Emergence of Memory in Reactive Agents Equipped with Environmental Markers

Ji Ryang Chung, *Student Member, IEEE,* and Yoonsuck Choe, *Member, IEEE*

***Abstract*—In the neuronal circuits of natural and artificial agents, memory is usually implemented with recurrent connections, since recurrence allows past agent state to affect the present, on-going behavior. Here, an interesting question arises in the context of evolution: how reactive agents could have evolved into cognitive ones with internalized memory? Our idea is that reactive agents with simple feedforward circuits could have achieved behavior comparable to internal memory if they can drop and detect external markers (e.g. pheromones or excretions) in the environment. We tested this idea in two tasks (ball-catching and food-foraging task) where agents needed memory to be successful. We evolved feedforward neural network controllers with a dropper and a detector, and compared their performance with recurrent neural network controllers. The results show that feedforward controllers with external material interaction show adequate performance compared to recurrent controllers in both tasks. This means that even memoryless feedforward networks can evolve behavior that can solve tasks requiring memory, when material interaction is allowed. These results are expected to help us better understand the possible evolutionary route from reactive to cognitive agents.**

***Index Terms*—Environmental memory, pheromone, emergent behavior, neuroevolution**

I. Introduction

In the neuronal networks of natural or artificial agents, memory is usually implemented with recurrent connections. It is well established that for a neural network to show some level of memory, it needs recurrent connections [1], [2]. Also, the brain does not have a strictly hierarchical organization: it has rich sets of reciprocal projections and loops (see e.g. [3]).

However, the nervous system of primitive creatures may have been limited to a feedforward topology, thus exhibiting only reactive (reflexive) behavior [4], [5], [6]. How could this kind of primitive nervous system evolve to be equipped with a full memory system, conferring the agent the ability to perform cognitive tasks? A clue to this question can be found in the olfactory system. The most immediate sensory information that led to internalized memory may have been olfaction, or chemoreception. It is one of the most fundamental capacities already existing in the simplest early animals [7]. Also, even the earliest creatures must have had chemical dropping behavior in the form of secretion or excretion because these are natural bi-product of metabolism. Combined, the two functions could have led to a memory capacity. Wadhams and Armitage showed that a highly specialized chemotaxis sytem exists in primitive creatures like the bacteria, which can distinguish more than 50 different types of proteins and a large portion of their genome is dedicated to the encoding in the chemotaxis system [8]. Furthermore, Tang-Martinez proposed the possibility that self-generated olfactory cues could have been used to develop elementary intelligence to recognize kin by *phenotype matching* [9]. Because this discrimination of kinship is essential for the survival of the organism, even sea squirts without brain can identify their kins using similar chemical cues [10]. Kin recognition does not require any long term memory: simple genetic encoding will suffice (e.g., the MHC-based recognition in mice and tadpoles). It is also suggested that chemical cues may have evolved into communication signals [11], [12]. Sorensen and Stacey(1999) and Wyatt(2003) contemplated that leaking hormones or other metabolites could have been the origin of pheromone. Furthermore, cognitive uses of such chemical signal could have evolved into internalized neuromodulators (cf. [13]). These studies support the biological feasibility of the two functions discussed above; olfaction and secretion.

Another interesting connection between olfaction and memory can be found in the relationship between the olfactory system and the hippocampus in the mammalian brain. Researchers including [14] have shown the anatomical and functional proximities between the olfactory bulb and the hippocampus. The olfactory bulb is a structure in mammalian forebrain where the olfactory pathway starts, and the hippocampus is a brain region involved in spatial memory. Anatomically, the olfactory bulb is located only a couple of synapses away from the hippocampus. Moreover, unlike other senses, olfactory sense bypasses the thalamus and directly feeds into the limbic system which includes the hippocampus. Because of these facts, hippocampus was once believed to be part of the olfactory system. This unique adjacency between the olfactory and the memory system is further emphasized in Proustian retrieval of autobiographical memory [15]. This is a phenomenon that autobiographical memory can be best retrieved by olfactory cues. Researchers believe that the possible involvement of the olfactory bulb in memory consolidation, due to its anatomical closeness to the amygdala and the hippocampus, makes olfaction also involved in memory formation [16].

In addition, embryological evidence suggests genetic kinship between the olfactory bulb and the hippocampus. In an effort to show that neurogenesis is not restricted to the embryonic period but also occurs in the adult mammalian nervous system, Frisén *et al.* found that neurogenesis in adults is most often observed in the olfactory bulb and the hippocampus, but rarely elsewhere [17]. Machold *et al.* and

Ji Ryang Chung and Yoonsuck Choe are with the Department of Computer Science and Engineering, Texas A&M University, 3112 TAMU College Station, TX, 77843-3112, USA e-mail: jchung@cs.tamu.edu, choe@tamu.edu.

Palma *et al.* tested the requirement for hedgehog signaling in the telencephalon and the subventricular zone [18], [19]. They mutated the Sonic hedgehog (Shh) gene, a ligand in the hedgehog signaling pathway regulating vertebrate organogenesis, to examine the resulting postnatal abnormalities. They found that Shh mutation affected the hippocampus but surprisingly it also affected the olfactory bulb. These results altogether imply a close genetic relationship between the olfactory bulb and the hippocampus.

Our main idea is that allowing reactive agents (with feedforward neural circuits) to produce and detect external markers could have served as an intermediate stage between reactive and cognitive behavior (cf. [20], [21] on the use of "inert" matter for augmenting cognition). We focus particularly on the use of chemical markers because of the potential relationship between olfaction (chemoreception) and memory. We investigate the evolution of memory by showing how reactive, feedforward networks can evolve a memory function, through utilizing external markers dropped and detected in the environment. We equiped a feedforward neural network controller with the ability to drop and detect external markers in the environment and trained it using neuroevolution. The network controller was tested in two different tasks, ball-catching task and food-foraging task. Such tasks require spatial memory, which is known to exist in the hippocampus [22], [23], [24], thus the tasks provide a mini-domain to relate our results to the discussion above. Because our idea is that this kind of agent could have been an evolutionary bridge between purely reactive agents and fully memory-capable agents, we expect to see its performance close or parallel to recurrent network controllers. The rest of the paper is organized as follows. Sec. II presents the method and results from the ball-catching task, and Sec. III, those from the food-foraging task. We will discuss interesting points arising from this research (Sec. IV), and conclude our paper in Sec. V.

II. Task I: Catching Falling Balls

In order to test if the use of external markers can work as well as the use of recurrent memory, we used a delayed-response task inspired by [25], [26] (Fig. 1). An agent controlled by a neural network moves horizontally at the bottom of the 2D environment while trying to catch falling balls (the movement is in 1D). The environment size was 400×400. The agent had an array of five range sensors with limited radius (=200). Two balls are dropped from the top at different speeds. They can be sensed if they come into contact with the range sensors. The goal of the agent is to catch both balls. The initial position of the balls can vary within the range of the agent's sensors, with two constraints: first, they are to be located on the two different sides (left and right) of the agent's initial position and they must also be horizontally separated far enough to meet the memory requirement of the task (if the balls are too close, they will remain within the sensor range at all times). Memory is necessary to be successful in this task, as can be seen in Fig. 1.

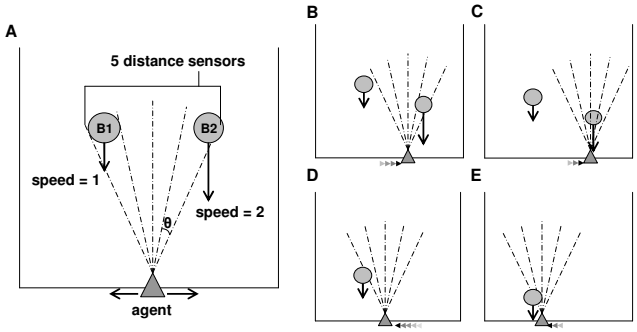


Fig. 1. Ball Catching Task. The task illustrates a scenario where memory may be needed. A. The agent is initially placed at the horizontal center of the environment. Two balls are falling at different speeds (the right ball is faster in this example). B. When the agent moves to catch the faster ball, the slower one goes out of the agent's view. C. The agent catches the first ball. D. The agent must move back to the slower ball without any input from the range sensors (this will require memory of where the other ball was when the agent lost contact of it). E. The agent catches the second ball.

A. Methods

Three agents, each controlled by a different type of neural network, were used in the experiment: (1) feedforward network, (2) recurrent network, and (3) feedforward network with external marker dropper/detector ("dropper network").

1) Feedforward Network: The first controller is made of a simple fully connected feedforward neural network containing three layers - input, hidden, and output (Fig. 2). Five inputs of the input layer were from five sensors. Sensor inputs were inversely proportional to the distance between the agent and the ball detected by the sensor:

$$I_i = 1 - Dist_i/Length_i \tag{1}$$

where $Length_i$ denotes the length of the range sensors which was set to 200 for all 5 sensors and $Dist_i$ is the distance between the agent and the ball detected by the i-th sensor. $Dist_i$ was 200 when no ball is detected, which makes $I_i = 0$. The input values are propagated to the hidden layer and then to the output layer. (Note that bias units were not used [in all three agents] to avoid default behavior that can artificially crank up the performance.) Activation values of the two outputs (O_1 and O_2 in Fig. 2) decide the movement of the agent (the horizontal location in the next time step, Loc). The agent takes a unit-sized step to its left if $O_1 > O_2$, to the right if $O_1 < O_2$, and stays if $O_1 = O_2$. The speed of the agent is fixed to a unit distance per time step. The equations for this baseline agent take the following standard form (see e.g., [27]):

$$\begin{aligned} H_j &= \sigma\left(\sum_{i=1}^{N_{\text{in}}} v_{ji} I_i\right) \qquad j = 1, ..., N_{\text{hid}} \\ O_k &= \sigma\left(\sum_{j=1}^{N_{\text{hid}}} w_{kj} H_j\right) \qquad k = 1, ..., N_{\text{out}} \\ Loc(t+1) &= \begin{cases} Loc(t) - 1 & O_1(t) > O_2(t) \\ Loc(t) + 1 & O_1(t) < O_2(t) \\ Loc(t) & O_1(t) = O_2(t) \end{cases} \end{aligned} \tag{2}$$

where I_i, H_j and O_k are the activations of the i-th input, j-th hidden, and k-th output neurons; v_{ji} the input-to-hidden weights and w_{kj} the hidden-to-output weights; $\sigma(\cdot)$ the sigmoid activation function; and N_{in}, N_{hid}, and N_{out} are the number of input, hidden, and output neurons whose values were 5, 3, and 2 respectively. The unit-distance of the movement of the agent was set to 1.

