17

Coordinated Hunting Based on Spiking Neural Network for Multi-robot System

Xu Wang, Zhiqiang Cao, Chao Zhou, Zengguang Hou and Min Tan
Laboratory of Complex Systems and Intelligence Science
Institute of Automation, Chinese Academy of Sciences
Beijing 100190
China

1. Introduction

Multi-robot systems have been extensively studied and the coordination among the robots becomes a hotspot. Among the typical research tasks of multi-robot system, coordinated hunting with unknown irregular motion of the evader or target has attracted more and more attentions due to its potential application to military, safe guard etc.

The spiking neural network (SNN), considered as the third generation of neural network (Maass & Bishop, 1999), has attracted many attentions. Spikes (pulses) are used to deliver the information between neurons, i.e. SNN processes the information in the form of spikes, which brings temporal structure and extends the functionality of SNN (Kasabov, 2010). Besides the rate coding, inspired by the results of biological experiments, some coding strategies based on spike timing have been proposed, such as time-to-first-spike coding, phase coding, correlations/synchrony coding et al. (Maass & Bishop, 1999). Also, Maass et al. present some useful spiking neuron models, such as spike response model (SRM), integrate-and-fire model (IF), Hodgkin-Huxley Model and so on (Maass & Bishop, 1999). For these coding methods and neuron models, the Hebb learning is proposed to adapt the weights between neurons based on the temporal difference between input and output spikes (Kempter et al., 1999). Nowadays, the controllers based on SNN have been introduced to many applications, such as phase/frequency correlations recognizing (Kiselev, 2009), movement prediction from real-world images (Burgsteiner et al., 2005), movement generation of the robot arm (Joshi & Maass, 2005), etc. Especially, SNN has been applied to the control of the mobile robot (Floreano & Mattiussi, 2001; Roggen et al., 2003; Hagras, 2004; Qu et al., 2009).

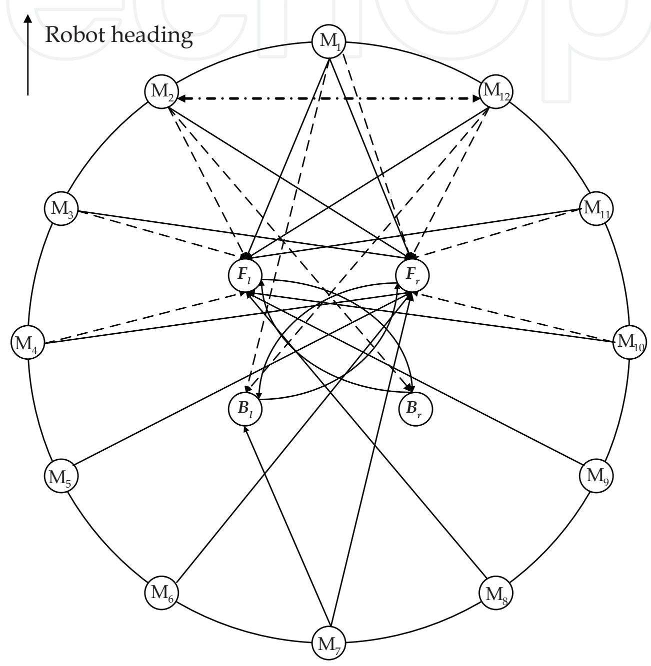
In this chapter, a robot controller based on spiking neural network (SNN) is proposed for the coordinated hunting of multi-robot system. The controller utilizes 12 direction-sensitive modules to encode and process the inputs including the environment, target and coordination information by the time-to-first-spike coding and then, the motor neurons generate the control signals for the motors according to the winner-take-all strategy. Also, the Hebbian learning with a stochastic form is applied to adjust the connection weights.

The rest of the chapter is organized as follows. In Section 2, the structure of the controller is given and the inputs encoding, coordination between robots and motor neurons are described in detail. Section 3 demonstrates the simulations. Finally, Section 4 concludes the work.

