Continuous-Spaced Action Selection for Single- and Multi-Robot Tasks Using Cooperative Extended Kohonen Maps

Kian Hsiang Low†, Wee Kheng Leow†, and Marcelo H. Ang, Jr.‡
†Dept. of Computer Science, National University of Singapore
3 Science Drive 2, Singapore 117543, Singapore
‡Dept. of Mechanical Engineering, National University of Singapore
10 Kent Ridge Crescent, Singapore 119260, Singapore
†lowkh, leowwk@comp.nus.edu.sg, ‡mpeangh@nus.edu.sg

Abstract - *Action selection is a central issue in the design of behavior-based control architectures for autonomous mobile robots. This paper presents an action selection framework based on an assemblage of self-organizing neural networks called Cooperative Extended Kohonen Maps. This framework encapsulates two features that significantly enhance a robot's action selection capability: self-organization in the continuous state and action spaces to provide smooth, efficient and fine motion control; action selection via the cooperation and competition of Extended Kohonen Maps so that more complex motion tasks can be achieved. Qualitative and quantitative comparisons for both single- and multi-robot motion tasks show that our framework can provide better action selection than do action superposition methods.*

1 Introduction

A central issue in the design of behavior-based control architectures for autonomous agents is the formulation of effective mechanisms to coordinate the behaviors. These mechanisms determine the policy of conflict resolution between behaviors, which involves behavioral cooperation and competition to select the most appropriate action. Developing such an action selection mechanism (ASM) is non-trivial due to realistic constraints such as environmental complexity and unpredictability, and resource limitations, which include computational and cognitive capabilities of the agent, incomplete knowledge of the environment, and time constraints.

This paper describes a neural network-based ASM for autonomous non-holonomic mobile robots. Our motivation is to develop a motion control strategy that can perform distributed multi-robot surveillance in unknown, dynamic, and unpredictable environments. By implementing the ASM using an assemblage of self-organizing neural networks, it induces the following key features that significantly enhance the agent's action selection capability:

Self-organization of continuous state and action spaces

As emphasized in recent autonomous agent research utilizing dynamical systems theory [2, 11] and reinforcement learning [7, 15], an agent's ASM should operate in continuous state and action spaces so that its interaction with the complex, unpredictable environment can be versatile and robust. In particular, a high degree of smoothness, flexibility and precision in motion control is essential for efficiently executing sophisticated tasks and interacting with humans. This characteristic can only be achieved with *continuous response encoding* (i.e., infinite set of responses) of very low-level velocity/torque control of motor/joint actuators. Our proposed ASM uses self-organizing neural networks to map continuous state space to continuous motor control space, so as to produce fine, smooth and efficient motion control.

In contrast, ASMs that employ *discrete response encoding* (i.e., finite, enumerated set of responses) [4, 14] produce high-level motion commands (e.g., forward, left, right) that are usually too coarse for fine, smooth robot control. Consequently, the robot may fail to negotiate unforeseen complex obstacles.

Action selection by cooperation and competition of self-organizing neural networks

There are four general classes of ASMs: behavior arbitration [4,9], action voting [13,14], action superposition [1,3,6], and sensor fusion [5, 8]. Our method is similar to sensor fusion. However, instead of fusing sensory inputs, each sensory input activates a separate self-organizing neural network, and the neural networks cooperate and compete to produce a final action. The versatility of our method will be shown in Section 3.

In contrast, the other three classes of ASMs tend to underutilize the sensory inputs that can potentially yield useful information for selecting rational actions. For example, a robot that uses action superposition ASM cannot pass through a narrow doorway because the forward action induced by the goal is cancelled by the backward action to avoid obstacles.

2 Action Selection Framework

2.1 Overview

Our proposed ASM is implemented by connecting an ensemble of Extended Kohonen Maps (EKMs), which are extensions of the Kohonen Self-Organizing Map. In addition to encoding a set of input weights that self-organize the sensory input space, the EKM neurons also produce outputs that vary with the incoming sensed inputs.

