Evolutionary Design of Cooperative Predation Strategies

Alexander Von Moll
Control Science Center of Excellence
Air Force Research Laboratory
WPAFB, OH, USA
ORCiD: 0000-0002-7661-5752

Pavlos Androulakakis, Zachariah Fuchs
Dept. of Elect. Eng. & Comp. Systems
University of Cincinnati
Cincinnati, OH, USA

Dieter Vanderelst
Depts. of Psych., Bio. Sciences
and Mech. & Mat. Eng.
University of Cincinnati
Cincinnati, OH, USA

***Abstract*—In this paper a scenario is considered in which a group of predators cooperate to maximize the number of prey captures over a finite time horizon on a two-dimensional plane. The emphasis is on developing predator strategies, and thus the behavior of the prey agents is fixed to a Boids-like flocking model which incorporates avoidance of nearby predators. At each time instant, the predators have control authority over their heading angle; however, we restrict the headings to be governed by one of five different pre-specified behaviors. No communication occurs between the predator agents – each agent determines its own control without knowledge of what the other predators will implement; thus, the predator strategies are fully decentralized. The full state space of the system is collapsed to a set of features which is independent of the number of prey. An evolutionary algorithm is used to evolve an anchor point controller wherein the anchor point lies in the feature space and has a particular predator behavior associated, thus providing a candidate solution to the question of "what to do when". The two predator case is the focus in this work, although the ideas could be applied to larger groups of predators. The strategies resulting from the evolutionary algorithm favor aiming at the nearest prey mostly, and also avoiding having the predators getting too close and then pursuing the same prey. Thus useless behaviors are generally not present among the elite at the end of the evolutionary process.**

I. Introduction

Cooperation is essential to the survival and success of many different species of animals. Remarkable feats are made possible through cooperative behaviors which emerge from relatively simple processes at the individual level. For example, consider the nest building, foraging, and decision making abilities that insect 'societies' are capable of [1], while at the same time only possessing a modest amount of processing power. Similarly, cooperation among autonomous vehicles (or more abstractly, agents) may enable new types of tasks to be accomplished or better performance on existing tasks. Particularly interesting is the use of cooperation within a group in adversarial scenarios *between* groups or species. In nature, these scenarios include collective defense such as meerkat mobbing [2], distributed nest defense [3], and musk oxen who press together with their horns facing outward [4], [5]; cooperative predation such as the yellowsaddle goatfish [6], wolves [7], and dolphins [8]; and cooperative sensing such as predator inspection performed by guppies [9].

Biological systems have evolved such innovative ways of cooperating that they often serve as the inspiration for optimization algorithms [5], robotic control algorithms [10], [11], [12], and methods for designing controllers (e.g. via evolutionary algorithms) [12], [13], [14]. Regarding the design of controllers, the difficulty often lies in the dimensionality of either the state space or the control space. Genetic algorithms, for example, tend to settle in local optima when the chromosome encoding the controller is large. One approach to reducing the size of control space is to constrain the controller at the outset based on heuristics or expert knowledge (c.f. [14], [15]). This approach finds its roots in Connell's ideas of minimalist robotics [16], wherein he discussed how a controller comprised of a few simple behaviors could mostly account for the seemingly complex behavior of a snail. Of course, the principle of simplicity in robotic control was heavily inspired by Braitenberg Vehicles whose two sensors are directly connected to its two motors; the different ways of connecting sensors to motors results in fundamentally different, but understandable behaviors [17].

The focus, in this paper, is on a particular type of predator-prey interaction, though aspects of some of the other natural cooperative behaviors were influential in the problem setup and technical approach. An analogous relationship is that between a pursuer and an evader which is prevalent in controls and optimization literature (see, e.g., [18]). Often, differential game theory is employed to obtain saddle-point equilibrium strategies for the pursuer and the evader depending on the particular cost functional being considered. This type of analysis is typically only possible for systems with simple dynamics and/or few numbers of agents. For example, the two-pursuer one-evader game of min/max capture time was solved in [19] wherein the two pursuers employ a "pincer" maneuver to reduce the capture time of the evader w.r.t. either of their individual capture times. Similarly, Breakwell et al. [20] solved the one-pursuer two-evader game of min/max capture time (of the second evader captured). Other cost functionals have been explored in the one-against-two game which drastically changes the equilibrium control strategies [21].

We consider, here, a similar style of pursuit-evasion (or predator-prey) game with similar (single-integrator) dynamics but with much larger numbers of prey. Also, while the mentioned literature pits two against one (or vice versa), we explore a two-against-many scenario. The goal of the two predators is to maximize the collective number of prey captured within a specified time horizon. They do so without communicating or directly coordinating with their fellow teammate, whereas teammates in a differential game context are essentially treated as a single entity. Thus our control strategy is inherently decentralized, which more closely represents a biological system. In order to focus on the cooperation of the predators, in particular, the prey control strategy is fixed to a Boids-like controller (c.f. [22]). As a result, the predators can learn to exploit the underlying prey flocking behavior by affecting the *shape* of the prey distribution. The predator control is based on a nearest-neighbor approach based on [14], though we consider the weighted nearest-neighbor here. Each predator measures the (meta) state of the system and implements the baseline behavior associated with the nearest anchor point.

These approaches and concepts are applicable to behavior generation for non-player characters (NPCs) as well as military training simulations. In particular, a neural network-based controller was evolved using a genetic algorithm for a real-time strategy (RTS)-like game in [13]. To contrast, the controller structure we employ here is not a neural network and we are focused on movement and positioning (as opposed to game elements like resource gathering). A neural network controller generated via genetic algorithm was also employed in [23] where the emphasis was on so-called "multi-modal" behaviors. One of the games considered was a fight or flight game where at any given time the player is executing one or other particular behavior; this concept of a multi-modal behavior or controller is quite related to this paper. Reference [24] addresses generation of NPC behaviors but from the angle of planning (e.g., via meta-heuristic search) which occurs on a much larger timescale than the type of control we focus on in this paper. Finally, the application of artificial intelligence and game concepts to military training simulations is demonstrated in [25] on task allocation and planning problems with an emphasis on "explainability".