This agent is expected to fail in the given task because it cannot remember the existence of the slower ball at the time it catches the faster one. Right after catching the faster ball, the agent receives no input to drive further movement (Fig. 1C). The purpose of showing this agent is just to provide the basic structure for the other agents and establish baseline performance.

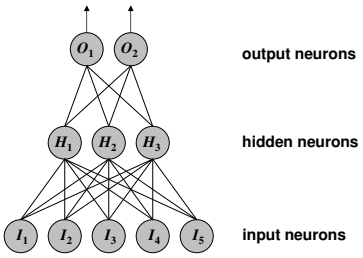


Fig. 2. Baseline Feedforward Agent.

2) Recurrent Network: A memory mechanism is built into the second agent by recurrent connections feeding previous hidden states back to the hidden layer (Fig. 3). This type of recurrent neural network known as Elman Tower is an extension of the basic Elman network, which is one of the most commonly referenced recurrent networks [2]. Also, other researchers used recurrent networks to model the hippocampus (see e.g. [28]). Because it is one of the simplest recurrent networks, the Elman network is a good candidate for an immediate descendent of feedforward networks, as a possible first step in evolution from a memoryless to a memory-capable system. Usually the more the number of hidden state feedbacks from the past, the more powerful the network is. The number of hidden state feedbacks was either 3 or 7, depending on the experiment. Recurrent connections can be defined as in the equation below:

$$H_j(t) = \sigma\left(\sum_{i=1}^{N_{\text{in}}} v_{ji} I_i(t) + \sum_{m=1}^{N_{\text{mem}}} \sum_{l=1}^{N_{\text{hid}}} \lambda^m u_{jl}(m) H_l(t-m)\right) \quad j = 1, ..., N_{\text{hid}} \tag{3}$$

where $u_{jl}(m)$ is the recurrent connection weight from the m-th previous hidden state vector $H_l(t-m)$. A constant decay rate ($\lambda = 0.7$) was used to penalize the effect of older state vectors, that is, $H(t-m) = \lambda H(t-(m-1))$. All other terms were identical to those explained above.

3) Dropper Network (Feedforward Network with Dropper/Detector): Finally, the last one, the one of our interest,

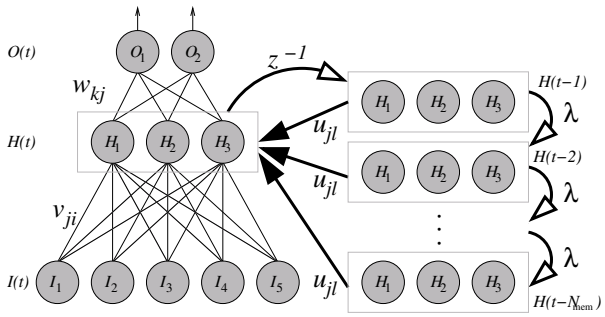


Fig. 3. Memory-Equipped Recurrent Agent. N_{mem} previous hidden state vectors are fed back to the hidden layer (filled arrows). The open arrows indicate a 1-step delayed copying operation (λ indicates the decay rate). Hidden-to-output connection is the same as that of the baseline agent. See the text for the definition of other terms.

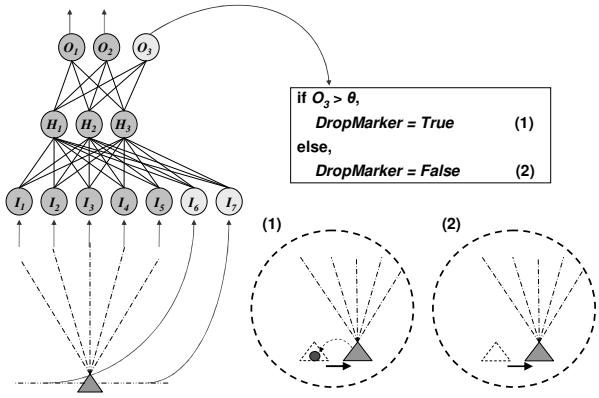
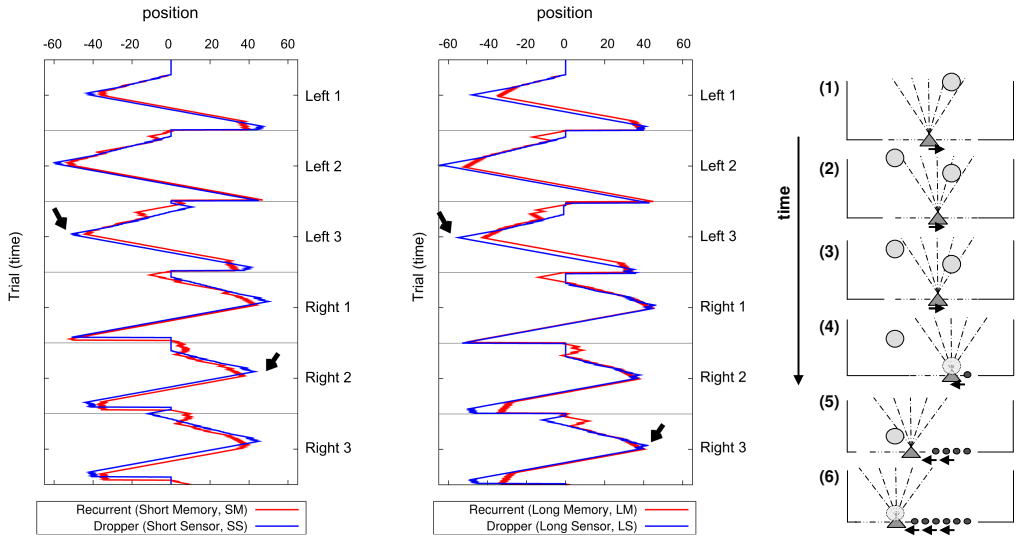


Fig. 4. Feedforward Agent with External Marker Dropper/Detector ("Dropper Network"). The architecture is identical to the feedforward agent, with the difference being the two additional inputs to detect the external markers (I_6:left, I_7:right) , and an added output for dropping markers (O_3).

is the agent using external markers (imagine leaving a bread-crumb trail). The underlying neural network is a feedforward network, identical to the baseline agent. The only difference is that it has two more inputs and it has one more output (Fig. 4). The additional inputs are from supplementary sensors which can detect external markers (I_6 [marker detected on the left] and I_7 [marker detected on the right] in Fig. 4). The additional output (O_3) is for the motor action to drop a marker in the environment. Because the primitive animal this agent is modeling already has external sensors, there is no evolutionary overhead to have additional mechanisms like these. That is, primitive animals may use already existing sensors to sense external markers in the environment. Throwing markers is a mere modeling of a behavior they may already have, e.g., excretion. Below is how this additional output works:

$$DropMarker = \begin{cases} \text{True} & \text{if } O_3 > \theta \\ \text{False} & \text{otherwise} \end{cases} \tag{4}$$

If the value of O_3 is greater than the threshold θ, the agent drops a marker in its current position before it moves to the next position. This threshold is not fixed but is also learned through genetic search. Note that this whole scheme could be seen as just another round-about way of adding recurrence,



A. Short Capacity/Range B. Long Capacity/Range C. Dropper Strategy

Fig. 5. Agent Trajectories. A&B. The trajectories of the recurrent (red curve) and the dropper agents (blue curve) are shown for different capacity/range (A: high/long, B: low/short). In each plot, six trials are shown, with three "fast left ball" (marked Left 1 to 3) and three "fast right ball" conditions (marked Right 1 to 3). Each trial shows 200 time steps. Agents using external markers (SS & LS) always slightly overshoot (few examples marked by $\rightarrow$) the horizontal position of the first ball they catch (blue curves). C. A sketch is shown of a typical evolved strategy observed in dropper agents (SS) in the faster right ball case. (1) Agent detects the faster ball first. (2) Agent detects both balls while moving toward the faster one. (3) The slower ball goes out of the agent's range sensor's scope. (4) Agent overshoots the ball and starts to drop markers. (5) The markers repel the agent away until the slower ball is detected by the sensor. (6) The slower ball comes within the range sensor's scope and the agent catches it.

however it is a simulation of an *indirect* external loop and thus is different from direct, internal, recurrent neural circuits. The recurrence created by the dropper network is *indirect* because it does not use the dynamics of the network, but uses sensory-motor interaction, and is thus *second-hand*. Moreover, while it is true that I_6 or I_7 activates as O_3 goes on, O_3's on/off condition is relative to an evolved threshold value, and the dropping event also specifies the location of the marker in the environment (this latter aspect is more prominent in the foraging task). Therefore, this is *indirect*, different from the *direct* neuronal relay in a recurrent network architecture.

B. Experiments and Results

The learning of connection weights of the agents is achieved through genetic search (neuroevolution). The fitness for an agent is set inversely proportional to the sum of the horizontal separations between itself and each ball when the balls hit the ground. The tasks were given to each agent 12 times (24 balls), where the ball to the right of the agent is falling faster in the first 6 tasks and vice versa for the 6 latter tasks. Fitness values of the 12 tasks are added to form the overall fitness for the agent. Best performing agents of the population in the current generation survive to the next generation. One-point crossover with probability 0.9 and mutation with rate 0.04 was applied to these best-performing agents to modify the connection weights of the current population. The halting criterion of the evolution was when any single agent in the current population is successful in catching more than 23 balls out of 24 (about 96% success rate). When the evolution step reaches a preset maximum, a fresh new agent pool was created by randomizing their weights and a new evolution process initiated.