Our ASM framework consists of four modules (Fig. 1). The *target localization* EKMs in the *target reaching* module are activated by the presence of targets within the robot's target sensing range. Each EKM receives a sensed target location and outputs corresponding excitatory signals to the motor

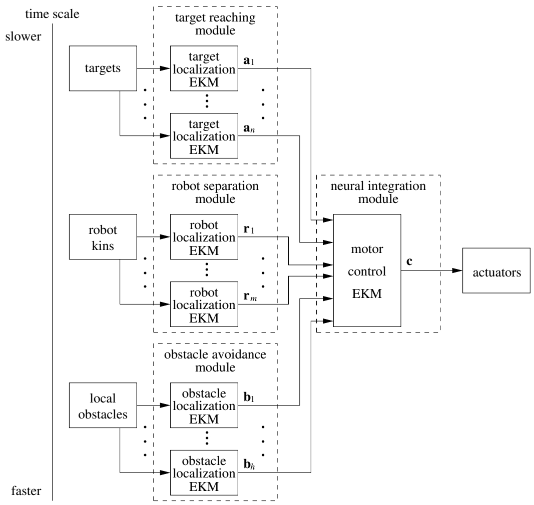


Figure 1: An action selection framework that is implemented by an ensemble of EKMs.

control EKM in the neural integration module at and around the locations of the sensed targets.

The *obstacle localization* EKMs in the *obstacle avoidance* module are activated by the presence of obstacles within the robot's obstacle sensing range. Each EKM receives a sensed obstacle location and outputs corresponding inhibitory signals to the motor control EKM in the neural integration module at and around the locations of the sensed obstacles. The *robot localization* EKMs in the *robot separation* module work in a similar fashion as the obstacle localization EKMs except that they process the sensed robot locations.

The *motor control* EKM in the *neural integration* module serves as the sensorimotor interface, which integrates the activity signals from the EKMs for cooperation and competition to produce an appropriate motor signal to the actuators.

All the modules operate asynchronously at different rates. The neural integration module is activated as and when neural activities are received. One noteworthy aspect of our framework is that no communication between robots is needed for the robots to cooperate in the tracking of multiple moving targets. In this paper, we demonstrate that robots, which are able to discriminate between targets, obstacles and robot kins, are adequate for achieving the cooperative task.

2.2 Target Reaching

The target reaching module adopts an egocentric representation of the sensory input vector $\mathbf{u}_p = (\alpha, d)^T$ where α and d are the direction and distance of a target relative to the robot's current location and heading. It uses the target localization EKM to self-organize the sensory input space $\mathcal{U}$. Each neuron i in the EKM has a sensory weight vector $\mathbf{w}_i = (\alpha_i, d_i)^T$ that encodes a region in $\mathcal{U}$ centered at $\mathbf{w}_i$. Each target that appears within the robot's sensory range activates a different target localization EKM. The same target can activate a different EKM at a different time. Based on each incoming sensory input $\mathbf{u}_p$ of the target location, the target localization EKM outputs excitatory signals to the motor control EKM in the neural integration module (Section 2.4).

