.*

Fig. 1 shows the architecture of a stochastic game between the controller and the attacker, where at each time step the controller chooses action $c(t)$ as the C-component using strategy σ_C, and the attacker chooses action $d(t)$ as the D-component using strategy σ_D. The next state of the plant is determined by the composed action $(c(t), d(t))$ based on the current state $s(t)$ and the dynamics of the plant model f.

Our main interest is in evaluating the resilience of a control algorithm σ_C (a controller can be viewed as a strategy in our framework) to an attack algorithm σ_D. The key assumption that the controller and the attacker make is the existence of a *cost function* $J : S \mapsto \mathbb{R}^+$ such that

$$G := \{s \mid J(s) \leqslant \varphi \text{ for some very small } \varphi > 0\}.$$

Given such a cost function J, the controller works by minimizing the cost of states reachable in one or more steps, as is done in model predictive control

(MPC). Since the cost function is highly nonlinear, the controller uses an optimization procedure based on randomization to search for a minimum. Hence, our controller is a randomized procedure. One possible attack strategy we consider (for an advanced attacker) is based on the cost function as well: the attacker tries to maximize the cost of reachable states.

4 The Adaptive-Horizon MPC Algorithm

We now present our new *adaptive-horizon* MPC algorithm we call AMPC. We will use this algorithm as the controller strategy in the stochastic games we play on MDPs. We will also consider attacker strategies that use AMPC. AMPC is an MPC procedure based on particle-swarm optimization (PSO) [8]. The MPC approach can be used for achieving a V-formation, as was outlined in [19, 20]. These earlier works, however, did not use an adaptive dynamic window, and did not consider the adversarial control problem.

The main algorithm of AMPC performs step-by-step control of a given MDP $\mathcal{M}$ by looking h steps ahead—i.e. it uses a *prediction horizon* of length h—to determine the next optimal control action to apply. We use PSO to solve the optimization problem generated by the MPC procedure.

For V-formation, define the cost of $\boldsymbol{a}^h$ as the minimum cost J (Eq. 2) obtained within h steps by applying the sequence $\boldsymbol{a}^h$ of h accelerations on $\mathcal{M}$. Formally, we have

$$\texttt{Cost}(\mathcal{M}, \boldsymbol{a}^h, h) = \min_{1 \leqslant \tau \leqslant h} J(s^{\tau}_{\boldsymbol{a}^h}) \tag{3}$$

where $s^{\tau}_{\boldsymbol{a}^h}$ is the state after applying the τ-th action of $\boldsymbol{a}^h$ to the initial state of $\mathcal{M}$.[4] For horizon h, PSO searches for the best sequence of 2-dimensional acceleration vectors of length h, thus having $2hB$ parameters to be optimized. The number of particles p used in PSO is proportional to the number of parameters to be optimized, i.e., $p = 2\beta hB$, where β is a preset constant.

The AMPC procedure is given in Algorithm 1. A novel feature of AMPC is that, unlike classical MPC which uses a fixed horizon h, AMPC adaptively chooses an h depending on whether it is able to reach a cost that is lower than the current cost by our chosen quanta Δ_i, $0 \leqslant i \leqslant m$, for m steps.

AMPC is hence an adaptive MPC procedure that uses level-based horizons. It employs PSO to identify the potentially best next actions. If the actions $\boldsymbol{a}^h$ improve (decrease) the cost of the state reached within h steps, namely $\texttt{Cost}(\mathcal{M}, \boldsymbol{a}^h, h)$, by the predefined Δ_i, the controller considers these actions to be worthy of leading the flock towards, or keeping it in, a V-formation.[5]

In this case, the controller applies the first action to each bird and transitions to the next state of the MDP. The threshold Δ_i determines the next level

[4] The initial state of $\mathcal{M}$ is being used to store the "current" state of the MDP as we execute our algorithm.

[5] We focus our attention on bird flocking, since the details generalize naturally to other MDPs that come with a cost function.