In addition to the initial comparison of the performance between the three agent types, we wanted to know the effect of the capacity of the external marker detectors on the performance of the external-marker-using agent. Therefore, two external-marker-using agents with different marker detector range (one with the same length as the distance sensors, the other with 1/4 the length) have been tested. For fair comparison, we also varied the capacity of the memory-equipped agent (one with the memory order N_{mem} [the number of hidden state vector feedbacks] set to 3, and the other to 7).

As expected, the baseline agent in Fig. 2 could never succeed in the task. It could catch only one of the balls (i.e., success rate was 50% at best). The agent stops upon catching the first ball because it does not receive any further input at that point (Fig. 1C). Because this agent was tested only to emphasize the memory requirement of this delayed-memory task, we will not show the detailed result here. The following results are from 4 groups of agents (2 network types $\times$ 2 capacity differences). Table I summarizes the 4 groups.

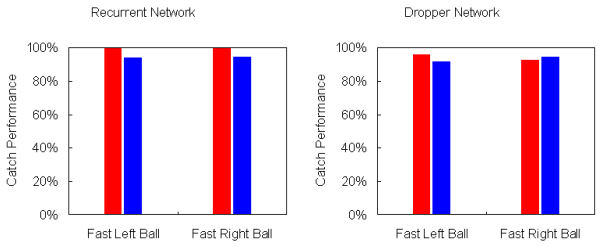
TABLE I
EXPERIMENT DESCRIPTION.

Agent	Mechanism	Capacity/Range
SM (Short Memory)	Recurrent Network	Low
LM (Long Memory)	Recurrent Network	High
SS (Short Sensor)	Dropper Network	Low
LS (Long Sensor)	Dropper Network	High

Fig. 5 shows the agents' trajectories in the four experiments.

In the figure, the y-axis represents time and the x-axis relative horizontal location (0 marks the initial position of the agent, and - and + the right and the left of that position). The balls on the left falls faster in the first 6 tests and the balls on the right falls faster in the 6 remaining tests. All the agents in the 4 different groups were successful in catching all 12 pairs of balls.

We compared the performance by running 200 pairs of the balls (100 in each faster-left and faster-right ball cases) for 5 agents in each experiment. Results show that all four types of agents were successful in solving the task with a success rate of at least 91.5%. Fig. 6 summarizes the results. In the figure, the average performance of the agents under faster left ball and faster right ball conditions are shown. Recurrent network and dropper network both show above 90% performance in all cases (both fast-left and fast-right conditions, and for both high/long [red bars] and low/short capacities [blue bars]).



A. Recurrent (LM & SM) B. Dropper (LS & SS)

Fig. 6. Average Performance of the Three Agent Types. The recurrent and the dropper networks show above 90% performance (high capacity/range [red], low capacity/range [blue]).

Between-group performance comparison reveals the recurrent network's slight performance superiority over the dropper network (Fig. 7). Nevertheless, what is important here is that the dropper networks show much higher performance than the baseline feedforward networks. Feedforward networks only showed near 50% performance when both fast-left and fast-right conditions are averaged: only catch left ball, catch left and right ball with equal probability, or only catch right ball (data not shown).

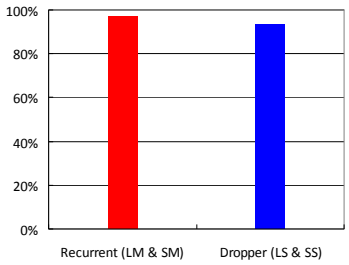
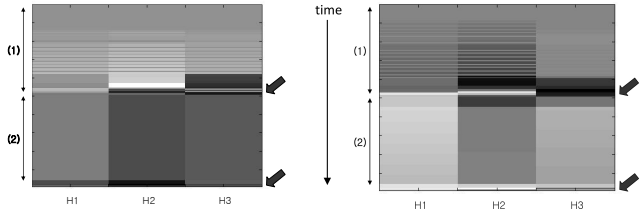


Fig. 7. Performance Comparison between Recurrent and Dropper Networks. Average catch performances are plotted. Recurrent networks (M=0.969) show a catch performance greater than dropper networks (M=0.935). The difference between two networks is significant (p<.001, n=4000).

How did the feedforward agents using external markers become almost as successful as the recurrent agents? To answer this question, we analyzed the behavior of the feedforward agent to see how they use external markers as a memory-aid. The strategy they used is illustrated in Fig. 5C. As described in the figure, feedforward agents overshoot the position of the faster-falling balls and then begin to throw external markers. This overshooting patterns of the agents using external markers are clearly observed in their trajectories in Fig. 5A and B. When there is no input from the distance sensors after they catch the ball, they use the external markers in an aversive manner to track back away from the markers until the slower-ball gets detected by their distance sensors.

To see how the use of external markers is related to memory, we looked at the hidden state activations of the agents. Fig. 8 shows example (LS) of the hidden state activations of the dropper networks throughout the task. We can discriminate the hidden states in "ball-driven" movements from those in "marker-driven" movements. However, the distinction between faster-left-ball and faster-right-ball cases were ambiguous. We further compared the hidden state activations for all four cases (SM, LM, SS, and LS) when the agents are moving back to the second ball after catching the first one because those are the times memory is being used for the recurrent network agents. The hidden state activations in this phase are plotted in Fig. 9.



A. Faster Left Ball B. Faster Right Ball

Fig. 8. Example of Hidden States Activation in LS Experiment (see Table I). The hidden states for "ball-driven" movements (A-(1) and B-(1)) and for "marker-driven" movements (A-(2) and B-(2)) are distinctive. Ball-catching moments are marked by $\rightarrow$.

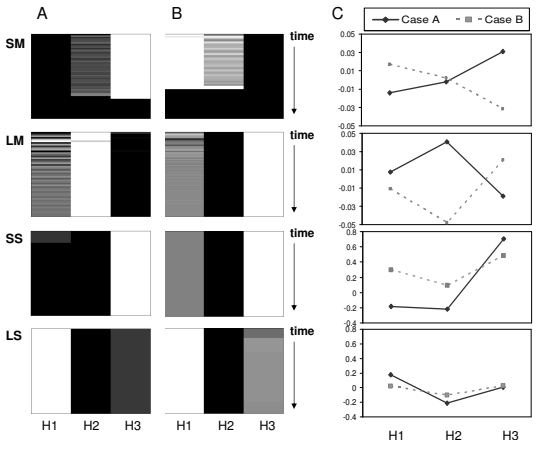


Fig. 9. Hidden State After First Ball is Caught. Case A: faster left ball. Case B: faster right ball. SM, LM, SS, and LS mean Short Memory, Long Memory, Short Sensor, Long Sensor, respectively (Table I). H1, H2, and H3 show the hidden layer activation level.

For the memory-equipped agents using recurrent networks, the difference in the activation patterns of hidden neurons between Case A (faster left ball) and Case B (faster right

ball) is evident. For example, in the recurrent network agents (Fig. 9, rows LM and SM), the hidden state activation patterns in Case A are near inverse of those in Case B (compare the average activation plot to the right). This hidden unit representation is the internal state of the agent (see e.g. [2]), and since they show distinct states, we can say that the recurrent agents indeed remember the location of the ball (left or right) distinctively. However for the dropper network agents (rows LS and SS), the activation patterns are similar in both Case A and B. This is because the distinction is already presented by the location of the self-dropped markers in the environment (markers on the left-side of the agent in Case A and the right-side of the agent in Case B) and does not need to be represented internally. Thus, the spatial information required to solve the task is located in two different places– inside the brain for the recurrent network or in the environment in the form of external markers for the dropper network.

III. Task II: Foraging Food

To test if the dropper network's performance can be generalized beyond a simple memory task, we extended the task domain to a 2D map. As the task environment becomes more complex, the agent must be able to express richer context. This task was a biologically plausible food-foraging task (Fig. 10). Same as with the ball-catching task, we tested recurrent and the dropper networks and compared their performance. We left out the feedforward network since its performance only established the baseline in the previous section.

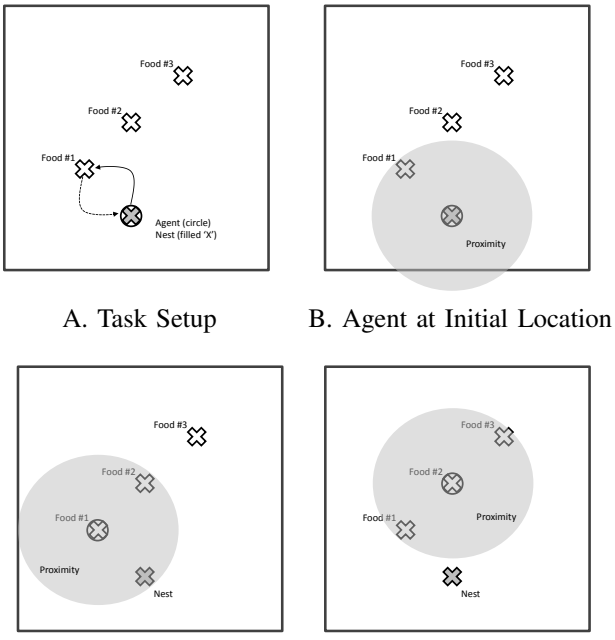


Fig. 10. Food Foraging Task Definition. A. Agent's initial location (gray X denotes nest location and blank X's denote food locations). B-D. Shaded area denotes the agent's sensor range. Agent's location is indicated by a circle.