Target Localization

For each sensory input $\mathbf{u}_p$, $p = 1, \ldots, n$,

1. Determine the winning neuron s in the pth target localization EKM. Each winning neuron s is the one whose sensory weight vector $\mathbf{w}_s = (\alpha_s, d_s)^T$ is nearest to the input $\mathbf{u}_p = (\alpha, d)^T$:
$$D(\mathbf{u}_p, \mathbf{w}_s) = \min_{i \in \mathcal{A}(\alpha)} D(\mathbf{u}_p, \mathbf{w}_i). \qquad (1)$$
The difference $D(\mathbf{u}_p, \mathbf{w}_i)$ is a weighted difference between $\mathbf{u}_p$ and $\mathbf{w}_i$:
$$D(\mathbf{u}_p, \mathbf{w}_i) = \beta_\alpha(\alpha - \alpha_i)^2 + \beta_d(d - d_i)^2 \qquad (2)$$
where β_α and β_d are constant parameters. The minimum in Eq. 1 is taken over the set $\mathcal{A}(\alpha)$ of neurons encoding very similar angles as α:
$$|\alpha - \alpha_i| \leq |\alpha - \alpha_j|, \quad \text{for each pair } i \in \mathcal{A}(\alpha), j \notin \mathcal{A}(\alpha) \ . \qquad (3)$$
In other words, direction has priority over distance in the competition between EKM neurons. This method allows the robot to quickly orientate itself to face the target while moving towards it. In the EKM, each neuron encodes a location $\mathbf{w}_i$ in the sensory input space $\mathcal{U}$. The region of $\mathcal{U}$ that encloses all the neurons is called the *local workspace* $\mathcal{U}'$. Even if the target falls outside $\mathcal{U}'$, the nearest neuron can still be activated (Fig. 2a).
2. Compute output activity a_i of neuron i in the p-th target localization EKM.
$$a_i = G_a(\mathbf{w}_s, \mathbf{w}_i) \qquad (4)$$
The function G_a is an elongated Gaussian:
$$G_a(\mathbf{w}_s, \mathbf{w}_i) = \exp\left[-\left(\frac{\alpha_s - \alpha_i}{\sigma_{a\alpha}}\right)^2 - \left(\frac{d_s - d_i}{\sigma_{ad}}\right)^2\right]. \qquad (5)$$
Parameter σ_{ad} is much smaller than $\sigma_{a\alpha}$, making the Gaussian distance-sensitive and angle-insensitive. These parameter values elongate the Gaussian along the direction perpendicular to the target direction α_s (Fig. 2b). This elongated Gaussian is the *target field*, which plays an important role in avoiding local minima during obstacle avoidance.

2.3 Obstacle Avoidance and Robot Separation

The obstacle avoidance module uses the obstacle localization EKMs that are self-organized in the same way as the target localization EKMs (Section 2.5). Each neuron i in the obstacle localization EKM has the same input weight vector $\mathbf{w}_i$ as the neuron i in the target localization EKM. The robot has h directed distance sensors around its body for detecting obstacles. Hence, each activated sensor encodes a fixed direction α_j and a variable distance d_j of the obstacle relative to the robot's heading and location. Each sensed input $\mathbf{u}_j = (\alpha_j, d_j)^T$ induces an obstacle localization EKM to output inhibitory signals to the motor control EKM in the neural integration module (Section 2.4).

Obstacle Localization

For each sensory input $\mathbf{u}_j$, $j = 1, \ldots, h$,

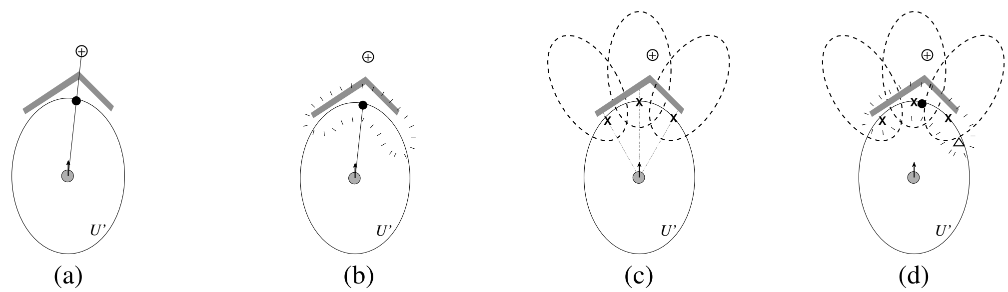


Figure 2: Cooperative EKMs. (a) In response to the target $\oplus$, the nearest neuron (black dot) in the target localization EKM (ellipse) of the robot (gray circle) is activated. (b) The activated neuron produces a target field (dotted ellipse) in the motor control EKM. (c) Three of the robot's sensors detect obstacles and activate three neurons (crosses) in the obstacle localization EKMs, which produce the obstacle fields (dashed ellipses). (d) Subtraction of the obstacle fields from the target field results in the neuron at $\triangle$ to become the winner in the motor control EKM, which moves the robot away from the obstacle.

