Genet Program Evolvable Mach (2012) 13:493–536
DOI 10.1007/s10710-012-9166-5

Evolving team behaviors with specialization

G. S. Nitschke · A. E. Eiben · M. C. Schut

Received: 16 September 2011 / Revised: 18 April 2012 / Published online: 16 May 2012

Abstract This article evaluates *Collective Neuro-Evolution* (CONE), a cooperative co-evolutionary method for solving collective behavior tasks and increasing task performance via facilitating behavioral specialization in agent teams. Specialization is used as a problem solving mechanism, and its emergence is guided and regulated by CONE. CONE is comparatively evaluated with related methods in a simulated evolutionary robotics pursuit-evasion task. This task required multiple pursuer robots to cooperatively capture evader robots. Results indicate that CONE is appropriate for evolving *specialized* behaviors. The interaction of specialized behaviors produces behavioral heterogeneity in teams and collective prey capture behaviors that yield significantly higher performances compared to related methods.

Keywords Behavioral specialization · Neuro-evolution · Pursuit-evasion · Simulation · Multi-robot systems

1 Introduction

In nature numerous examples of *collective behavior systems* are observable. Collective behavior systems are defined as those that are composed of many individuals, where cooperative task accomplishment is required in order for individuals and the group to survive. Many artificial collective behavior systems

G. S. Nitschke (✉)
Ikegami Lab, Department of General Systems Studies, University of Tokyo, Tokyo, Japan
e-mail: geoff@sacral.c.u-tokyo.ac.jp

A. E. Eiben · M. C. Schut
Computational Intelligence Group, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
e-mail: gusz@cs.vu.nl

M. C. Schut
e-mail: schut@cs.vu.nl

have used design principles which draw their inspiration from examples of specialization in nature. Examples include complex ecological communities such as social insect colonies [12, 13, 17, 78, 94, 103], biological neural networks [3], multi-cellular organisms [40], economies of a nation and companies [1, 73, 90]. In endeavors to replicate the success of biological collective behavior systems, such as in the engineering of robotic swarms [93], it is highly desirable to reproduce the underlying mechanisms of biological collective (cooperative) behavior [11]. One such mechanism is *behavioral specialization* [75], referring to agent behaviors that are beneficial for solving specific types of tasks [5, 57, 72, 98, 102].

Consider the biological collective behaviors of certain ant species, where workers specialize to varying tasks, adapting their behavior to varying situations and adopting complementary roles [91]. Such emergent specialization has been used as a biologically inspired design principle in real world multi-agent optimization and control tasks. For example, agent-based resource allocation models regulate the frequency of specialized behavior activation in response to dynamically arising tasks [13, 17, 33]. A specific example is the application of response threshold models of division of labor to solve dynamic task-scheduling problems [11, 12].

This article presents a *cooperative co-evolution* method, *Collective Neuro-Evolution* (CONE). CONE evolves agent behavioral specialization most appropriate for solving collective behavior tasks. Cooperative co-evolution [105] has been demonstrated as versatile and applicable to a broad range of complex, continuous, and noisy tasks [18, 48]. Using multiple *species* (genotype populations) is a natural representation for many collective behavior tasks [9, 14, 76]. Also, behavioral specialization is often effectuated by evolved species in response to task and environment constraints [54, 88].

This article establishes the effectiveness of CONE via demonstrating its capability for effectuating beneficial forms of behavioral specialization in an *Evolutionary Robotics* (ER) [82] pursuit-evasion [7] task. Previous research has demonstrated that the pursuit-evasion task benefits from behavioral specialization [9, 74, 109].

In the theme of pursuit-evasion research that uses specialization as a problem solving mechanism, Haynes and Sen et al. [42, 43, 44, 45] compared *Genetic Programming* (GP) [50] approaches for evolving cooperative prey capture in agent teams. Evolved prey capture behaviors relied upon emergent behavioral specialization, rather than domain specific knowledge. In related GP research, Luke and Spector [57] applied various GP methods to evolve collective prey capture behaviors in homogeneous versus heterogeneous agent teams (that is, teams comprised of behaviorally specialized agents). Heterogeneity and homogeneity was determined by different GP breeding strategies within one population of GP trees. Results indicated that heterogenous breeding evolved specialized behaviors that outperformed homogenous breed teams. In research that combined *Neuro-Evolution* (NE) [108] and cooperative co-evolution, Yong and Miikkulainen [109] evolved specialized behaviors in cooperative prey capture strategies in agent teams. Specialized roles complemented each other to form cooperative prey capture behaviors.

In work that used physical robots in pursuit-evasion, Blumenthal and Parker et al. [8, 9, 10] combined *Punctuated Anytime Learning* [84] and co-evolution to exploit differences in predator robot morphologies. This co-evolution resulted in the derivation of behavioral specialization as robots interacted to solve a collective prey capture task. In other pursuit-evasion work that used robot teams, Potter and Meeden [88] applied cooperative co-evolution to evolve controllers in behaviorally homogeneous versus heterogeneous teams. Experiments found that as task difficulty increased, heterogeneity and specialization became essential for task accomplishment.

These different approaches to solving the pursuit-evasion task in simulated (agent) and physical (embodied) systems, demonstrate that given appropriate task and environment constraints, specialization emerges as a problem solving mechanism. This notion is the defining motivation of this article's research.

The efficacy of CONE has thus far been demonstrated in a multi-rover collective behavior task [76] and a multi-robot collective construction task [77]. CONE combines NE and cooperative co-evolution [87] to adapt *Artificial Neural Networks* (ANN) controllers [41], operating in collective behavior tasks [75]. NE is the adaptation of ANNs using artificial evolution [108]. NE has been advocated as an appropriate means of controller adaptation in continuous and partially observable task environments [31].

One advantage of NE is that details about how a task is to be solved does not need to be specified *a priori* by the system designer. Rather, a simulator is used to derive, evaluate and adapt controller behaviors for a given task [65]. NE has been successfully applied to solve a disparate range of collective behavior tasks including multi-agent computer games [96], RoboCup [104], and cooperative transport [80], and collective construction [77] in robot teams.

Cooperative co-evolution methods work via decomposing a given task into composite sub-tasks that are *cooperatively* solved by a set of artificial species [105]. Cooperative co-evolution methods use cooperation between multiple genotype populations (species) and competition between genotypes (individuals) in a species to derive solutions. The use of multiple genotype populations provides a natural representation for many collective behavior tasks. Furthermore, such multi-population representations aid in facilitating behavioral specialization in multi-agent models [75].

The advantages of cooperative co-evolution include versatility and applicability to a broad range of complex, continuous, and noisy tasks. For example, cooperative co-evolution (combined with NE) has been applied to solve collective behavior tasks such as multi-agent game playing [14], collective prey-capture [9, 88] by robot teams, and multi-rover search [76].

In such cases, the combination of cooperative co-evolution and NE was an appropriate means of adapting the collective behaviors of ANN controllers, as well as facilitating specialization in controller behavior. An overview of all cooperative co-evolution methods that have been applied to solve collective behavior tasks is beyond the scope of this article. However, Wiegand [105] includes a relatively recent review of cooperative co-evolution.

This research evaluates CONE in comparison with two related cooperative co-evolution methods, *Multi-Agent Enforced Sub-Populations* (MESP) [109], and the *Cooperative Co-Evolutionary Algorithm* (CCGA) [86] in the context of ER experiments. CCGA and MESP were selected since both are appropriate for facilitating specialized ANN controller behaviors in simulated multi-robot [88], and multi-agent [14] collective behavior tasks.

This article contributes self-regulating *genotypic* and *behavioral* metrics that effectuate behavioral specialization in simulated robot teams, where such specialization increases team fitness. The *research goal* was to demonstrate the efficacy of CONE for evolutionary controller design of collective behaviors, via effectuating behavioral specialization. The *research hypothesis* was that CONE's genotype and behavioral specialization metrics adaptively regulate recombination and facilitate behavioral specialization resulting in CONE evolved teams yielding a higher average fitness, compared to CCGA and MESP.

Genotypic and behavioral metrics have been used in previous ER experiments to encourage genotype and behavioral diversity [53, 54, 67–70]. Such studies indicated that genotype, and especially behavioral, based metrics resulted in substantial improvements in solution convergence rates in a wide range of ER tasks. Also, there is empirical evidence that genotypic and behavior based metrics allow a wider range of ER tasks to be solved [24]. Diversity mechanisms such as *fitness sharing* [35, 92], *crowding* [58, 59], *multi-objective evolutionary algorithms* [15, 22, 24, 69, 70, 99], or *novelty search* [51] have been used to encourage genotype and behavioral diversity and increase task performance in various task domains, including ER [71].

In this research, behavioral specialization in teams is encouraged via genotypic and behavioral distance metrics that regulate inter-population genotype recombination, based upon agent (controller) behavioral and genotype similarities. This directs the emergence and propagation of beneficial agent specializations, resulting in the evolution of effective collective prey capture behaviors.

This article's goal is to demonstrate CONE's capability to facilitate behavioral specialization in collective behavior tasks that *require* and *benefit from* specialization. Pursuit-evasion (this article's case study) is one such task. This article's case study was an ER task that simulated robot teams (predators) that had to immobilize (capture) one or two other robots (prey) in a bounded environment. This pursuit-evasion task required different predators in a team to assume complementary behaviorally specializations in order for the team to achieve an optimal or near optimal task performance [74]. A thorough analysis of the pursuit-evasion task was conducted to support this.

Pursuit-evasion results, and a subsequent analysis indicates that CONE's computational and algorithmic complexity, coupled with the pursuit-evasion task (a task environment that encourages behavioral specialization) results in the evolution of team behaviors that could not be evolved by related controller adaptation methods. That is, a thorough analysis of CONE elucidates the contribution and impact of many of CONE's algorithmic mechanisms to the task performance of CONE evolved teams. The algorithmic mechanisms examined are those that distinguished CONE from related cooperative co-evolution controller adaptation methods. Such mechanisms included the genotype and specialization

difference metrics to regulate inter-population recombination, and an elite controller evaluation procedure. The aim of this analysis is to demonstrate CONE as a general algorithm for facilitating behavioral specialization in agent teams. Importantly, CONE's advantages are not limited to the pursuit-evasion task, but have been demonstrated in other tasks that benefit from the collective problem solving of behaviorally specialized agents [76, 77].

It is supposed that the efficacy of CONE is not limited to simulated ER tasks such as pursuit-evasion, multi-rover surveillance and collective gathering and construction. It is anticipated that CONE is applicable to a diverse range of multi-agent (collective behavior) tasks, where different agents in the system are required to adopt complementary specialized functions in order to form effective collective behavior solutions. For example, it is envisioned that non-linear process control in reactors [21, 20], and electrical power grids [34] is within the purview of CONE's problem solving capabilities.

2 Methods: collective neuro-evolution (CONE)

CONE is a controller design method [36, 87] that uses cooperative co-evolution to adapt behaviors in a team of ANNs (agent controllers). Given n genotype populations (species), n controllers are evolved. Controllers are evaluated according to how well they solve a collective behavior task (requiring agents to cooperate). Each controller is a recurrent feed-forward ANN with one hidden layer fully connected to input and output layers. CONE evolves the input–output connection weights of hidden layer neurons, and within each species combines the fittest neurons into complete ANN controllers.

CONE extends *Multi-Agent Enforced Sub-Populations* (MESP) [109], with two novel contributions. First, CONE solves collective behavior tasks via purposefully evolving behavioral specialization in agents. When these specialized agent behaviors interact, the team is able to increase task performance and solve collective behavior tasks that could not otherwise be solved. Second, CONE employs *Genotype* and *Specialization Difference Metrics* (GDM and SDM, respectively) to regulate inter-population genotype recombination. Based upon genotype similarities and the success of evolving behavioral specializations, the GDM and SDM direct the evolution of specialized agent behaviors and composite collective behaviors. The GDM and SDM differ from related genotype [107] and behaviorial [4] metrics, in that the GDM and SDM effectuate behavioral specialization in the collective behaviors of agent teams.

From a technical standpoint, the GDM and SDM design motivation was to dynamically ascertain a degree of recombination between genotype populations that is appropriate for evolving specialized behaviors (and collective behaviors) most suited to the given task. From a biological standpoint, the GDM and SDM design was inspired by the *genotypic cluster definition* [60] and simulates the gradualism and fuzziness of the speciation process [63]. That is, the GDM and SDM work under the assumption that *isolation barriers* between populations delimiting species boundaries, undergo evolution. This means that isolation barriers and speciation are constantly changing [19, 38, 61].

The use of the GDM and SDM as mechanisms to regulate inter-population recombination is also supported by research on *partially heterogeneous* populations. A partially heterogeneous population is comprised of groups that are, on average, more genetically similar (but not identical) to individuals of their own group, comparative to the rest of the population [101]. In this article, such groups are defined as *species*. The impact of partial genetic heterogeneity on the evolution of group behaviors, especially with respect to the evolution of multiple, complementary specialized behaviors has received little investigation in evolutionary multi-agent research. However, Luke and Hohn [56] and Luke [55] suggest that partial genetic heterogeneity in an evolving agent group can lead to specialized behaviors. This is supported by studies in biology [39, 52]. Also, Perez et al. [85], and Waibel et al. [101] indicated that team fitness increases were related to selection within genetically related agents. As an extension of this notion, the GDM and SDM suppose that recombining genetically *and* behaviorally related agents increases team task performance, or allows the team to solve tasks that could not otherwise be solved. The design choices for CONE's representation and iterative process were motivated by the following results. First, MESP and related methods have successfully solved collective behavior tasks [14, 109]. Second, evolutionary methods applied to genetically heterogeneous agent teams (where agents are defined by different genotypes) often results in agents evolving specialized behaviors [6, 56]. This is especially the case for cooperative co-evolutionary methods [32, 76].

Parameter calibration experiments (Sect. 3.5) determined the most appropriate parameter values for CONE in the pursuit-evasion task (Sect. 3).

2.1 Representation: multi-population structure

As with related NE methods [36, 86], CONE segregates the genotype space into n populations so as to evolve n controllers. CONE mandates that ANN_i ($1 \leq i \leq n$) is derived from population P_i, where P_i contains u_i sub-populations (that is, initial number of sub-populations in population i). Figure 1 exemplifies the use of sub-populations in CONE. ANN_1 and ANN_2 (evolved from populations 1 and 2, respectively) has three hidden layer neurons, whilst ANN_3 (evolved from population 3) has four hidden layer neurons. Thus, populations 1 and 2 consist of three sub-populations, for evolving the three neurons in ANN_1 and ANN_2. Where as, population 3 uses four sub-populations for evolving the four neurons in ANN_3. ANN_i is derived from P_i via selecting one genotype from each sub-population and decoding these genotypes into hidden layer neurons (Fig. 2). ANN_i consists of w input neurons, and v output neurons, fully connected to all hidden layer neurons.

The CONE process is driven by cooperation and competition within and between sub-populations and populations. Competition exists between genotypes in a sub-population that compete for a place as a hidden layer neuron in the fittest controller. Cooperation exists between sub-populations, in that fittest genotypes selected from each sub-population must cooperate as a controller. There was also cooperation between controllers since controllers must cooperate to solve a collective behavior task.