Three food items were located in an environment of size 300×300, and the agent starts from the nest location and explores for food. The goal is to consume all three food items in the environment. When a food item is found, the agent needs to find its way back to the nest to consume the food item (The agent has limited food delivery capacity that it can carry only one food at a time). This type of behavior is often observed in animals because they feed the young, cache food for later use, or avoid competition by not consuming the food on-site [29], [30], [31]. The agent has sensors with a limited range of 40, whose inputs are calculated in the same manner as Equation 1. The memory requirement of the task is imposed as follows. From the nest, the agent can only detect food#1 (Fig. 10B). When it moves close to food#1, then food#2 comes within its sight (Fig. 10C). However, it needs to come back to the nest to consume food#1 and this makes food#2 invisible again. Therefore, the agent needs to memorize where the next food item was (in this example, food#2). The same applies to food#3 (Fig. 10D) and the agent has to move back and forth between the food item and the nest at least three times. Moreover, the nest does not generate any sensory cue at all, thus the nest location also needs to be remembered by default. The agent has a limited life span which increases only when it successfully consumes a food item.

A. Methods

The agent model used in this task is illustrated in Fig. 11. The agent interacts with the environment with 8 pairs of sensors distributed uniformly around the center at an interval of $45°$, 8 of which are receptors sensitive to the external markers the agent drops. The other 8 are distance sensors sensitive to food items. Both types of sensors are limited in range subject to the constraint mentioned in the foraging task description, and the magnitude of the sensor signal is inversely proportional to the distance to the detected object.

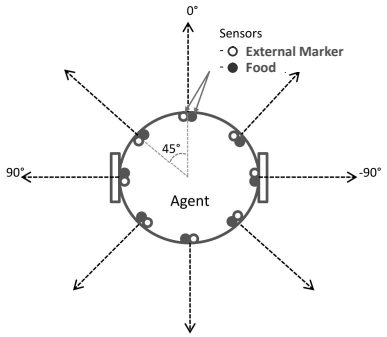


Fig. 11. Agent Model. Simple agent with directional/ranged sensors for food and external markers.

Recurrent and dropper networks used in this task have similar topologies as described in the ball-catching task, with minor differences in the number of input and output units. Three network outputs (O_1, O_2, and O_3) indicate x-, y-, and orientation offsets, which determine the next location and orientation of the agents:

$$\theta(t+1) = \theta(t) + 180° \times O_3(t) \quad (5)$$

$$x(t+1) = x(t) + Speed \times O_1(t) \times \cos(\theta(t+1)) + Speed \times O_2(t) \times \cos(\theta(t+1) + 90^\circ) \quad (6)$$

$$y(t+1) = y(t) + Speed \times O_1(t) \times \sin(\theta(t+1)) + Speed \times O_2(t) \times \sin(\theta(t+1) + 90^\circ) \quad (7)$$

where $\theta(t)$, $x(t)$, and $y(t)$ are the orientation and xy location of the agent at time t (Fig. 12). $Speed$, the speed of the agent, was set to 3. The agent can freely navigate within the environment while bouncing off when it comes into contact with the wall (Fig. 13). As in the previous task, default movements are blocked by removing the bias units so that the agent does not move without sensory input. This is intended to maximize the memory requirement of the task. Also, one more output unit is added to a typical feedforward network to allow marker dropping behavior (Fig. 14). To analyze the effect of 'forgetfulness' of the memory, three different memory decay rates were tested for both the recurrent and the dropper networks. Table II summarizes the experimental setup. We tried three different memory decay rates of the recurrent network (λ=1.0, 0.99, and 0.7). For each of them, three previous hidden state vector sizes (N_{mem}=5, 10, and 20, see Equation 3 and Fig. 3) were tested to identify their effects on the performance. The evaporation rate (ρ) of the chemical markers used by the dropper network was also varied as the memory decay rate of the recurrent networks: $M(t) = \rho M(t-1)$, where ρ=1.0, 0.99, and 0.7, and $M(t)$ is the marker strength at time t. Volatility is a natural property of chemicals analogous to the recency effect (forgetfulness) of internal memory. If so, using this property can increase performance by prioritizing newer events, as the recency effect does [32]. The same genetic algorithm discussed in the above section was used to train the networks.

TABLE II
EXPERIMENT DESCRIPTION.

Memory Decay	Agent		
	Recurrent		Dropper
0% loss per step	λ=1.0	N_{mem}=5 N_{mem}=10 N_{mem}=20	ρ=1.0
1% loss per step	λ=0.99	N_{mem}=5 N_{mem}=10 N_{mem}=20	ρ=0.99
30% loss per step	λ=0.7	N_{mem}=5 N_{mem}=10 N_{mem}=20	ρ=0.7

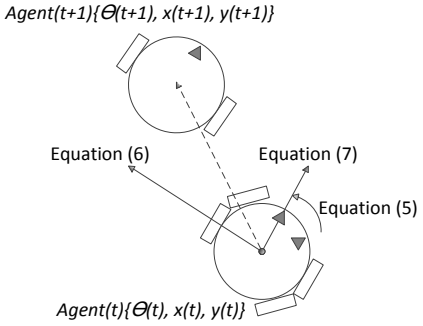


Fig. 12. Agent Movement. $Agent(t)$ denotes the agent's orientation and location at time t. It turns and moves according to Equation(5), (6), and (7).

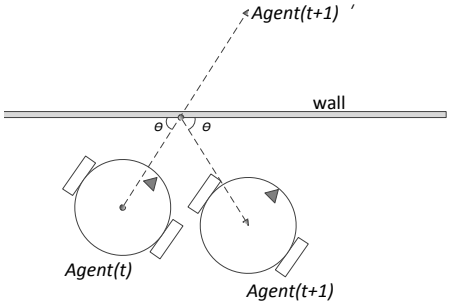


Fig. 13. Agent Bounces off the Wall. $Agent(t+1)'$ denotes the agent's next location calculated using Equation (5), (6), and (7). As the agent contacts a wall on its way to $Agent(t+1)'$, it bounces off the wall and the next location is changed to $Agent(t+1)$.

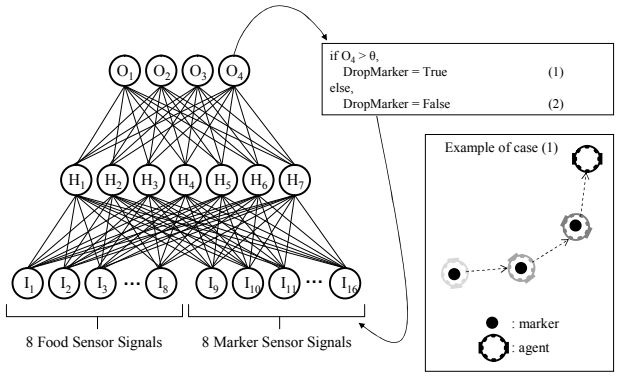
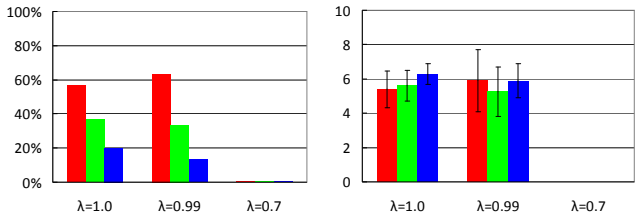


Fig. 14. Neural Network of the Dropper Agent. An output unit (O_4) is added to the typical feedforward network to allow external marker dropping behavior. The box shows an example of leaving external markers when the agent moves to a next location. The recurrent network counterpart had a similar structure as Fig. 3, with no external marker sensors.

B. Experiments and Results

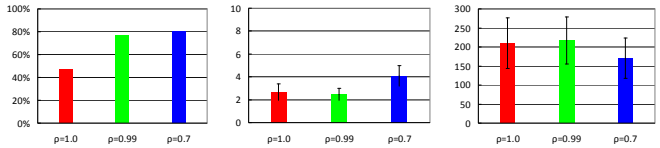
A total of 30 evolutionary trials were given to each type of network controller. A trial is recorded as successful if the agent consumes all three food items in the environment. Fig. 15 and Fig. 16 show the quantitative results of the experiment. It is clear that the growing number of hidden state feedback degrades the performance of the recurrent networks (Fig. 15A). This is because of the additional parameters that the recurrent agents have to tune. For example, the recurrent agent with 5 hidden state feedbacks has 8×7 (input to hidden weights)+5×7×7 (hidden feedback weights)+7×3 (hidden to output weights)=322 parameters while the dropper agent has only 16×7 (input to hidden weights)+7×4 (hidden to output weights)+1 (threshold for O_4)=141 parameters. When the number of hidden state feedback is 20 for the recurrent agent, the number of parameters grows to 1057. Comparing the success rates between λ=1.0 and λ=0.99, small memory decay did not seem to have a significant impact for the

performances of the recurrent networks. Only the success rate of the one with 5 hidden states feedbacks slightly grew as λ changes from 1.0 to 0.99. Inverse correlation is observed between the success rate and the travel distance when $\lambda = 1.0$. However, such correlation is not observed when $\lambda = 0.7$ (Fig. 15B). When the memory decay was 30% per step, no recurrent network was successful. On the other hand, the performances of the dropper networks grew once we let the markers evaporate (Fig. 16). The dropper networks show fairly high (80%) success rate even with high marker evaporation rate (ρ=0.7). No agent with random network weights in either network type (recurrent and dropper) was found successful (data not shown). For the traveled distances, the recurrent networks moved longer distances than the dropper networks, regardless of the different memory decay rate (Fig. 15B and Fig. 16B). When the evaporation rate changed from ρ=0.99 to ρ=0.7, the travel distance of the dropper network increased and fewer markers were dropped. Further qualitative analysis elucidates the quantitative differences.



A. Success Rate B. Distance Traveled

Fig. 15. Quantitative Data of the Successful Recurrent Networks. Colors indicate different sizes of the hidden state feedback (red: N_{mem}=5; green: N_{mem}=10; blue: N_{mem}=20, see Equation 3 and Fig. 3). A. Performance degrades as the number of hidden states feedback grows. B. Distance metric is a ratio of the distance traveled and the minimum distance needed to reach a goal. Overall, the succesful recurrent networks traveled long distances (min avg.=5.249 with λ=0.99, N_{mem}=10; max avg.=6.277 with λ=1.0, N_{mem}=20).