1. Determine the winning neuron s in the j-th obstacle localization EKM. The obstacle localization EKM is activated in the same manner as Step 1 of Target Localization (Section 2.2).
2. Compute output activity b_i of neuron i in the j-th obstacle localization EKM:

$$b_i = G_b(\mathbf{w}_s, \mathbf{w}_i) \tag{6}$$

where

$$\begin{aligned} G_b(\mathbf{w}_s, \mathbf{w}_i) = &\quad \exp\left[-\left(\frac{\alpha_s - \alpha_i}{\sigma_{b\alpha}}\right)^2 - \left(\frac{d_s - d_i}{\sigma_{bd}(d_s, d_i)}\right)^2\right] \\ \sigma_{bd}(d_s, d_i) = &\quad \begin{cases} 3.5 & \text{if } d_i \geq d_s \\ 0.035 & \text{otherwise.} \end{cases} \end{aligned} \tag{7}$$

The function G_b is a Gaussian stretched along the obstacle direction α_s so that motor control EKM neurons beyond the obstacle locations are also inhibited to indicate inaccessibility (Fig. 2c). If no obstacle is detected, $G_b = 0$. In the presence of an obstacle, the neurons in the obstacle localization EKMs at and near the obstacle locations will be activated to produce *obstacle fields* (Eq. 6). The neurons nearest to the obstacle locations have the strongest activities.

The separation between a robot and its other kins is achieved with robot localization EKMs. These EKMs work in the same way as obstacle localization EKMs, except that they produce wider *robot kin fields*. This has the effect of keeping a robot away from targets that are close to other robot kins. As a result, the overlap in the coverage of targets between robots is minimized.

2.4 Neural Integration

The neural integration module uses a motor control EKM to integrate the activities from the neurons in the target, obstacle and robot localization EKMs. It is trained to partition the sensory input space $\mathcal{U}$ into locally linear regions. Each neuron i in the motor control EKM is self-organized in the same way as the localization EKMs by encoding the same input weight vector $\mathbf{w}_i$ as the neuron i in those EKMs. It also has a set of output weights which encode the outputs produced by the neuron. However, unlike existing direct-mapping methods [12, 16] that perform discrete response encoding (Section 1), the output weights $\mathbf{M}_i$ of neuron i of the motor control EKM represents control parameters in the parameter space $\mathcal{M}$ instead of the actual motor control vector. The control parameter matrix $\mathbf{M}_i$ is mapped to the actual motor control vector $\mathbf{c}$ by a linear model (Eq. 10). Compared to direct-mapping EKM, indirect-mapping EKM can provide finer and smoother robot motion control (Section 2.5). With indirect-mapping EKM, motor control is performed as follows:

Motor Control

1. Compute activity e_i of neuron i in the motor control EKM.

$$e_i = \sum_{p=1}^{n} a_{pi} - \sum_{j=1}^{h} b_{ji} - \sum_{q=1}^{m} r_{qi} \tag{8}$$

where a_{pi} is the excitatory input from neuron i of the p-th target localization EKM (Section 2.2), b_{ji} is the inhibitory input from neuron i of the j-th obstacle localization EKM and r_{qi} is the inhibitory input from neuron i of the q-th robot localization EKM (Section 2.3).

2. Determine the winning neuron k in the motor control EKM. Neuron k is the one with the largest activity:

$$e_k = \max_i e_i \ . \tag{9}$$

3. Compute motor control vector $\mathbf{c}$:

$$\mathbf{c} = \mathbf{M}_k \mathbf{z}_k \tag{10}$$

where

$$\mathbf{z}_k = \frac{\displaystyle\sum_{i \in \mathcal{N}(k)} G(|e_i - e_k|)\mathbf{w}_i}{\displaystyle\sum_{i \in \mathcal{N}(k)} G(|e_i - e_k|)} \tag{11}$$

$G(|e_i - e_k|)$ is a Gaussian with its peak located at neuron k and $\mathcal{N}(k)$ defines a small set of neurons in the neighborhood of neuron k. At the goal state at time T, $\mathbf{z}_k(T) = (\alpha, 0)^T$ for any α.