The contributions of this paper are listed as follows: (i) specification of baseline cooperative predator strategies; (ii) specification of many-against-many metastates; (iii) an anchor point control architecture based on weighted nearest anchor point; and (iv) a decentralized predator control strategy which outperforms a baseline strategy. Section II contains the problem formulation. Sections III and IV describe the prey model and predator model, respectively. The controller structure is also specified in Section IV. Section V specifies the evolutionary algorithm along with all of the genetic operators particular to the control structure we employ. Simulation results are contained in Section VI, and Section VII concludes the paper.

II. Problem Formulation

The environment in which the scenario takes place is a flat plane with no boundaries or obstacles. Let the predator positions be given as $\mathbf{P}_1 = (x_{\mathbf{P}_1}, y_{\mathbf{P}_1}), \mathbf{P}_2 = (x_{\mathbf{P}_2}, y_{\mathbf{P}_2}) \in \mathbb{R}^2$. Similarly, the prey positions are denoted by $\mathbf{E}_j = (x_{\mathbf{E}_j}, y_{\mathbf{E}_j}) \in \mathbb{R}^2$ for $j \in 1, \ldots, M$, and $M \in \mathbb{Z}$. Let the full system state be denoted as $\mathbf{x} = \begin{bmatrix}\mathbf{P}_1^\top & \mathbf{P}_2^\top & \mathbf{E}_1^\top & \ldots & \mathbf{E}_M^\top\end{bmatrix}^\top$. In general, it is assumed that the maximum predator and prey speeds, v_P and v_E, are such that $v_P > v_E$, and that agent speeds are homogeneous within each respective group. All the agents move with simple motion, i.e.,

$$\dot{\mathbf{P}}_i = v_P \begin{bmatrix}\cos\psi_i \\ \sin\psi_i\end{bmatrix}, \quad \dot{\mathbf{E}}_j = \bar{v}_E \begin{bmatrix}\cos\phi_j \\ \sin\phi_j\end{bmatrix},$$

where $\psi_i, \phi_j \in [0, 2\pi]$ are the instantaneous heading angle of predator i and prey j, respectively, and $\bar{v}_E \in [0, v_E]$. Note that the predator is assumed to always move with its maximum speed whereas the prey is allowed to slow down (subject to the prey model discussed in detail in Section III). For the purposes of numerical simulation, the scenario takes place in discretized time and the agents' headings are computed/updated simultaneously. Thus the discrete time step $t_k = k\Delta t \in [0, T]$, where T is the time horizon of the simulation and is an integer multiple of Δt, and $k \in \{1, 2, \ldots, T/\Delta t\}$. Throughout the remainder of the paper, we generally refer to the current positions of the agents without explicitly stating its dependence on time (e.g. $\mathbf{E}_j$ rather than $\mathbf{E}_j(t_k)$).

Capture is said to occur when a predator comes within a distance d_c of a prey agent. An indicator function $\mathbb{I}_{\text{cap}}(i, j, t_k)$ is used to represent predator i ($i \in \{1, 2\}$) capturing prey j in time step t_k:

$$\mathbb{I}_{\text{cap}}(i, j, t_k) = \begin{cases} 1 & \text{if } \|\mathbf{P}_i - \mathbf{E}_j\| \leq d_c, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $\|\cdot\|$ is the Euclidean norm. Once a prey agent has been captured, it is effectively removed from the scenario and the capturing predator is free to move on to other targets immediately thereafter.

The predators have a shared goal of maximizing the number of prey captured over a time horizon of T simulation seconds,

$$U\left(\psi_1\left(\mathbf{x}(t)\right), \psi_2\left(\mathbf{x}(t)\right)\right) = \sum_{i=1}^{2} \sum_{j=1}^{M} \sum_{t_k} \mathbb{I}_{\text{cap}}(i, j, t_k), \tag{2}$$

where $\psi_i\left(\mathbf{x}(t)\right)$ is the state-feedback control law of predator i. In practice, (2) must be modified to exclude double-counted captures (i.e. when prey j is within d_c of both predators). Because the utility is shared, this scenario is not an example of by-product mutualism, wherein cooperation arises from selfish acts, as in [6]. With the prey behavior fixed, the aim of this study is to design a state-feedback controller for the predators.

III. Prey Model

The behavior of the prey agents is governed by a Boids-like model (c.f. [22]); this section describes the details of the model. In order to mimic the flocking behavior of animals, the Boids model introduces inertia into the agents' motion and models various influences as virtual forces. The virtual forces governing the behavior of the prey are alignment, cohesion, and separation [22] as well as avoidance (of nearby predators). Another augmentation to the original Boids model is the inclusion of a finite sensing radius, r, for the prey. That is, a prey agent only knows, or takes into consideration, the positions of predators and prey within r distance of its current position. Here, we treat the prey agents as if they have unit mass.

Three sets are used in formulating the virtual forces. The first is the set of prey agents within the sensing radius r,

$$\mathcal{R}_j := \{j' \mid j' \neq j, \|\mathbf{E}_j - \mathbf{E}_{j'}\| \leq r\}. \tag{3}$$

Next is the set of prey agents within a desired minimum separation distance s,

$$\mathcal{S}_j := \{j' \mid j' \neq j, \|\mathbf{E}_j - \mathbf{E}_{j'}\| \leq s\}. \tag{4}$$

Last is the set of predators within the sensing radius r,

$$\mathcal{P}_j := \{i \mid \|\mathbf{E}_j - \mathbf{P}_i\| \leq r\}. \tag{5}$$

The current centroid of prey positions and velocity vectors within distance r of prey j are respectively given by

$$\bar{\mathbf{E}}_j = \frac{1}{|\mathcal{R}_j|} \sum_{j' \in \mathcal{R}_j} \mathbf{E}_{j'}, \quad \dot{\bar{\mathbf{E}}}_j = \frac{1}{|\mathcal{R}_j|} \sum_{j' \in \mathcal{R}_j} \dot{\mathbf{E}}_{j'}. \tag{6}$$