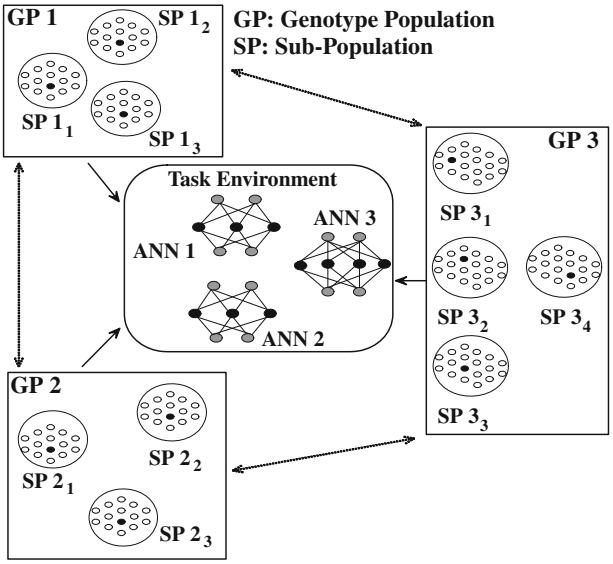

Fig. 1 CONE example. An ANN is evolved from each genotype population. All ANNs are evaluated in a collective behavior task (task environment). *Double ended arrows* indicate self-regulating recombination between populations. *ANN* artificial neural network, *GP X* genotype population X, *SP* X_z sub-population z in genotype population X

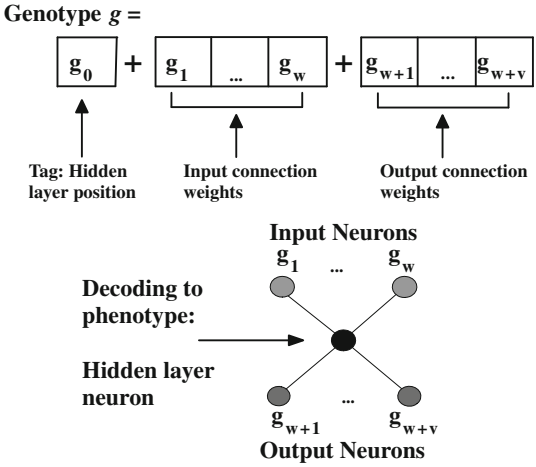

Fig. 2 CONE genotype. A genotype g directly maps to a hidden layer neuron. A genotype has w genes for a neuron's input connection weights, and v genes for the neuron's output weights. A tag (g_0) specifies the neuron's position in a controller's hidden layer, and hence the neuron's sub-population, within a given population (species)

2.2 Behavioral specialization

An integral part of CONE is defining and measuring controller specialization. A controller's *behavioral specialization* (S) is defined by the frequency with which a controller switches between executing distinct motor outputs (actions) during its lifetime. The S metric is an extension of that defined by Gautrais et al. [33]. The S metric was selected since it is applicable to individual controller behaviors,

accounts for a partitioning of a controller's work effort among different actions, and is simple enough to work within CONE. The S metric is also general enough to define specialization as the case where controllers regularly switch between different actions, spending an approximately equal portion of its lifetime on each action, but where there is a slight preference for one action. Equation 1 specifies the calculation of S, that is, the frequency with which a controller switches between each of its actions in its lifetime, A is the number of times the controller switches between different actions, and N is the total number of possible action switches. At least two distinct agent actions are assumed.

$$S = \frac{A}{N} \tag{1}$$

An S value close to zero indicates a high degree of specialization. In this case, a controller specializes to one action, and switches between this and its other actions with a low frequency. An S value close to one indicates a *low degree of specialization*. In this case, a controller switches between some or all of its actions with a high frequency. A perfect specialist ($S = 0$), is a controller that executes the same action for the duration of its lifetime ($A = 0$). An example of a non-specialist ($S = 0.5$) is where a controller spends half of its lifetime switching between two actions. For example, if $A = 3$, $N = 6$, then the controller switches between each of its actions every second iteration. Controllers are labeled as *specialized* if S is less than a given *Behavioral Specialization Threshold* (BST). Otherwise, controllers are labeled as *non-specialized*.

2.3 Adaptation of algorithmic parameters

The GDM and SDM adaptively regulated inter-population genotype recombination as a function of controller fitness progress. Regulating inter-population is integral to the CONE adaptive process. That is, CONE works on the premise that recombining populations containing *genetically similar* genotypes and produce behaviorally similar behaviors (with beneficial specializations) will result in the evolution of increasingly beneficial specialized behaviors.

As part of the regulation process, two dynamic algorithmic parameters, the *Genetic Similarity Threshold* (GST), and *Specialization Similarity Threshold* (SST) were used by the GDM and SDM, respectively. The initial GST and SST values are floating point values randomly initialized in the range: [0.0, 1.0]. Whenever the GST value is adapted by the GDM, a static value (δGST) is either added to, or subtracted from the GST value. Similarly, when the SST value is adapted by the SDM, a static value (δSST) is either added to, or subtracted from the SST value. The following describes the GDM and SDM.

2.3.1 Genotype difference metric (GDM)

The GDM is a heuristic that adaptively regulates inter-population recombination of genetically similar genotypes. Two genotypes $\bar{a}$ and $\bar{b}$ are considered genetically

similar if their average weight difference is less than *GST* [107]. The GST value, and hence inter-population genotype recombination, is adapted as a function of the number of previous *W* recombinations and a team's average fitness progress (the fittest *n* controllers). The following rules regulated the GST value and thus the number of inter-population recombinations.

1. If recombinations between populations have increased over the previous *W* generations, and fitness has stagnated or decreased, then decrement the GST value, so as to restrict the number of recombinations.
2. If recombinations between populations have decreased or stagnated, and fitness has stagnated or decreased over the last *W* generations, then increment the GST value, to increase the number of recombinations.

Similar genotypes in different populations may encode very different functionalities. Recombining such genotypes may produce neurons that do not work together as a controller. The SDM addresses this problem.

2.3.2 *Specialization difference metric (SDM)*

The SDM adaptively regulates inter-population recombination based on behavioral specialization similarities exhibited by controllers. The SDM ensures that only the genotypes that make up controllers with similar behaviors are recombined. The SDM defines the specialized behaviors of controllers ANN_i and ANN_j to be similar if the following conditions are true:

1. $| S(ANN_i) - S(ANN_j) | < SST$, where, S (Eq. 1 in Sect. 2.2) is the *degree of behavioral specialization* exhibited by ANN_i and ANN_j.
2. If ANN_i and ANN_j have the same *specialization label*.

The *specialization label* is the most executed action of a specialized controller. The SST value is adapted as a function of controller behavioral specialization (S) similarities and a team's average fitness progress. The following rules were used to regulate the SST value.

1. If the S of at least one of the fittest controllers has increased over the last V generations, and average team fitness stagnates or is decreasing over this same period, then decrement the SST value. Thus, if the fittest controllers have an average S that is too high for improving team fitness, then recombination between populations is restricted.
2. If the S of at least one of the fittest controllers has decreased over the last V generations, and average fitness stagnates or is decreasing over this same period, then increment the SST value. Thus, if the fittest controllers have an average S that is too low to improve team fitness, then allow for more recombination between populations.

2.4 Collective neuro-evolution (CONE) process

This section overviews CONE's iterative cooperative co-evolution process.

1. *Initialization*. n populations are initialized. Population P_i $(i \in \{1, \ldots, n\})$ contains u_i sub-populations. Sub-population P_{ij} contains m genotypes. P_{ij} contains genotypes encoding neurons assigned to position j in the hidden layer of ANN_i (ANN_i is derived from P_i).
2. *Evaluate all genotypes*. Systematically select each genotype g in each sub-population of each population, and evaluate g in a complete controller. This controller (containing g) is evaluated with n-1 other controllers (n is the number of controllers in a team). Other controllers are constructed via randomly selecting a neuron from each sub-population of each of the other populations. Evaluation results in a fitness being assigned to g.
3. *Evaluate elite controllers*. For each population, systematically construct a fittest controller via selecting from an elite portion of genotypes in each sub-population. Controller fitness is equated to *utility*. Utility is the average fitness of the genotypes for a controller's hidden layer. Groups of the fittest n controllers are evaluated together in task simulations until all genotypes in the elite portion of each population have been assigned a fitness. For each genotype, this fitness overwrites previously calculated fitness.
4. *Parent selection*. If the two fittest controllers ANN_i and ANN_j constructed from the elite portions of P_i and P_j have sufficiently similar *behavioral specializations* (Sect. 2.2) then P_i and P_j become candidates for recombination. For P_i and P_j to be recombined, both ANN_i and ANN_j must have the same specialization label (Sect. 2.2). That is, both ANN_i and ANN_j must be behaviorally specialized to the same action. Between P_i and P_j each pair of sub-populations is tested for *genetic similarity* (average weight difference is less than GST). Genetically similar sub-populations are recombined. For sub-populations that are not genetically similar to others, recombination occurs *within* the sub-population. Similarly, for populations that are not behaviorally similar to other populations, recombination occurs *within* all sub-populations of the population.
5. *Recombination*. When sub-populations pairs are recombined, the genotype elite portion in each sub-population is ranked by fitness. Genotypes with the same fitness rank are recombined. For recombination *within* a sub-population, each genotype in the sub-population's elite portion is systematically selected and recombined using one-point crossover [27], with another randomly selected genotype from the sub-population's elite portion.
6. *Mutation*. After recombination, *burst mutation* with a *Cauchy distribution*[1] [37] is applied to each gene of each genotype with a given probability.
7. *Parameter adaptation*. If the fitness of at least one of the n fittest controllers has not progressed in:
   (a) V generations: Adapt *Genetic Similarity Threshold* (GST).
   (b) W generations: Adapt *Specialization Similarity Threshold* (SST).
8. *Stop condition*. Reiterate steps [2, 7] until a desired collective behavior task performance is achieved, or the process has run for X generations.

[1] Herein, referred to as *burst mutation*.

3 Experimental setup

The *pursuit-evasion* task required a *predator* team to collectively capture at least one *prey* in a multi-robot simulation. Predator and prey robots were simulated Khepera mobile robots [66]. Prior to being placed in pursuit-evasion experiments, the prey was evolved with an evasion behavior. Prey did not move deterministically, so it was impossible for predators to consistently predict prey movement. Also, prey capture was made more difficult via giving the prey an advantage of greater speed. Nitschke [74] elucidated that at least two predators were required to accomplish this task, and that predators in a team adopting complementary behavioral specializations yielded the benefit of increasing the time for which a prey was captured. The beneficial forms of predator behavioral specialization, and collective prey capture behaviors were not known *a priori*, and were thus evolved by CCGA, MESP or CONE.

3.1 Continuous simulation environment

The environment was a 1,000 cm $\times$ 1,000 cm continuous area, and was simulated using an extended version of the *EvoRobot Khepera simulator* [79]. Each simulation iteration, a robot (predator or prey) could orientate itself between [0, 360] degrees with respect to its current heading. Robot orientation was calculated according to the speed of each wheel. Assumptions made by the simulation model are described in previous research [74, 79]. Figure 3 depicts an example of three predators and one prey in the simulation environment.

3.2 Predator and prey robots

3.2.1 Prey: sensors and actuators

Prey used eight infrared proximity sensors ([SI-0, SI-7] in Fig. 4a) on its periphery, and a light on its top (L-0 in Fig. 4a). This light was detectable by predator light sensors, and thus attracted predators. When an obstacle came within range of a prey's proximity sensor, that sensor was activated with a value proportional to the distance to the obstacle. Sensor values were normalized within the range: [0.0, 1.0], via dividing the sensor value by the maximum value. Prey were also equipped with two wheels ([MO-0, MO-1] in Fig. 4) that controlled its speed and orientation. Motor output values (MO-0, MO-1) were normalized in the range: [$-$1.0, 1.0], and controlled wheel speed and direction. Where, [MO-0, MO-1] = 0.0, denote no wheel speed, and [MO-0, MO-1] = 1.0, denote maximum wheel.

3.2.2 Prey controller

Prey sensory inputs were mapped to motor outputs using a feed-forward *Artificial Neural Network* (ANN) controller (Fig. 5). Eight sensory input and two motor outputs were fully connected to five *Hidden Layer* (HL) neurons. Sensory inputs encoded the state of the eight infrared proximity sensors. Motor outputs encoded

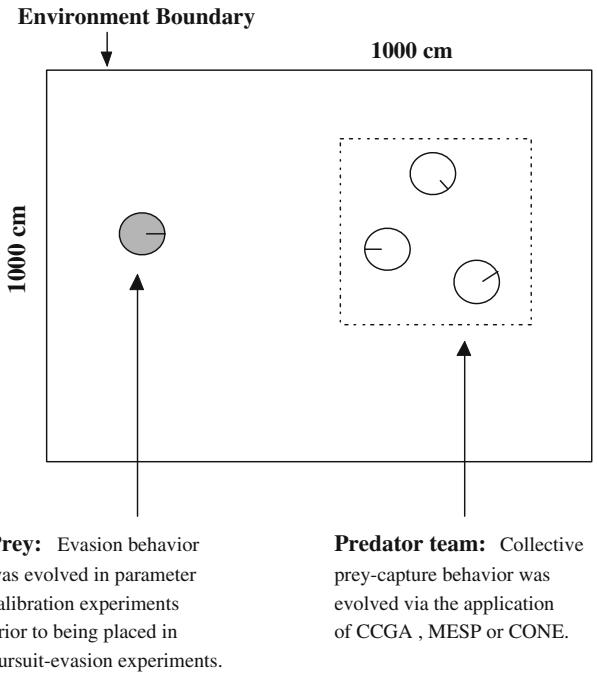

Fig. 3 Simulation environment example. CCGA, MESP or CONE were applied to evolve n (three in this example) predator controllers and collective pursuit behavior

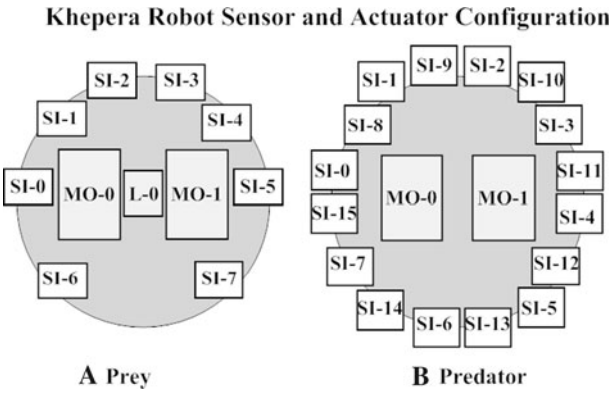

Fig. 4 *Robot Sensor and Actuator Configuration*. Prey (**a**) have only proximity sensors as well as a light. Predators (*b*) have light and proximity sensors. $L-0$: Light, $SI-x$: Sensory input x, $MO-y$: Motor output y

wheel speed. Output values were computed via applying the sigmoid function [47], and multiplying the output value by 1.20. This set prey speed to 20 % faster than predator speed. Also, before prey were placed in pursuit-evasion experiments, the prey controller was evolved with an evasion behavior. Prey controller evolution, and

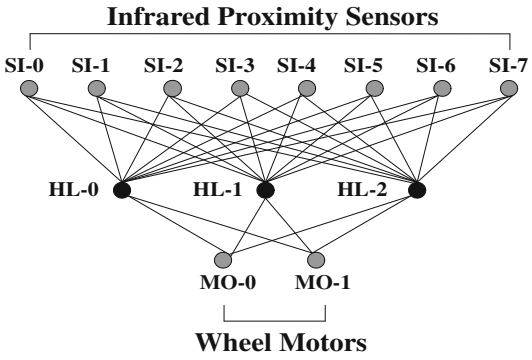

Fig. 5 *Prey Feed-Forward ANN Controller*. Connection weights and the number of hidden layer neurons were evolved prior to being placed in the pursuit-evasion experiments. $SI - x$: Sensory input x, $MO - y$: Motor output y, $HL - z$: Hidden layer neuron z

ascertaining an appropriate number of HL neurons, was done during parameter calibration (section 3.5).