Algorithm 1: AMPC: Adaptive-Horizon Model Predictive Control

Input : $\mathcal{M}, \varphi, h_{max}, m, B, \beta, \texttt{Cost}$
Output: $\{\boldsymbol{a}^i\}_{1 \leqslant i \leqslant m}$ *// optimal control sequence*

Initialize $\ell_0 \leftarrow J(s_0)$; $\widehat{J} \leftarrow \inf$; $i \leftarrow 1$; $h \leftarrow 1$; $p \leftarrow 2\beta hB$; $\Delta_0 \leftarrow (\ell_0 - \varphi)/m$;
while $(\ell_{i-1} > \varphi) \wedge (i < m)$ **do**
  // find and apply first best action out of the horizon sequence of length h
  $[\boldsymbol{a}^h, \widehat{J}] \leftarrow \texttt{particleswarm(Cost}, \mathcal{M}, p, h\texttt{)}$;
  if $\ell_{i-1} - \widehat{J} > \Delta_i \vee h = h_{max}$ **then**
    // if a new level or the maximum horizon is reached
    $\boldsymbol{a}^i \leftarrow \boldsymbol{a}^h_1$; $\mathcal{M} \leftarrow \mathcal{M}^{\boldsymbol{a}^i}$; *// apply the action and move to the next state*
    $\ell_i \leftarrow J(s(\mathcal{M}))$; *// update ℓ_i with the cost of the current state*
    $\Delta_i \leftarrow \ell_i/(m-i)$; *// update the threshold on reaching the next level*
    $i \leftarrow i+1$; $h \leftarrow 1$; $p \leftarrow 2\beta hB$; *// update parameters*
  else
    $h \leftarrow h+1$; $p \leftarrow 2\beta hB$; *// increase the horizon*
  end
end

$\ell_i = \texttt{Cost}(\mathcal{M}, \widehat{\boldsymbol{a}}^h, h)$, where $\widehat{\boldsymbol{a}}^h$ is the optimal action sequence. The prediction horizon h is increased iteratively if the cost has not been decreased enough. Upon reaching a new level, the horizon is reset to one (see Algorithm 1).

Having a horizon $h > 1$ means it will take multiple transitions in the MDP to reach a solution with sufficiently improved cost. However, when finding such a solution with $h > 1$, we only apply the first action to transition the MDP to the next state. This is explained by the need to allow the other player (the environment or an adversary) to apply their action before we obtain the actual next state. If no new level is reached within h_{max} horizons, the first action of the best $\boldsymbol{a}^h$ using horizon h_{max} is applied.

The dynamic threshold Δ_i is defined as in [10]. Its initial value Δ_0 is obtained by dividing the cost range to be covered into m equal parts, that is, $\Delta_0 = (\ell_0 - \ell_m) \,/\, m$, where $\ell_0 = J(s_0)$ and $\ell_m = \varphi$. Subsequently, Δ_i is determined by the previously reached level ℓ_{i-1}, as $\Delta_i = \ell_{i-1}/(m-i+1)$. This way AMPC advances only if $\ell_i = \texttt{Cost}(\mathcal{M}, \widehat{\boldsymbol{a}}^h, h)$ is at least Δ_i apart from ℓ_{i-1}.

This approach allows us to force PSO to escape from a local minimum, even if this implies passing over a bump, by gradually increasing the exploration horizon h. We assume that the MDP is controllable. A discrete-time system S is said to be *controllable* if for any given states s and t, there exist a finite sequence of control inputs that takes S from s to t [13]. We also assume that the set G of goal states is non-empty, which means that from any state, it is possible to reach a state whose cost decreased by at least Δ_i. Algorithm 1 describes our approach in more detail.

Theorem 1 (AMPC Convergence). *Given an MDP $\mathcal{M} = (S, A, T, J)$ with positive and continuous cost function J, and a nonempty set of target states*

$G \subset S$ with $G = \{s \mid J(s) \leqslant \varphi\}$. If the transition relation T is controllable with actions in A, then there exists a finite maximum horizon h_{max} and a finite number of execution steps m, such that AMPC is able to find a sequence of actions $a_1, \ldots, a_m$ that brings a state in S to a state in G with probability one.

