Multirobot Behavior Synchronization through Direct Neural Network Communication

In: ***Proceedings of the 5th International Conference on Intelligent Robotics and Applications (ICIRA-2012).*** **New York, NY: Springer-Verlang, 2012.**

David B. D'Ambrosio, Skyler Goodell, Joel Lehman, Sebastian Risi, and Kenneth O. Stanley

Department of Electrical Engineering and Computer Science
University of Central Florida
Orlando, Florida 32816-2362, USA
{ddambro,goodsky,jlehman,srisi,kstanley}@eecs.ucf.edu
http://eplex.cs.ucf.edu/

Abstract. Many important real-world problems, such as patrol or search and rescue, could benefit from the ability to train teams of robots to coordinate. One major challenge to achieving such coordination is determining the best way for robots on such teams to communicate with each other. Typical approaches employ hand-designed communication schemes that often require significant effort to engineer. In contrast, this paper presents a new communication scheme called the *hive brain*, in which the neural network controller of each robot is directly connected to internal nodes of other robots and the weights of these connections are evolved. In this way, the robots can evolve their own internal "language" to speak directly brain-to-brain. This approach is tested in a multirobot patrol synchronization domain where it produces robot controllers that synchronize through communication alone in both simulation and real robots, and that are robust to perturbation and changes in team size.

Keywords: Evolutionary Algorithms, HyperNEAT, Multirobot Teams, Coordination, Communication, Artificial Neural Networks

1 Introduction

As robot technology has matured and large teams of robots have become more commonplace, a research question of growing importance is how to best *coordinate* such robotic teams. While one approach is to coordinate robot teams centrally, scaling such an approach to many robots and mitigating the inherent challenges of limited bandwidth and unreliable communication in the real world may prove problematic [1]. Thus this paper instead focuses on treating robots as autonomous *communicating* agents, which notably has proven a robust and scalable strategy in nature [2, 3]. For example, insect colonies and human society itself operate by this principle.

Importantly, communication between agents enlarges the scope of their possible behaviors by enabling coordination and sharing of knowledge. In this way,

teams of communicating robots may have greater potential than those without communication. However, an open question in this context is how to best implement an artificial communication system that allows autonomous robotic agents to coordinate their behavior. That is, it is unclear a priori how exactly robots should pass information to each other. In nature, communication between organisms takes diverse forms. For example, ants emit and detect pheromone signals [2], bees dance and recognize visual dancing patterns [3], and humans vocalize and interpret the complex auditory signals comprising speech. Interestingly, in an artificial system it is possible to consider communication systems impossible or unlikely to be exploited by nature.

In particular, if individual robotic agents' policies are represented by artificial neural networks (ANNs), communication between agents can be implemented by connections *between* such networks. In other words, one agent's brain can directly feed into another's. By analogy, one way of understanding such a system is to imagine it as a form of telepathy; information from one agent's brain can flow into another's. The advantage of such an approach is that it bypasses the complexity of signal transduction. That is, it is unnecessary to encode a message first into an orthogonal form such as scent, movement, or sound before transmitting it, and it is symmetrically unnecessary for the recipient to decode it; a bee with such connections to other bees would not need to dance to indicate to others where to find food. While the laws of physics prevent such direct connections between the brains of biological organisms, distributed implementations of ANNs have no such limitation (though of course inter-network connections may incur some communication delay).

This insight motivates the novel approach presented in this paper, called the *hive brain*, in which agents are controlled by interconnected ANNs. This new approach for creating communicating multirobot teams is built upon the foundation of an established evolutionary algorithm called *multiagent HyperNEAT* (MAHN [4]) that is extended to represent such ANN interdependence and can scale to evolve teams with many robots. The hive brain extension allows one robot's ANN to interconnect with others' to enable communication between them. Interestingly, in this way the communication scheme itself can evolve.

This paper investigates a particular kind of collective behavior that such a hive brain can facilitate. It is inspired by an interesting physical phenomenon called *odd sympathy* [5], which is the tendency of pendulum clocks to synchronize when mounted near each other. The cause of such synchronization is that vibrations from one clock affect the other through the medium on which they are mounted. In other words, a small amount of physical *information* is transferred between the pendulums that results in a larger macrolevel effect, i.e. synchronization. An interesting analogy for robot teams would arise if communicating small amounts of information could likewise cause them to synchronize, which might have practical applications for teams that must cooperate in tasks sensitive to timing.