3.2.3 Predator sensors and actuators

Predators were equipped with eight infrared proximity sensors ([SI-0, SI-7] in Fig. 4b), as well as eight light ([SI-8, SI-15] in Fig. 4b) sensors, on its periphery (Fig. 4b). When an obstacle (another predator or wall) came within range of a proximity sensor, that sensor was activated with a value proportional to the distance to the obstacle. Likewise, when a prey came within range of a light sensor, that sensor was activated with a value proportional to the distance to the prey. Sensor values were normalized within the range: [0.0, 1.0]. Predators were also equipped with two wheel motors ([MO-0, MO-1] in Fig. 4) that controlled its speed and orientation.

3.2.4 Predator controller

A predator controller was a recurrent ANN [28], used to emulate short term memory, which was found to be a prerequisite for collective prey capture. A HL of sigmoidal neurons fully connects 22 sensory input neurons to six HL neurons to two motor output neurons (Fig. 6). Input neurons encode the state of eight infrared proximity sensors and eight light sensors ([SI-0, SI-15] in Fig. 6), as well as the HL activation values from the previous simulation iteration ([SI-16, SI-20] / [SI-16, SI-21] in Fig. 6). Motor outputs ([MO-0, MO-1] in Fig. 6) encode the speed of the two wheels. The output value of each motor neuron updates the speed of the corresponding wheel at each simulation iteration. For each pursuit-evasion experiment (Sect. 3.6), the number of HL neurons was first evolved by parameter calibration experiments (Sect. 3.5).

3.2.5 Defining and measuring specialized predator behaviors

Predator controllers did not produce motor outputs that directly corresponded to distinct behaviors. Rather, varying wheel speeds and robot orientations produced

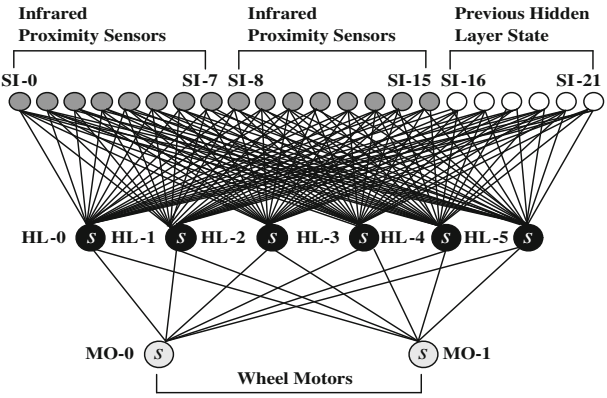

Fig. 6 *Predator Recurrent ANN Controller*. Connection weights were evolved by CCGA, MESP or CONE in pursuit-evasion experiments. The number of hidden layer neurons were evolved in parameter calibration experiments. $SI - x$: Sensory input x, $MO - y$: Motor output y, $HL - z$: Hidden layer neuron z

behaviors that were not easily distinguishable. Thus, it was necessary to identify distinct emergent predator behaviors during the CCGA, MESP, and CONE processes. Given that distinct predator behaviors were identified, the *behavioral specialization metric* (Sect. 2.2) was applied (during the CCGA, MESP, and CONE processes) to determine if emergent behaviors were specialized. However, before executing CCGA, MESP, and CONE to evolve collective prey capture behaviors (Sect. 3), it was first necessary to identify the *emergent predator behaviors*. Section 3.4 (pre-experimental phase) details experiments used to identify distinct emergent predator behaviors.

3.3 Validating the complexity of CONE

These validation experiments elucidated the CONE process via applying it to solve pursuit-evasion tasks, ranging from simple to complex. These tasks demonstrated CONE's benefits compared to CCGA, MESP, and a *Conventional Neuro-Evolution* (CNE) method. Experiments indicated that CONE's algorithmic complexity is mandated to evolve teams to solve collective behavior tasks requiring behavioral specialization and heterogeneity in teams.

Validation experiment 1: Required one predator to capture one prey, and behavioral specialization is not beneficial for task accomplishment. Prey capture was when a predator collided with a prey. Predator fitness equalled the number of simulation iterations before prey capture occurs.

Validation experiment 2: Required two predators to capture a prey. Cooperation was required but behavioral specialization was not beneficial or required for task accomplishment. Prey capture was when two predators collided with a prey at the same simulation iteration. Team fitness (both predators) equalled the number of iterations before prey capture occurs.

Validation experiment 3: Required three predators to capture a prey. Cooperation was required and specialization was beneficial for task accomplishment. Prey capture was when the three predators pushed against the prey such that it was immobilized. In this task it was possible for the prey to escape the hold of the predators by virtue of turning about and trying to move off in various directions. Predator fitness (for each predator) equalled the total number of simulation iterations for which a prey was held immobilized by the predators. Previous research [74] demonstrated that at least two predators are required to accomplish this task, where predators must use complementary behavioral specializations.

3.3.1 Validation experiments: experimental setup

The environment, predator and prey controllers, and experimental setup was the same as the pursuit-evasion experiments (Sect. 3.6). A CONE validation experiment was the application of CNE, CCGA, or CONE to either task 1, 2, or 3, where predator controller connection weights were adapted by CNE, CCGA, or CONE. For a consistent comparison, CNE used the same parameters as CCGA, MESP and CONE (Table 3). For each validation experiment, an average fitness over 20 simulation runs was calculated.

3.3.2 CNE: conventional neuro-evolution

CNE [106] is an evolutionary process illustrated in Fig. 7. A *Genetic Algorithm* (GA) [27] is applied to evolve a genotype population. Each genotype encodes an ANN controller. A controller receives sensory inputs (observations) from its environment and maps inputs to motor outputs (actions). A fitness is then assigned based on the controller's evaluation in the task (environment). CNE was selected as

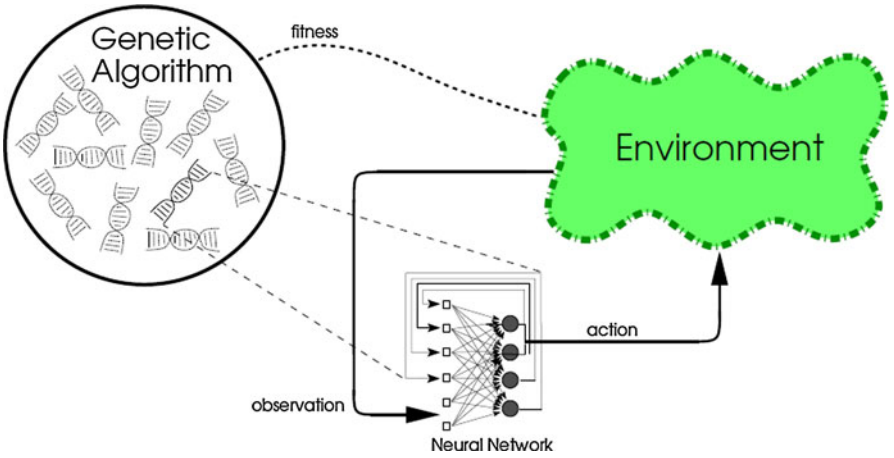

Fig. 7 *Conventional Neuro-Evolution (CNE).* Complete ANN controllers are evolved from one genotype population. Figure adapted from Gomez [36]

the first comparative method since it has been successfully applied to evolve simulated robot controllers in previous research [74].

3.3.3 CCGA: cooperative co-evolutionary algorithm

The *Cooperative Co-evolving Genetic Algorithm* (CCGA) [86, 87] uses a GA to cooperatively co-evolve n individuals (genotypes) from n populations (species). Each species is genetically isolated, so recombination only occurs *within* each species. Figure 8 illustrates an example of the CCGA model, using only two species. A set of n genotypes (one selected from each species) are evaluated based on how well they cooperate to solve a given task. CCGA was selected as the second comparative method since it is has been previously applied to evolve behaviorally specialized robot controllers [88]. CCGA is also included in the pursuit-evasion experiments (Sect. 3.6).

3.3.4 CNE / CCGA applied to validation task 1

CNE and CCGA encoded a complete controller as one genotype. A genotype was a set of floating point values encoding all sensory inputs plus all motor output weights connected to a controller's hidden layer. In validation task 1, only one predator controller ($n = 1$) was evolved. Thus, CNE and CCGA followed the same algorithmic process, using one population of 600 genotypes. The CNE and CCGA evolutionary processes consisted of the following steps.

3.3.4.1 Initialization CNE and CCGA began by initializing each gene in each genotype to a random value in the range: [0.0, 1.0].

3.3.4.2 Genotype evaluation Each genotype in the population was systematically, decoded into a predator controller, and evaluated in validation task 1. To ensure rigorous controller evaluation, each controller was evaluated in 10 different *epochs* (one predator *lifetime*). An epoch was one simulation scenario that tested different predator and prey orientations and starting positions in the environment. An epoch was 1,000 simulation iterations. Predator fitness was calculated as an average over all epochs of a predator's lifetime.

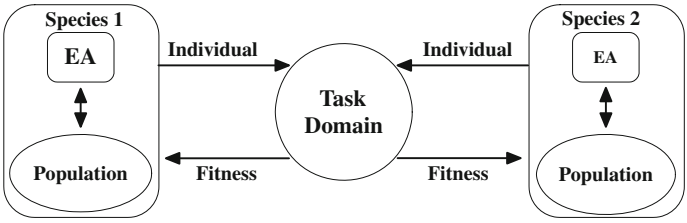

Fig. 8 *Cooperative Co-evolutionary Genetic Algorithm (CCGA).* CCGA allows for n species to be co-evolved. Figure adapted from Potter [85]

3.3.4.3 Parent selection At the end of a generation, all genotypes in the population were ranked by fitness. Each genotype in the population's *elite portion* (fittest 20 %) was systematically selected and paired with a second genotype (randomly selected from the population's elite portion).

3.3.4.4 Recombination and replacement Each genotype parent pairing was recombined using one-point crossover [27]. All parent pairs produced enough child genotypes to completely replace the current population.

3.3.4.5 Mutation Burst mutation [37] was applied to each gene of each new genotype with a probability of 0.05.

The next generation of CNE or CCGA then began (with *genotype evaluation*). After 500 generations, the fittest genotype in the population represented the predator controller best suited to solve validation task 1.

3.3.5 CONE / MESP applied to validation task 1

CONE and MESP evolved one predator controller ($n = 1$). This controller contained six hidden layer neurons ($u = 6$), evolved from six sub-populations. Each sub-population contained 100 genotypes. CONE and MESP worked via evolving hidden neuron (input-output connection weights) from separate sub-populations and combining evolved neurons as a controller's hidden layer (Sect. 2). The CONE and MESP processes comprised the following steps.

3.3.5.1 Initialization The CONE/MESP process initialized each gene of each genotype with a random value in the range: [0.0, 1.0].

3.3.5.2 Evaluate genotypes For CONE and MESP, each genotype g in each of the six sub-populations was systematically selected, and g was evaluated in the context of a complete controller tested in validation task 1. A controller is constructed from g and five other genotypes randomly selected from the other five sub-populations. Genotype g fitness equalled fitness assigned to the controller (containing g). To ensure rigorous evaluation, each controller was evaluated in 10 *epochs*.

3.3.5.3 Evaluate elite controllers This step in the process was only used by CONE. After all genotypes in all sub-populations were evaluated, a second round of fitness evaluations were executed, in which *elite controllers* were evaluated. An elite controller was constructed via systematically selecting the fittest genotypes from each sub-population's *elite portion*. Elite controller fitness (*utility*), equalled the average fitness of genotypes comprising the controller's hidden layer. Each elite controller was evaluated in validation task 1, where each genotype (neuron) was assigned the same fitness as assigned to the controller. Each new genotype fitness evaluation overwrote the previously calculated fitness. The process of constructing elite controllers from each sub-population's elite portion continued until all genotypes in each sub-population's elite portion had been assigned a new fitness.

3.3.5.4 Parent selection In validation task 1, only one predator was evolved, so for CONE, there was no inter-population parent selection and recombination. Rather, selection occurred *within* each sub-population's elite portion. MESP does not use inter-population selection and recombination, so selection always occurred within sub-populations. Hence, for both MESP and CONE, each elite portion genotype was systematically selected and paired with another genotype (randomly selected from the same sub-population elite portion).

3.3.5.5 Recombination and replacement After parents had been paired within each sub-population, parents were recombined using one-point crossover [27]. Enough child genotypes were produced to replace the current sub-population.

3.3.5.6 Mutation After CONE / MESP recombination, burst mutation [37] was applied to each gene of each genotype with a 0.05 degree of probability.

The next generation of CONE / MESP then began (with *evaluate genotypes*). After 500 generations, the fittest controller (constructed via selecting the fittest genotype from each sub-population), was the controller best suited to solve validation task 1.

3.3.6 CCGA applied to validation task 2 and 3

For validation tasks 2 and 3, two or three predators were evolved from two or three populations ($n = [2, 3]$), respectively. Task 2 and 3, used populations containing 300 and 200 genotypes, respectively. CCGA's cooperative co-evolution process consisted of the following steps.

3.3.6.1 Initialization CCGA initialized each gene in each genotype to a random value in the range: [0.0, 1.0].

3.3.6.2 Genotype evaluation Within each population, the same evaluation procedure was used as for CCGA in validation task 1 (Sect. 3.3.4). However, since tasks 2 and 3 evolved two and three controllers (from two and three populations, respectively), controllers were evaluated based on how effectively they cooperated to solve these tasks. Consider that, for task 2, a controller (containing genotype g, to be evaluated), where g was derived from population 1. This controller (ANN_0) was evaluated in task 2, together with a second controller derived from population 2 (ANN_1). ANN_1 was derived via randomly selecting a second genotype from population 2. After the evaluation of ANN_0 and ANN_1 in task 2, a fitness was assigned to g. After all genotypes in population 1 were evaluated (in the context of ANN_0), then the genotype evaluation procedure was repeated for all genotypes in population 2. That is, each genotype (g) in population 2 was evaluated in ANN_1, where g was assigned a fitness based on how well ANN_1 and ANN_0 cooperated to solve task 2. The same genotype evaluation procedure was used for task 3, the difference being that three populations (controllers) were evaluated. To ensure rigorous controller evaluation, each genotype g was evaluated in 10 epochs.