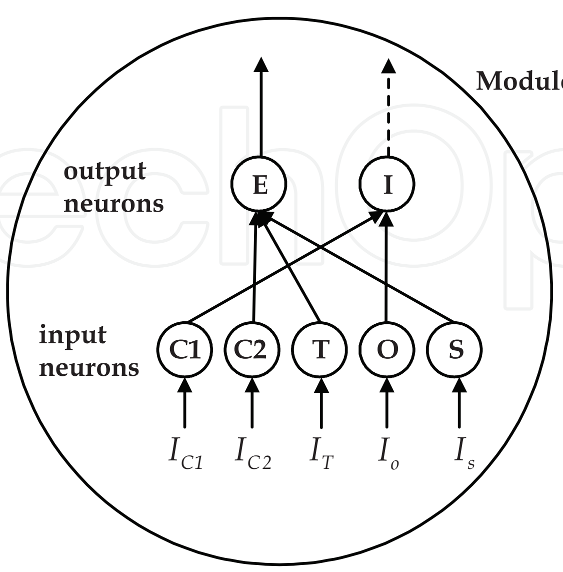
2. The controller based on spiking neural network

2.1 The structure of the controller

The SNN based controller is composed of 12 modules, denoted by $M_i, i = 1, \cdots, 12$, as well as four motor neurons, as shown in Figure 1(a). The module M_i is direction-sensitive and it is



(a) the controller structure



(b) the configuration of the module

Fig. 1. The frame of the controller

corresponding to the direction $\frac{(i-1)\pi}{6}$ with respect to the heading of the robot, such that they are considered to be uniformly distributed around the robot. The module is shown in Figure 1(b), and it consists of two layers. Layer 1 is composed of input neurons {S, O, T, C1, C2} that encode the sensor input I_s, obstacle input I_o, target input I_T, coordination inputs I_{C1} and I_{C2} according to the time-to-first-spike coding, respectively. Layer 2 generates the outputs with two output neurons E (excitory) and I (inhibitory) that represent that the robot should turn to or away from the direction of the module, respectively. In Figure 1(a), the lines denote the connections from the neuron E of every module to the motor neurons, while dashed lines represent the connections from the neuron I to the motor neurons.

The four motor neurons denoted by (F_l, B_l, F_r, B_r) represent left forward motor neuron, left backward motor neuron, right forward motor neuron and right backward motor neuron, respectively. They are divided into two pairs (F_l, B_l) and (F_r, B_r) according to the related motor. There exist connections between two pairs for the information transmission.

The motor neurons corresponding to each motor receive the outputs of the modules along with the outputs of other motor neurons and fire spikes to generate the control signal by the winner-take-all strategy.

2.2 Input signals of the module

The input signals are encoded into corresponding spikes by time-to-first-spike coding and can be formulated as follows:

a. Sensor input I_s

I_s is encoded according to the formula

$$t_s = round(T(1-\frac{I_s}{I_{\lim}}))+1 \tag{1}$$

where t_s is the time when the spike is generated with the constant T=100ms, and the input I_s is the data from the corresponding sensor bounded by an up-bound $I_{\lim}$. Here, we assume that there are 16 ultrasonic sensors evenly distributed on the peripheral ring of the robot and "corresponding" means the sensor is closest to the direction of the module. Also, the sensor data caused by the target is filtered such that the target does not affect the sensors information. The function round (•) adjusts the variable into its nearest integer.

b. Obstacle input I_o

I_o also comes from the corresponding sensor, and it is encoded by

$$t_T = round(\frac{5T}{I_{\lim}}\max(I_o - I_{thred},0))+1 \tag{2}$$

where $I_o = I_s$, and the constant I_{thred} can be considered as the threshold for the obstacle neuron. The function max(•) makes sure that $t_T \geq 1$.