Thus this paper applies the hive brain approach for the first time to evolving robot teams that synchronize their movements over time. Teams that success-

fully synchronize are artificially evolved in a computer simulation, demonstrating that the hive brain can facilitate "odd sympathy"-like behavior in robot controllers. Furthermore, evolved teams are successfully transfered to the real world in Khepera III robots, illustrating the real-world potential of the technique. The conclusion is that the hive brain is an interesting new technique for evolving communicating teams of robots that merits further exploration.

2 Background

This section reviews past work in cooperative multiagent learning that requires communication and the NEAT and HyperNEAT methods applied in the experiments presented in this paper.

2.1 Cooperative Multiagent Learning

There are two primary traditional approaches to training multiple agents to collaborate to solve a given task. The first, multiagent reinforcement learning (MARL), encompasses several specific techniques based on off-policy and on-policy temporal difference learning [6–8]. The basic principle that unifies MARL techniques is to identify and reward promising cooperative states and actions among a team of agents [9, 10]. The other major approach, cooperative coevolutionary algorithms (CCEAs), is an established evolutionary method for training teams of agents that must work together [11, 12, 10]. The main idea is to maintain one or more populations of candidate agents, evaluate them in groups, and guide the creation of new candidate solutions based on their joint performance. However, while these approaches are effective in a number of domains [6–10], their focus is usually not on agents with explicit communication channels.

2.2 Communicating Robots

While communication among robots is not always necessary, tasks that require coordination without a central controller can benefit from robots that are able to share information about the world.

One such class of problems, which has been the subject of many studies, is known as the consensus problem [13]. In the consensus problem multiple agents must reach an *agreement* about the current state of the world, which requires shared information. Consensus schemes have been applied to various multiagent tasks, ranging from vehicle formations [14], coupled oscillators [15], and robot position synchronization [16]. While previous work on the consensus problem has focused on analytical approaches [13], if the individual robotic agents' policies are represented as ANNs, the agents should in principle be able to reach consensus through minimal communication as well.

Different ANN-based approaches to communication have been investigated in the past, in which the communication between the agents takes diverse forms. In Yong and Mikkulainen [17], each agent receives directly the position of the other

agents as input. Di Palolo [18] studied agents that cooperate acoustically, while the focus of Floreano et al.'s [19] work was evolving robots that communicate by emitting light. However, unlike the approach introduced in this paper, all these approaches rely on signal transduction (i.e. a message is encoded, sent and then decoded) between a sender and a receiver.

Nevertheless, if the agents in a distributed system are controlled by neural networks, communication between agents could potentially also be implemented by direct connections between such networks. In other words, one agent's brain can directly feed into another's, which makes encoding and decoding messages unnecessary. The HyperNEAT approach should allow such controllers to be evolved because it can easily represent such ANN interdependence. The next section reviews the Neuroevolution of Augmenting Topologies (NEAT) approach, the foundation of HyperNEAT.

2.3 Neuroevolution of Augmenting Topologies

The HyperNEAT approach extended in this paper is itself an extension of the original NEAT algorithm that evolves increasing large ANNs. NEAT starts with a population of simple networks that then *increase in complexity* over generations by adding new nodes and connections through mutations. By evolving ANNs in this way, the topology of the network does not need to be known a priori; NEAT searches through increasingly complex networks to find a suitable level of complexity. Because it starts simply and gradually adds complexity, it tends to find a solution network close to the minimal necessary size. However, as explained next, it turns out that directly representing connections and nodes as explicit genes in the genome cannot scale up to large brain-like networks. For a complete overview of NEAT see Stanley and Miikkulaninen [20].

2.4 HyperNEAT

Many neuroevolution methods are *directly encoded*, which means each component of the phenotype is encoded by a single gene, making the discovery of repeating motifs expensive and improbable. Therefore, indirect encodings [21–23] have become a growing area of interest in evolutionary computation and artificial life.