A. Success Rate B. Distance Traveled C. Marker Drops

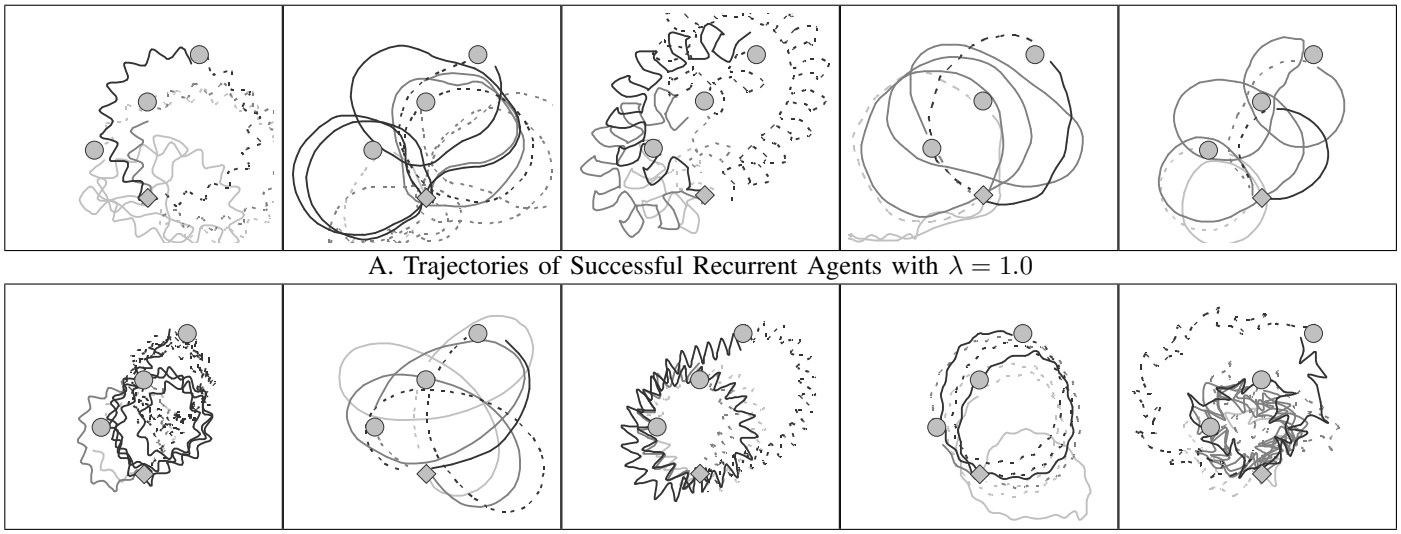
Fig. 16. Quantitative Data of Successful Dropper Agents. A. Large difference in success rate is observed once markers evaporate (14/30 for ρ=1.0; 23/30 for ρ=0.99; 24/30 for ρ=0.7). B and C. Agent moved significantly longer distance and dropped less number of chemical markers when evaporation rate was the highest (ρ=0.7). For the distance metric used in B, see Fig. 15B.

Fig. 17 plots the traces of successful recurrent agents with two different memory decay rates (λ=1.0 and 0.99). The agents do not have explicit knowledge about the nest location, because the nest does not radiate any information and the agents do not have a mechanism to change their states based on whether they visited the nest or not. Moreover, the hidden state feedbacks affect the movements of the recurrent agents, regardless of their current locations. Therefore, it is difficult for the recurrent agents to change their states to turn abruptly at the nest location. Under this condition, the fittest strategy found by artificial evolution for the recurrent agents was a repeated circular movement. Even though this strategy enabled the agent to be successful in the task, the qualities of the solutions were poor, as can be seen from the behavioral trajectories shown in Fig. 17. As we can see in this plot, the agents seemed to blindly scan through the task arena repeatedly drawing circles. The solutions they found almost seemed to be out of pure luck, since the trajectories do not show any goal-directedness. As a result, the orders of the food items found by this strategy were often different from the order that the food items were presented (1-2-3). For example, the order that the food items were found in the 4th panel of Fig. 17A and in the 2nd panel of Fig. 17B is 2-1-3.

Fig. 18 plots the traces of the dropper agents with three different marker evaporation rates. For the dropper agents, this task is not too difficult because the markers explicitly contain spatial information. For example, after foraging a food item, the dropper agents are guaranteed to get to the nest just by following the marker trail. Also, the memory source is local and thus dependent on the current location of the agents because they can sense only the markers within their sensor radius. Regardless of their current location and orientation, the dropper agents only need to evolve their connection weights between their sensors and actuators (input and output neurons) to generate movement so that they can maintain the distance to keep some of the markers still visible at the next location. As a result, only few repeating circular movements are observed, and the dropper agents usually came back to the nest with a food item once they were out for foraging. Compared to Fig. 17, the trajectories of the dropper agents are more goal-directed and this is why the average travel distance of the dropper agents is much shorter than that of the recurrent agents. When ρ=0.7, the markers evaporate too fast and the dropper agents can use the most recent markers only. The limited marker information made the dropper agents to generate wiggling trajectories, thus requiring longer travel distance. To fully account for the effect of evaporating markers, we performed a detailed comparison between the best dropper agents with ρ=1.0 and ρ=0.99. These two ρ values were selected because their qualitative results showed the most difference. Dropper agents with the shortest step sizes were selected as being the best because they have the most economical strategy in solving the same task.

One-to-one comparisons of the strategy and the trajectory between the best-performing agents in each marker evaporation rate show divergence in the evolved behaviors. Fig. 19 confirmed more compact trajectory developed by agents using evaporating markers (A-(1) vs. B-(1)). B-(2) and B-(3) show the evaporation of chemical markers in fading color. As the chemicals evaporate, their sensitivity fades and the attraction toward the non-existing (already-consumed) food becomes weaker and less competitive than the attraction to the remaining food. Because of the difference in the volatile characteristic of markers, the two groups developed different strategies for the task. The strategies are summerized below:

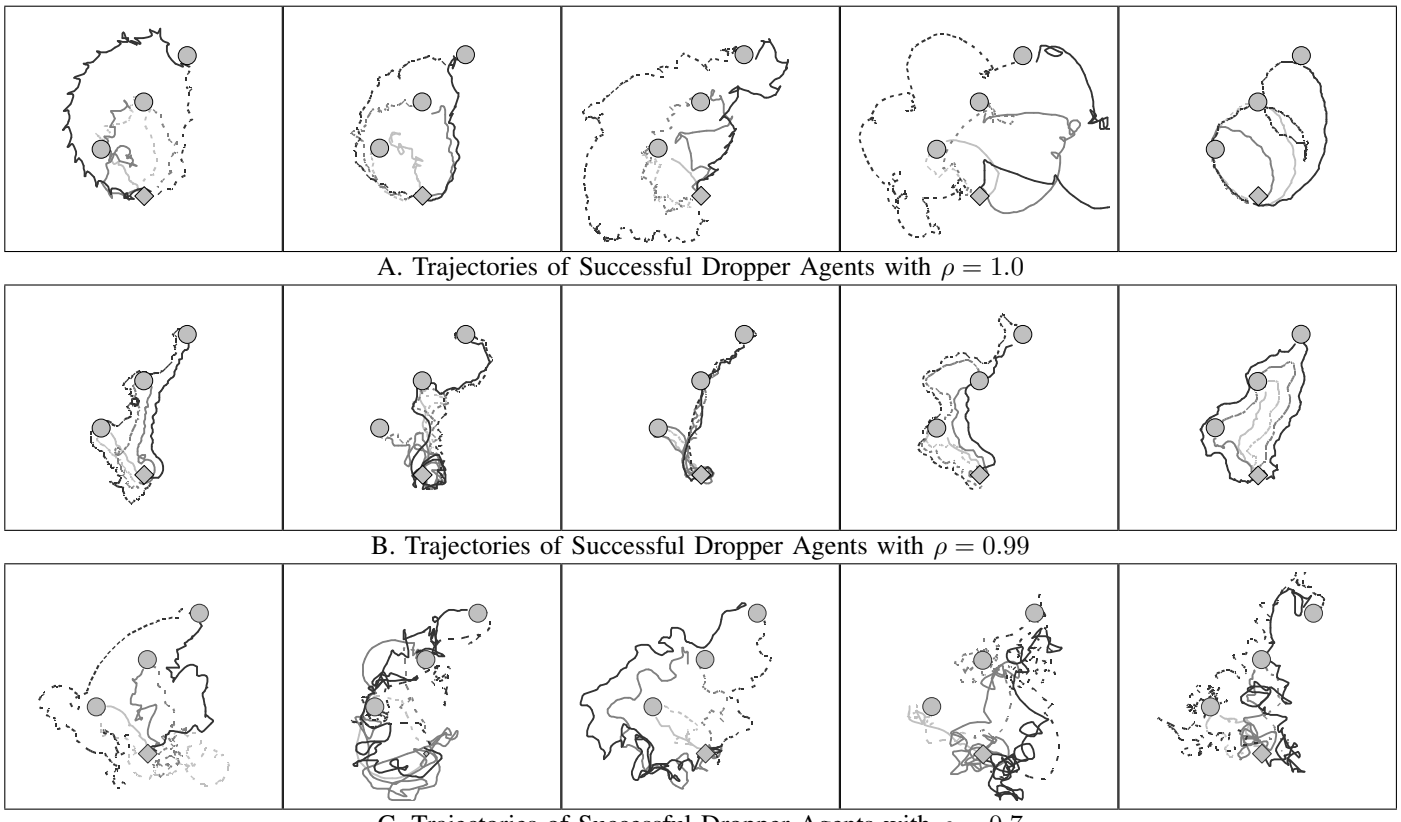
- Dropper networks using non-evaporating markers (ρ=1.0)
  1) Approach food item from right to left (Fig. 19, A-



A. Trajectories of Successful Recurrent Agents with $\lambda = 1.0$

B. Trajectories of Successful Recurrent Agents with $\lambda = 0.99$

Fig. 17. Trajectories of Successful Recurrent Agents. A and B. Agents are moving in circles to get to the food items and the nest. In general, the trajectories do not show any goal-directedness (move directly toward food or toward nest). The gray diamonds at the bottom of each panel indicate the nest and the gray circles denote food items. The grayscale intensity of the line denotes the temporal order of each trajectory (the darker, the more recent). The line type (solid/dashed) distinguishes between the nest-to-food/food-to-nest trajectories.