3.3.6.3 Parent selection In each population, all genotypes were ranked by fitness. Each genotype in a population's elite portion was systematically selected and paired with a second genotype (randomly selected from the elite portion).

3.3.6.4 Recombination and replacement For each population, each genotype parent pairing was recombined using one-point crossover [27]. All parents produced enough child genotypes to completely replace the given population.

3.3.6.5 Mutation In each population, burst mutation [37] was applied to each gene of each genotype with a 0.05 degree of probability.

The next CCGA generation then began (with *genotype evaluation*). After 500 generations, the fittest genotype was selected from each of the n populations (where, $n = [2, 3]$). These 2 or 3 genotypes (controllers) represented the predator team best suited to solve validation tasks 2 or 3, respectively.

3.3.7 CONE and MESP applied to validation task 2 and 3

For validation tasks 2 and 3, two and three predators were evolved from two and three populations ($n = [2, 3]$), respectively. Since a predator controller contained six hidden layer neurons, each population contained six sub-populations. For task 2, each population contained 300 genotypes, and each sub-population contained 50 genotypes. For task 3, each population contained 200 genotypes, and each sub-population contained 33 genotypes. CONE and MESP used the following process.

3.3.7.1 Initialization Each CONE / MESP gene of each genotype was initialized to a random value in the range: [0.0, 1.0].

3.3.7.2 Evaluate genotypes Within each CONE / MESP population, the same genotype evaluation procedure, as used for validation task 1, was used (Sect. 3.3.5). However, for tasks 2 and 3, two and three controllers were evolved from two or three populations (respectively), and evaluated based on how well they cooperated to accomplish the task. Consider that, for task 2, a controller (containing genotype g, to be evaluated), was derived from population 1. This controller (ANN_0) was evaluated in the task together with a second controller derived from population 2 (ANN_1). ANN_1 was derived via randomly selecting one genotype from each of the sub-populations in population 2. After the evaluation of ANN_0 and ANN_1 in task 2, a fitness was assigned to genotype g. After all genotypes in all sub-populations of population 1 had been evaluated (in ANN_0), then the genotype evaluation procedure was repeated for all genotypes in population 2. That is, each genotype g in each sub-population of population 2 was evaluated in ANN_1, where genotype g was assigned a fitness based on how well ANN_0 and ANN_1 cooperated to solve task 2. The same genotype evaluation procedure was used for task 3, except that three populations (controllers) were evaluated. For rigorous controller evaluation each genotype g was evaluated in 10 epochs.

3.3.7.3 Evaluate elite controllers Elite controller evaluation was only used by CONE, and the same procedure as used for validation task 1 (Sect. 3.3.5), was used for tasks 2 and 3. However, for tasks 2 and 3, two and three elite controllers derived from two and three populations (respectively), were cooperatively evaluated. Consider that, for task 2, an elite controller (containing genotypes: $[g_0, g_5]$), selected from the elite portions of the six sub-populations in population 1, were to be evaluated. This elite controller (ANN_0) was evaluated in task 2 together with a second elite controller derived from population 2 (ANN_1). ANN_1 was similarly constructed via selecting one genotype from the elite portion of each sub-population in population 2. After the evaluation of ANN_0 and ANN_1 in task 2, a new fitness was assigned to $[g_0, g_5]$, over-writing the previous fitness of these elite portion genotypes. After all genotypes in the elite portions of the sub-populations of population 1 had been evaluated (in ANN_0), then the elite controller evaluation procedure was repeated for population 2. That is, genotypes: $[g_0, g_5]$, selected from the elite portions of the sub-populations of population 2 were evaluated in ANN_1, where $[g_0, g_5]$ were assigned a fitness based on the effectiveness of the elite controllers ANN_1 and ANN_0 cooperating to solve validation task 2. This evaluation procedure was also used for task 3, except that three populations were evaluated.

3.3.7.4 Parent selection, recombination and replacement If CONE used more than one population (as when applied to validation tasks 2 and 3), then genotype recombination occurred *within* sub-populations or *between* populations. For MESP, parent selection, recombination, and replacement only occurred *within* each population's sub-population, following the same procedure as used in validation task 1 (Sect. 3.3.5).

The *Specialization Difference Metric* (SDM) was applied to determine if recombination occurred between populations (Sect. 2.2). For example, in task 2 (where CONE used two populations P_0 and P_1), the SDM was applied as follows. If the SDM calculated that the two fittest controllers ANN_0 and ANN_1 (derived from the elite portions of the sub-populations in P_0 and P_1) had sufficiently similar *behavioral specializations*, then the sub-populations of P_0 and P_1 became candidates for recombination. The degree of behavioral similarity required for controllers to be sufficiently similar was determined by the *Specialization Similarity Threshold* (SST) parameter (Sect. 2.2). The SST was adapted in the final step of each generation of the CONE process.

As a further condition for recombination between P_0 and P_1, all pairs of sub-populations (between populations) were tested for *genetic similarity* by the *Genotype Difference Metric* (GDM). For example, consider one sub-population SP_0 in P_0 and a second sub-population SP_1 in P_1. If the average weight difference between SP_0 and SP_1 was less than the *Genotype Similarity Threshold* (GST), then SP_0 and SP_1 were recombined (Sect. 2.3). The recombination of SP_0 and SP_1 occurred as follows. A genotype was randomly selected from the elite portion of SP_0 and SP_1 and one-point crossover [27] applied to create two child genotypes. The first child genotype was assigned as the first replacement genotype in SP_0, and the second child genotype was assigned as the first replacement genotype in SP_1. This

process was iteratively continued until all genotypes in both sub-populations had been replaced by child genotypes created from the SP_0 and SP_1 elite portions.

The GST was adapted in the final step of each generation of the CONE process. If the GDM calculated that SP_0 and SP_1 were *not genetically similar*, then recombination occurred *within* SP_0 and SP_1 (Sect. 3.3.5). That is, as in the case of CONE applied to validation task 1, each sub-population's elite portion genotype was systematically selected and paired with another genotype (randomly selected from the elite portion). These elite portion genotype pairings were then recombined, producing enough child genotypes to replace the given sub-population. Recombination also occurred *within* all sub-populations (of P_0 and P_1), if the SDM calculated that populations P_0 and P_1 were *not behaviorally similar*. For task 3, the same procedure was used, except that the SDM and GSM were applied between three populations.

3.3.7.5 Mutation Burst mutation [37] was applied to each CONE / MESP gene of each genotype with a 0.05 degree of probability.

3.3.7.6 Parameter adaptation This step was only used by the CONE process. The GST and SST parameter values regulated genotype recombination between populations. Thus, if the fitness of at least one of the n fittest controllers had not progressed in 10 generations (V in Table 3) the GST was adapted. Similarly, if the fitness of at least one of the n fittest controllers had not progressed in 20 generations (V in Table 3), the SST was adapted. CONE parameter adaptation is described in Sect. 2.3.

The next generation of CONE and MESP then began (with *genotype evaluation*). After 500 generations, the fittest controller derived from n ($n = [2, 3]$) populations represented a team of two or three predators best suited to solve validation tasks 2 or 3, respectively. As a further validation of CONE's complexity, the CONE process was also executed on validation tasks 2 and 3, but without the *evaluation of elite controllers*. The evaluation of elite controllers, in addition to the GDM and SDM, was a key difference between CONE and related cooperative co-evolution methods such as CCGA and MESP.

3.3.8 CONE validation experiments: results discussion

Independent t-tests gauged average fitness differences between CNE, CCGA, and CONE evolved teams. Validation experiments used the same procedure for statistical comparison as the pursuit-evasion experiments (Sect. 4.5).

Validation task 1: CNE, CCGA, MESP and CONE evolved predators that yielded a statistically comparable average fitness. Task 1 prey capture occurred when the predator collided with the prey. The prey-capture behavior that most frequently emerged for CNE, CCGA, MESP and CONE evolved predators was that the predator would move straight, switching directions randomly when a wall was encountered. When the prey came within sensor range, the predator would move directly towards the prey and attempt to collide with it.

Validation task 2: CNE, CCGA, MESP and CONE evolved predator teams yielded a statistically comparable average fitness. Prey capture occurred when both predators were pushing against the prey at the same time. The prey-capture behavior that most frequently emerged for CNE, CCGA, MESP and CONE evolved teams was that both predators would move straight, and turn in a random direction when a wall was encountered. When the prey came within sensor range, a predator would move directly towards it. In some simulation instances, the two predators would collide with the prey and capture it.

Validation task 3 results: CONE evolved teams yielded the highest average fitness, with statistical significance, compared to CNE, CCGA and MESP evolved teams. CCGA and MESP evolved teams yielded a higher average fitness, with statistical significance, compared to CNE evolved teams, though comparable to each other. For task 3, different prey-capture behaviors were evolved by CNE, CCGA, MESP and CONE. However, in experiments that executed CONE without the *elite controller evaluation* step, CONE evolved teams yielded a team fitness comparable to CCGA and MESP evolved teams. This result is discussed in Sect. 4.9.

3.3.8.1 CNE evolved homogenous team behaviors In CNE evolved teams, all three predators used the same behavior to form a collective prey-capture strategy. The CNE evolved behavior that most frequently emerged was for the predators to move straight, turning away in a random direction when a wall was encountered. When the prey came within sensor range of a predator, that predator would move directly towards it. If the prey came within sensor range of all three predators then all predators would concurrently move towards the prey whilst avoiding each other. At certain simulation iterations this would strategy would result in the prey being immobilized between the three predators. However, the prey would often escape the hold of predators. This resulted in CNE evolved teams having the lowest average fitness.

3.3.8.2 CCGA and MESP evolved heterogenous team behaviors In CCGA and MESP evolved teams, the three predators evolved dissimilar behaviors, that collectively formed a prey-capture strategy. CCGA and MESP evolved teams yielded a comparable average team fitness for all three validation tasks. For both CCGA and MESP, the strategy that most frequently emerged was similar to the *pursuer-blocker* prey-capture behavior (Sect. 4.1) observed in the pursuit-evasion experiments. In CCGA and MESP evolved teams, it was the interaction of predators with complementary behaviors that resulted in a significantly higher average fitness (compared to CNE evolved teams). The behaviors adopted by predators in CCGA and MESP evolved teams were calculated as being *non-specialized* (Sect. 2.2). That is, each of the CCGA and MESP evolved predators switched between the *pursuer* and *blocker* behavioral roles (Sect. 4.1) with a high frequency.

3.3.8.3 CONE evolved heterogenous team behaviors Similar to CCGA evolved teams, the three CONE evolved predators used dissimilar behaviors. The prey-capture strategy that most frequently emerged was similar to the *role-switcher*

prey-capture behavior (Sect. 4.3) observed in the pursuit-evasion experiments. In CONE evolved teams, it was the interaction of predators with complementary *specialized* behaviors that resulted in a higher average fitness (compared to CNE, MESP and CCGA evolved teams). That is, CONE evolved predators switched between *idle*, *knocker* and *flanker* roles (Sect. 4.3) with a low frequency, and thus tended to maintain a single role for the duration of their lifetime. As with CCGA, CONE's multi-population architecture encouraged emergent heterogeneity in evolved team behaviors. However, CONE's regulated inter-population genotype recombination based on behavioral specialization similarities of controllers and genetic similarities between populations was successful in evolving other advantageous forms of specialization.

Section 4 presents experiments validating the role of CONE's *Specialization* and *Genotype Difference Metrics* (SDM and GDM) in evolving beneficial forms specialization, and a discussion of the contribution of emergent behavioral specialization to collective prey-capture behaviors and predator team fitness.

3.4 Evolving collective prey-capture behavior: experimental phases

Each experiment placed a team of two to six predators in the simulation environment, with one or two prey, and applied CCGA, MESP or CONE to evolve the team's collective prey capture behavior. Experiments measured the impact of a *team type* (Table 1) and *controller design method* (CCGA, MESP, or CONE) on predator team fitness. The *experimental objective* was to ascertain which method achieved the highest task performance for all team types, and to investigate the contribution of behavioral specialization to collective prey capture performance. *Team fitness* was calculated as the the average time for which a prey was captured. It was assumed that each predator contributed equally to prey capture, and thus received an equal fitness reward for prey capture. Team fitness was an average calculated over all epochs of a team's lifetime, and over all simulation runs.

The structural credit assignment problem [2] is often evident in multi-robot tasks [62, 97], but was avoided in the context of this team fitness function. As with

Table 1 Team types

Team type	Team composition
1	Two predators and one prey
2	Three predators and one prey
3	Four predators and one prey
4	Five predators and one prey
5	Six predators and one prey
6	Two predators and two prey
7	Three predators and two prey
8	Four predators and two prey
9	Five predators and two prey
10	Six predators and two prey

Pursuit-evasion experiments (pre-experimental and experimental phase) tested 10 team types (compositions of predator and prey)

Quinn's [89] work on NE and behaviorally heterogenous teams, a fitness function that assigns an equal fitness share to each robot in the team [16] sufficed for the NE controller adaptation methods and task tested in this study. This fitness assignment approach had the advantage of ensuring that there is no conflict between an individual controller's (robot's) goal to maximize its own fitness, and the team's goal to maximize its fitness. Thus, each robot's individual fitness could only be increased via increasing team fitness [95]. Furthermore, the accuracy of fitness assigned to each individual controller was improved by evaluating each individual in the context of multiple teams in multiple task scenarios. This fitness function has been successfully applied in previous research [74]. Experiments used the following phases.

3.4.1 Parameter calibration phase

Prior to the pursuit-evasion experiments, simulation parameters, prey controller weights, and the number of HL neurons used by predator and prey controllers, were derived in parameter calibration experiments (section 3.5).

3.4.2 Pre-experimental phase

A first set of pursuit-evasion experiments applied CCGA, MESP, and CONE to evolve collective prey capture behavior for each team type (table 1). The purpose of this phase was for the experimenter to observe prey capture behaviors that emerged during the evolutionary processes of CCGA, MESP and CONE. Observed emergent prey-capture behaviors were identified according to sensory-motor activation values. Thus, whenever a prey was collectively captured (by at least two predators), the sensory input values and corresponding motor output values of each predator (that captured the prey) was recorded for the period of prey capture. These recorded ranges of sensory-motor values were then given *behavioral labels*. Section 4 presents the identified predator behaviors and their labels. Each (CCGA, MESP, and CONE) experiment was executed for 20 simulation runs. Each run consisted of 500 generations. One generation was a predator/prey *lifetime*. Each lifetime lasted for 10 epochs. An epoch was a simulation scenario that tested different predator and prey orientations and starting positions in the simulation environment. Each epoch consisted of 1000 simulation iterations.

3.4.3 Experimental phase

A second set of pursuit-evasion experiments applied CCGA, MESP, and CONE to evolve collective prey capture behavior for each team type. The experimental phase setup was the same as that used for the pre-experimental phase. Given that distinct predator behaviors were identified in the pre-experimental phase, the *Behavioral Specialization Metric* (Sect. 2.2) was applied to ascertain if emergent behaviors were specialized. Also, during the CONE evolutionary process, the *Specialization Difference Metric* was applied to regulate inter-population recombination based upon behavioral specializations exhibited by CONE evolved predator behaviors.