One such indirect encoding designed explicitly for neural networks is the Hypercube-based NEAT (HyperNEAT) approach [24, 25], which is itself an indirect extension of the directly-encoded NEAT approach [26, 20] reviewed in the previous section. This section briefly reviews HyperNEAT; a complete introduction is in Stanley et al. [24] and Gauci and Stanley [25]. Rather than expressing connection weights as distinct and independent parameters in the genome, HyperNEAT allows them to vary across the phenotype in a regular pattern through an indirect encoding called a *compositional pattern producing network* (CPPN; [27]), which is like an ANN, but with specially-chosen activation functions.

CPPNs in HyperNEAT *encode* the connectivity patterns of ANNs as a *function of geometry*. That is, if an ANN's nodes are embedded in a geometry, i.e.

assigned coordinates within a space, then it is possible to represent its connectivity as a single evolved function of such coordinates. In effect the CPPN paints a pattern of weights across the geometry of a neural network. To understand why this approach is promising, consider that a natural organism's brain is physically embedded within a three-dimensional geometric space, and that such embedding heavily constrains and influences the brain's connectivity. Topographic maps (i.e. ordered projections of sensory or effector systems such as the retina or musculature) are realized within brains that preserve geometric relationships between high-dimensional sensor and effector fields [28, 29]. In other words, there is important information *implicit* in geometry that can only be exploited by an encoding informed by geometry.

In particular, geometric *regularities* such as symmetry or repetition are pervasive throughout the connectivity of natural brains. To similarly achieve such regularities, CPPNs exploit activation functions that induce regularities in HyperNEAT networks. The general idea is that a CPPN takes as input the geometric coordinates of two nodes embedded in the *substrate*, i.e. an ANN situated in a particular geometry, and outputs the weight of the connection between those two nodes. In this way, a Gaussian activation function by virtue of its symmetry can induce symmetric connectivity and a sine function can induce networks with repeated elements. Note that because the size of the CPPN is decoupled from the size of the substrate, HyperNEAT can compactly encode the connectivity of an arbitrarily large substrate with a single CPPN.

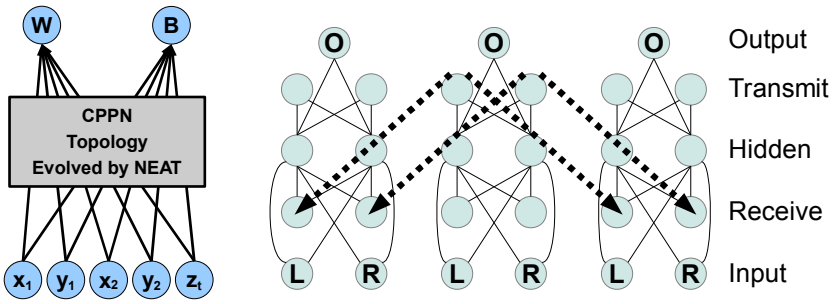
HyperNEAT also allows the evolution of controllers for *teams* of agents. This multiagent HyperNEAT algorithm was first introduced by D'Ambrosio and Stanley [30] and D'Ambrosio et. al [4]. It is designed to work with homogeneous and heterogeneous teams; however, in this paper the tasks only necessitates the homogeneous case. While previous experiments with multiagent HyperNEAT did not involve communication between the agents, the next section introduces such a model, which should allow the agents to synchronize their movements over time.

3 Hive Brain Approach

While HyperNEAT has been applied to multiagent problems in the past [4, 31, 30], these previous experiments did not involve communication between the agents. This paper extends HyperNEAT by allowing it to define inter-network connections that act as communication channels between agents. Such communication can be advantageous in situations where agents must cooperate closely or come to consensus, such as in this paper. Because the agents communicate through regular ANN connections, the experimenter does not need to define a specific "language" for communication, and in fact, the agents can devise one of their own that is easily integrated into their neural architecture.

For simplicity, the teams in this paper are composed of homogeneous agents, that is, agents who all have the same control policy. Thus HyperNEAT needs only to create a single ANN and communication scheme and copy this plan to all

agents in the team. The substrate (i.e. the controller ANN and its geometry) for a single agent is shown in figure 1b and is made up of five layers: input, receive, hidden, transmit, and output. The input, hidden, and output layers are familiar ANN constructs, but the transmitting and receiving layers are additions that facilitate communication between agents. The transmit layer takes input from one agent's hidden layer and sends it to another agent's receiving layer, which in turn inputs into the target agent's hidden layer. In this way messages can be passed among the entire team of agents. The weights of these connections are encoded by the CPPN through the inclusion of CPPN input z_t (figure 1a) that defines the target agent of a connection. In this paper communication is limited to left and right neighbors, so z_t is defined as -1 for left neighbors, 0 for intra-agent connections, and 1 for the right neighbor, although more general communication schemes are possible.