Proof. In each (macro-) step of horizon length h, from level ℓ_{i-1} to level ℓ_i, AMPC decreases the distance to φ by $\Delta_i \geqslant \Delta$, where $\Delta > 0$ is fixed by the number of steps m chosen in advance. Hence, AMPC converges to a state in G in a finite number of steps, for a properly chosen m. AMPC is able to decrease the cost in a macro step by Δ_i by the controllability assumption and the fairness assumption about the PSO algorithm. Since AMPC is a randomized algorithm, the result is probabilistic. Note that the theorem is an existence theorem of h_{max} and m whose values are chosen empirically in practice.

The adaptive MPC procedure, AMPC, is a key contribution of our work. Recall that traditional MPC uses a fixed finite horizon to determine the best control action. In contrast, AMPC dynamically chooses the horizon depending on the severity of the action played by the opponent (or environment). AMPC is inspired by the optimial plan synthesis procedure we recently presented in [10], which dynamically configures the amount of the effort it uses to search for a better solution at each step. In [10] the monolithic synthesis procedure was adaptive (and involved dynamically changing several parameters), whereas here the control procedure is adaptive and the underlying optimization is non-adaptive off-the-shelf procedure, and hence the overall procedure here is simpler.

Note that AMPC is a general procedure that performs adaptive MPC using PSO for dynamical systems that are controllable, come with a cost function, and have at least one optimal solution. In an adversarial situation two players have opposing objectives. The question arises what one player assumes about the other when computing its own action, which we discuss next.

5 Stochastic Games for V-Formation

We describe the specialization of the stochastic-game verification problem to V-formation. In particular, we present the AMPC-based control strategy for reaching a V-formation, and the various attacker strategies against which we evaluate the resilience of our controller.

5.1 Controller's Adaptive Strategies

Given current state $(\boldsymbol{x}(t), \boldsymbol{v}(t))$, the controller's strategy σ_C returns a probability distribution on the space of all possible accelerations (for all birds). As mentioned above, this probability distribution is specified implicitly via a randomized algorithm that returns an actual acceleration (again for all birds). This randomized algorithm is the AMPC algorithm, which inherits its randomization from the randomized PSO procedure it deploys.

When the controller computes an acceleration, it assumes that the attacker does *not* introduce any disturbances; i.e., the controller uses the following model:

$$\begin{aligned} \boldsymbol{x}_i(t+1) &= \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1) \\ \boldsymbol{v}_i(t+1) &= \boldsymbol{v}_i(t) + \boldsymbol{a}_i(t) \end{aligned} \tag{4}$$

where $\boldsymbol{a}(t)$ is the only control variable. Note that the controller chooses its next action $\boldsymbol{a}(t)$ based on the current configuration $(\boldsymbol{x}(t), \boldsymbol{v}(t))$ of the flock using MPC. The current configuration may have been influenced by the disturbance $\boldsymbol{d}(t-1)$ introduced by the attacker in the previous time step. Hence, the current state need not to be the state predicted by the controller when performing MPC in step $t-1$. Moreover, depending on the severity of the attacker action $\boldsymbol{d}(t-1)$, the AMPC procedure dynamically adapts its behavior, i.e. the choice of horizon h, in order to enable the controller to pick the best control action $\boldsymbol{a}(t)$ in response.

5.2 Attacker's Strategies

We are interested in evaluating the resilience of our V-formation controller when it is threatened by an attacker that can remove a certain number of birds from the flock, or manipulate a certain number of birds by taking control of their actuators (modeled by the displacement term in Eq. 1). We assume that the attack lasts for a limited amount of time, after which the controller attempts to bring the system back into the good set of states. When there is no attack, the system behavior is the one given by Eq. 4.

Bird Removal Game. In a BRG, the attacker selects a subset of R birds, where $R \ll B$, and removes them from the flock. The removal of bird i from the flock can be simulated in our framework by setting the displacement $\boldsymbol{d}_i$ for bird i to ∞. We assume that the flock is in a V-formation at time $t=0$. Thus, the goal of the controller is to bring the flock back into a V-formation consisting of $B-R$ birds. Apart from seeing if the controller can bring the flock back to a V-formation, we also analyze the time it takes the controller to do so.