Alignment is based on the principle that the prey naturally seek to align their velocity with the overall direction of travel of the flock,

$$\mathbf{F}_{\text{ali}_j} = \frac{\dot{\bar{\mathbf{E}}}_j}{\|\dot{\bar{\mathbf{E}}}_j\|}. \tag{7}$$

Cohesion provides some influence for prey to gravitate towards the centroid of prey positions which prevents the flock from splitting apart,

$$\mathbf{F}_{\text{coh}_j} = \frac{\bar{\mathbf{E}}_j - \mathbf{E}_j}{\|\bar{\mathbf{E}}_j - \mathbf{E}_j\|}. \tag{8}$$

Because the agents represent some physical entity, a force designed to maintain a desired minimum separation distance, s, is necessary,

$$\mathbf{F}_{\text{sep}_j} = \frac{1}{|\mathcal{S}_j|} \sum_{j' \in \mathcal{S}_j} \frac{\mathbf{E}_j - \mathbf{E}_{j'}}{\|\mathbf{E}_j - \mathbf{E}_{j'}\|^2} \tag{9}$$

In this case, the contribution due to each neighboring prey is distance-weighted to prioritize nearer violators of the desired minimum separation distance. Finally, avoidance forces the prey to flee from predators within the sensing radius r,

$$\mathbf{F}_{\text{avo}_j} = \frac{1}{|\mathcal{P}_j|} \sum_{i \in \mathcal{P}_j} \frac{\mathbf{E}_j - \mathbf{P}_i}{\|\mathbf{E}_j - \mathbf{P}_i\|^2}. \tag{10}$$

Again, closer predators provide more force.

With all of the necessary forces defined, the (candidate) velocity update law of the prey is given as,

$$\begin{aligned}\dot{\mathbf{E}}'_j(t_k) = \; & \dot{\mathbf{E}}_j(t_{k-1}) + \Delta t \left(w_{\text{ali}} \mathbf{F}_{\text{ali}}\right. \\ & \left. + w_{\text{coh}} \mathbf{F}_{\text{coh}} + w_{\text{sep}} \mathbf{F}_{\text{sep}} + w_{\text{avo}} \mathbf{F}_{\text{avo}}\right),\end{aligned} \tag{11}$$

where the w's are weights corresponding to the level at which the prey are influenced by each force. Then the candidate velocity is passed through a saturation function to ensure it does not exceed the maximum prey speed,

$$\dot{\mathbf{E}}_j = \begin{cases} \dot{\mathbf{E}}'_j & \text{if } \|\dot{\mathbf{E}}'_j\| \leq v_E, \\ \dot{\mathbf{E}}'_j \frac{v_E}{\|\dot{\mathbf{E}}'_j\|} & \text{otherwise.} \end{cases} \tag{12}$$

IV. Predator Model

The overall predator control approach is based on the idea of specifying simple atomic behaviors and then letting the predator decide which behavior to implement in each time step. Unlike the prey, the predators have full access to the state, $\mathbf{x}$ (the positions of each agent). Each predator selects its own atomic behavior to implement – that is, they do not collaborate on which behavior to select. We emphasize, again, that this approach is decentralized, in contrast to more explicit cooperative maneuvers in studies like [15] which require communication.

A. Atomic Behaviors

The first atomic behavior is the `pursue` behavior wherein the predator, $\mathbf{P}_i$, aims directly at the nearest prey agent. The index of the nearest prey agent is given by

$$j^* = \arg\min_j \|\mathbf{P}_i - \mathbf{E}_j\|, \tag{13}$$

and the associated heading angle is

$$\psi_{\text{pur}} = \operatorname{atan2}\left(y_{\mathbf{E}_{j^*}} - y_{\mathbf{P}_i}, x_{\mathbf{E}_{j^*}} - x_{\mathbf{P}_i}\right), \tag{14}$$

where atan2 is the four-quadrant inverse tangent function. In the one-pursuer, one-evader differential game of $\min\max$ capture time with simple motion this is, in fact, the equilibrium strategy for the pursuer (c.f. [18]). Note, this may not be optimal for the predator since the prey is not necessarily implementing its equilibrium strategy. Nonetheless, it is robust to any prey strategy. It is also useful here as the one-on-one pursuit-evasion game may be considered to be a subproblem to the overall scenario.

Next, the `converge` and `diverge` behaviors are based on mixing pure convergence (i.e. aiming directly towards the predator centroid) or divergence (i.e. aiming directly away from the predator centroid) with aiming at the prey centroid. Inclusion of `diverge` is partly based on [26], wherein the author utilized an intra-predator repulsive force in controlling a group of pursuers pursuing a single evader in a decentralized fashion. Let M' be the number of prey currently living; the angles from the predator to the prey centroid and predator centroid are given as

$$\psi_{\bar{\mathbf{E}}} = \operatorname{atan2}\left(\left(\sum_j \frac{y_{\mathbf{E}_j}}{M'}\right) - y_{\mathbf{P}_i}, \left(\sum_j \frac{x_{\mathbf{E}_j}}{M'}\right) - x_{\mathbf{P}_i}\right), \tag{15}$$

$$\psi_{\bar{\mathbf{P}}} = \operatorname{atan2}\left(\left(\sum_{i'} \frac{y_{\mathbf{E}_{i'}}}{N}\right) - y_{\mathbf{P}_i}, \left(\sum_{i'} \frac{x_{\mathbf{E}_{i'}}}{N}\right) - x_{\mathbf{P}_i}\right), \tag{16}$$

respectively. Then the `converge` and `diverge` behaviors are governed by

$$\psi_{\text{con}} = \operatorname{atan2}\left(m \sin\psi_{\bar{\mathbf{P}}} + \sin\psi_{\bar{\mathbf{E}}}, m\cos\psi_{\bar{\mathbf{P}}} + \cos\psi_{\bar{\mathbf{E}}}\right), \tag{17}$$

$$\psi_{\text{div}} = \operatorname{atan2}\left(-m \sin\psi_{\bar{\mathbf{P}}} + \sin\psi_{\bar{\mathbf{E}}}, -m\cos\psi_{\bar{\mathbf{P}}} + \cos\psi_{\bar{\mathbf{E}}}\right), \tag{18}$$

where m is the mixing weight; the larger m the more the behaviors approach pure convergence/divergence.