Section 4 presents the average fitness, for each team type, of CCGA, MESP and CONE evolved teams.

3.4.4 Post-experimental phase

Predator behaviors classified as specialized (in the experimental phase) were observed and assigned specialization labels by the experimenter. A predator was assigned a specialization label, if that predator executed a given specialized behavior for more than 50 % of its lifetime in the last generation of the CCGA, MESP or CONE evolutionary process. Specialization labels were thus the *most executed specialized behavior* for each predator in the fittest CCGA, MESP or CONE evolved teams. If a given predator did not executed a specialized behavior for most of its lifetime, then that predator was assigned a *non-specialized* label. Specialization (and non-specialization) labels were assigned for the purpose of comparing the behavioral composition of the fittest CCGA, MESP, and CONE evolved teams. The behavioral specializations adopted by predators in the fittest CCGA, MESP, and CONE evolved teams was then correlated with average team fitness. Section 4 presents the specialization labels assigned to individual predator behaviors in the fittest teams. Section 4 also presents collective prey capture behavior labels assigned by the experimenter.

3.5 Parameter calibration phase

Parameter calibration experiments were executed for CONE, CCGA, and MESP, and each team type. Table 2 presents the parameters that were calibrated. Prior to running parameter calibration experiments, it was first necessary to evolve prey evasion behavior. This evolved prey behavior was then used in parameter calibration experiments. Calibrated parameters and the evolved prey behavior were

Table 2 Parameter calibration

Parameter	Value range	Range interval
Robot movement range	[0.01, 0.51]	0.05
Light/proximity detection sensor range	[0.01, 0.10]	0.01
Simulation runs	[10, 30]	2
Iterations per epoch (Robot lifetime)	[500, 1500]	100
Generations	[50, 150]	10
Epochs	[2, 20]	2
Mutation (per gene) probability	[0.0, 0.11]	0.01
Fitness stagnation V (CONE)	[5, 15]	1
Fitness stagnation W (CONE)	[10, 25]	1
Population elite portion	[5, 55]	5 %
Hidden Layer (HL) neurons	[1, 10]	1

Values tested for the pursuit-evasion task

Table 3 Pursuit-evasion parameters

Simulation runs/environment size	20 / 1000 cm x 1000 cm
Team types (predator/prey composition)	[1, 10]
Robots (predators/prey)	Khepera
Proximity/light sensor range	22 cm (Predator/Prey)
Predator/Prey team size	[2, 6] / [1, 2]
Generations/Epochs/Epoch iterations	500 / 50 / 1000
Mutation probability/Range	0.05 / [-1.0, +1.0]
Mutation/Crossover operator	Burst (Cauchy distribution) / Single point
Fitness stagnation V/W	10 / 20 Generations (CONE)
Behavioral Specialization Threshold	0.5 (CONE)
Genotype Similarity Threshold (GST)	[0.0, 1.0] (dynamic/CONE)
Specialization Similarity Threshold (SST)	[0.0, 1.0] (dynamic/CONE)
Genotype population elite portion	25 %
Weight (gene) range	[-10.0, +10.0]
Predator ANN Input/Output/HL neurons	22 / 2 / 6
Total number of genotypes	600 (CCGA, MESP, CONE)
Genotype structure	Hidden layer neuron input-output weights (MESP/CONE), All weights (CCGA)
Genotype representation	Floating point value vector
Genotype populations	[2, 6] (CCGA, MESP, CONE)
Genotype length	24 (CONE, MESP), 144 (CCGA)
Genotypes per population	[300, 200, 150, 120, 100] (CCGA, CONE: [2, 6] predators), 100 (MESP)

Simulation settings. *HL* hidden layer

then used in the pre-experimental and experimental phases (Sect. 3.4). Table 3 presents the calibrated parameter values.

3.5.1 Prey controller evolution

Prior to being placed in pursuit-evasion experiments, the prey ANN controller was evolved with an evasion behavior. This controller evolution adapted the number of HL neurons and input-output connection weights.

One prey and two predators (coded with heuristic pursuit behaviors) were placed in a simulation environment. CONE was applied to the prey to evolve an evasion behavior. Two predators were used as previous research [74] demonstrated two to be the minimum for this pursuit-evasion task. A predator's heuristic behavior was such that it moved directly towards the prey, or towards the prey's last known

location, or in a straight line in a random direction. When faced with a wall, the predator moved away in an opposite random direction. Initially, the prey controller used one HL neuron. A CONE experiment was executed for 20 simulation runs. Each run was 100 generations. One generation was a predator/prey *lifetime*. Each lifetime was 10 epochs. Each epoch was 500 simulation iterations. An epoch was a simulation scenario that tested different predator and prey orientations and starting positions in the environment. CONE experiments were re-run five times. Each time another HL neuron was added to the prey's controller. Five HL neurons was the minimum number required to evolve an evasion behavior that consistently evaded predators in all simulation runs. The fittest evolved prey controller was used in the parameter calibration and then the pursuit-evasion experiments.

3.5.2 Pursuit-evasion parameter calibration

Each of the parameters in Table 2 were systematically selected and varied within 100 % of its value range at 20 % intervals. Thus, 10 different parameter values were tested for each parameter. When a given value was selected, other parameter values were fixed at the median value in their range. The impact of given parameter values was ascertained via running CCGA, MESP, and CONE for 500 generations, and 20 simulation runs (for each team type). Each of the parameters given in Table 2 were calibrated independently. Thus, parameter inter-dependencies were not taken into account, since the complexities of parameter interactions could not be adequately explored using this parameter calibration scheme. Investigating the parameter interactions during calibration remains a current research topic [26]. However, the impact of the *behavioral specialization threshold*, the number of *HL neurons* and *simulation runs* are briefly outlined here. Varying these parameters had the greatest impact on team fitness (for CCGA, MESP and CONE evolved teams).

3.5.2.1 Behavioral specialization threshold Calibration experiments found that deceasing the specialization threshold to below 0.5 resulted in fewer controllers being classified as specialized and thus fewer specialized controller recombinations. This reduced recombination of specialized controllers and beneficial behaviors between populations. Increasing the specialization threshold above 0.6 resulted in more controllers being classified as specialized and thus more controllers being recombined between populations. This in turn resulted in the propagation of specialized behaviors that were not necessarily beneficial. The overall impact of a specialization threshold value outside the range [0.5, 0.6] was a decreasing task performance for all team types tested.

3.5.2.2 HL neurons Calibration experiments determined that for CCGA, MESP, and CONE evolved teams (for all team types), an appropriate number of HL neurons was five, seven, and seven, respectively. For a fair method comparison, predator controllers used six hidden layer neurons (Fig. 6).

3.5.2.3 Simulation runs Calibration experiments determined that 20 evolutionary runs was sufficient to derive an appropriate estimate of average team fitness CCGA, MESP, and CONE evolved teams. Less than 20 runs was found to be insufficient, and more than 20 runs too time and computationally expensive.

3.6 Pursuit-evasion experiments

After parameter calibration and prey controller evolution, CCGA, MESP and CONE were applied to evolve collective pursuit behavior for each team type (Table 1). The n populations used by CCGA, MESP and CONE were initialized with genotypes comprised of genes with random values in the range: [0.0, 1.0]. For team sizes of [2, 6], each population was initialized with 300, 200, 150, 120 or 100 genotypes, respectively. For CCGA, each population contained genotypes corresponding to complete ANN controllers [86]. At each generation of CCGA, the fittest genotype was selected from each population, and evaluated as n controllers (a predator team) in the pursuit evasion task. Each genotype was encoded as a vector of 144 floating point values, representing sensory input and motor output weights connected to the HL in each predator controller. That is, 24 sensory inputs multiplied by six HL neurons plus two motor outputs (Fig. 6). For MESP and CONE, each population contained genotypes that were decoded into individual HL neurons. Section 2 describes the procedure used to construct complete ANN controllers from each population, and to evaluate n controllers in the pursuit-evasion task. Each genotype was encoded as a vector of 24 floating point values, representing the sensory input and motor output weights connected to one HL neuron (Fig. 6).

3.7 Evolving collective pursuit behavior with CCGA

For a team of n predators, where $n \in [2, 6]$), n populations were initialized. For a predator team size of $n = \{300, 250, 200, 150, 120, 100\}$, run for 500 generations, the number of evaluations E, was:

$CCGA_E = \{300, 200, 150, 120, 100\}$ (genotypes per population) $* n = [2, 6]$ (populations) $*$ 500 (generations) $*$ 50 (epochs per generation);
$CCGA_E = \{15\ 300\ 000\ (n = [2, 6])\}$;

This includes 300 000 CCGA evaluations required to the evaluate the elite portion (fittest 20 %) of genotypes in each population. For each population, elite portion genotypes were systematically selected and evaluated together with genotypes randomly selected from the elite portions of the other populations. These elite portion evaluations were included so as the total number of evaluations per CCGA experiment would equal that of MESP and CONE.

3.8 Evolving collective pursuit behavior with MESP / CONE

For a team size of $n = [2, 6]$, executed for 500 generations, the number of evaluations E, was:

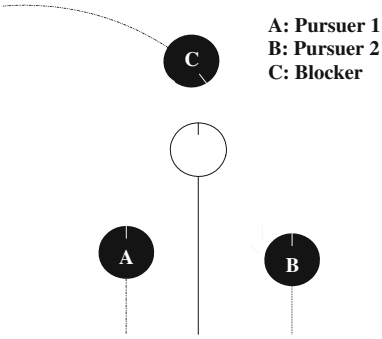

Fig. 9 Pursuer-blocker behavior. A tangential *bar* in a *circle* indicates the current heading of a predator or prey robot. *Black circles*: predators. *White circle*: prey

$MESP/CONE_E = \{300, 200, 150, 120, 100\}$ (genotypes per population) $* n = [2, 6]$ (populations) $*$ 500 (generations) $*$ 50 (epochs per generation);

$MESP/CONE_E = \{15\ 300\ 000\ (n = [2, 6])\}$;

This number of evaluations includes 300 000 evaluations required to evaluate controller utility (section 2.4) of the fittest 20 % of controllers.

4 Results and discussion

This section describes collective prey capture behaviors evolved by CCGA, MESP, and CONE for each team type (Table 1).

4.1 CCGA / MESP / CONE evolved behavior: pursuer-blocker

Pursuer-blocker was a prey capture behavior that emerged in approximately 80 % of CCGA, 60 % of MESP and 40 % of CONE experiments. Figure 9 depicts an example of the pursuer-blocker behavior using team type 2. Predators A and B are the *pursuers*, positioned behind and to either side of the prey. Predator C assumed the role of a *blocker*. When the prey moved within light sensor range of predator C, this predator moved directly towards the prey. The prey then turned to avoid predator C, however its evasion was halted by *pursuers*, and the prey was captured by the three predators. Pursuer-blocker was most effective for team types 2 and 3. Team types 4 and 5 yielded comparatively poor results due to physical interference that occurred between predators as they collectively approached a prey. Pursuer-blocker failed with team type 1, as two predators were insufficient for prey capture.

4.2 CCGA / MESP / CONE evolved behavior: spider-fly

Spider-fly[2] was a prey capture behavior that emerged in approximately 20 % of CCGA, 50 % of MESP, and 20 % of CONE experiments. Figure 10 presents an example of the *spider-fly* behavior using team type 2. At simulation time t the prey

[2] *Spider-fly* is from related research of Nolfi and Floreano [81], where evolved predator behaviors exhibited spider like predatory behavior (when capturing a fly).

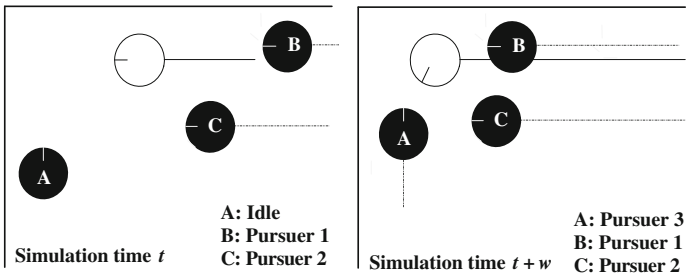

Fig. 10 Spider-fly behavior. A tangential *bar* in a *circle* indicates the current heading of a predator or prey. *Black circles* predators. *White circle* prey

followed a wall, and was pursued by predators B and C. As the prey reached the corner and turned about, predator A (previously idle in close proximity to the corner) became active. The result was that at simulation time $t + w$ the prey was immobilized between the corner and predators A, B, and C. The *spider-fly* behavior was most effective when using team types 1 to 3. Team types 4 to 6 failed in the early stages of the CONE evolutionary process ($\leq$ 250 generations) due to physical interference that occurred between predators as they collectively approach a prey. However, team types 4 to 6 often succeeded in later stages of the CONE process ($>$ 250 generations) given that the fourth, fifth and sixth predators evolved so as to assume *idle* behaviors (Sect. 4.5).

4.3 CONE evolved behavior: role-switcher

Role-switcher was a prey capture behavior that emerged in approximately 80 % of experiments applying CONE. Figures 11 and 12 illustrate two versions of role switcher using team types 2 to 3. Several versions of the role-switcher behavior emerged. However, only two are described and illustrated here for clarity. In each version, different predators adopted multiple and complementary behavioral roles, and switched between these roles to maintain the effectiveness of the prey capture behavior. These behavioral roles were named: *flanker*, *knocker* and *idle*. A *flanker* was a predator that remained in close proximity to the left or right hand rear side of

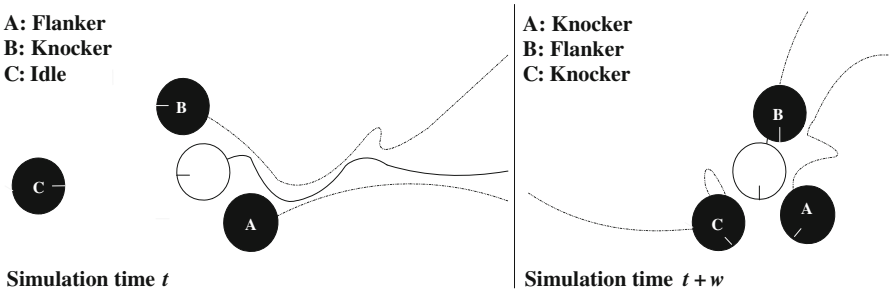

Fig. 11 Role switcher behavior 1. A tangential *bar* in a *circle* indicates the current heading of a predator or prey. *Black circles* predators. *White circle* prey

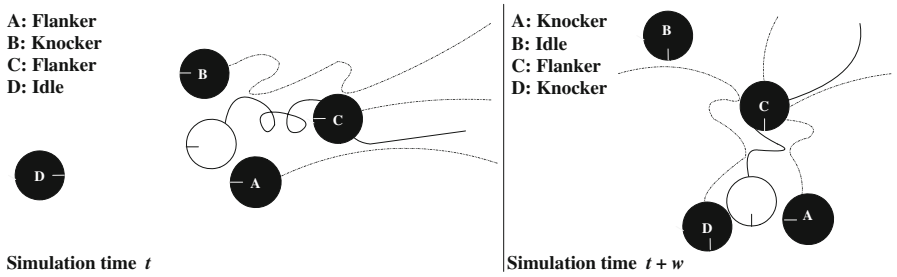

Fig. 12 Role switcher behavior 2. A tangential *bar* in a *circle* indicates the current heading of a predator or prey robot. *Black circles* predators. *White circle* prey

a prey. A flanker repeatedly collided with a prey so as to force the prey's movement in a particular direction. A *knocker* was a predator that consistently moved in a semi-circular motion so as to repeatedly collide with the prey, and thus slow its movement. An *idle* predator did not move. The role switcher strategy was effective for team types [2, 5] and [7, 10]. Given that at least three predators were within sensory range of a prey, the closest three participated in the role-switcher behavior, whilst the other predators remained idle. Section 4.5 discusses idle behavior emergence. Two predators (team type 1) were insufficient to immobilize a prey in this case.