A. Trajectories of Successful Dropper Agents with $\rho = 1.0$

B. Trajectories of Successful Dropper Agents with $\rho = 0.99$

C. Trajectories of Successful Dropper Agents with $\rho = 0.7$

Fig. 18. Trajectories of the Successful Dropper Agents. A. When the markers do not evaporate (ρ =1.0), the dropper networks move in circles with growing radius. B. The dropper agents with ρ =0.99 generate very compact trajectories. C. The dropper agents show 'wiggling' trajectories with greater memory decay (ρ =0.7). See Fig. 17 for plotting conventions.

(1)).

2) Follow markers laid on its left side (Fig. 19, A-(1)).
3) If food item is detected,
   a) If marker is detected rather far, follow the food item while throwing markers (Fig. 19, A-(2)).
   b) If marker is detected very near, follow the marker without throwing another one (Fig. 19, A-(3)).

- Dropper networks using evaporating markers (ρ=0.99)
  1) Always throw markers (Fig. 19, B-(1)).
  2) If food item is detected,
     a) If marker signal is week (far or old), follow the

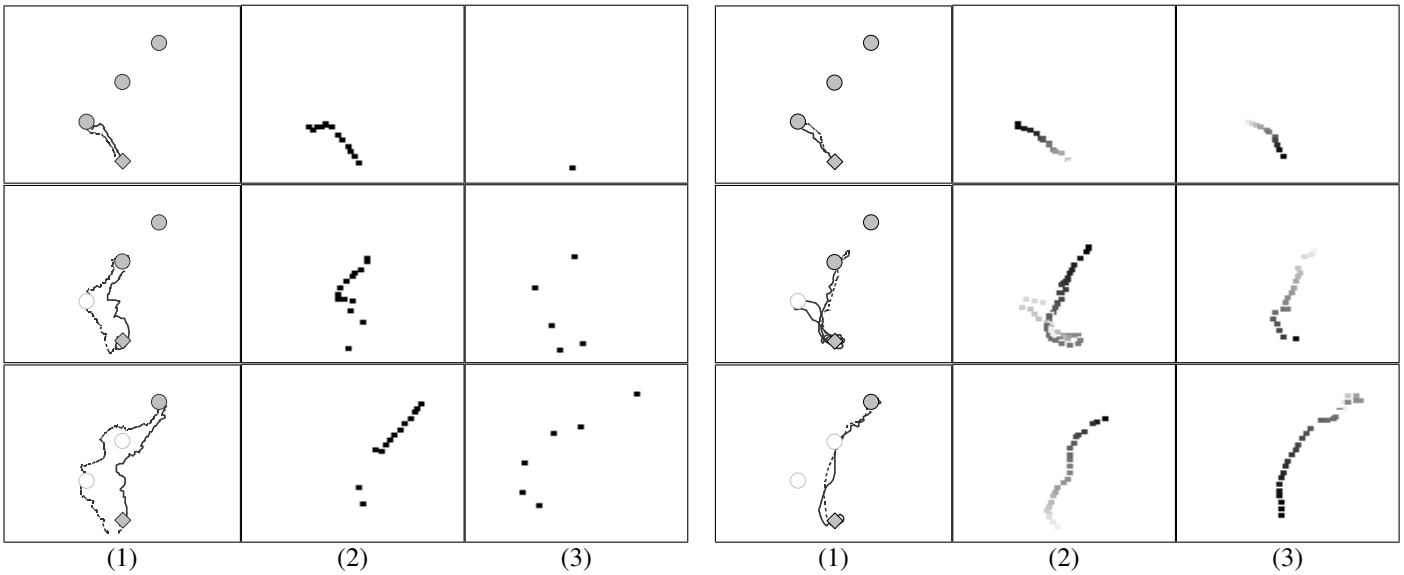


Fig. 19. Agent trajectory and chemical trails (solid: nest-to-food, dashed: food-to-nest trajectory). Agents using evaporating chemical markers developed simpler set of rules.

food item (Fig. 19, B-(2)).

b) If marker signal is strong (close and recent), follow the marker (Fig. 19, B-(3)).

The strategies laid out above show that agents using non-evaporating markers (ρ=1.0) evolved additional rules to drop/detect markers in a particular direction (left side in this example), whereas the agents using evaporative markers has simpler drop/detect rule. This is due to the evaporating property of the markers. Because the evaporative markers contain temporal information, it does not have to develop additional rules to avoid conflicts among the markers. As the markers evaporate, their sensitivity fades and the attraction toward the non-existing (already-consumed) food item becomes weaker than the attraction to the remaining food item. However, because the non-evaporating markers cannot encode the passage of time, a tie-breaking function for conflicting markers had to be developed by the agent. To summarize, utilizing self-generated olfactory-like cues enables the agent to exploit the volatility of markers, a natural property of chemical markers. As a result, the dropper agent can distribute, in the environment, information necessary to solve a task, and its interaction with the environment enables it to communicate with itself in the past. This kind of *stigmergy* lowers the complexity of the solution (see [33] for a different kind of stigmergy utilizing altered behavior in other agents in the environment).

We again took a look at the dropper agent's hidden state activations. Fig. 20 shows the hidden state activations of the best agent using evaporative markers. We can observe significantly more fluctuations in the hidden states activations when the agent is generating "food-driven" movement (red-marked area). We measured the degree of fluctuation using a simple metric:

$$Fluctuation = \sum_{i=2}^{N_t} \sum_{j=1}^{N_{\text{hid}}} \text{abs}(H_{i,j} - H_{i-1,j})/N_t \quad (8)$$

where $H_{i,j}$ is the activation value of j^{th} hidden neuron at time i, N_t is the maximum step size, and N_{hid} is the number of hidden neurons. $\text{abs}(\cdot)$ is the absolute value function. The difference in the fluctuation amount in "food-driven" and "marker-driven" movements is shown in Fig. 21. To see this difference from a different perspective, Fig. 20C-(2) was replotted as a line chart (Fig. 21B). Even though it is hard to tell the details of the detected markers, the agent's internal states can show visible distinction during different behaviors ("food-driven" vs. "marker-driven"), which is similar to the previous result in the ball-catching task.

IV. Discussion

The main novelty of our work is to have shown that reactive feedforward neural network can exhibit behavior requiring memory when coupled with a simple external material interaction mechanism. The results have a significant implication on how we should begin to understand the evolution of internalized memory. It is surprising that the feedforward network can decide when to drop the external marker, and what to do when a marker is detected, thus using the environment as an open canvas.

There are several existing works that share key mechanisms presented in our work, but there are important differences. Below, we will review three prominent approaches, (1) epistemic structures, (2) behavior-based robotics, and (3) ant colony optimization (and the use of artificial pheromones in general), and discuss how our work provides unique contributions.

The most notable is the work by Chandrasekharan and Stewart [34], [21] where "epistemic structures" (similar to pheromones) are dynamically deposited in the environment by

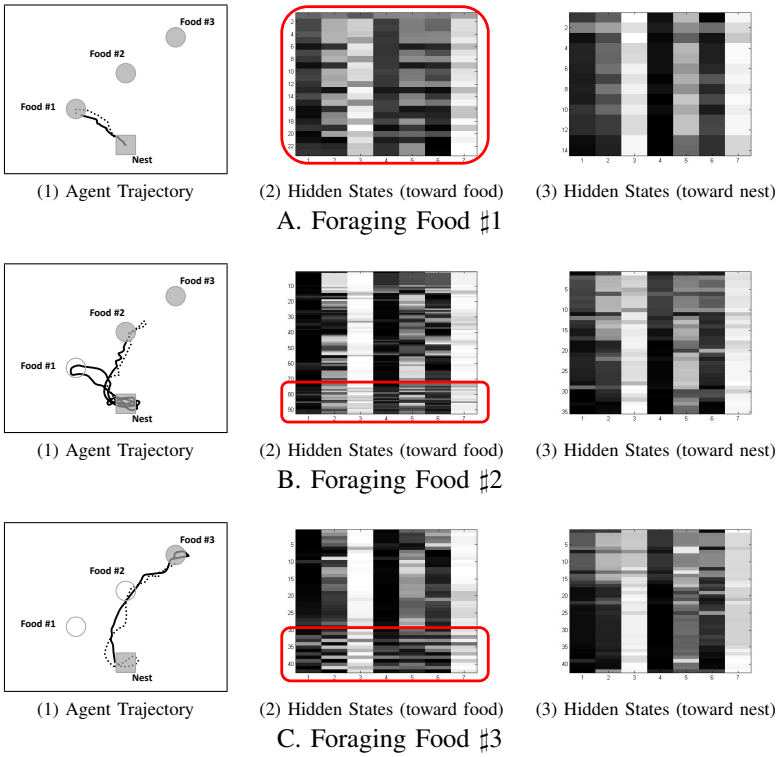


Fig. 20. Hidden State Activations. (1) Same as Fig. 19. (2) Areas bounded by the red box denote the hidden state activation during "food-driven" movements, where hidden states fluctuate more vigorously.