The last two behaviors, `drive` and `flank`, both make use of an angle, ψ_{den}, which represents the angle from the predator to the highest distance-weighted density of prey. Kernel Density Estimation (KDE) is used to compute an estimate of ψ_{den}. First, the angles from the predator to each living prey are computed as

$$\psi_{i,j} = \operatorname{atan2}\left(y_{\mathbf{E}_j} - y_{\mathbf{P}_i}, x_{\mathbf{E}_j} - x_{\mathbf{P}_i}\right). \tag{19}$$

Then the distribution of $\psi_{i,j}$ is smoothed via the following estimator:

$$\hat{f}_h(\psi) = \frac{1}{hM'}\sum_j \frac{1}{\|\mathbf{P}_i - \mathbf{E}_j\|} K\left(\frac{\psi - \psi_{i,j}}{h}\right), \tag{20}$$

where $h > 0$ is the bandwidth of the estimator and K is the kernel. For this study, the bandwidth is set using Silverman's Rule [27], and the standard Gaussian distribution is used for the kernel. Finally, the angle from the predator to the highest distance-weighted density of prey is defined as

$$\psi_{\text{den}} = \arg\max_{\psi} \hat{f}_h(\psi). \tag{21}$$

The `drive` behavior is governed by

$$\psi_{\text{dri}} = \psi_{\text{den}}, \tag{22}$$

thus the predator aims always in the direction of maximum distance-weighted prey density. The main purpose of including this behavior is to avoid the potential pitfall of wasting time pursuing a single prey agent away from an advantageous cluster of prey.

Influence over the distribution or shape of the prey flock is the main purpose of the `flank` behavior. In general, it is better for the predators when the prey are highly concentrated; however, once a predator approaches, the flock will evade and disperse. Thus the `flank` behavior is designed to aim the predator in a direction tangential to the flock. Let $i' = \arg\min_i \|\mathbf{P}_i - \mathbf{P}_{i'}\|$ be the index of the predator closest to the ith predator.

$$\psi_{\text{fla}} = \psi_{\text{den}} - \frac{\pi}{2} \cdot \operatorname{sign}\left(\begin{bmatrix}\cos\psi_{\text{den}} \\ \sin\psi_{\text{den}}\end{bmatrix} \times (\mathbf{P}_{i'} - \mathbf{P})\right) \tag{23}$$

The last term ensures that the tangential direction is one which points away from the nearest predator.

Figure 1 shows a pictorial representation of each of the predator behaviors. Although `pursue` and `drive` (and, similarly `diverge` and `flank`) appear quite similar, the presence of additional prey can effect more obvious distinctions.

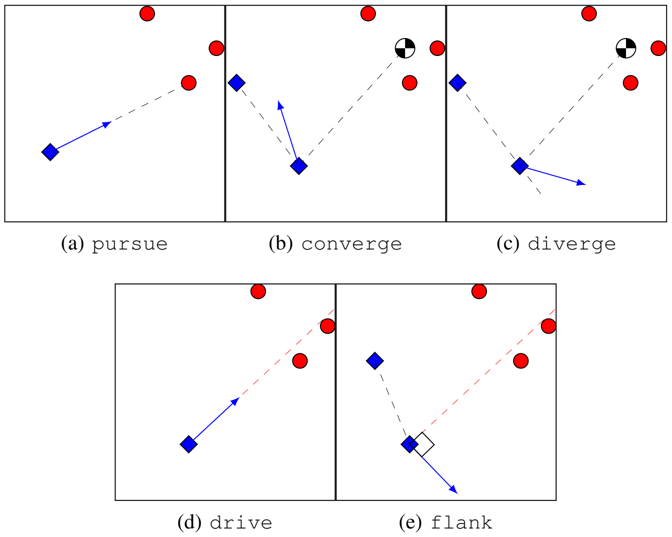

Fig. 1. Demonstration of each of the five predator behaviors. The red dashed line indicates the direction of maximum distance-weighted density of prey, ψ_{den} .

B. Meta State

The overall state of the system at any given time step is essentially the positions of all of the predators and all of the prey agents who have not yet been captured. One may consider including another state for the prey which specifies whether a particular agent is alive or dead, but ultimately the positions of dead prey agents ought not have any bearing on the predator's decisions. Aside from the variability in the size of the state space, the main issue is that even with a few living prey agents the number of dimensions is quite large. In an effort to avoid the curse of dimensionality, and to be able to function in the presence of many prey agents, it is prudent to collapse the positional state information into a smaller (meta) state space.

Let the centroid of *all* currently living prey be $\bar{\mathbf{E}} = \frac{1}{M'}\sum_j \mathbf{E}_j$. For the ith predator, we define the meta state as

$$\hat{\mathbf{x}}_i(t_k) = \begin{bmatrix} \min_j \|\mathbf{P}_i - \mathbf{E}_j\| \\ \|\mathbf{P}_1 - \bar{\mathbf{E}}\| \\ \vdots \\ \|\mathbf{P}_N - \bar{\mathbf{E}}\| \\ \|\mathbf{P}_i - \mathbf{P}_{\neq i}\| \\ \vdots \end{bmatrix}, \tag{24}$$

where N is the number of predators. The meta state is comprised of the distance to the nearest prey, the distances from each predator to the prey centroid, and the distances to each other predator; $\hat{\mathbf{x}}_i \in \mathbb{R}^{2N}$. For $N = 2$ predators the size of the meta state space is 4, regardless of the number of prey in the simulation.