(a) CPPN (Encodes Substrate)

(b) Substrate (ANNs)

Fig. 1: Hive brain substrate. The CPPN (a) that encodes the connection weights in a hive substrate (b) is augmented with a z_t input that determines the target of the connection. The hive substrate (b) includes input, output, and hidden layers. However two of the hidden layers are designated as transmitting and receiving layers that are used for communication. The flow of information between agents is shown by the dashed lines. The inputs are the left and right sensors and the output is interpreted as a motor command, which are discussed in Section 4.

4 Synchronization Experiment

To demonstrate the ability to communicate, the robots are tested on their ability to synchronize their movements as they patrol a room. This domain is motivated by the natural phenomenon called odd sympathy [5], in which closely situated pendulums tend to synchronize their motions through tiny vibrations. In effect these vibrations are like simple messages; thus the hive brain communication

scheme should in principle be able to produce similar results. In this domain, synchronization is defined for the robots as *moving in unison* such that their patrol trajectories begin and end at the same time. To focus on the issue of synchronization, the robots are restricted to moving left and right within a rectangular room. Thus an optimal solution would be for all robots to reverse direction at the same time before hitting a wall. Overall, the robots should oscillate back and forth between walls in synchrony, thereby covering the room reliably and systematically. The challenge is that the robots are not *started* in synchrony, so they must cooperate to achieve it, which tests the ability of the hive to produce a coordinated result. An interesting advantage of this approach is that it does not require explicit positional information to be encoded in the agents, which may not be easily obtained in the real world due to sensor ambiguity.

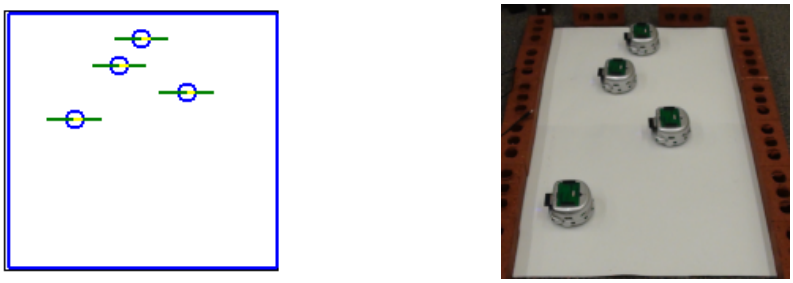
Each robot has left and right rangefinder sensors that return the distance of the robot to a wall, up to 30 cm, normalized between 0.0 and 1.0. The robot has a single effector that determines both the direction and speed of motion: an output between 0 and 0.5 will cause the robot to move left at a speed between 2.5 cm/sec and 5.0 cm/sec based on the value of the output. Similarly, an output between 0.5 and 1.0 will cause the robot to move right at a velocity within that same range. To determine the output for a given timestep, each network is activated four times: enough so that information about an agent's current input can travel to the output, but not enough times for information about neighbors' inputs to reach the output, effectively creating a communication delay. Importantly, the robots cannot see each other and can only see walls, so they *must* rely on communication to properly synchronize.

Training (i.e. evolution) occurs in a robot simulator (figure 2a) where teams of four agents are trained on five different initial configurations of robots between two walls between which they must patrol. These configurations are created by lining the robots up in the center of the room and then staggering their positions by different amounts. This approach forces the robots to find a general syncing policy that can work from multiple starting configurations. Teams are trained by HyperNEAT with a multiobjective reward scheme through NSGAII [32] with three objectives: (1) minimize the distance between each robot and its neighbors, (2) patrol back and forth in the room, and (3) maintain genomic diversity (calculated through NEAT speciation [20]). The distance objective is calculated by, at each time step, summing the distance from each robot to the next robot in the line and then dividing by the number of robots minus one. The average of these normalized sums over all time steps is the distance objective. The patrolling objective is simply the average number of patrol cycles performed by each robot divided by 25 (the maximum number of cycles possible in the given time).