Definition 6. *In a* Bird Removal Game *(BRG), the attacker strategy σ_D is defined as follows. Starting from a V-formation of B birds, i.e., $J(s_0) \leqslant \varphi$, the attacker chooses a subset of R birds, $R \ll B$, by uniform sampling without replacement. Then, in every round, it assigns each bird i in the subset a displacement $\boldsymbol{d}_i = \infty$, while for all other birds j, $\boldsymbol{d}_j = 0$.*

Random Displacement Game. In an RDG, the attacker chooses the displacement vector for a subset of R birds uniformly from the space $[0, M] \times [0, 2\pi]$ with $R \ll B$. This means that the magnitude of the displacement vector is picked from the interval $[0, M]$, and the direction of the displacement vector is picked from the interval $[0, 2\pi]$. We vary M in our experiments. The subset of R birds that are picked in different steps are not necessarily the same, as the attacker makes this choice uniformly at random at runtime as well.

The game starts from an initial V-formation. The attacker is allowed a fixed number of moves, say 20, after which the displacement vector is identically 0 for all birds. The controller, which has been running in parallel with the attacker, is then tasked with moving the flock back to a V-formation, if necessary.

Definition 7. *In a* Random Displacement Game *(RDG), the attacker strategy σ_D is defined as follows. Starting from a V-formation of B birds, i.e., $J(s_0) \leqslant \varphi$, in every round, it chooses a subset of R birds, $R \ll B$, by uniform sampling without replacement. It then assigns each bird i in the subset a displacement $\boldsymbol{d}_i$ chosen uniformly at random from $[0, M] \times [0, 2\pi]$, while for all other birds j, $\boldsymbol{d}_j = 0$. After T rounds, all displacements are set to* 0.

AMPC Game. An AMPC game is similar to an RDG except that the attacker does not use a uniform distribution to determine the displacement vector. The attacker is advanced and strategically calculates the displacement using the AMPC procedure. See Figure 1. In detail, the attacker applies AMPC, but assumes the controller applies zero acceleration. Thus, the attacker uses the following model of the flock dynamics:

$$\begin{aligned} \boldsymbol{x}_i(t+1) &= \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1) + \boldsymbol{d}_i(t) \\ \boldsymbol{v}_i(t+1) &= \boldsymbol{v}_i(t) \end{aligned} \tag{5}$$

Note that the attacker is still allowed to have $\boldsymbol{d}_i(t)$ be non-zero for only a small number of birds. However, it gets to choose these birds in each step. It uses the AMPC procedure to simultaneously pick the subset of R birds and their displacements. The objective of the attacker's AMPC is to maximize the cost.

Definition 8. *In an* AMPC game*, the attacker strategy σ_D is defined as follows. Starting from a V-formation of B birds, i.e., $J(s_0) \leqslant \varphi$, in every round, it uses AMPC to choose a subset of R birds, $R \ll B$, and their displacements $\boldsymbol{d}_i$ for bird i in the subset from $[0, M] \times [0, 2\pi]$; for all other birds j, $\boldsymbol{d}_j = 0$. After T rounds, all displacements are set to* 0.

Theorem 2 (AMPC resilience in a C-A game). *Given a controller-attacker game, there exists a finite maximum horizon h_{max} and a finite maximum number of game-execution steps m such that AMPC controller will win the controller-attacker game in m steps with probability 1.*

Proof. Since the flock MDP (defined by Eq. 1) is controllable, the PSO algorithm we use is fair, and the attack has a bounded duration, the proof of the theorem follows from Theorem 1.

Remark 1. While Theorem 2 states that the controller is expected to win with probability 1, we expect winning probability to be possibly lower than one in many cases because: (1) the maximum horizon h_{max} is fixed in advance, and so is (2) the maximum number of execution steps m; (3) the underlying PSO algorithm is also run with bounded number of particles and time. Theorem 2 is an existence theorem of h_{max} and m, while in practice one chooses fixed values of h_{max} and m that could be lower than the required values.