C. Controller Structure

In order to limit the complexity of this initial study, we consider the predators to be homogeneous – that is, they share the same controller, ψ, which is a function of the meta state. Thus let $\psi_i(\mathbf{x}) := \psi(\hat{\mathbf{x}}_i)$ for $i \in \{1,2\}$. In order to evolve the predators' feedback controller, we need to be able to parameterize it in such a way that will allow us to apply crossing and mutation operations to it. For the research, we accomplish this by using an anchor point method [14].

Anchor points $a := [\hat{\mathbf{x}}, w, u]$ are comprised of a position (in the meta state space) $\hat{\mathbf{x}}$, a weight w, and a control u. In this case, the control is selected as one of the atomic behaviors described in Section IV-A:

$$u \in \{\psi_{\text{pur}}, \psi_{\text{con}}, \psi_{\text{div}}, \psi_{\text{dri}}, \psi_{\text{fla}}\}\,. \tag{25}$$

A set of anchor points $S = [a_1, a_2, \ldots a_N]$ can be used to parameterize a feedback controller by using the anchor points as the basis for a weighted nearest neighbor switching controller.

$$[\hat{\mathbf{x}}^*, w^*, u^*] = \underset{[\hat{\mathbf{x}}_i, w_i, u_i] \in S}{\arg\min} \; w_i \|\hat{\mathbf{x}} - \hat{\mathbf{x}}_i\| \tag{26}$$

Figure 2 shows a 2D example of a feedback controller

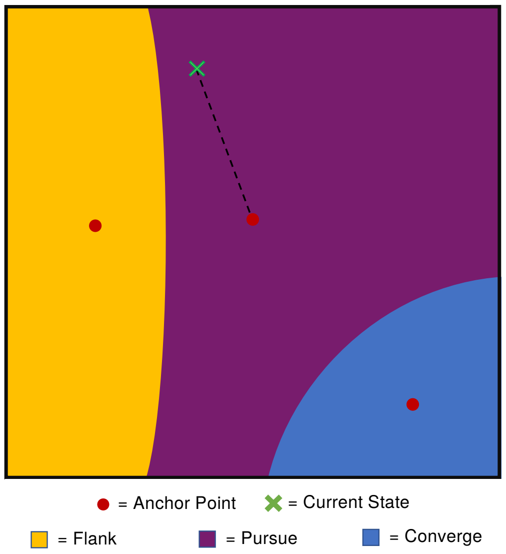

Fig. 2. Example of an anchor point feedback controller in a fictitious 2-dimensional meta state space.

parameterized by a set of 3 anchor points. At any given state $\hat{\mathbf{x}}$, the control can be computed by finding the weighted nearest anchor point, a^*, by (26) and implementing the associated control, u^*. The higher the weight associated with a particular anchor point the further it will appear, in this case.

V. Evolutionary Algorithm

The evolutionary algorithm (EA) used in this research follows a canonical EA format. We begin at generation G_0 with a initial population of P candidate controllers, each of which is comprised of $n_l \sim \text{Uniform}(1, n)$ anchor points that are randomly placed in the 4-dimensional meta state space with a randomly assigned behavior and a random weight, $w \sim \text{Uniform}(0, 1)$. This initial population is evaluated, assigned a fitness, and entered as the first generation of the evolutionary loop. In each generation, the population is crossed, mutated, and evaluated using the methods outlined in this section. Once a predetermined number of generations is completed, the EA ends and the candidate controller with the highest fitness is considered the best evolved controller.

A. Crossing

Crossing begins by selecting two unique parent controllers from the current generation. Each parent has an equal probability of being selected. The child controller is created by using a uniform crossover technique in which each child anchor is randomly selected from the corresponding anchors of the two parents. This process is repeated until P children are created. The new set of children controllers are then added into the current population to create a combined population of size $2P$.

B. Mutation

After the crossing is complete, the combined population (comprised of both parents and children) is passed through a mutation operation. There are two distinct types of mutation that can occur; *major* and *minor* mutations. The probability for these mutations to occur to a given anchor point is given by their respective mutation rates: μ_M for major mutation and μ_m for minor mutation.

Major mutations randomly change the value of the control associated with a given anchor point. For example, if an anchor point has a control value of `pursue`, a major mutation can change it to a control value of `drive`. This can have a drastic effect on the performance of the controller and therefor is assigned a relatively small probability of happening: $\mu_M = 1\%$.

Minor mutations shift the position of a given anchor point a small amount in a random direction. Given an anchor point $\hat{\mathbf{x}}$ with weight w, the mutated version $\hat{\mathbf{x}}'$, w' can be computed as

$$\hat{\mathbf{x}}' = \hat{\mathbf{x}} + \mathbf{R}, \qquad w' = w + \text{Uniform}(0,1)$$

where $\mathbf{R} \in \mathbb{R}^4$ is a vector of uniform random numbers in the range 0 to 1. Minor mutations are designed to slightly modify the boundaries between the different regions of control and thus will usually have a relatively small effect on the controllers performance. Minor mutations are applied to anchor points with a probability of $\mu_m = 10\%$

C. Fitness Evaluation

After the crossing and mutation operations have been completed, the combined population is then evaluated and each candidate controller is assigned a fitness. The fitness is defined as the average utility of 9 different simulations (all with the same settings) with various initial conditions. Figure 3 shows all 9 of the sets of initial conditions. These same 9 configurations are used to assess the fitness in every generation. All of the configurations begin with the prey concentrated in a circular ball. In the first column of Fig. 3, the prey start with zero velocity; the second column has the prey all moving in the same direction at max velocity; and the third column has