An ideal team should quickly synchronize their behaviors so that the distances are minimized and continually patrol the room without crashing or getting stuck. Because teams start out desynchronized it is impossible to completely maximize both objectives, so a team with a normalized sum of 1.9 or greater is considered to have solved the problem in the simulator. The third objective,

diversity, is only included to increase the efficiency of the optimization process, i.e. it is not the explicit goal of the experiment, so it is not included in the performance measure. The multiobjective approach was found useful in this domain because the two tasks of minimizing distance and patrolling are simple to accomplish on their own, but deceptively complex to accomplish simultaneously.

Selected teams were later transferred to actual Khepera III robots to perform the same task in the real-world (figure 2b). These robots have several IR rangefinder sensors: the rear sensor serves as the "left" sensor from simulation, and because there is no direct front sensor, the average of the two front-most sensors is used to determine the value of the "right" sensor. The robots are placed on a posterboard surface between two walls 68.5cm apart that are made of red bricks. In addition to the training size of four robots, the scalability of the solutions was also tested in the real world by testing teams of three and five robots from the *same* CPPN that was trained on only four. Furthermore, the robustness of solutions was tested by manually desynchronizing the robots once they already synchronized themselves.



(a) Simulated Robots (b) Actual Robots

Fig. 2: Synchronization Domain. In this domain, robots are placed inside a room such that their positions are initially staggered. The goal is for them to move such that their motions become synchronized.

4.1 Experimental Parameters

Because HyperNEAT differs from original NEAT only in its set of activation functions, it uses the same parameters [20]. The experiment was run with a modified version of the public domain SharpNEAT package [33]. The size of the population was 500 with 20% elitism. The number of generations was 300. Sexual offspring (50%) did not undergo mutation. Asexual offspring (50%) had 0.96 probability of link weight mutation, 0.03 chance of link addition, and 0.01 chance of node addition. The coefficients for determining species similarity were 1.0 for nodes and connections and 0.1 for weights. The available CPPN activation functions were sigmoid, Gaussian, absolute value, and sine, all with equal probability

of being added to the CPPN. Parameter settings are based on standard SharpNEAT defaults and prior reported settings for NEAT [20, 34]. They were found to be robust to moderate variation through preliminary experimentation.

5 Results

From 20 simulated runs of evolution, 17 runs evolved a solution with performance above or equal to the success threshold of 1.90. On average the algorithm took 41 generations to find such a solution (stddev = 52). Of these solutions, all but two maintained a performance of 1.90 or greater when tested on untrained team sizes of three and five in simulation.

The best-performing team in each run that produced a solution was transferred to real Khepera III robots. Of the 17 runs that were successful in simulation, 15 were able to duplicate that behavior in real robots (figure 3). When robots were agitated during evaluation, i.e. removed from formation and then returned to the room out of sync, the teams could quickly resynchronize their motions. Additionally, the two teams that had the highest sum of objectives during simulation were further tested on their ability to synchronize in real robots when scaled to teams of three and five robots (on which they were not trained) and were able to successfully synchronize. Videos of the robots synchronizing can be found at `http://eplex.cs.ucf.edu/demos/hive-brain-patrol`.

Every working solution demonstrated a similar strategy: The team moves in the same direction towards a wall until the first agent detects the wall with its rangefinder. At that point all other agents either move towards the wall to catch up or oscillate near the wall while waiting for other agents to gather. Once enough agents gather at the wall or enough time passes all agents ultimately flip direction and run the same procedure on the opposite wall. Over several such iterations the agents lock into phase. This behavior was observed both in solutions running in the simulator and the solutions transfered to the actual Khepera III robots.

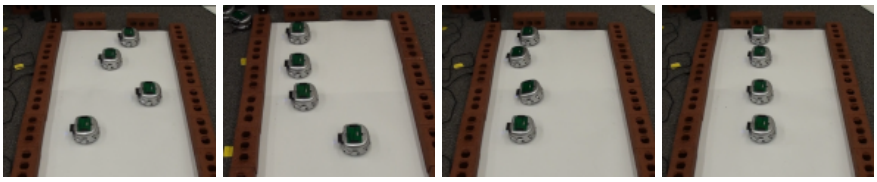


Fig. 3: Synchronization Example. The robots start out disorganized and unaligned, but using communication they are able to synchronize their movements.