6 Statistical MC Evaluation of V-Formation Games

As discussed in Section 3, the stochastic-game verification problem we address in the context of the V-formation-AMPC algorithm is formulated as follows. Given a flock MDP $\mathcal{M}$ (we consider the case of $B = 7$ birds), acceleration actions $\boldsymbol{a}$ of the controller, displacement actions $\boldsymbol{d}$ of the attacker, the randomized strategy $\sigma_C : S \mapsto PD(C)$ of the controller (the AMPC algorithm), and a randomized strategy $\sigma_D : S \mapsto PD(D)$ for the attacker, determine the probability of reaching a state s where the cost function $J(s) \leqslant \varphi$ (V-formation in a 7-bird flock), starting from an initial state (in this case this is a V-formation), in the underlying Markov chain induced by strategies σ_C, σ_D on $\mathcal{M}$.

Since the exact solution to this reachability problem is intractable due to the infinite/continuous space of states and actions, we solve it approximately with classical statistical model-checking (SMC). The particular SMC procedure we use is from [7] and based on an *additive* or *absolute-error* (ε, δ)*-Monte-Carlo-approximation scheme*. This technique requires running N i.i.d. game executions, each for a given maximum time horizon, determining if these executions reach a V-formation, and returning the average number of times this occurs.

Each of the games described in Section 5 is executed 2,000 times. For a confidence ratio $\delta = 0.01$, we thus obtain an additive error of $\varepsilon = 0.1$. We use the following parameters in the game executions: number of birds $B = 7$, threshold on the cost $\varphi = 10^{-3}$, maximum horizon $h_{max} = 5$, number of particles in PSO $p = 20hB$. In BRG, the controller is allowed to run for a maximum of 30 steps. In RDG and AMPC game, the attacker and the controller run in parallel for 20 steps, after which the displacement becomes 0, and the controller has a maximum of 20 more steps to restore the flock to a V-formation.

To perform SMC evaluation of our AMPC approach we designed the above experiments in C and ran them on the Intel Core i7-5820K CPU with 3.30 GHz and with 32GB RAM available.

Table 1. Results of 2,000 game executions for removing 1 bird with $h_{max} = 5$, $m = 40$

	Ctrl. success rate, %	Avg. convergence duration	Avg. horizon
Bird 4	99.9	12.75	3.64
Bird 3	99.8	18.98	4.25
Bird 2	100	10.82	3.45

6.1 Discussion of the Results

To demonstrate the resilience of our adaptive controller, for each game introduced in Section 5, we performed a number of experiments to estimate the probability of the controller winning. Moreover, for the runs where the controller

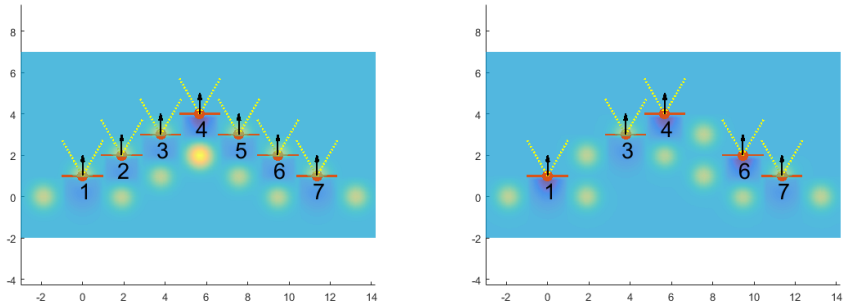

Fig. 2. Left: numbering of the birds. Right: configuration after removing Bird 2 and 5. The red-filled circle and two protruding line segments represent a bird's body and wings. Arrows represent bird velocities. Dotted lines illustrate clear-view cones. A brighter/darker background color indicates a higher upwash/downwash.

Table 2. Results of 2,000 game executions for removing 2 birds with $h_{max} = 5$, $m = 30$

	Ctrl. success rate, %	Avg. convergence duration	Avg. horizon
Birds 2 and 3	0.8	25.18	4.30
Birds 2 and 4	83.1	11.11	2.94
Birds 2 and 5	80.3	9.59	2.83
Birds 2 and 6	98.6	7.02	2.27
Birds 3 and 4	2.0	22.86	4.30
Birds 3 and 5	92.8	11.8	3.43

wins, the average number of steps required by the controller to bring the flock to a V-formation is reported as *average convergence duration*, and the average length of the horizon used by AMPC is reported as *average horizon*.