4.3.1 Role switcher behavior 1

Figure 11 illustrates an example of the first version of *role switcher* operating with team type 2. At simulation time t the prey turns left 90 degrees to evade predators A and B. Predator B switches its behavior from a knocker to a flanker role, and predator C switches from an idle to a knocker role. The result is that predators stay in close proximity to the prey, and at time $t + w$, capture it within a triangular formation.

4.3.2 Role switcher behavior 2

Figure 12 illustrates an example of the second version of the *role switcher* behavior, operating with team type 3. At simulation time t the prey turns left 90 degrees to evade predators A and B. Predator B switches its behavior to an idle role, whilst predator A switches to a knocker role. At the same time predator D switches from an idle to a knocker role, whilst predator C maintains its flanker role. The result is that at simulation time $t + w$ predators A, C and D capture the prey within a triangular formation.

4.3.3 Role switcher behavior 3

Figure 13 exemplifies the third version of the *role switcher* behavior, operating with team type 2. At time t the prey turns left 180 degrees to evade predators A and B,

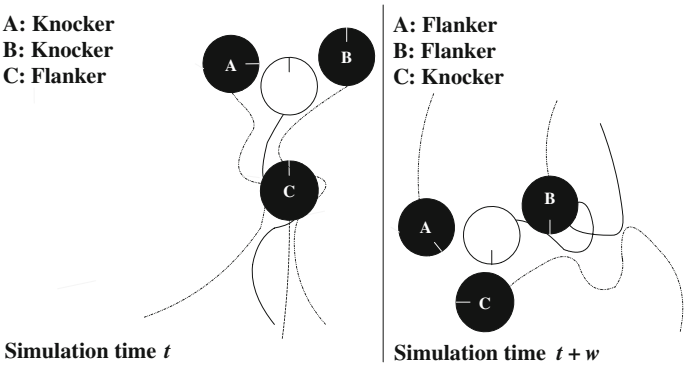

Fig. 13 Role switcher behavior 3. A tangential *bar* in a *circle* indicates the current heading of a predator or prey robot. *Black circles* predators. *White circle* prey

both predators A and B switch their behaviors from knocker to flanker roles. At the same time predator C switches its behavior from a flanker to a knocker role. At time $t + w$ the predators manage to immobilize the prey within a triangular formation.

4.4 Pursuit-evasion experiments testing two prey

Experiments testing two to six predators with two prey were also conducted. Figure 14 presents average fitness (CCGA, MESP and CONE), as lower for experiments testing two prey (team types [6, 10]), compared to experiments testing one prey (team types [1, 5]). This resulted from predators frequently switching pursuit behavior. If two prey were in close proximity to each other, then predators often switched between pursuing each. This inconsistent pursuit behavior, and

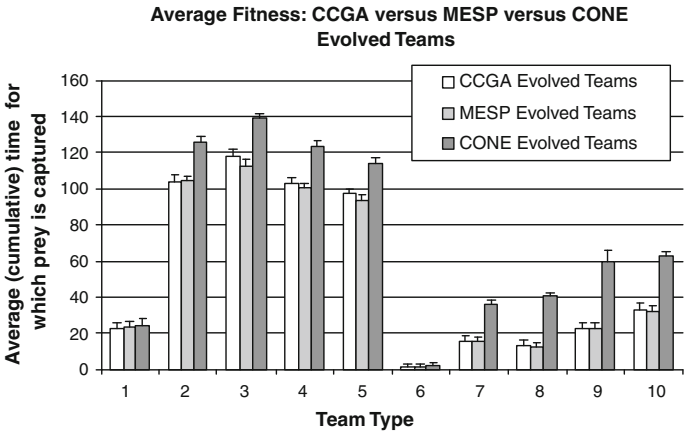

Fig. 14 Average (predator) team fitness. Average time (simulation iterations) for which the prey were captured by CCGA, MESP, and CONE evolved teams

predators avoiding collisions with each other when in prey pursuit, decreased the chance of collective prey capture.

4.5 Pursuit-evasion task experimental analysis

To draw conclusions, statistical tests were used to gauge average fitness (time for which a prey is captured) differences between CCGA, MESP and CONE evolved teams. Figure 14 presents the average fitness yielded by CCGA, MESP, and CONE evolved teams for all team types (Table 1).

- The Kolmogorov-Smirnov (KS) test [30] was applied to each data set. The KS test found that all data sets conformed to normal distributions.
- Independent *t* tests [30] were applied to ascertain if there was a statistically significant difference between average team fitness.

Bonferroni multiple significance test correction [25] was used to overcome the problem of t-tests reporting a spurious significance of difference as a result of pairwise comparisons between multiple data sets. The threshold for statistical significance was 0.016, and the null hypothesis was that data sets did not significantly differ. T-tests were applied in pair-wise comparisons to test for significant difference between the following average fitness data sets. Average fitness was calculated via selecting the fittest team for a given method, team type and simulation run. An average fitness (of fittest teams) was then calculated for 20 runs executed for each team type and method.

- CCGA versus MESP evolved teams.
- CCGA versus CONE evolved teams.
- MESP versus CONE evolved teams.

Statistical test results partially support the second hypothesis of this article: *Behavioral specialization facilitated by CONE beneficially contributes via CONE evolved teams yielding a higher average fitness, compared to CCGA and MESP evolved teams*. That is, for 80 % of team types, CONE evolved teams with a higher fitness (statistically significant) compared to CCGA and MESP evolved teams. For team types using two predators (1 and 6), CONE evolved teams yielded a comparable fitness to CCGA and MESP evolved teams.

The higher performance of CONE evolved teams is supported by the *idle* predator behavior. It is theorized that the idle behavior emerged as a means of reducing physical interference between predators. The idle behavior was an essential component of the *role switcher* prey capture behavior. Teams that used the idle behavior (as part of the role switcher prey capture behavior) increased their effectiveness and thus fitness. In CONE evolved teams, idle behaviors emerged for all team types, except 1 and 6 (those using two predators). This was the case since the role switcher behavior only functioned with three to six predators (Sect. 4.3). Furthermore, the idle behavior did not emerge in CCGA or MESP evolved teams. This suggests that CONE was able to facilitate a form of behavioral specialization (the idle behavior) appropriate for supporting an effective prey capture behavior

(role switcher). Section 4.6 discusses the contribution of the idle behavior to collective prey capture.

4.6 CONE difference metric / elite controller experiments

This section discusses the contribution of the *Genotype Difference Metric* (GDM) and *Specialization Difference Metric* (SDM) for facilitating behavioral specialization and increasing CONE evolved team fitness. This section also discusses the contribution of *elite controller evaluation* (Sect. 2.4) to the task performance of CONE evolved teams. To this end CONE was re-executed with the following variants to the original experimental setup.

1. *CONE-1:* Teams were evolved by CONE without the GDM. The SDM for regulating inter-population genotype recombination remained active.
2. *CONE-2:* Teams were evolved by CONE without the SDM. The GDM remained active.
3. *CONE-3:* Teams were evolved by CONE without the GDM and SDM.
4. *CONE-4:* Teams were evolved by CONE without elite controller evaluation (step 3 in the CONE process, Sect. 2.4).

Figure 15 presents the results of applying CONE-1, CONE-2, CONE-3, and CONE-4 to evolve team types: [1, 10]. Team fitness was averaged over 20 runs. For comparison, results attained by CONE (original experimental setup) evolved teams are also presented. Statistical comparisons indicated that teams evolved by CONE without the GDM (CONE-1), SDM (CONE-2), and both the GDM and SDM (CONE-3), yielded a significantly lower task performance comparable to CONE evolved teams, for team types: [3, 10]. Thus, both the GDM and SDM were beneficial in terms of increasing average team fitness.

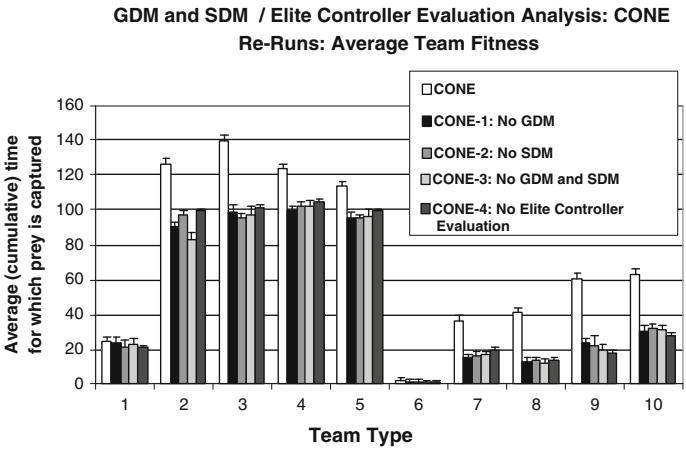

Fig. 15 GDM/SDM/Elite Controller Analysis Results. Average prey capture time (fitness) of teams evolved by CONE, CONE-1, CONE-2, CONE-3 and CONE-4

Statistical tests indicated that CONE-3 evolved teams yielding an average fitness comparable to CCGA and MESP evolved teams. Also, Fig. 15 presents CONE-1 and CONE-2 as yielding higher task performances comparative to CONE-3, for all team types. Statistical tests supported CONE-1 and CONE-2 evolved teams as yielding a significantly higher average fitness compared to CONE-3 evolved teams. This indicates that the GDM and SDM, yield benefits when working independent of each other. However, for team types 1 and 6, predators evolved by CONE-1 and CONE-2 yielded an average fitness comparable to CONE evolved teams. This was theorized to be a result of the role-switcher prey capture behavior not emerging in team types 1 and 6. Thus, for team types 1 and 6, there was no need for CONE to evolve more complex behavioral specializations suited to the role switcher behavior.

This result further supports hypothesis 2. That is, CONE's genotypic and behavioral regulation mechanisms facilitate specialization necessary for achieving a higher team fitness compared to CCGA and MESP evolved teams. The advantage of behavioral specialization for capturing two prey also supports hypothesis 2. That is, for team types: [7, 10] there was greater difference in average team fitness between CONE and CONE-1, CONE-2, and CONE-3 evolved teams compared to the difference calculated for team types: [2, 5]. This indicates that the contribution of behavioral specialization facilitated by both the GDM and SDM increases with the task's complexity. This, together with the statistical comparison of average fitness yielded by CONE versus CCGA and MESP evolved teams, completely supports hypothesis 2.

These results are also supported by related research that analyzed the role of GDM and SDM in the evolution of rover controllers and behavioral specialization in a multi-rover search task [76]. Thus, the GDM and SDM experiments conducted for the pursuit-evasion and multi-rover tasks demonstrated some generality of the GDM and SDM for encouraging behavioral specialization that in turn increased team fitness. This is supported by the different approaches used to define agent behaviors in the pursuit-evasion versus the multi-rover task. Predator behaviors in the pursuit-evasion task were defined by the interaction of specific motor output values and the correlation with specific sensory input values (Sect. 3.2.4). Where as, rover behaviors in the multi-rover task were defined by a series of motor outputs that produced a series of distinct behaviors [76]. In both tasks, the GDM and SDM were applicable to agent behaviors and behavioral specialization was measured

Thus, *CONE difference metric* experiments determined that specialization in CONE evolved teams was encouraged by CONE's genotypic and behavioral difference metrics. In this pursuit-evasion case study, the key emergent specialization was the idle behavior, which in turn facilitated a key prey capture behavior: Role-Switcher. The Role-Switcher behavior in turn allowed CONE evolved teams to achieve a higher fitness (statistically significant), compared to CCGA and MESP evolved teams, for most team types.

In terms of the contribution of elite controller evaluation, statistical tests indicated that CONE-4 evolved teams yielded an average fitness that was significantly lower than CONE evolved teams. This held true for team types: [2, 3, 4, 5, 7, 8, 9, 10] (Fig. 15). For these same team types, average fitness was

comparable to CCGA and MESP evolved teams. As with the experiments running CONE-1, CONE-2, and CONE-3, all methods evolved teams with comparable task performances for team types 1 and 6. This was a result of the lack of any need for CONE to evolve behavioral specializations, and complex collective prey-capture behaviors for team types 1 and 6.

4.7 Idle behavior exclusion experiments

The idle behavior is theorized to have emerged as a means to reduce physical interference between predators as they collectively moved to capture a prey. The idle behavior thus increased the effectiveness of prey capture.

To test this supposition, additional experiments were conducted in which CCGA, MESP, and CONE were re-run. Each method was re-executed for 20 simulation runs using the experimental setup specified in Sect. 3, but including the following adjustment. During the evolutionary process, predators received a fitness penalty equal to the number of simulation iterations that they remained stationary. These experiments were designed to discourage the emergence of idle behavior, and were called *idle behavior excluded* experiments.

These experiments yielded two main results elucidating the idle behavior's contribution. First, the *Spider-Fly* prey capture behavior (Sect. 4.2) did not emerge in CCGA, MESP, or CONE evolved teams. Second, the *Role-Switcher* prey capture behavior (Sect. 4.3) did not emerge in CONE's adaptation of predator teams. Rather, it was only the *Pursuer-Blocker* prey-capture behavior (Sect. 4.1) that emerged in the *idle behavior excluded* experiments. That is, the Pursuer-Blocker behavior did not use the idle behavior, where as, for the Spider-Fly and Role-Switcher behaviors the idle behavior was an important contribution. This statement is supported by the result of the fittest teams (evolved by CCGA, MESP, and CONE) in the *idle behavior exclusion* experiments yielding a comparable average fitness. Also, the average fitness was approximately 46, 53, and 60 % lower for CCGA, MESP and CONE evolved teams, respectively. This was in comparison to teams evolved using the original experimental setup (Sect. 3). This result held for all team types, except 1 and 6. The reason is that the Spider-Fly and Role-Switcher behaviors did not emerge in team types 1 and 6, where as, the Pursuer-Blocker behavior emerged in all team types.