Another capability that derives from this kind of team is the ability to synchronize regardless of the initial direction of each agent. With no further training the teams are able to synchronize against alternating opposite walls in real

robots, creating a staggered patrolling pattern that covers more of the room at any given time (as can be seen in the videos at the above address), demonstrating that a variety of behaviors can be synchronized from this approach, and suggesting the potential for learning more complex patrol routes in the future.

6 Discussion

The results demonstrate that the hive brain approach to communication does allow agents to learn on their own to communicate effectively. Thus even with only the ability to communicate with direct neighbors the team is coming to a simple consensus about when it should leave the wall, which clearly requires communication and is the subject of many studies [35].

The hive brain approach was also often successful in real robots despite sensor and actuator noise (and manual desynchronization), implying that the policies discovered can be robust. The ability for teams to scale to new sizes, despite only being trained with four agents, also suggests that a general synchronization policy is consistently found. Perhaps most interestingly, the multiagent hive brain makes up its own communication strategy to solve the task, without any a priori programming.

These results open up a number of possible directions for future research based on the hive brain. First, a natural step would be to exploit multiagent HyperNEAT to create heterogeneous teams of communicating agents, whose communication can also thereby be heterogeneous. The robots in this paper were homogeneous and had predefined partners with which to communicate; however it would also be possible to allow HyperNEAT to decide with whom they need to talk to accomplish the task. Preliminary experiments suggest that letting every agent talk to every other agents produces too much cross-talk, resulting in poor performance. However, approaches like HyperNEAT-LEO [36] could potentially allow HyperNEAT to define appropriate patterns of communication on its own. There is also substantial room to explore how the communication connections are configured; in this paper there are explicit transmitting and receiving layers for communication, but other approaches such as directly connecting output layers to hidden layers could prove useful depending on the domain. Finally, the hive brain approach could be combined with other learning algorithms in addition to HyperNEAT.

7 Conclusion

This paper demonstrated a new multiagent communication technique called the hive brain, in which the ANNs of agents are directly connected to each other. The initial demonstration of this approach in this paper allowed simulated agents to synchronize their movements through communication and the strategies that were discovered were verified in actual Khepera III robots. This new technique opens the door to significant additional research and applications by suggesting a new way of thinking about robot communication.

Acknowledgments. This work was supported through grants from the US Army Research Office (Award No. W911NF-11-1-0489) and DARPA Computer Science Study Group Phase III (Award No. N11AP20003). This paper does not necessarily reflect the position or policy of the government, and no official endorsement should be inferred.

References

1. Busoniu, L., Babuska, R., De Schutter, B.: A comprehensive survey of multiagent reinforcement learning. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on **38**(2) (2008) 156–172
2. Jackson, D.E., Ratnieks, F.L.: Communication in ants. Current biology **16**(15) (2006) R570–R574
3. Riley, J., Greggers, U., Smith, A., Reynolds, D., Menzel, R.: The flight paths of honeybees recruited by the waggle dance. Nature **435**(7039) (2005) 205–207
4. D'Ambrosio, D., Lehman, J., Risi, S., Stanley, K.O.: Evolving policy geometry for scalable multiagent learning. In: Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2010), International Foundation for Autonomous Agents and Multiagent System (2010) 731–738
5. Bennett, M., Schatz, M.F., Rockwood, H., Wiesenfeld, K.: Huygens's clocks. Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences **458**(2019) (2002) 563–579
6. Bowling, M., Veloso, M.: Multiagent learning using a variable learning rate. Artificial Intelligence **136**(2) (2002) 215–250
7. Hu, J., Wellman, M.P.: Multiagent reinforcement learning: theoretical framework and an algorithm. In: Proc. 15th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA (1998) 242–250
8. Santana, H., Ramalho, G., Corruble, V., Ratitch, B.: Multi-agent patrolling with reinforcement learning. Autonomous Agents and Multiagent Systems, International Joint Conference on **3** (2004) 1122–1129
9. Busoniu, L., Schutter, B.D., Babuska, R.: Learning and coordination in dynamic multiagent systems. Technical Report 05-019, Delft University of Technology (2005)
10. Panait, L., Luke, S.: Cooperative multi-agent learning: The state of the art. Autonomous Agents and Multi-Agent Systems **3**(11) (November 2005) 383–434
11. Ficici, S., Pollack, J.: A game-theoretic approach to the simple coevolutionary algorithm. Lecture notes in computer science (2000) 467–476
12. Panait, L., Wiegand, R., Luke, S.: Improving coevolutionary search for optimal multiagent behaviors. Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI) (2003) 653–658
13. Ren, W., Beard, R., Atkins, E.: A survey of consensus problems in multi-agent coordination. American Control Conference, 2005. Proceedings of the 2005 (June 2005) 1859–1864 vol. 3
14. Fax, J.A., Murray, R.M.: Information flow and cooperative control of vehicle formations. Automatic Control, IEEE Transactions on **49**(9) (2004) 1465–1476
15. Sepulchre, R., Leonard, N.: Collective motion and oscillator synchronization. Electrical Engineering **309** (2004) 189–205
16. Rodriguez-Angeles, A., Nijmeijer, H.: Mutual synchronization of robots via estimated state feedback: A cooperative approach. IEEE Trans. on Control Systems Technology **12**(4) (2004) 542–554