c. Target input I_T

I_T is the relative angle θ_T between the module direction and the line from the target to the robot. It is encoded to the spike time t_T as

$$t_T = round(T(1-\frac{\max(f(\theta_T),0)}{0.8}))+1 \tag{3}$$

with

$$f(x) = 2e^{(-\frac{\pi x^2}{4})} - 1.2e^{(-\frac{\pi x^2}{16})}$$

where t_T only concerns the relative angle and does not care about the relative distance. Obviously, $\max_x(f(x)) = 0.8$ and we divide the function by 0.8 to normalize the result. The function $f(\bullet)$ makes the module sensitive to the target near the direction of the module.

d. Coordination input I_{C1}

I_{C1} is introduced to prevent the robots from being close to each other. It contains two parts: the relative angle θ_C^j between the direction of the module and one detected robot R_j and the relative distance d_C^j between the two robots.

$$t_{C1} = round(T(1 - \min(\max(\sum_j \frac{g(d_C^j)f(\theta_C^j)}{0.8}, 0), 1))) + 1 \quad (4)$$

with

$$g(x) = e^{-(\frac{x}{2500})^2}$$

where $f(\bullet)$ as mentioned above deals with the relative angle θ_C^j while $g(\bullet)$ is responsible for the relative distance d_C^j. Eq. (4) makes the module be more sensitive to the companion robots near its direction.

e. Coordination input I_{C2}

I_{C2} is activated only when there exists at least one neighboring robot R_n whose Boolean attraction tag is 1. Every robot has its own table of all robots' attraction tags and neighboring robots topology. By broadcasting, each robot sends its information including whether it detects the target, its tag and neighbor information such that the tables of robots can be updated. The attraction tag of a robot is updated as follows:

If some neighborships among the robots break or some robots lose the track of the target based on the updated table

```
{
    If there exists a route from it to the target according to the neighboring robots
    topology
    { its tag is 1; }
    Else
    { its tag is 0; }
  }
  Else
  {
  If one tag of its neighbors is 1 or it detects the target
  { the tag is 1; }
  Else
```

{ the tag is 0; }
}
When I_{C2} is activated, it can be encoded as

$$t_{C2} = round(T(1-\min(\max(\sum_n \frac{g(d_C^n - 3000) f(\theta_C^n)}{0.8}, 0), 1))) + 1 \quad (5)$$

where the relative distance d_C^n between the robots is encoded with a translation, as shown in $g(d_C^n - 3000) = e^{-(\frac{d_C^n - 3000}{2500})^2}$.

2.3 The weights and outputs of the module

For a module, the weights between neurons represent the delay from neurons to neurons. Thus, when delay is small, the connection between neurons is considered to be strong. For every module, the initial weights for the neurons in the module are nearly the same, which will be shown in the simulation section. The output neurons E_i and I_i of the module $M_i, i = 1, \cdots, 12$ fire at the time t_E^i and t_I^i (when their first input spikes arrive). t_E^i and t_I^i can be calculated as follows:

$$t_j^i = \min_k (t_k^i + w_{k,j}^i), j = E, I; k = S, O, T, C1, C2 \quad (6)$$

where $w_{k,j}^i$ is the corresponding weight (i.e. delay) from the input neuron k to the output neuron j in the module M_i.
Furthermore, there exist mutual inhibitions between the module M_2 and M_{12} (dot and dash line in Figure 1(a)), which can be calculated as follows:

$$\begin{cases} t_I^{12} = 10T & t_I^2 < t_I^{12} \\ t_I^2 = t_I^{12} = 10T & t_I^2 = t_I^{12} \\ t_I^2 = 10T & t_I^2 > t_I^{12} \end{cases} \quad (7)$$

2.4 The motor neuron

The motor neurons fire at the time $t_i^j, i = r, l; j = f, b$ when their first input spikes arrive. t_i^j is bounded by T. Especially, when there is no input spike during the period (0, T], t_i^j is set to be T. Furthermore, there are mutual connections between the left forward motor neuron F_l and the right backward motor neuron B_r, as well as the right forward motor neuron F_r and the left backward motor neuron B_l. Thus, the information from one side can pass to the other side. The motor neurons are divided into two pairs according to the motor they belong to. The competence (i.e. winner-take-all strategy) is used in each pair to decide whether the motor rotates forward or backward. Then we have

$$v_i = \frac{10\,\mathrm{sgn}(t_i^b - t_i^f)(1 - \min(t_i^f, t_i^b)/T)}{4}, i = r, l \quad (8)$$

where v_i is the velocity of the motor i.