The numbering of the birds in Tables 1 and 2 is given in Figure 2. Bird-removal scenarios that are symmetric with the ones in the tables are omitted. The results presented in Table 1 are for the BRG game with $R = 1$. In this case, the controller is *almost always* able to bring the flock back to a V-formation, as is evident from Table 1. Note that removing Bird 1 (or 7) is a trivial case that results in a V-formation.

In the case when $R = 2$, shown in Table 2, the success rate of the controller depends on *which two birds are removed*. Naturally, there are cases where dropping two birds does not break the V-formation; for example, after dropping Birds 1 and 2, the remaining birds continue to be in a V-formation. Such trivial cases are not shown in Table 2. Note that the scenario of removing Bird 1 (or 7) and one other bird can be viewed as removing one bird in flock of 6 birds, thus not considered in this table. Among the other nontrivial cases, the success rate of controller drops slightly in four cases, and drops drastically in remaining two

Table 3. Results of 2,000 game executions for random displacement and AMPC attacks with $h_{max} = 5$ and $m = 40$ (attacker runs for 20 steps)

Range of noise	Ctrl. success rate, %	Avg. convergence duration	Avg. horizon
	Random displacement game		
$[0, 0.50] \times [0, 2\pi]$	99.9	3.33	1.07
$[0, 0.75] \times [0, 2\pi]$	97.9	3.61	1.11
$[0, 1.00] \times [0, 2\pi]$	92.3	4.14	1.18
	AMPC game		
$[0, 0.50] \times [0, 2\pi]$	97.5	4.29	1.09
$[0, 0.75] \times [0, 2\pi]$	63.4	5.17	1.23
$[0, 1.00] \times [0, 2\pi]$	20.0	7.30	1.47

cases. This suggests that attacker of a CPS system can incur more damage by being prudent in the choice of the attack.

Impressively, whenever the controller wins, the controller needs about the same number of steps to get back to V-formation (as in the one-bird removal case). On average, removal of two birds results in a configuration that has worse cost compared to an BRG with $R = 1$. Hence, the adaptive controller is able to make bigger improvements (in each step) when challenged by worse configurations. Furthermore, among the four cases where the controller win rate is high, experimental results demonstrate that removing two birds positioned asymmetrically with respect to the leader poses a stronger, however, still manageable threat to the formation. For instance, the scenarios of removing birds 2 and 6 or 3 and 5 give the controller a significantly higher chance to recover from the attack, 98.6% and 92.8%, respectively.

Table 3 explores the effect of making the attacker smarter. Compared to an attacker that makes random changes in displacement, an attacker that uses AMPC to pick its action is able to win more often. This again shows that an attacker of a CPS system can improve its chances by cleverly choosing the attack. For example, the probability of success for the controller to recover drops from 92.3% to 20.0% when the attacker uses AMPC to pick displacements with magnitude in $[0, 1]$ and direction in $[0, 2\pi]$. The entries in the other two columns in Table 3 reveal two even more interesting facts.

First, in the cases when the controller wins, we clearly see that the controller uses a longer look-ahead when facing a more challenging attack. This follows from the observation that the average horizon value increases with the strength of attack. This gives evidence for the fact that the adaptive component of our AMPC plays a pivotal role in providing resilience against sophisticated attacks. Second, the average horizon still being in the range 1-1.5, means that the adaptation in our AMPC procedure also helps it perform better than a fixed-horizon MPC procedure, where usually the horizon is fixed to $h \geqslant 2$. When a low value of h (say $h = 1$) suffices, the AMPC procedure avoids unnecessary calculation that using a fixed h might incur.