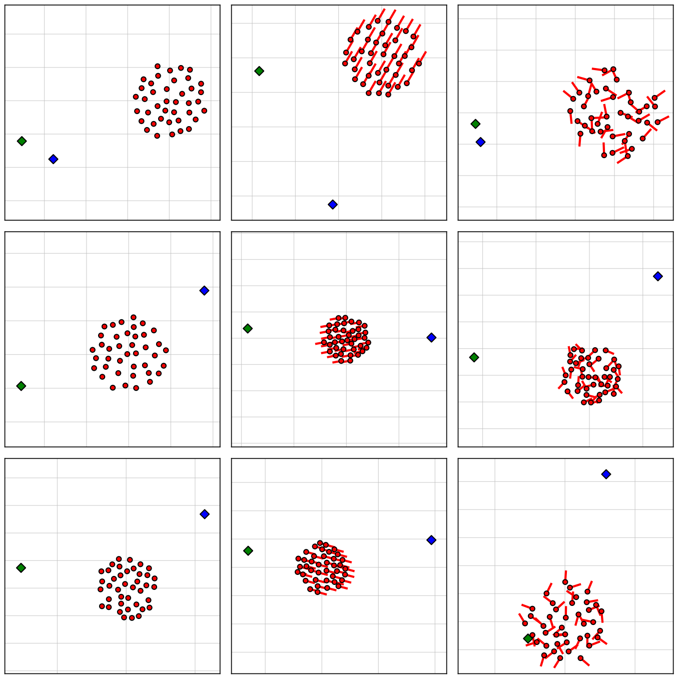

Fig. 3. Suite of initial conditions to simulate to determine fitness of an individual controller.

the prey moving with random velocities. For the predators, the first row of Fig. 3 starts the predators relatively close together; the second row places the two predators nearly opposite one another w.r.t. the prey and at similar distances; and the third row starts the predators nearly opposite one another but with one of the predators closer to the prey. The purpose of the different initial conditions is to expose the controller to a variety of scenarios in an effort to avoid over-fitting to a particular configuration. Once all the agents are assigned a fitness, the top 30% of the combined population are selected along with $0.4P$ individuals chosen from the bottom 70% to pass on to the next generation.

Table I summarizes the settings used for the EA simulation in the following section. Note the odd value for population size is due to the number of cores available on the computer in which the EA was run. In each generation, the fitness evaluation is performed in parallel (one individual per core).

TABLE I
EA PARAMETER SETTINGS

Parameter	Value	Description
n	10	maximum number of anchor points
μ_m	0.1	minor mutation probability
μ_M	0.01	major mutation probability
G_max	100	number of generations
P	190	population size

VI. RESULTS

This section contains the results of the EA as well as a Monte Carlo simulation comparing the best evolved controller to two baseline controllers across many different initial conditions not previously seen by the EA. Table II contains the simulation parameters used in all of the experiments. Note that the EA-learned controller was trained for these particular settings of prey virtual force weights. Thus the EA-learned controller is specifically tuned to this particular prey behavior. The cohesion weight, w_coh, in particular, is quite high in comparison to the other forces. This is to encourage the prey to cluster together more since we are most interested in examining how predators should approach/maneuver around clusters of prey. For most cases, T is not large enough for the predators to capture all of the prey.

TABLE II
SIMULATION PARAMETER SETTINGS

Parameter	Value	Description
M	40	number of prey
N	2	number of predators
Δt	0.01	timestep
T	15	final time
v_E	0.1	max evader speed
v_P	0.2	max pursuer speed
r	0.6	prey sensing range
s	0.05	desired separation
d_c	0.01	capture distance
w_coh	7	weight for cohesion force
w_ali	0.1	weight for alignment force
w_sep	1	weight for separation force
w_avo	2	weight for avoiding the predator

Figure 4 shows the evolution of the best and average utility (over the whole population) for 100 generations. As indicated by the consistent gap between the best and average fitnesses at each generation, some diversity within the population is maintained throughout the evolution. By the end of the 100 generations, the best evolved controller's utility is 20% higher than the baseline controller, which always uses the `pursue` behavior (aim at the nearest prey).

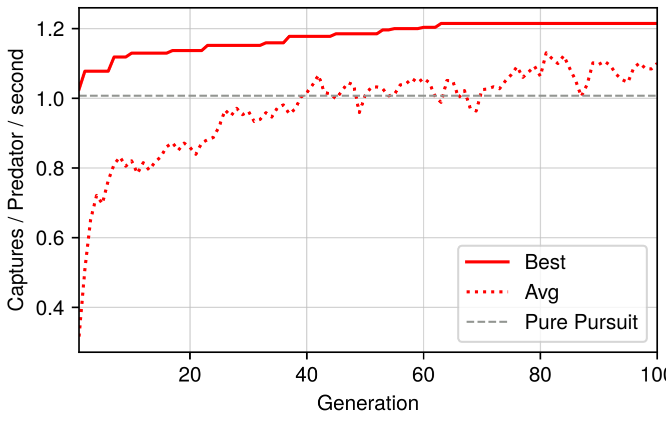

Fig. 4. Evolutionary algorithm results - best and average normalized fitness for each generation. The gray line is the average performance of the baseline Pure Pursuit controller on the test suite.

Figure 5 contains an animation of the best evolved controller starting from a particular initial condition (seen during evolution). For most of the simulation, the predators utilize the `pursue` behavior; Table III summarizes the amount of time spent using each of the behaviors. Interestingly, the best