In activating the motor control EKM (Fig. 2d), the obstacle fields are subtracted from the target field (Eq. 8). If the target lies within the obstacle fields, the activation of the motor control EKM neurons close to the target location will be suppressed. Consequently, another neuron at a location that is not inhibited by the obstacle fields becomes most highly activated (Fig. 2d). This neuron produces a control parameter that moves the robot away from the obstacle. While the robot moves around the obstacle, the target and obstacle localization EKMs are continuously updated with the current locations and directions of the target and obstacles. Their interactions with the motor control EKM produce fine, smooth, and accurate motion control of the robot to negotiate the obstacle and move towards the target until it reaches the goal state $\mathbf{z}_k(T)$ at time step T. In the case of multi-robot tracking task, the robots act like obstacles to other robots, thus separating them from each other.

2.5 Self-Organization of EKMs

In contrast to most existing methods, online training is adopted for the EKMs. Initially, the EKMs have not been trained and the motor control vectors $\mathbf{c}$ generated are inaccurate. Nevertheless, the EKMs self-organize, using these control vectors $\mathbf{c}$ and the corresponding robot displacements $\mathbf{v}$ produced by $\mathbf{c}$, to map $\mathbf{v}$ to $\mathbf{c}$ indirectly. As the robot moves around and learns the correct mapping, its sensorimotor control becomes more accurate. At this stage, the same online training mainly fine tunes the indirect mapping. The self-organized training algorithm (in obstacle-free environment) is as follows:

Self-Organized Training

Repeat

1. Get sensory input $\mathbf{u}_p$.
2. Execute target reaching procedure and move robot.
3. Get new sensory input $\mathbf{u}'_p$ and compute actual displacement $\mathbf{v}$ as a difference between $\mathbf{u}'_p$ and $\mathbf{u}_p$.
4. Use $\mathbf{v}$ as the training input to determine the winning neuron k (same as Step 1 of Target Reaching).
5. Adjust the weights $\mathbf{w}_i$ of neurons i in the neighborhood $\mathcal{N}_k$ of the winning neuron k towards $\mathbf{v}$:
$$\Delta \mathbf{w}_i = \eta\, G(k,i)(\mathbf{v} - \mathbf{w}_i) \tag{12}$$
where $G(k,i)$ is a Gaussian function of the distance between the positions of neurons k and i in the EKM, and η is a constant learning rate.
6. Update the weights $\mathbf{M}_i$ of neurons i in the neighborhood $\mathcal{N}_k$ to minimize the error e:
$$e = \frac{1}{2} G(k,i) \|\mathbf{c} - \mathbf{M}_i \mathbf{v}\|^2 . \tag{13}$$
That is, apply gradient descent to obtain
$$\Delta \mathbf{M}_i = -\eta \frac{\partial e}{\partial \mathbf{M}_i} = \eta\, G(k,i)(\mathbf{c} - \mathbf{M}_i \mathbf{v})\mathbf{v}^T . \tag{14}$$

The target, obstacle and robot localization EKMs self-organize in the same manner as the motor control EKM except that Step 6 is omitted. At each training cycle, the weights of the winning neuron k and its neighboring neurons i are modified. The amount of modification is proportional to the distance $G(k,i)$ between the neurons in the EKM. The input weights $\mathbf{w}_i$ are updated towards the actual displacement $\mathbf{v}$ and the control parameters $\mathbf{M}_i$ are updated so that they map the displacement $\mathbf{v}$ to the corresponding motor control $\mathbf{c}$. After self-organization has converged, the neurons will stabilize in a state such that $\mathbf{v} = \mathbf{w}_i$ and $\mathbf{c} = \mathbf{M}_i\mathbf{v} = \mathbf{M}_i\mathbf{w}_i$. For any winning neuron k, given that $\mathbf{z}_k = \mathbf{w}_k$, the neuron will produce a motor control output $\mathbf{c} = \mathbf{M}_k\mathbf{w}_k$ which yields a desired displacement of $\mathbf{v} = \mathbf{w}_k$. If $\mathbf{z}_k \neq \mathbf{w}_k$ but close to $\mathbf{w}_k$, the motor output $\mathbf{c} = \mathbf{M}_k\mathbf{z}_k$ produced by neuron k will still yield the correct displacement if linearity holds within the input region that activates neuron k. Thus, given enough neurons to produce an approximate linearization of the sensory input space $\mathcal{U}$, indirect-mapping EKM can produce finer and smoother motion control than direct-mapping EKM.