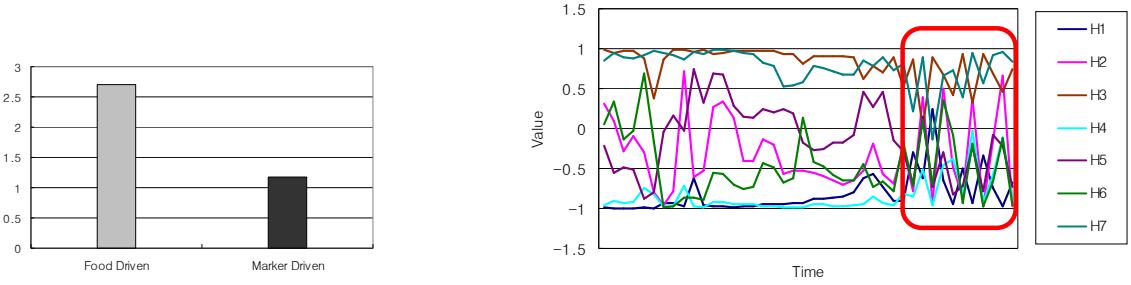


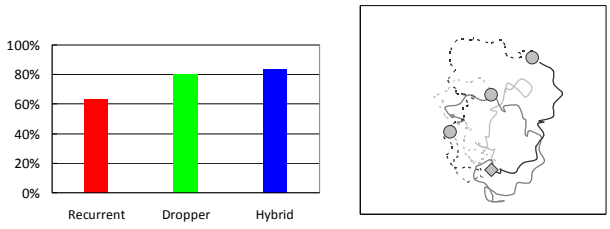
Fig. 21. Difference in Hidden State Activation. A. Amount of fluctuation is calculated according to the equation 4. B. Line chart of Fig. 20C-(2). Area marked in red denotes the same area in Fig. 20C-(2).

agents, controlled by genetic algorithms or by Q-learning. In this case, the goal was to reduce "tiredness" (cognitive load), so an explicit link to memory was not made. Furthermore, the task itself only contained one nest and one food source, thus memory capacity was not directly tested (see [35] for an extension of this to multiple food sources). Despite these differences, our work and that of Chandrasekharan and Stewart share the important concept that external markers can contribute to cognitive function even in reactive agents.

One tenet of behavior-based robotics is that instead of internalized representation, the environment itself can be used as a representation [36]. We share this view, but we go one step further. In behavior-based robotics, the representation in the environment is rarely altered, whereas in our work, the agent, although reactive in itself, actively alters the environment. In a sense, behavior-based robotics uses the environment as a representation, while our approach creates representations in the environment. As a result, our dropper network can develop behavior that can solve memory tasks. Braitenberg and many other researchers have studied the importance of sensory-motor interaction between the agent and its environment in behavioral modification [37], [38], [39], [33]. It seems like it is only through this agent-environment coupling that the agent can break away from simple reactive behavior.

Another existing work with strong similarity to our work is the active field of ant colony optimization [40], [41], [42], [43], [44]. Ant colony optimization commonly uses artificial pheromone-like markers just like our dropper network does, and has been used successfully in many tough optimization tasks. However, the main use of pheromones in ant colony

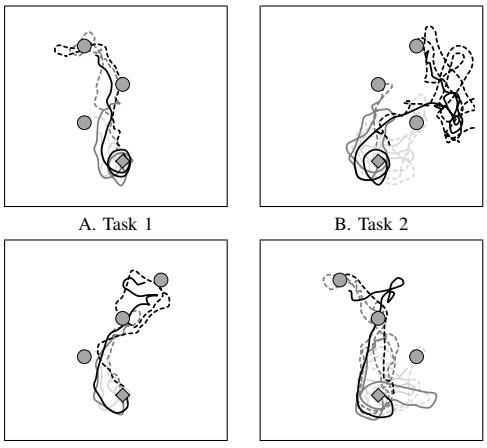
optimization is for social function (i.e., to enable communication among different individuals in the swarm). This is in contrast with our own work presented in this paper where the pheromones (markers) are used for the agent's own individual purpose. Moreover, existing ant colony optimization works did not address a substantial property of the pheromone. Pheromones have two important properties: spatial and temporal. The location and time of a pheromone drop denote the locus and instant of importance, respectively. Nontheless, most studies on pheromone agents overlooked the temporal property by predetermining the pheromone deposition rules. Their agents throw pheromone unconditionally either at each state transition (online step-by-step pheromone trail update) or as they trace back their path after the solution is found (online delayed pheromone trail update) [45], [46]. If traditional pheromone agent with unconditional marker throwing behavior is used in the foraging task, the increased number of markers will create ambiguities. Similar to what the dropper networks with non-evaporative markers did to resolve ambiguity, they may also have to evolve more difficult strategies with additional rules. Therefore, it would be an interesting direction for future work to compare the performance between self-conditioned and unconditional marker throwing behaviors. Moreover, because the external markers with richer information can lower the complexity of the evolved strategy, it will also be interesting to confirm whether the strategy also gets simpler with additional types of markers.



A. Success Rate B. Sample Trajectory

Fig. 22. Preliminary Result of Hybrid Networks. A. Success rate comparison. Colors indicate different network types (red: recurrent; green: dropper; blue: hybrid). The best performing configuration of each network is shown (recurrent: λ=0.99, N_{mem}=5; dropper: ρ=0.7; hybrid: λ=0.99, N_{mem}=5, ρ=0.7). See Fig. 15 and Fig. 16 for details. B. Trajectory of a sampled hybrid network. See Fig. 17 for plotting conventions.

An interesting insight arises when we compare the external and internal implementation of memory. External memory is more persistent, while internal memory is more transient, due to its dynamic nature, and they may have specialized in these respective kinds of tasks. Probably this is why both systems are preserved in modern animals. Therefore, it would be an interesting future work to observe the evolution of hybrid networks with both recurrent connection and marker dropping/detecting ability. In fact, our preliminary results suggest that a hybrid can outperform the non-hybrid versions (Fig. 22). It will also be intriguing to analyze if there exists a certain condition that makes one type of memory (external or internal) more preferable to another. Finally, the low performance of recurrent agents may also be due to the difficulty of evolving networks with a huge number of tunable parameters (the connection weights). The dropper network is almost the same as the feedforward network, while the function is much more advanced. This kind of economy could surely have been exploited by the process of evolution.



A. Task 1 B. Task 2

C. Task 3 D. Task 4

Fig. 23. Trajectories of A Successful Dropper Agent in Multiple Tasks. A single dropper agent throwing markers evaporative markers (ρ =0.99) was successful in solving multiple tasks with various food locations. See Fig. 17 for plotting conventions.

One possible criticism is that the tasks themselves do not require elaborate spatial memory, and can be tackled with taxon navigation. Taxon navigation uses a reactive stimulus-response strategy, with which agents generate direct homing onto landmarks. Taxon navigation requires a visible landmark at all times, whereas in our tasks, due to the limited sensor range, landmarks (or objects other than the self-generated markers) are not always visible, thus the task itself requires spatial memory. However, it is still possible that our agents used taxon navigation since all three food items were always at the same location throughout trials. To exclude this possibility, we evolved our dropper agent by varying the food item locations. The initial results were encouraging (Fig. 23), where the same evolved agent was able to solve all four different food item configurations, thus showing the agent's spatial memory function. The reactive part of the agent indeed employs a taxon navigation strategy, but the landmarks in this case are both food items and agent-generated markers. Thus, as a whole (i.e., including the marker dropper/sensing capability), the agent cannot be said to be following a taxon-based strategy.

As we briefly mentioned in the introduction, our work has strong implications on understanding the relationship between the olfactory system, the hippocampus, and the neuromodulator networks. In our view, olfaction is a form of external memory (dropper/detector network), while the hippocampus is a form of internal memory (recurrent network). Neuromodulators can be seen as an intermediate step between external and internal memory, where markers (neuromodulators) can be thought of as being dropped *inside* the brain, rather than outside, without explicit recurrent neuronal linkage (cf. [13]). Theoretical work also indicates that the boundary between the internal and the external can be blurred [47] (note: in this case, the main focus is on the reward structure).

In fact, developing a neuroevolution framework to include such possibility is a promising direction we are currently ex-

ploring. We strongly believe this intermediate step is necessary to simulate the evolution of memory, from external to fully internal. This kind of evolution can be readily implemented in neuroevolution algorithms such as Neuroevolution of Augmenting Topologies (NEAT) [48], with the addition of marker dropping and detection (both external and internal).

Another line of future work is to gain insights on the exact computational nature of our dropper/detector agents by measuring the information flow in the sensorimotor loop, borrowing from [49]. Results in Fig. 8, Fig. 9, Fig. 20, and Fig. 21 suggest that distributed memory can be reflected in the internal representation of dropper networks, which is similar to emergent meta-level cognition in dynamical systems [50], [25], [51]. Our work shares a common view with these earlier works that embodiment plays an important role in forming such internally represented cognition. We will also scale up the memory capacity by introducing different types of markers and detectors, and to have multiple agents interact with each other using these markers.

Finally, internal marker dropping/detecting can be thought of as a primitive form of mechanism that supports self-aware/self-effecting behavior (SASE agents, [52]). Thus, our work shows an interesting way to approach autonomous mental development.

V. Conclusion

The main contribution of our work is to have shown computationally that the use of external markers enables reactive agents to perform successfully in tasks requiring memory. With other studies listed in the introduction and the discussion, our result reinforces the concept that memory may have its evolutionary origin in the olfactory system (enabling material interaction) and that an important first step toward the evolution of cognition could have been the use of external markers.

Acknowledgments

This is a significantly expanded version of our earlier papers that appeared in [53] and [54]. New results (high vs. low memory capacity and detailed behavioral trajectories) and analysis (hidden layer dynamics) have been added, and the introduction and discussion substantially expanded. We would like to thank the anonymous reviewers and Sanjay Chandrasekharan for their constructive comments.

References

[1] M. I. Jordan, *Attractor Dynamics and Parallelism in a Connectionist Sequential Machine*. IEEE, 1990.

[2] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, pp. 179–211, 1990.

[3] D. J. Felleman and D. C. Van Essen, "Distributed hierarchical processing in primate cerebral cortex," *Cerebral Cortex*, vol. 1, pp. 1–47, 1991.

[4] K. G. Field, G. J. Olsen, D. J. Lane, S. J. Giovannoni, M. T. Ghiselin, E. C. Raff, N. R. Pace, and R. A. Raff, "Molecular phylogeny of the animal kingdom," *Science*, vol. 239, pp. 748–753, 1988.

[5] N. King, "The unicellular ancestry of animal development," *Developmental Cell*, vol. 7, pp. 313–325, 2004.