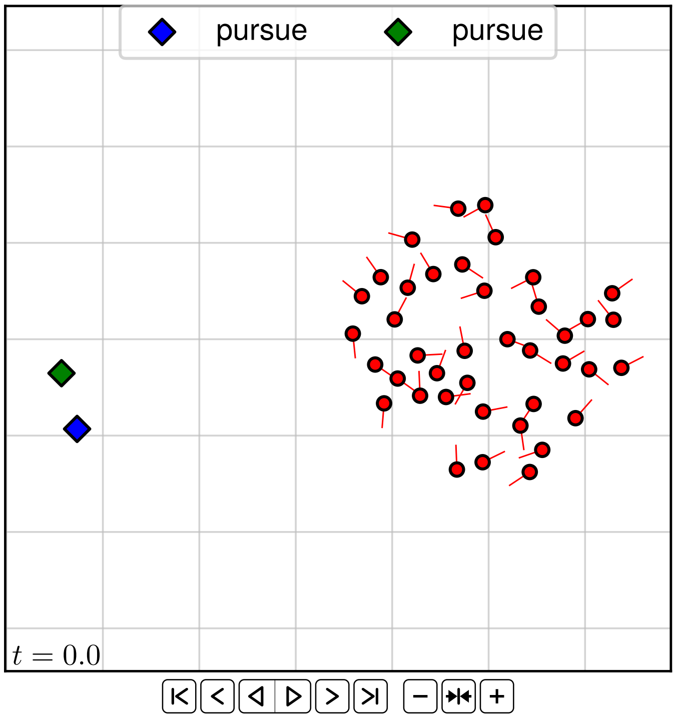

Fig. 5. Simulation of the best controller evolved after 100 generations on one of nine initial conditions in which it was tested. Also available at avonmoll.github.io/files/pred_prey.gif

TABLE III
SUMMARY OF BEHAVIORS USED IN SIMULATION SHOWN IN FIG. 5

Behavior	$\mathbf{P}_1$ Percentage of Time	$\mathbf{P}_2$ Percentage of Time
`pursue`	99.6	93.4
`converge`	0	0
`diverge`	0	0
`drive`	0	0.13
`flank`	0.4	6.47

evolved controller's anchor points are comprised only of the `pursue`, `drive`, and `flank` behaviors. Neither `converge` nor `diverge` are even present in the controller. It appears that these two behaviors were mostly "evolved out" of the population (or at least moved to a remote area of the meta state space, thereby limiting its activation). Because individual predators are capable of capturing individual prey agents alone, the `converge` behavior is almost never necessary. As far as `diverge` goes, the `flank` behavior appears to be a slightly more useful means of creating separation between the predators when they come too close. Although `drive` and `flank` are used, they are used sparingly, suggesting they are useful only in specific circumstances. Nonetheless, the results in Table III suggest that behaviors that are active for a small amount of time can have a large influence on the overall utility.

Because the controllers in the EA saw the same 9 initial conditions in each generation, over fitting is a concern. It is possible that the controllers learned only how to handle these 9 initial conditions and may generalize poorly for other scenarios. In order to corroborate the performance gain of the evolved controller over the baseline (as shown in Fig. 4) a Monte Carlo experiment is run over 1000 different (previously unseen) initial conditions. Figure 6 contains the results of the Monte Carlo experiment. Compared to the Pure Drive controller (i.e. always use `drive`), the evolved controller typically captures three times the number of prey. The utility gain over the Pure Pursuit controller is more modest – the peak of the histogram occurs at 1, meaning the two controllers have the same utility. Pure Pursuit outperforms the evolved controller in less than 14% of the experiments, and the loss is never more than 23%. The evolved controller performs 100% better than Pure Pursuit for several cases. Pure Pursuit's biggest drawback is that it's possible for the predators to end up quite close to one another; thereafter the two predators make the same moves, always pursuing the same prey, which is clearly a waste. Although the evolved controller only saw 9 initial conditions during evolution it was able to generalize reasonably well, generally matching or outperforming the baseline controllers.

VII. CONCLUSION

We considered a predator-prey scenario in which the prey's behavior is governed by a Boids-like flocking model and the predators cooperate to maximize the number of captures within the time horizon. One of the purposes of the paper was to demonstrate the efficacy of controllers based on a small set of pre-specified behaviors for a many-on-many type of adversarial engagement. Moreover, we employed an evolutionary algorithm to determine the appropriate base behavior to implement as a function of a meta state space. The meta state space we used here was independent of the number of prey and thus the predator controller is scalable. In essence, the meta state space is just a feature space; it is possible that other features or functions of features could be useful

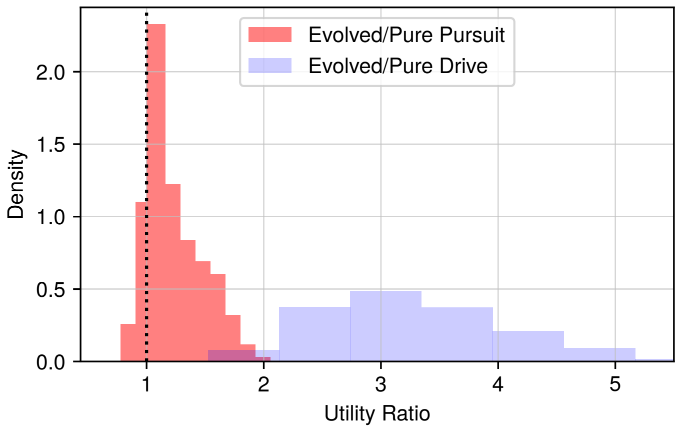

Fig. 6. Monte Carlo results for 1000 simulations - histograms of the utility ratio of the best evolved controller compared to two baseline controllers (always `pursue` and always `drive`).

in determining an appropriate behavior for this particular scenario. We introduced the notion of weighting onto the existing anchor point control architecture in order to have finer control over the partitioning of the meta state space. Based on the simulation results, the optimal (or approximately optimal) behavior for the two predator case is to aim at the nearest prey while avoiding getting too close to the other predator. The evolved controller was shown to have this type of behavior, generally, and thus it performed much better than always aiming at the nearest prey, in most cases.

Analytical solutions to many-on-many pursuit evasion problems like this one do not exist. However, approximately optimal solutions are often intuitive. The approach used in this study yielded an overall strategy that deconflicted the predators when necessary and otherwise employed a control that's optimal under several assumptions. It is possible, perhaps, to design a controller which plans a sequence of targets to pursue that is more optimal than always aiming at the nearest. However, that approach would be much more computationally complex; deconfliction would also require either explicit communication or heuristics to avoid wasting resources. Thus the advantages of our approach are that it is scalable, decentralized, and simple.