17. Yong, C.H., Miikkulainen, R.: Coevolution of role-based cooperation in multiagent systems. IEEE Transactions on Autonomous Mental Development **1** (2010) 170–186
18. Di Paolo, E.A.: Behavioral coordination, structural congruence and entrainment in a simulation of acoustically coupled agents. Adaptive Behavior **8**(1) (2000) 27–48
19. Floreano, D., Mitri, S., Magnenat, S., Keller, L.: Evolutionary conditions for the emergence of communication in robots. Current Biology **17**(6) (2007) 514 – 519
20. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation **10** (2002) 99–127
21. Bongard, J.C., Pfeifer, R. Morpho-functional Machines: The New Species (Designing Embodied Intelligence). In: Evolving complete agents using artificial ontogeny. Springer-Verlag (2003) 237–258
22. Hornby, G.S., Pollack, J.B.: Creating high-level components with a generative representation for body-brain evolution. Artificial Life **8**(3) (2002) 223–246
23. Stanley, K.O., Miikkulainen, R.: A taxonomy for artificial embryogeny. Artificial Life **9**(2) (2003) 93–130
24. Stanley, K.O., D'Ambrosio, D.B., Gauci, J.: A hypercube-based indirect encoding for evolving large-scale neural networks. Artificial Life **15**(2) (2009)
25. Gauci, J., Stanley, K.O.: Autonomous evolution of topographic regularities in artificial neural networks. Neural Computation (2010) 38 To appear.
26. Stanley, K.O., Miikkulainen, R.: Competitive coevolution through evolutionary complexification. Journal of Artificial Intelligence Research **21**(1) (2004) 63–100
27. Stanley, K.O.: Compositional pattern producing networks: A novel abstraction of development. Genetic programming and evolvable machines **8**(2) (2007) 131–162
28. Udin, S., Fawcett, J.: Formation of topographic maps. Annual review of neuroscience **11**(1) (1988) 289–327
29. Hubel, D.H., Wiesel, T.N.: Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of Physiology **160** (1962) 106–154
30. D'Ambrosio, D.B., Stanley, K.O.: Generative encoding for multiagent learning. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008), New York, NY, ACM Press (2008)
31. D'Ambrosio, D., Lehman, J., Risi, S., Stanley, K.O.: Task switching in multiagent learning through indirect encoding. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS 2011), Piscataway, NJ, IEEE (2011)
32. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. Evolutionary Computation, IEEE Transactions on **6**(2) (2002) 182–197
33. Green, C.: SharpNEAT homepage. `http://sharpneat.sourceforge.net/` (2003–2006)
34. Stanley, K.O., Miikkulainen, R.: Competitive coevolution through evolutionary complexification. **21** (2004) 63–100
35. Ren, W., Beard, R., Atkins, E.: A survey of consensus problems in multi-agent coordination. In: American Control Conference, 2005. Proceedings of the 2005, IEEE (2005) 1859–1864
36. Verbancsics, P., Stanley, K.: Constraining connectivity to encourage modularity in HyperNEAT. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation, ACM (2011) 1483–1490