2.5 Learning

For the SNN based controller, the weights are adjusted by the Hebbian learning (Kempter et al., 1999) in a stochastic form. Denote w_{ij} the weight from the neuron j to neuron i. When learning occurs, a random number ς is generated following the uniform distribution in the [0, 1].
If ς is smaller than $|2r_{learn}W(t_i - t_j)|$ with

$$W(\Delta t) = \begin{cases} \exp(\frac{\Delta t}{\tau^{syn}})[(1-\frac{\Delta t}{\tilde{\tau}_+}) - (1-\frac{\Delta t}{\tilde{\tau}_-})] & \Delta t \geq 0 \\ \exp(-\frac{\Delta t}{\tau_+}) - \exp(-\frac{\Delta t}{\tau_-}) & \Delta t < 0 \end{cases} \tag{9}$$

the weight will be adjusted. Here constants $\tau^{syn} = 5$, $\tau_+ = 1$, $\tau_- = 20$. Considering that weights are integers, the change of weight is also an integer. In this controller, the weight will change by 1 or -1, i.e.

$$w_{ij} = \begin{cases} w_{ij} - 1 & \varsigma < |2r_{learn}W(t_i - t_j)|, t_i > t_j \\ w_{ij} + 1 & \varsigma < |2r_{learn}W(t_i - t_j)|, t_i < t_j \\ w_{ij} & otherwise \end{cases} \tag{10}$$

where $r_{learn} = 0.001$ is the learning rate.

3. Simulations

In this part, the simulations will be given to testify the feasibility of the proposed controller based on SNN. The mobile robots are modeled as two-wheel mobile robots and can be seen as circles with a diameter 500mm. The target (evader) is also considered as a circle. The target has the same controller structure as shown in Figure 1(a), and each module has a repulsing input to make it be away from the robots just like I_{C1} with $g(x) = e^{-(\frac{x}{3000})^2}$ while there are no coordination inputs and target input. The parameters of the controller in simulations are as follows: $I_{\lim} = 2000mm$ and the maximum range of detecting other robots is 5000mm; $I_{thred} = 200mm$ for M_1 and $300mm$ for other modules.
The initial weights between the input neurons and the output neurons in the module are as follows: {S→E, O→I, T→E, C1→I, C2→E} are {38,8,13,8,18}, {40,10,15,11,20}, {40, 10,15,13,20}, {40,10,15,16,20}, {40,10,15,10,20}, {40,10,15,10,20}, {40,10,15,10,20}, {40,10,15,10,20}, {40,10,15,10,20}, {40,10,15,16,20}, {40,10,15,13,20}, {40,10,15,11,20} for modules M_1 to M_{12}, respectively. The initial values of the weights connecting the modules to the motor neurons are 10, while the initial values are 4 for the weights of the motor neurons' mutual connections.

In following simulations, the start points of the robots are denoted by "S", while "G" represents the stop points. When the distance between the target and a robot is smaller than 1000mm, the hunting task is considered to be completed.

In simulation 1, we consider three robots to complete the hunting task in an environment which is enclosed by 4 walls (the black line in the Figure 2). The result of simulation 1 is shown in Figure 2, and the states of the robots are shown in Figure 3. Every robot has three states: 0, 1, 2. State 2 represents the target being observed by the robot itself and state 1 depicts that the robot builds a neighborship route with a robot that observes the target while the target is not detected by itself, otherwise, its state is 0. At first, only robot 3 can observe the target. The target is out of the detection range of robot 1 and robot 2. Then, robot 2 will follow robot 3 according to the updated tag table which includes the information that robot 3 can observe the target. After that, robot 1 also follows robot 2 because it finds that the tag of robot 2 is 1. A route from robot 1 to the target in the neighboring robots topology is built. After the robot 2 observes the target by itself, the target is pursued by robots 2 and 3, while robot 1 is still in following. Finally, the target is caught successfully.