3 Experiments and Discussions

3.1 Robot Motion in Complex, Unpredictable Environments

This section presents a qualitative evaluation of the action selection capabilities of a non-holonomic mobile robot endowed with cooperative EKMs for goal-directed, collision-free motion in complex, unpredictable environments. The experiments were performed using Webots, an embodied simulator for Khepera mobile robots, which incorporated 10% noise in its sensors and actuators. 12 directed long-range sensors were also modelled around its body of radius 2.5 cm. Each sensor had a range of 17.5 cm, enabling the detection of obstacles at 20 cm or nearer from the robot's center, and a resolution of 0.5 cm to simulate noise.

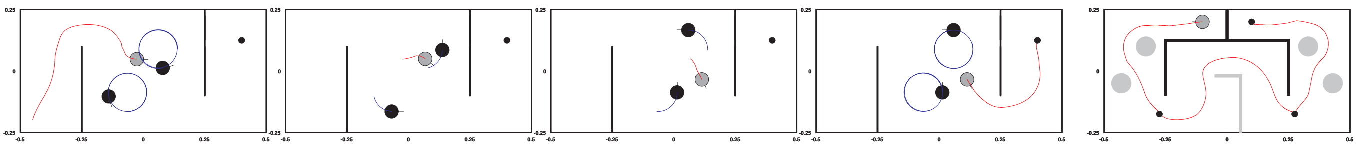
Two tests were conducted to demonstrate the capabilities of cooperative EKMs in performing complex motion tasks. The environment for the first test consisted of three rooms connected by two doorways (Fig. 3(a)–(d)). The middle room contained two obstacles moving in anticlockwise circular paths. The robot began in the left-most room and was tasked to move to the right-most room. Test results show that the robot was able to negotiate past the extended walls and the dynamic obstacles to reach the goal.

The environment for the second test consisted of three rooms connected by two doorways and an unforeseen static obstacle (Fig. 3(e)). The robot began in the top corner of the left-most room and was tasked to move into the narrow corner of the right-most room via checkpoints plotted by a planner. The robot was able to move through the checkpoints to the goal by traversing between narrowly spaced convex obstacles in the first and the last room, and overcoming an unforeseen concave obstacle in the middle room. This result further confirms the effectiveness of cooperative EKMs in handling complex, unpredictable environments.

Similar tests have also been performed on robots that use action superposition method. In those tests, the robots were trapped by the narrow doorways between closely spaced obstacles, and were unable to go through the doorways.

3.2 Cooperative Multi-Robot Observation of Multiple Moving Targets

This section presents qualitative and quantitative tests of the action selection capabilities of a team of robots, each fitted



(a) (b) (c) (d) (e)

Figure 3: (a–d) Motion of robot (gray) in an environment with two unforeseen obstacles (black) moving in anticlockwise circular paths. The robot successfully negotiated past the extended walls and the dynamic obstacles to reach the goal (small black dot). (e) Motion of robot (dark gray) in an environment with an unforeseen static obstacle (light gray). The robot successfully navigated through the checkpoints (small black dots) located at the doorways to reach the goal.