In the cases where success rate was low (Row 1 and Row 5 in Table 2, and Row 3 of the AMPC game in Table 3), we conducted additional 500 runs for each case and observed improved success rates (2.4%, 9% and 30.8% respectively) when we increased h_{max} to 10 and m to 40. This shows that success rates of AMPC improves when given more resources, as predicted by Theorem 1.

7 Related Work

In the field of CPS security, one of the most widely studied attacks is *sensor spoofing*. When sensors measurements are compromised, state estimation becomes challenging, which inspired a considerable amount of work on attack-resilient state estimation [4, 6, 14–16]. In these approaches, resilience to attacks is typically achieved by assuming the presence of redundant sensors, or coding sensor outputs. In our work, we do not consider sensor spoofing attacks, but assume the attacker gets control of the displacement vectors (for some of the birds/drones). We have not explicitly stated the mechanism by which an attacker obtains this capability, but it is easy to envision ways (radio controller, attack via physical medium, or other channels [2]) for doing so.

Adaptive control, and its special case of adaptive model predictive control, typically refers to the aspect of the controller updating its process model that it uses to compute the control action. The field of adaptive control is concerned with the discrepancy between the actual process and its model used by the controller. In our adaptive-horizon MPC, we adapt the lookahead horizon employed by the MPC, and not the model itself. Hence, the work in this paper is orthogonal to what is done in adaptive control [1, 11].

Adaptive-horizon MPC was used in [5] to track a reference signal. If the reference signal is unknown, and we have a poor estimate of its future behavior, then a larger horizon for MPC is not beneficial. Thus, the horizon was determined by the uncertainty in the knowledge of the future reference signal. We consider cost-based reachability goals here, which allows us to choose a horizon in a more generic way based on the progress toward the goal. More recently, adaptive horizons were also used in [9] for a reachability goal. However, they chose a large-enough horizon that enabled the system to reach states from where a pre-computed local controller could guarantee reachability of the goal. This is less practical than our approach for establishing the horizon.

A key focus in CPS security has also been detection of attacks. For example, recent work considers displacement-based attacks on formation flight [12], but it primarily concerned with detecting which UAV was attacked using an unknown-input-observer based approach. We are not concerned with detecting attacks, but establishing that the adaptive nature of our controller provides attack-resilience for free. Moreover, in our setting, for both the attacker the and controller the state of the plant is completely observable. In [17], a control policy based on the robustness of the connectivity graph is proposed to achieve consensus on the velocity among a team of mobile robots, in the present of non-cooperative robots that communicate false values but execute the agreed upon commands.

In contrast, we allow the attacker to manipulate the executed commands of the robots. The cost function we use is also more flexible so that we can encode more complicated objectives.

We are unaware of any work that uses statistical model checking to evaluate the resilience of adaptive controllers against (certain classes of) attacks.

8 Conclusions

We have introduced AMPC, a new model-predictive controller that unlike MPC, comes with provable convergence guarantees. The key innovation of AMPC is that it dynamically adapts its receding horizon (RH) to get out of local minima. In each prediction step, AMPC calls PSO with an optimal RH and corresponding number of particles. We used AMPC as a bird-flocking controller whose goal is to achieve V-formation despite various forms of attacks, including bird-removal, bird-position-perturbation, and advanced AMPC-based attacks. We quantified the resiliency of AMPC to such attacks using statistical model checking. Our results show that AMPC is able to adapt to the severity of an attack by dynamically changing its horizon size and the number of particles used by PSO to completely recover from the attack, given a sufficiently long horizon and execution time (ET). The intelligence of an attacker, however, makes a difference in the outcome of a game if RH and ET are bounded before the game begins.

Future work includes the consideration of additional forms of attacks, including: *Energy attack*, when the flock is not traveling in a V-formation for a certain amount of time; *Collisions*, when two birds are dangerously close to each other due to sensor spoofing or adversarial birds; and *Heading change*, when the flock is diverted from its original destination (mission target) by a certain degree.

Acknowledgments. Research supported in part by the Doctoral Program Logical Methods in Computer Science and the Austrian National Research Network RiSE/SHiNE (S11412-N23) project funded by the Austrian Science Fund (FWF) project W1255-N23, AFOSR Grant FA9550-14-1-0261 and NSF Grants CCF-1423296, CNS-1423298, IIS-1447549, CNS-1446832, CNS-1445770, CNS-1445770.