The *idle behavior exclusion* experiments indicated that idle behavior was beneficial to the evolution of effective prey capture behaviors. This was especially the case for the Role-Switcher behavior. That is, Role-Switcher only emerged in CONE evolved teams and emerged for all team types (except 1 and 6). It was supposed that Role-Switcher was primarily responsible for the higher average team fitness (of CONE evolved teams) observed for all team types (except 1 and 6). Experiments described in Sect. 4.8 test this supposition that the emergence of the Role-Switcher behavior led to the significantly higher task performance observed for CONE evolved teams.

An important implication of these results, is that only robots that are indispensable for task accomplishment, participate in a collective behavior. The *idle behavior exclusion* experiments indicated that teams not using the idle behavior

yielded lower fitness. This was a result of physical interference occurring between predators as they collectively approached a prey. This interference disrupted the formation of collective prey capture behaviors. This result has important implications for the engineering of multi-robot behaviors. For example, in multi-robot tasks, robots that are non-essential could stand-by and conserve energy for the contingency that if one or more other robots are damaged or destroyed, the idle robots could assume the vacated behavioral roles and complete the task.

Also, the emergence of the idle behavior is relevant to biological literature. For example, a large number of unproductive *lazy* workers have been observed in various species of social insect societies [83]. The presence of idle workers was found to be a byproduct of individuals switching between different behavioral specializations during the accomplishment of a collective behavior task. Thus, only individuals that are *strictly needed* work on a given task [29, 64].

4.8 Behavioral lesion experiments

This section discusses the contribution of the *Role-Switcher* behavior to the higher average team fitness of CONE evolved teams compared to CCGA and MESP evolved teams (Fig. 14). To elucidate the contribution of Role-Switcher a *behavioral lesion* experiments were conducted. This entailed removing the predator behaviors that constituted the Role-Switcher prey capture behavior from the behavioral repertoire of CONE evolved teams. Such teams were known as *lesioned teams*. The fittest CONE evolved teams (for each team type) were modified into *lesioned teams*. The behavioral lesion experiments re-executed the fittest CONE evolved teams in non-adaptive pursuit-evasion experiments that tested each team type. Each behavioral lesion experiment was executed for 20 simulation runs, where each run equalled one predator/prey lifetime. Team fitness of *lesioned teams* was calculated over all simulation runs (for a given team type) and compared to that calculated for CONE evolved teams using the original experimental setup (Sect. 3).

In the post-experimental phase (Sect. 3.4), Role-Switcher (Sect. 4.3) was observed as consisting of specialized predator behaviors labeled: *flanker, knocker*, and *idle*. Thus, *lesioned teams* were the fittest CONE evolved teams (for each team type) with *flanker* and *knocker* behavioral specializations disabled. Only these behaviors were removed since they were used in Role-Switcher. Whenever the sensory-motor correlation that defined either of these behavioral specializations (Sect. 3.2.5) was detected then a *hard-wired heuristic* controller was activated. The time for which the heuristic controller remained active equalled the average prey capture time calculated for the Role-Switcher behavior (for the fittest CONE evolved teams using each team type). Heuristically controlled predators moved directly towards a prey when within light sensor range, or towards a prey's last known location, and in a random direction if the prey was not found at its last known location.

Table 4 presents the average fitness (calculated over 20 runs) of the lesioned teams as percentages of the original average fitness of CONE evolved teams. For comparison the average fitness of the fittest CCGA and MESP evolved teams is included. Table 4 indicates that the fittest CONE evolved teams (for each team type) yielded a lower (statistically significant) average team fitness when Role-Switcher

Table 4 Average Fitness of CONE Lesioned Teams. Presented values are percentages of the original fitness of the fittest CCGA, MESP, and CONE evolved teams (for each team type). Team types 1 and 6 are not included since the *Role-Switcher* behavior did not emerge in the CONE evolved teams using these team types

Team type	CCGA (%)	MESP (%)	CONE (%)
2	65	52	42
3	64	48	36
4	67	51	41
5	56	59	49
7	52	45	40
8	59	48	36
9	55	60	37
10	58	63	36

was removed. The importance of the Role-Switcher behavior is further supported by the lower (statistically significant) average fitness of lesioned teams compared to CCGA and MESP evolved teams.

Thus the *behavioral lesion* experiments demonstrated that Role-Switcher was critical to the fitness achieved by CONE evolved teams. That is, without the Role-Switcher behavior, the fittest CONE evolved teams performed poorly in comparison to the fittest CCGA and MESP evolved teams. This result supports the central claim of this article. That is, that CONE facilitates behavioral specialization, where such specialization leads to collective behaviors and team fitness that could not otherwise be achieved.

The ramification of these results is that behavioral specialization emerges in response to the task and environment constraints for the benefit of collective behavior task accomplishment. This research demonstrated this to be the case for the pursuit-evasion task. Previous research using a multi-rover collective behavior task [76] also indicated this to be the case.

4.9 Experiments that validate the complexity of CONE

To establish the necessity of CONE's algorithmic complexity, and to elucidate CONE's principled approach, additional experiments were executed. These *CONE validation experiments* (Sect. 3.3) demonstrated the efficacy CONE's complex cooperative co-evolution approach via a comparative application (with CNE and CCGA) to three validation pursuit-evasion tasks.

Validation task 1 tested only one predator and prey, and no specialization or cooperation was required for prey-capture. CONE evolved predators yielded a task performance comparable to CNE and CCGA evolved teams. This was a result of validation task 1 requiring only one predator with a simple non-specialized behavior for prey-capture.

Validation task 2 tested two predators and one prey, where predator cooperation, but no specialization was needed for prey-capture. CONE evolved predator teams yielded a task performance comparable to CNE and CCGA evolved teams. This was

a result of behaviorally homogenous teams being adequate for prey-capture. Validation task 2 did not require the evolution of behavioral heterogeneity. Thus, CONE's multi-population cooperative co-evolution architecture did not offer any benefits in this task. Similarly, CONE's GDM and SDM (designed to facilitate behavioral during the cooperative co-evolution of teams) were not beneficial since specialization was not required to accomplish validation task 2.

Validation task 3 tested three predators and one prey, where cooperation was required and specialization was beneficial for prey-capture. This task was the same as the pursuit-evasion experiment that tested *team type 2* (Table 1, Sect. 3.4). Validation task 3 (and the pursuit-evasion experiments) demonstrated that CONE evolved predator teams achieved a comparatively high task performance via evolving behavioral heterogeneity. That is, different predators adopted complementary behavioral specializations to collectively form cooperative prey-capture behaviors. This behavioral heterogeneity emerged since, the requirements of validation task 3 mandated the evolution of more complex collective prey-capture behaviors. In this task, CONE's cooperative co-evolution process together with its GDM and SDM evolved forms of behavioral specializations, team behavioral heterogeneity and a collective prey-capture behavior not observed in the comparative cooperative co-evolution methods (CCGA and MESP). This statement is supported by experiments that re-run CONE without the GDM and SDM (Sect. 4.6). Section 4.6 discusses the contribution of the GDM and SDM to CONE evolved prey-capture behaviors, as well as the higher task performance of CONE evolved teams.

Furthermore, CONE was re-executed without *elite controller evaluation*, for all pursuit-evasion experiments. This CONE variant was labeled *CONE-4* (Sect. 4.6). This was an important step in verifying the complexity in CONE's principled approach, since, elite controller evaluation was one of the key differences between CONE and related cooperative co-evolution methods such as CCGA and CONE. The results of CONE-4 experiments indicated that without elite controller evaluation, CONE, CCGA, and MESP evolved teams all yielded comparable average task performances (Fig. 15).

Thus, these validation experiments demonstrated that the complexity of CONE is necessary if CONE is to evolve high performance team behaviors in tasks that require cooperation, where such cooperation is formed from interacting behavioral specializations. Validation task 3 (replicated from the pursuit-evasion experiments) is a prime example of such a task. These validation experiments also elucidated the CONE process, in a step by step application to three pursuit-evasion tasks ranging from simple to complex.

5 Conclusions and future directions

This article evaluated a cooperative co-evolution method (CONE: *Collective Neuro-Evolution*) that effectuated behavioral specialization as a problem solving mechanism in teams of ANN controllers applied to solve a collective behavior task. The research hypothesis was that CONE facilitates behavioral specializations that

benefit the fitness of CONE evolved teams. To test this hypothesis, CONE and related cooperative co-evolution methods were applied to evolve collective prey capture behaviors in a multi-robot pursuit-evasion task. Results indicated that CONE evolved behavioral specializations that were integral to the formation of collective prey capture behaviors. Such collective behaviors achieved a higher fitness (significantly significant), compared to teams evolved by related methods. This study, together with previous work [76], established that behavioral specialization emerges in response to collective behavior task and environment constraints, for the benefit of task accomplishment. In these specific tasks, CONE's genotypic and behavioral difference metrics for regulating inter-population genotype recombination, were key to facilitating such behavioral specialization. More generally, it is theorized that CONE's multiple populations, cooperative co-evolution nature, and its genotypic and behavioral difference metrics are all essential components for effectuating beneficial forms of behavioral specialization in a range of collective behavior tasks. However, this supposition is the subject of ongoing research.

Furthermore, a number of future research directions are envisioned as part of this ongoing research. In order to more thoroughly investigate more complex forms of emergent specialization (behavioral and morphological), as it relates to predator-prey behaviors, we intend to investigate more realistic (three dimensional) simulation environments. For example, a relevant future research direction would be to investigate the impact of morphological adaptations on the co-evolution of aerial pursuit and evasion behaviors, as occurs with various bird species in nature [23, 46, 100]. The goal of such research would be to elucidate how certain behavioral and morphological specializations in predators and prey impact upon the co-evolution of pursuit and evasion strategies and relate them to those observed in nature.

Given that CONE has been demonstrated in collective tasks that require relatively simple forms of behavioral specialization, for small to medium sized agent teams [76], future work will focus upon applying CONE to collective behavior tasks that require and benefit from more complex forms of specialization. For example, morphological specialization in robotic swarm systems [49]. Future work will also investigate the potential for generalizing the principles of CONE to teams of embodied or simulated agents that do not use ANN controllers. Different controller types (for example, rule-based controllers) will test the multi-population structure, co-evolution process, and genotypic and behavioral difference metrics as a means of effectuating specialization and increasing collective behavior task performance. Thus a prevailing future goal is to ascertain if CONE is applicable to a broad range of collective behavior tasks, and if other (non neural) controller types yield similar benefits.

References

1. H. Abdel-Rahman, When do cities specialize in production. Reg. Sci. Urban. Econ. **26**(1), 1–22 (2001)
2. A. Agogino, K. Tumer, Efficient evaluation functions for multi-rover systems, in *Proceedings of the Genetic and Evolutionary Computation Conference* (Springer, New York, 2004), pp. 1–12

3. K. Baev, *Biological Neural Networks*. (Birkuser, Berlin, 1997)
4. T. Balch. Behavioral Diversity in Learning Robot Teams. PhD Thesis. College of Computing, Georgia Institute of Technology, Altanta, USA (1998)
5. T. Balch, Taxonomies of multi-robot task and reward, in *Robot teams: From diversity to polymorphism* (A K Peters, Natick, USA, 2002), pp. 23–35
6. G. Baldassarre, S. Nolfi, D. Parisi, Evolving mobile robots able to display collective behavior. Artif. Life **9**(1), 255–267 (2003)
7. M. Benda, V. Jagannathan, R. Dodhiawalla, *On optimal cooperation of knowledge sources* (Tech. Rep. BCS-G2010-28) (Boeing AI Center, Boeing Computer Services, Bellevue, USA, 1986)
8. J. Blumenthal, G. Parker, Co-evolving team capture strategies for dissimilar robots, in *Proceedings of the Artificial Multi-Agent Learning Symposium* (AAAI Press, Arlinton, Virginia, 2004), pp. 15–23
9. J. Blumenthal, G. Parker, Competing sample sizes for the co-evolution of heterogeneous agents, in *Proceedings of the International Conference on Intelligent Robots and Systems* (IEEE, Sendai, Japan, 2004), pp. 1438–1443
10. J. Blumenthal, G. Parker, Punctuated anytime learning for evolving multi-agent capture strategies, in *Proceedings of the Congress on Evolutionary Computation* (IEEE Press, Portland, USA, 2004), pp. 1820–1827
11. E. Bonabeau, M. Dorigo, G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. (Oxford University Press, Oxford, 1998)
12. E. Bonabeau, A. Sobkowski, G. Theraulaz, J. Deneubourg, Adaptive task allocation inspired by a model of division of labour in social insects, in *Bio-Computing and Emergent Computation* (World Scientific, Singapore, 1997), pp. 36–45
13. E. Bonabeau, G. Theraulaz, J. Deneubourg, Quantitative study of the fixed threshold model for the regulation of division of labour in insect societies. Proc. R. Soc. Lond. B **263**(1), 1565–1569 (1996)
14. B. Bryant, R. Miikkulainen, Neuro-evolution for adaptive teams, in *Proceedings of the Congress on Evolutionary Computation* (IEEE Press, Canberra, 2003), pp. 2194–2201
15. L. Bui, J. Branke, H. Abbass, Diversity as a selection pressure in dynamic environments, in *Proceedings of the Conference on Genetic and Evolutionary Computation* (ACM, Washington D.C, 2005), pp. 1557–1558
16. L. Bull, J. Holland, Evolutionary computing in multi-agent environments: Eusociality, in *Proceedings of the Second Annual Conference on Genetic Programming* (IEEE Press, San Francisco, 1997), pp. 347–352
17. N. Calderone, R. Page, Genotypic variability in age polyethism and task specialization in the honey bee. Behav. Ecol. Sociobiol. **22**(1), 17–25 (1988)
18. K. Chellapilla, D. Fogel, Evolving neural networks to play checkers without expert knowledge. IEEE Trans. Neural Netw. **10**(16), 1382–1391 (1999)
19. C. Clarke, P. Sheppard, Interactions between major genes and polygenes in the determination of the mimetic patterns of papilio dardanus. Evolution **17**, 404–413 (1963)
20. A. Conradie, R. Miikkulainen, C. Aldrich, Adaptive control utilising neural swarming, in *Proceedings of the Genetic and Evolutionary Computation Conference* (2002)
21. A. Conradie, R. Miikkulainen, C. Aldrich, Intelligent process control utilizing symbiotic memetic neuro-evolution, in *Proceedings of the Congress on Evolutionary Computation*. (IEEE Press, Honolulu, USA, 2002)
22. E. De Jong, R. Watson, J. Pollack, Reducing bloat and promoting diversity using multi-objective methods, in *Proceedings of the Congress on Genetic and Evolutionary Computation* (ACM, San Francisco, 2001), pp. 11–18
23. P. Domenici, J. Blagburn, J. Bacon, Animal escapology II: escape trajectory case studies. J. Exp. Biol. **214**(15), 2474–2494 (2011)
24. S. Doncieux, J. Mouret, Behavioral diversity measures for evolutionary robotics, in *Proceedings of the IEEE Congress on Evolutionary Computation* (Springer, Barcelona, Spain, 2010), pp. 1303–1310
25. C. Dunnett, A multiple comparisons procedure for comparing several treatments with a control. J. Am. Stat. As. **50**, 294–299 (1995)
26. A. Eiben, S. Smit, Parameter tuning for configuring and analyzing evolutionary algorithms. Swarm Evol. Comput. **1**(1), 19–31 (2011)
27. A. Eiben, J. Smith, *Introduction to Evolutionary Computing*. (Springer, Berlin, 2003)
28. J. Elman, Finding structure in time. Cogn. Sci. **14**(1), 179–211 (1990)