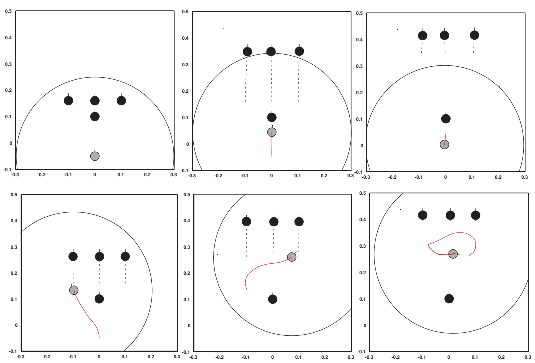


Figure 4: (Top row) Robot (gray) using action superposition ASM got stuck at the stationary target. Eventually, the three mobile targets moved out of the robot's sensing range (circle). (Bottom row) Robot using cooperative EKMs could negotiate past the stationary target to track all the targets.

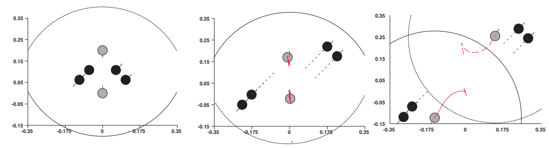
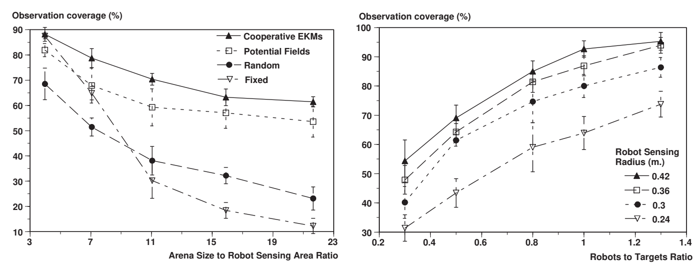


Figure 5: Cooperative tracking of moving targets. When the targets were moving out of the robots' sensory range, the two robots moved in opposite directions to track the targets. In this way, all targets could still be observed by the robots.



(a) (b)

Figure 6: Comparison of observation coverage for (a) robots using different tracking strategies in varying arena size, and (b) varying number of robots with different sensing ranges.

with cooperative EKMs, in cooperative tracking of multiple mobile targets. The motivation of this inherently cooperative task relates to that of security, surveillance, and reconnaissance such that the number of sensors and sensor positionings cannot be predetermined due to the following constraints: little information is known about the exact locations of the targets; the available sensor range is limited; the area to be observed is so large that it requires placing a lot of static sensors, which is not economical; the area is physically inaccessible before the actual deployment. All these conditions may cause the robots' sensors to be unable to cover the entire region of interest. The robots must therefore move dynamically in response to the targets' motion to maintain observation and maximize coverage.

Four tests were conducted using Webots simulator with settings similar to those in Section 3.1. The first test (Fig. 4) was performed to highlight the advantages of cooperative EKMs over an action superposition ASM known as potential fields [6] utilized by [10] for the same task. The robot utilizing potential fields got trapped by the static target while attempting to track all four targets. Eventually, the three mobile targets moved out of the robot's sensing range, causing the robot to observe only one out of four targets. In contrast, the robot fitted with cooperative EKMs was able to negotiate past the stationary target to track the three moving targets as well. All four targets were thus observed by the robot. The results of this test demonstrated that local minima situations could greatly decrease the coverage of targets by robots using potential fields. However, robots endowed with cooperative EKMs can still provide maximum coverage under these situations.

The next test (Fig. 5) illustrates how two robots endowed with cooperative EKMs cooperate to track four moving targets. When the targets were moving out of the robots' sensory range, the robot below chose to track the two targets moving to the bottom left while the robot above responded by tracking the two targets moving to the top right. In this manner, all targets could be observed by the robots. This test shows that the two robots can cooperate to track multiple moving targets without communicating with each other.

Two quantitative tests were conducted to determine the overall tracking performance of the robot team based on the following performance index [10]:

$$\text{observation coverage} = \sum_{t=1}^{T} 100\frac{n(t)}{NT} \qquad (15)$$

where N is the total number of targets, n is the number of targets being tracked at time t, and the experiment lasts T amount of time. For both tests, N and T were fixed respectively as 10 targets and 1000 time steps at intervals of 128 ms.