We aimed to show emergent cooperative predator behaviors along similar lines as those described in the referenced literature. The cooperation that was evolved in this study is intuitive, but the predators do not spend much time "herding" the prey or otherwise limiting prey dispersion. It is possible that, based on our definition of the utility and the settings of the prey model parameters, this type of behavior is unnecessary. Future research efforts may explore the simulation parameter space, consider heterogeneous predators (i.e. each predator has its own anchor points, and, perhaps, role), specifying different utility functionals (for example, requiring two predators to capture, or explicitly including prey dispersion in the utility), and, of course, studying larger scenarios with more predators.

ACKNOWLEDGMENT

This paper is based on work performed at the Air Force Research Laboratory (AFRL) *Control Science Center of Excellence*. Distribution Unlimited. 13 May 2020. Case #88ABW-2020-1747. The views expressed in this paper are those of the authors and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

REFERENCES

[1] C. Anderson, G. Theraulaz, and J.-L. Deneubourg, "Self-assemblages in insect societies," *Insectes Sociaux*, vol. 49, pp. 99–110, 5 2002.

[2] B. Graw and M. B. Manser, "The function of mobbing in cooperative meerkats," *Animal Behaviour*, vol. 74, pp. 507–517, 09 2007.

[3] R. Olendorf, T. Getty, and K. Scribner, "Cooperative nest defence in red–winged blackbirds: reciprocal altruism, kinship or by–product mutualism?" *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 271, pp. 177–182, 01 2004.

[4] D. C. Heard, "The effect of wolf predation and snow cover on musk-ox group size," *The American Naturalist*, vol. 139, pp. 190–204, 1 1992.

[5] S. L. Tilahun and H. C. Ong, "Prey-predator algorithm: A new metaheuristic algorithm for optimization problems," *International Journal of Information Technology & Decision Making*, vol. 14, pp. 1331–1352, 11 2015.

[6] M. Steinegger, D. G. Roche, and R. Bshary, "Simple decision rules underlie collaborative hunting in yellow saddle goatfish," *Proceedings of the Royal Society B: Biological Sciences*, vol. 285, p. 20172488, 1 2017.

[7] J. O. Sullivan, *Variability in the Wolf, a Group Hunter*. Elsevier, 1978, pp. 31–40.

[8] S. K. Gazda, R. C. Connor, R. K. Edgar, and F. Cox, "A division of labour with role specialization in group–hunting bottlenose dolphins (tursiops truncatus) off cedar key, florida," *Proceedings of the Royal Society B: Biological Sciences*, vol. 272, pp. 135–140, 1 2005.

[9] l. P. Croft, R. James, l. O. R. Thomas, C. Hathaway, D. Mawdsley, l. N. Laland, and J. Krause, "Social structure and co-operative interactions in a wild population of guppies (poecilia reticulata)," *Behavioral Ecology and Sociobiology*, vol. 59, no. 5, pp. 644–650, 3 2006.

[10] A. Weitzenfeld, A. Vallesa, and H. Flores, "A biologically-inspired wolf pack multiple robot hunting model," in *2006 IEEE 3rd Latin American Robotics Symposium*. IEEE, 10 2006.

[11] L. Strickland, K. Baudier, K. Bowers, T. P. Pavlic, and C. Pippin, "Bio-inspired role allocation of heterogeneous teams in a site defense task," in *Distributed Autonomous Robotic Systems*, N. Correll, M. Schwager, and M. Otte, Eds. Springer International Publishing, 2019, pp. 139–151.

[12] D. Floreano and F. Mondada, "Automatic creation of an autonomous agent: Genetic evolution of a neural network driven robot." The MIT Press, 1994, pp. 421–430.

[13] T. Haynes and S. Sen, "Evolving behavioral strategies in predators and prey," in *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1996, pp. 113–126.

[14] P. Androulakakis and Z. E. Fuchs, "Evolutionary design of engagement strategies for turn-constrained agents," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, 6 2017, pp. 2354–2363.

[15] L. G. Strickland, E. G. Squires, M. A. Day, and C. E. Pippin, "On coordination in multiple aerial engagement," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 6 2019.

[16] J. H. Connell, *Minimalist mobile robotics*. Elsevier, 2012, vol. 5.

[17] V. Braitenberg, *Vehicles: Experiments in synthetic psychology*. MIT press, 1986.

[18] R. Isaacs, *Differential Games: A Mathematical Theory with Applications to Optimization, Control and Warfare*. Wiley, New York, 1965.

[19] E. Garcia, Z. E. Fuchs, D. Milutinović, D. W. Casbeer, and M. Pachter, "A geometric approach for the cooperative two-pursuer one-evader differential game," *IFAC-PapersOnLine*, vol. 50, pp. 15 209–15 214, 2017.

[20] J. V. Breakwell and P. Hagedorn, "Point capture of two evaders in succession," *Journal of Optimization Theory and Applications*, vol. 27, pp. 89–97, 1979.

[21] Z. E. Fuchs, P. P. Khargonekar, and J. Evers, "Cooperative defense within a single-pursuer, two-evader pursuit evasion differential game," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 3091–3097.

[22] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *the 14th annual conference*. ACM Press, 1987.

[23] J. Schrum and R. Miikkulainen, "Evolving multi-modal behavior in npcs," in *2009 IEEE Symposium on Computational Intelligence and Games (CIG)*. IEEE, 9 2009.

[24] J. Drake, A. Safonova, and M. Likhachev, "Towards adaptability of demonstration-based training of npc behavior," in *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2017.

[25] M. Van Lent, W. Fisher, and M. Mancuso, "An explainable artificial intelligence system for small-unit tactical behavior," in *IAAI Emerging Applications*. AAAI, 2004.

[26] R. E. Korf, "A simple solution to pursuit games," in *Working Papers of The 11th International Workshop on DAI*, 1992, pp. 183–194.

[27] B. Silverman, *Density Estimation for Statistics and Data Analysis*, ser. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1986.