[6] L. W. Swanson, *Brain Architecture: Understanding the Basic Plan*. Oxford: Oxford University Press, 2003.

[7] J. G. Hildebrand, "Analysis of chemical signals by nervous systems," *Proceedings of National Academy of Sciences, USA*, vol. 92, pp. 67–74, 1995.

[8] G. H. Wadhams and J. P. Armitage, "Making sense of it all: bacterial chemotaxis," *Nature Reviews Molecular Cell Biology*, vol. 5, pp. 1024–1037, 2004.

[9] Z. Tang-Martinez, "The mechanisms of kin discrimination and the evolution of kin recognition in vertebrates: a critical re-evaluation," *Behavioural Processes*, vol. 53, pp. 21–40, 2001.

[10] D. W. Pfennig and P. W. Sherman, "Kin recognition," *Scientific American*, vol. 272, pp. 98–103, 1995.

[11] P. W. Sorensen and N. E. Stacey, "Evolution and specialization of fish hormonal pheromones," in *Advances in Chemical Signals in Vertebrates*, R. E. Johnston and P. W. Sorensen, Eds. New York: Plenum Publishers, 1999, pp. 15–47.

[12] T. D. Wyatt, *Pheromones and Animal Behavior*. Cambridge University Press, 2003.

[13] J. L. Krichmar, "The neuromodulatory system: a framework for survival and adaptive behavior in a challenging world," *Adaptive Behavior*, vol. 16, pp. 385–399, 2008.

[14] M. Doop, C. Mohr, B. Folley, W. Brewer, and S. Park, *Olfaction and the Brain*. Cambridge University Press, 2006.

[15] S. Chu and J. J. Downes, "Proust nose best: Odors are better cues of autobiographical memory," *Memory & Cognition*, vol. 30, pp. 511–518, 2002.

[16] A.-M. Mouly, U. Kindermann, R. Gervais, and A. Holley, "Involvement of the olfactory bulb in consolidation processes associated with long-term memory in rats," *Behavioral Neuroscience*, vol. 107, pp. 451–457, 1993.

[17] J. Frisén, C. B. Johansson, C. Lothian, and U. Lendahl, "Central nervous system stem cells in the embryo and adult," *CMLS, Cellular and Molecular Life Science*, vol. 54, pp. 935–945, 1998.

[18] R. Machold, S. Hayashi, M. Rutlin, M. D. Muzumdar, S. Nery, J. G. Corbin, A. Gritli-Linde, T. Dellovade, J. A. Porter, S. L. Rubin, H. Dudek, A. P. McMahon, and G. Fishell, "Sonic hedgehog is required for progenitor cell maintenance in telencephalic stem cell niches," *Neuron*, vol. 39, pp. 937–950, 2003.

[19] V. Palma, D. A. Lim, N. Dahmane, P. Sánchez, T. C. Brionne, C. D. Herzberg, Y. Gitton, A. Carleton, A. Álvarez Buylla, and A. R. Altaba, "Sonic hedgehog controls stem cell behavior in the postnatal and adult brain," *Development*, vol. 132, pp. 335–344, 2004.

[20] L. M. Rocha, "Eigenbehavior and symbols," *Systems Research*, vol. 13, pp. 371–384, 1996.

[21] S. Chandrasekharan and T. C. Stewart, "The origin of epistemic structures and proto-representations," *Adaptive Behavior*, vol. 15, pp. 329–353, 2007.

[22] J. O'Keefe and D. H. Conway, "Hippocampal place units in the freely moving rat: Why they fire where they fire," *Experimental Brain Research*, vol. 31, no. 4, pp. 573–590, 1978.

[23] J. L. Davis, H. Eichenbaum, and U. S. O. of Naval Research, *Olfaction: A Model System for Computational Neuroscience*. MIT Press, 1991.

[24] H. Eichenbaum, P. Dudchenko, E. Wood, M. Shapiro, and H. Tanila, "The hippocampus, memory, and place cells: Is it spatial memory or a memory space?" *Neuron*, vol. 23, pp. 209–226, 1999.

[25] R. D. Beer, "The dynamics of active categorical perception in an evolved model agent," *Adaptive Behavior*, vol. 11(4), pp. 209–243, 2003.

[26] R. Ward and R. Ward, "2006 special issue: Cognitive conflict without explicit conflict monitoring in a dynamical agent," *Neural Networks*, vol. 19, no. 9, pp. 1430–1436, 2006.

[27] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.

[28] P. A. Hetherington and M. L. Shapiro, "A simple network model simulates hippocampal place fields: Ii. computing goal-directed trajectories and memory fields," *Behavioral neuroscience*, vol. 111, pp. 20–34, 1993.

[29] A. F. Skutch, "Do tropical birds rear as many young as they can nourish?" *Ibis*, vol. 91, pp. 430–455, 1949.

[30] C. R. Carroll and D. H. Janzen, "Ecology of foraging by ants," *Annual Review of Ecology and Systematics*, vol. 4, pp. 231–257, 1973.

[31] C. C. Smith and O. J. Reichman, "The evolution of food caching by birds and mammals," *Annual Review of Ecology and Systematics*, vol. 15, pp. 329–351, 1984.

[32] A. D. Baddeley and G. J. Hitch, "The recency effect: Implicit learning with explicit retrieval?" *Memory and Cognition*, vol. 21, pp. 146–155, 1993.

[33] C. H. Yong and R. Miikkulainen, "Coevolution of role-based cooperation in multiagent systems," *IEEE Transactions on Autonomous Mental Development*, vol. 1, pp. 170–186, 2009.

[34] S. Chandrasekharan and T. Stewart, "Reactive agents learn to add epistemic structures to the world," in *CogSci2004*, K. D. Forbus, D. Gentner, and T. Regier, Eds. Hillsdale, NJ: Lawrence Erlbaum, 2004.
[35] A. E. A. Goosen, "The world inside your head: From structuring to representations to language," Master's thesis, Department of Artificial Intelligence, Radboud University Nijmegen, 2008.
[36] R. A. Brooks, "Intelligence without representation," *Artificial Intelligence*, vol. 47, pp. 139–159, 1991.
[37] V. Braitenberg, *Vehicles: Experiments in synthetic psychology*. Cambridge, MA: MIT Press, 1984.
[38] R. Pfeifer and C. Scheier, "Sensory-motor coordination: the metaphor and beyond," *Robotics and Autonomous Systems*, vol. 20, pp. 157–178, 1997.
[39] S. Nolfi, "Evolutionary robotics: exploiting the full power of self-organization," *Connection Science*, vol. 10, pp. 167–184, 1998.
[40] I. A. Wagner, M. Lindenbaum, and A. M. Bruckstein, "Distributed covering by ant-robots using evaporating traces," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 5, pp. 918–933, 1999.
[41] E. Bonabeau, M. Dorigo, and G. Theraulaz, "Inspiration for optimization from social insect behaviour," *Nature*, vol. 406, pp. 39–42, 2000.
[42] L. Panait and S. Luke, "A pheromone-based utility model for collaborative foraging," in *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*. Washington DC, USA: IEEE Computer Society, 2004, pp. 36–43.
[43] T. Ziemke, N. Bergfeldt, G. Buason, T. Susi, and H. Svensson, "Evolving cognitive scaffolding and environment adaptation: a new research direction for evolutionary robotics," *Connection Science*, vol. 16, pp. 339–350, 2004.
[44] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theoretical Computer Science*, vol. 344, no. 2-3, pp. 243–278, 2005.
[45] O. Cordón, F. Herrera, and T. Stützle, "A review on the ant colony optimization metaheuristic: basis, models and new trends," *Mathware and Soft Computing*, vol. 9 (2-3), pp. 141–175, 2002.
[46] R. J. Mullen, D. Monekosso, S. Barman, and P. Remagnino, "A review of ant algorithms," *Expert Systems with Applications*, vol. 36, pp. 9608–9617, 2009.
[47] S. Singh, R. L. Lewis, A. G. Barto, and J. Sorg, "Intrinsically motivated reinforcement learning: An evolutionary perspective," *IEEE Transactions on Autonomous Mental Development*, vol. 2, pp. 70–82, 2010.
[48] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, pp. 99–127, 2002.
[49] M. Lungarella and O. Sporns, "Mapping information flow in sensorimotor networks," *PLoS Comput Biol*, vol. 2, no. 10, pp. 1301–1312, 2006.
[50] R. D. Beer, "Dynamical approaches to cognitive science," *Trends in cognitive sciences*, vol. 4, pp. 91–99, 2000.
[51] M. Maniadakis and J. Tani, "Dynamical systems account form meta-level cognition," *10th International Conference on the Simuation of Adaptive Behavior (SAB-08)*, pp. 311–320, 2008.
[52] J. Weng, "Developmental robotics: Theory and experiments," *International Journal of Humanoid Robotics*, vol. 1, pp. 199–236, 2004.
[53] J. R. Chung, J. Kwon, and Y. Choe, "Evolution of recollection and prediction in neural networks," in *Proceedings of the International Joint Conference on Neural Networks*. Piscataway, NJ: IEEE Press, 2009, pp. 571–577.
[54] J. R. Chung and Y. Choe, "Emergence of memory-like behavior in reactive agents using external markers," in *21st International Conference on Tools with Artificial Intelligence (ICTAI09)*, Newark, NJ, USA, November 2-5 2009.

Ji Ryang Chung received the B.S. degrees from Seoul National University, Seoul, Korea, in 2004 in computer science and engineering. He is currently pursuing the doctoral degree in computer science and engineering at Texas A&M University.

His research involves the study of evolutionary route of internal memory system and the application of the concept to autonomous agents and multiagent systems.

Yoonsuck Choe Yoonsuck Choe is an associate professor in the Department of Computer Science and Engineering at Texas A&M University.

He received his B.S. degree in Computer Science from Yonsei University (Korea) in 1993, and his M.S. and Ph.D. degrees in Computer Sciences from the University of Texas at Austin in 1995 and 2001. His current research areas include computational neuroscience, neural networks, computational neuroanatomy, and biologically motivated vision.