29. J. Fewell, S. Bertram, Division of labor in a dynamic environment: response by honeybees (apis mellifera) to graded changes in colony pollen stores. Behav. Ecol. Sociobiol. **46**, 171–179 (1999)
30. B. Flannery, S. Teukolsky, W. Vetterling, *Numerical Recipes*. (Cambridge University Press, Cambridge, 1986)
31. D. Floreano, P. Dürr, C. Mattiussi, Neuroevolution: from architectures to learning. Evol. Intell. **1**(1), 47–62 (2008)
32. N. Garcia-Pedrajas, C. Hervas-Martnez, D. Ortiz-Boyer, Cooperative coevolution of artificial neural network ensembles for pattern classification. IEEE Trans. Evol. Comput. **9**(3), 271–302 (2005)
33. J. Gautrais, G. Theraulaz, J. Deneubourg, C. Anderson, Emergent polyethism as a consequence of increased colony size in insect societies. J. Theor. Biol. **215**(1), 363–373 (2002)
34. H. Ghezelayagh, K. Lee, Intelligent predictive control of a power plant with evolutionary programming optimizer and neuro-fuzzy identifier, in *Proceedings of the Congress on Evolutionary Computation* (IEEE Press, Honolulu, USA, 2002), pp. 1308–1313
35. D. Goldberg, Simple genetic algorithms and the minimal, deceptive problem, in *Genetic algorithms and simulated annealing* (Morgan Kaufman, Berlin, Germany, 1987), pp. 74–88
36. F. Gomez, *Robust Non-Linear Control Through Neuroevolution*. PhD thesis. (Department of Computer Sciences, The University of Texas, Austin, 2003)
37. F. Gomez, R. Miikkulainen, Incremental evolution of complex general behavior. Adapt. Behav. **5**(1), 317–342 (1997)
38. B. Grant, P. Grant, Mimicry and warning color at the boundary between races and species, in ed by D. Howard, S. Berlocher, *Endless Forms: Species and Speciation*. (Oxford University Press, Oxford, 1998), pp. 404–422
39. W. Hamilton, The genetical evolution of social behavior. J. Theor. Biol. **7**(1), 1–16 (1964)
40. D. Hawthorne, Genetic linkage of ecological specialization and reproductive isolation in pea aphids. Nature **412**(1), 904–907 (2001)
41. S. Haykin, *Neural Networks: A Comprehensive Foundation*. (Prentice Hall, Ontario, 1995)
42. T. Haynes, S. Sen, Evolving cooperation strategies, in *Proceedings of the First International Conference on Multi-Agent Systems* (MIT Press, Cambridge, 1995), pp. 450–459
43. T. Haynes, S. Sen, Co-adaptation in a team. Int. J. Comput. Intell. Org. **1**(4), 1–20 (1996)
44. T. Haynes, S. Sen, Evolving behavioral strategies in predators and prey, in *Adaptation and Learning in Multi-Agent Systems: Lecture Notes in Computer Science* (Springer, Berlin, 1996), pp. 113–126
45. T. Haynes, S. Sen, Crossover operators for evolving a team, in *Proceedings of Genetic Programming: Second Annual Conference* (Morgan Kaufmann, San Francisco, 1997), pp. 162–167
46. A. Hedenström, M. Rosén, Predator versus prey: on aerial hunting and escape strategies in birds. Behav. Ecol. **12**(2), 150–156 (2001)
47. J. Hertz, A. Krogh, R. Palmer, *Introduction to the Theory of Neural Computation*. (Addison-Wesley, Redwood City, 1991)
48. N. Keerativuttitumrong, N. Chaiyaratana, V. Varavithya, Multi-objective co-operative co-evolutionary genetic algorithm, in *Proceedings of Parallel Problem Solving from Nature* (Springer, Granada, 2002), pp. 288–297
49. S. Kernbach, L. Ricotti, J. Liedke, P. Corradi, M. Rothermel, Study of macroscopic morphological features of symbiotic robotic organisms, in *Proceedings of the International Conference on Intelligent Robots and Systems (Workshop on Self-Reconfigurable Robots)* (IEEE, Nice, 2008), pp. 18–25
50. J. Koza, Evolution of subsumption using genetic programming, in *Proceedings of the European Conference on Artificial Life* (MIT Press, Cambridge, 1992), pp. 110–119
51. J. Lehman, K. Stanley, Abandoning objectives: evolution through the search for novelty alone. Evol. Comput. **19**(2), 189–223 (1994)
52. L. Lehmann, L. Keller, The evolution of cooperation and altruism a general framework and a classification of models. J. Theor. Biol. **19**(1), 1365–1376 (2006)
53. L. Li, A. Martinoli, Y. Mostafa, Emergent specialization in swarm systems, in *Lecture notes in computer science: Vol. 2412. Intelligent data engineering and automated learning* (Springer, Berlin, 2002), pp. 261–266
54. L. Li, A. Martinoli, A. Yaser, Learning and measuring specialization in collaborative swarm systems. Adapt. Behav. **12**(3), 199–212 (2004)
55. S. Luke, Genetic programming produced competitive soccer softbot teams for robocup 97, in *Proceedings of the Third Annual Genetic Programming Conference* (Morgan Kaufmann, San Fransisco, 1998), pp. 204–222

56. S. Luke, C. Hohn, J. Farris, G. Jackson, J. Hendler, Co-evolving soccer softbot team coordination with genetic programming, in *RoboCup-97: Robot Soccer World Cup I* (Springer, Berlin, 1998), pp. 398–411
57. S. Luke, L. Spector, Evolving teamwork and coordination with genetic programming, in *Proceedings of the International Conference on Genetic Programming* (MIT Press, Stanford, USA, 1996), pp. 150–156
58. S. Mahfoud, *Handbook of Evolutionary Computation*. (Taylor and Francis, Amsterdam, 1997)
59. S.W. Mahfoud, *Niching Methods for Genetic Algorithms*. Ph. D. Dissertation. (Department of Computer Science, University of Illinois, Urbana, 1995)
60. J. Mallet, A species definition for the modern synthesis. Trends Ecol. Evol. **10**, 294–299 (1995)
61. J. Mallet, W. McMillan, C. Jiggins, Mimicry and warning color at the boundary between races and species, In: D. Howard, S. Berlocher (eds) *Endless Forms: Species and Speciation*, (Oxford University Press, New York, 1998) pp. 390–403.
62. M. Mataric, Reward functions for accelerated learning, in *Proceedings of the Eleventh International Conference on Machine Learning* (Morgan Kaufmann, San Francisco, USA, 2002), pp. 181–189
63. E. Mayr, *Animal species and evolution*. (Harvard University Press, Cambridge , 1963)
64. D. Merkle, M. Middendorf, Dynamic polyethism and competition for tasks in threshold reinforcement models of social insects. Adapt. Behav. **12**, 251–262 (2004)
65. R. Miikkulainen, Neuroevolution, in *Encyclopedia of Machine Learning* (Springer, New York, 2010)
66. F. Mondada, E. Franzi, P. Ienne, Mobile robot miniaturization: A tool for investigation in control algorithms, in *Proceedings of Third International Symposium on Experimental Robotics* (IEEE Press, Kyoto, 1993), pp. 501–513
67. H. Moriguchi, S. Honiden, Sustaining behavioral diversity in neat, in *Proceedings of the Conference on Genetic and Evolutionary Computation* (ACM Press, Portland, 2010), pp. 611–618
68. J. Mouret, Novelty-based multiobjectivization. In: S. Doncieux, N. Bredeche, J. Mouret (eds) *New Horizons in Evolutionary Robotics: Extended Contributions from the 2009 EvoDeRob Workshop*, (Springer, Berlin, 2011) pp. 139–154.
69. J. Mouret, S. Doncieux, Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity, in *Proceedings of the IEEE Congress on Evolutionary Computation* (ACM Press, Trondheim, 2009), pp. 1161–1168
70. J. Mouret, S. Doncieux, Using behavioral exploration objectives to solve deceptive problems in neuro-evolution, in *Proceedings of the Conference on Genetic and Evolutionary Computation* (IEEE Press, Montreal, 2009), pp. 627–634
71. J. Mouret, S. Doncieux, Encouraging behavioral diversity in evolutionary robotics: an empirical study. Evolutionary Computation **To appear** (2011)
72. A. Murciano, J. Millan, J. Zamora, Specialization in multi-agent systems through learning. Biol. Cybern. **76**(1), 375 382 (1997)
73. Y. Ng, X. Yang, Specialization, information, and growth: A sequential equilibrium analysis. Rev. Dev. Econ. **1**(1), 257–274 (1997)
74. G. Nitschke, Designing emergent cooperation: a pursuit-evasion game case study. Artif. Life Robotics **9**(4), 222–233 (2005)
75. G. Nitschke, M. Schut, A. Eiben, Emergent specialization in biologically inspired collective behavior systems, in *Intelligent Complex Adaptive Systems* (IGI, New York, 2007), pp. 100–140
76. G. Nitschke, M. Schut, A. Eiben, Collective neuro-evolution for evolving specialized sensor resolutions in a multi-rover task. Evol. Intell. **3**(1), 13–29 (2010)
77. G. Nitschke, M. Schut, A. Eiben, Evolving behavioral specialization in robot teams to solve a collective construction task. Swarm Evol. Comput. **2**(1), 25–38 (2011)
78. C. Noirot, J. Pasteels, Ontogenetic development and the evolution of the worker caste in termites. Experientia **43**(1), 851–860 (1987)
79. S. Nolfi, *Evorobot 1.1 User Manual. Technical Report*. (Institute of Cognitive Sciences, National Research Council, Rome, 2000)
80. S. Nolfi, G. Baldassarre, D. Parisi, Evolution of collective behaviour in a team of physically linked robots, in *Applications of Evolutionary Computing* (Springer, Berlin, 2003), pp. 581–592
81. S. Nolfi, D. Floreano, Co-evolving predator and prey robots: Do arm races arise in artificial evolution. Artif. Life **4**(4), 311–335 (1999)
82. S. Nolfi, D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. (MIT Press, Cambridge, 2000)

83. R. Page, S. Mitchell, Self organization and adaptation in insect societies. Philos. Sci. Assoc. **2**, 289–298 (1991)
84. G. Parker, Co-evolving model parameters for anytime learning in evolutionary robotics. Robot. Auton. Syst. **33**(1), 13–30 (2000)
85. A. Perez-Uribe, D. Floreano, L. Keller, Effects of group composition and level of selection in the evolution of cooperation in artificial ants, in *Advances of Artificial Life: Proceedings of the Seventh European Conference on Artificial Life* (Springer, Dortmund, 2003), pp. 128–137
86. M. Potter, *Design and Analysis of a Computational Model of Cooperative Coevolution*. (Computer Science Department, George Mason University, Fairfax, 1997)
87. M. Potter, K. De Jong, Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evol. Comput. **8**(1), 1–29 (2000)
88. M. Potter, L. Meeden, A. Schultz, Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists, in *Proceedings of the International Joint Conference on Artificial Intelligence* (AAAI Press, Seattle, 2001), pp. 1337–1343
89. M. Quinn, A comparison of approaches to the evolution of homogeneous multi-robot teams, in *Proceedings of the Congress Evolutionary Computation* (IEEE Press, Seoul, 2001), pp. 128–135
90. M. Resnick, *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*. (MIT Press, Cambridge, 1997)
91. G. Robinson, Regulation of division of labor in insect societies. Annu. Rev. Entomol. **37**(1), 637–665 (1992)
92. B. Sareni, L. Krahenbuhl, Fitness sharing and niching methods revisited. IEEE Trans. Evol. Comput. **2**(3), 97–106 (1998)
93. C. Schultz, L. Parker, in *Multi-robot Systems: From Swarms to Intelligent Automata* (Kluwer Academic Publishers, Washington DC, 2002)
94. H. Seligmann, Resource partition history and evolutionary specialization of subunits in complex systems. Biosystems **51**(1), 31–39 (1999)
95. F. Seredynski, Competitive coevolutionary multi-agent systems: The application to mapping and scheduling problems. J. Parallel Distrib. Comput. **47**(1), 39–57 (1997)
96. K. Stanley, B. Bryant, R. Miikkulainen, Real-time neuro-evolution in the nero video game. IEEE Trans. Evol. Comput. **9**(6), 653–668 (2005)
97. P. Stone, *Layered Learning in Multi-Agent Systems: A Winning Approach to Robotic Soccer*. (MIT Press, Cambridge, 2000)
98. G. Theraulaz, E. Bonabeau, J. Deneubourg, Response threshold reinforcement and division of labour in insect societies. Proc. R. Soc. Lond. B **265**(1), 327–332 (1998)
99. A. Toffolo, E. Benini, Genetic diversity as an objective in multi-objective evolutionary algorithms. Evol. Comput. **11**(2), 151–167 (2003)
100. van den P. Houta, K. Mathotb, L. Maasc, T. Piersma, Predator escape tactics in birds: linking ecology and aerodynamics. Behav. Ecol. **21**(1), 16–25 (2011)
101. M. Waibel, D. Floreano, L. Keller, Genetic team composition and level of selection in the evolution of cooperation. IEEE Trans. Evol. Comput. **13**(3), 648–660 (2009)
102. M. Waibel, D. Floreano, S. Magnenat, L. Keller, Division of labor and colony efficiency in social insects: effects of interactions between genetic architecture, colony kin structure and rate of perturbations. Proc. R. Soc. B **273**(1), 1815–1823 (2006)
103. T. Wenseleers, F. Ratnieks, J. Billen, Caste fate conflict in swarm-founding social hymenoptera: an inclusive fitness analysis. Evol. Biol. **16**(1), 647–658 (2003)
104. S. Whiteson, N. Kohl, R. Miikkulainen, P. Stone, Evolving keep-away soccer players through task decomposition, in *Proceeding of the Genetic and Evolutionary Computation Conference* (AAAI Press, Chicago 2003), pp. 356–368
105. R. Wiegand, *An Analysis of Cooperative Coevolutionary Algorithms*. PhD. Thesis. (George Mason University Press, George Mason University, Fairfax, 2004)
106. A. Wieland, Evolving neural network controllers for unstable systems, in *Proceedings of the International Joint Conference on Neural Networks* (IEEE Press, Seattle, 1991), pp. 667–673
107. M. Wineberg, F. Oppacher, The underlying similarity of diversity measures used in evolutionary computation, in *Proceedings of the Genetic and Evolutionary Computation Conference* (Springer, 2003), pp. 1493–1504
108. X. Yao, Evolutionary artificial neural networks. J. Neural Syst. **4**(3), 203–222 (1993)
109. C. Yong, R. Miikkulainen, Co-evolution of role-based cooperation in multi-agent systems. IEEE Trans. Auton. Mental Dev. **1**(3), 170–186 (2010)