References

1. Adetola, V., DeHaan, D., Guay, M.: Adaptive model predictive control for constrained nonlinear systems. Systems & Control Letters 58(5), 320–326 (2009)
2. Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., an, A.C., Roesner, F., Kohno, T.: Comprehensive experimental analyses of automotive attack surfaces. In: USENIX Security (2011)
3. Condliffe, J.: A 100-drone swarm, dropped from jets, plans its own moves. MIT Technology Review (Jan 2017)
4. Davidson, D., Wu, H., Jellinek, R., Ristenpart, T., Singh, V.: Controlling UAVs with sensor input spoofing attacks. In: Proceedings of WOOT'16, 10th USENIX Workshop on Offensive Technologies. Austin, TX (Aug 2016)

5. Droge, G., Egerstedt, M.: Adaptive time horizon optimization in model predictive control. In: American Control Conference (ACC), 2011. pp. 1843–1848. IEEE (2011)
6. Fawzi, H., Tabuada, P., Diggavi, S.N.: Secure estimation and control for cyber-physical systems under adversarial attacks. IEEE Trans. Automat. Contr. 59(6), 1454–1467 (2014), http://dx.doi.org/10.1109/TAC.2014.2303233
7. Grosu, R., Peled, D., Ramakrishnan, C.R., Smolka, S.A., Stoller, S.D., Yang, J.: Using statistical model checking for measuring systems. In: Proceedings of the International Symposium Leveraging Applications of Formal Methods, Verification and Validation. LNCS, vol. 8803, pp. 223–238. Springer (2014)
8. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of 1995 IEEE International Conference on Neural Networks. pp. 1942–1948 (1995)
9. Krener, A.J.: Adaptive horizon model predictive control. arXiv preprint arXiv:1602.08619 (2016)
10. Lukina, A., Esterle, L., Hirsch, C., Bartocci, E., Yang, J., Tiwari, A., Smolka, S.A., Grosu, R.: ARES: adaptive receding-horizon synthesis of optimal plans. In: Proc. of TACAS 2017: the 23rd International Conference on Tools and Algorithms for the Construction and Analysis of Systems. p. to appear. LNCS (2017)
11. Narendra, K.S.: Adaptive control using neural networks. In: Neural networks for control. pp. 115–142. MIT Press (1990)
12. Negash, L., Kim, S.H., Choi, H.L.: An unknown-input-observer based approach for cyber attack detection in formation flying UAVs. In: AIAA Infotech (2016)
13. Ogata, K.: Modern Control Engineering. Instrumentation and controls series, Prentice Hall (2010)
14. Pajic, M., Weimer, J., Bezzo, N., Tabuada, P., Sokolsky, O., Lee, I., Pappas, G.J.: Robustness of attack-resilient state estimators. In: 5th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS) (2014)
15. Park, J., Ivanov, R., Weimer, J., Pajic, M., Lee, I.: Sensor attack detection in the presence of transient faults. In: 6th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS) (2015)
16. Pasqualetti, F., Dorfler, F., Bullo, F.: Attack detection and identification in cyber-physical systems. IEEE Trans. on Automatic Control 58(11), 2715–2729 (2013)
17. Saulnier, K., Saldana, D., Prorok, A., Pappas, G.J., Kumar, V.: Resilient flocking for mobile robot teams. IEEE Robotics and Automation Letters 2(2), 1039–1046 (2017)
18. Shapley, L.S.: Stochastic games. Proceedings of the National Academy of Sciences 39(10), 1095–1100 (1953)
19. Yang, J., Grosu, R., Smolka, S.A., Tiwari, A.: Love thy neighbor: V-formation as a problem of model predictive control. In: LIPIcs-Leibniz International Proceedings in Informatics. vol. 59. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2016)
20. Yang, J., Grosu, R., Smolka, S.A., Tiwari, A.: V-formation as optimal control. In: Proceedings of the Biological Distributed Algorithms Workshop (2016)