The first test compared the mean observation coverage of robots adopting four different tracking strategies: Cooperative EKMs, potential fields method, fixed deployment, and random deployment. The environment or arena was an enclosed, octagonal, obstacle-free region that varied in size. The mobile targets were forward-moving Braitenberg obstacle-avoidance vehicles [3] that changed their direction and speed with 5% probability. Five robots, each with target and robot sensing radius of 0.3 m, were deployed in this task. The fixed deployment approach distributed stationary robots uniformly over the arena. The random deployment approach allowed the robots to move randomly in a manner similar to the moving targets. Test results in Fig. 6(a) reveal that, in very large arenas, tracking strategies that respond dynamically to the targets' motion (cooperative EKMs and potential fields) are significantly better than those that do not (fixed and random). In particular, cooperative EKMs offered the highest observation coverage.

The second test compared the mean observation coverage of the cooperative-EKM robots with different sensing ranges and number of robots. The size of the arena was 6.4 m^2, which corresponded to the largest arena used for the first test. Test results in Fig. 6(b) show that observation coverage increases with increasing number of robots and sensing range.

4 Conclusion

This paper describes an action selection framework based on an assemblage of cooperative and competitive EKMs. It can significantly enhance a robot's action selection capability by employing self-organization in continuous state and action spaces to provide smooth, efficient and fine motion control, and action selection at the neuronal level via the cooperation and competition of EKMs to yield more flexible and varied motor patterns for achieving complex motion tasks. Qualitative and quantitative comparisons of cooperative EKMs with action superposition ASMs have shown that cooperative EKMs can provide better action selection capability in both the single- and multi-robot motion tasks, even though action superposition ASMs also operate in continuous state and action spaces. Thus, robots that adopt our framework can perform better in multi-robot surveillance in unknown, dynamic, and unpredictable environments.

References

[1] R. C. Arkin. Motor schema-based mobile robot navigation. *International Journal of Robotics Research*, 8(4):92–112, 1989.

[2] R. D. Beer. A dynamical systems perspective on agent-environment interaction. *Artificial Intelligence*, 72(1-2):173–215, 1995.

[3] V. Braitenberg. *Vehicles: Experiments in Synthetic Psychology*. MIT Press, 1984.

[4] V. Decugis and J. Ferber. Action selection in an autonomous agent with a hierarchical distributed reactive planning architecture. In *Proceedings of 2nd International Conference on Autonomous Agents*, pages 354–361, 1998.

[5] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, 3:249–265, 1987.

[6] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1):90–98, 1986.

[7] J. del R Millán, D. Posenato, and E. Dedieu. Continuous-action Q-learning. *Machine Learning*, 49:249–265, 2002.

[8] H. P. Moravec and D. W. Cho. A Bayesian method for certainty grids. In *Working Notes: AAAI Spring Symposium on Robot Navigation*, pages 57–60, 1989.

[9] M. Nicolescu and M. J. Matarić. A hierarchical architecture for behavior-based robots. In *Proceedings of 1st International Joint Conference on Autonomous Agents and MultiAgent Systems*, volume 1, pages 227–233, 2002.

[10] L. E. Parker. Distributed algorithms for multi-robot observation of multiple moving targets. *Autonomous Robots*, 12(3):231–255, 2002.

[11] R. F. Port and T. van Gelder. *Mind as Motion: Explorations in the Dynamics of Cognition*. MIT Press, Cambridge, MA, 1995.

[12] R. P. H. Rao and O. Fuentes. Hierarchical learning of navigational behaviors in an autonomous robot using a predictive sparse distributed memory. *Machine Learning*, 31(1-3):87–113, 1998.

[13] J. Riekki and J. Röning. Reactive task execution by combining action maps. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 224–230, 1997.

[14] J. K. Rosenblatt. DAMN: A distributed architecture for mobile navigation. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(2-3):339–360, 1997.

[15] W. D. Smart and L. P. Kaelbling. Practical reinforcement learning in continuous spaces. In *Proceedings of International Conference on Machine Learning*, 2000.

[16] C. Touzet. Neural reinforcement learning for behavior synthesis. *Robotics and Autonomous Systems*, 22(3-4):251–281, 1997.