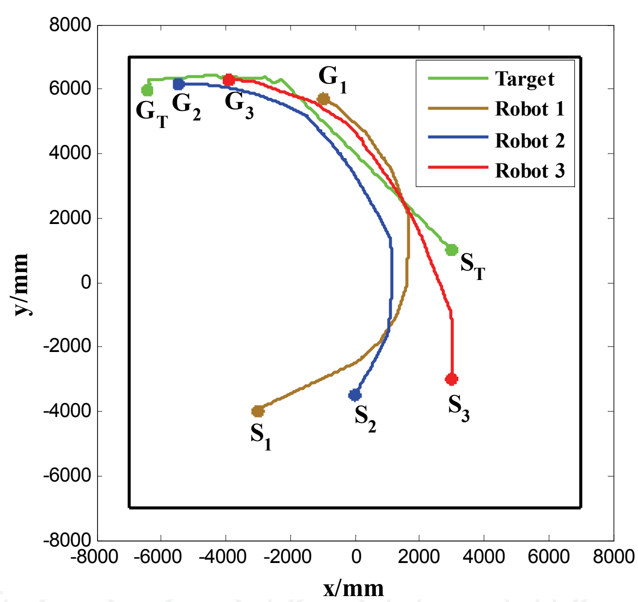
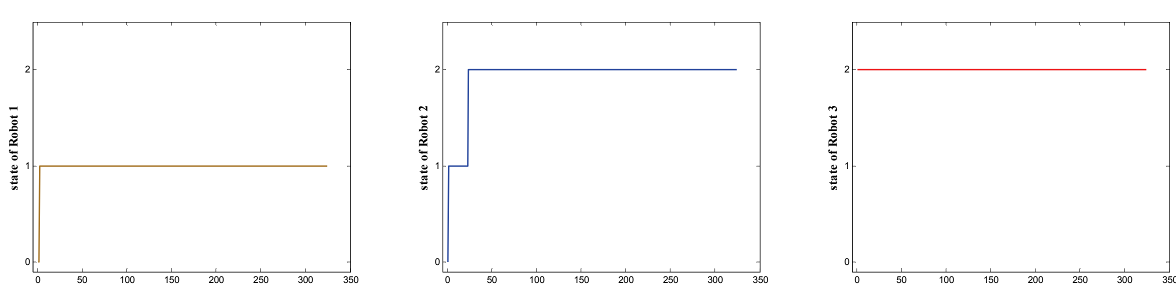


Fig. 2. The trajectories of the robots in simulation 1



(a) (b) (c)

Fig. 3. The states of the robots in simulation 1

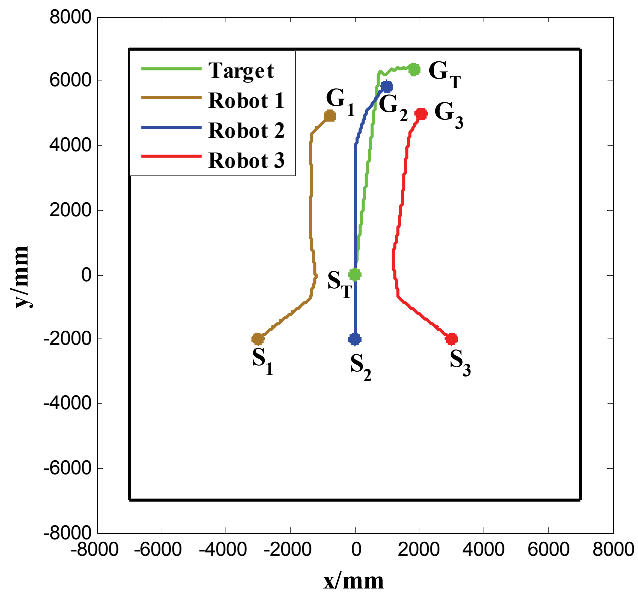


Fig. 4. The trajectories of the robots in simulation 2

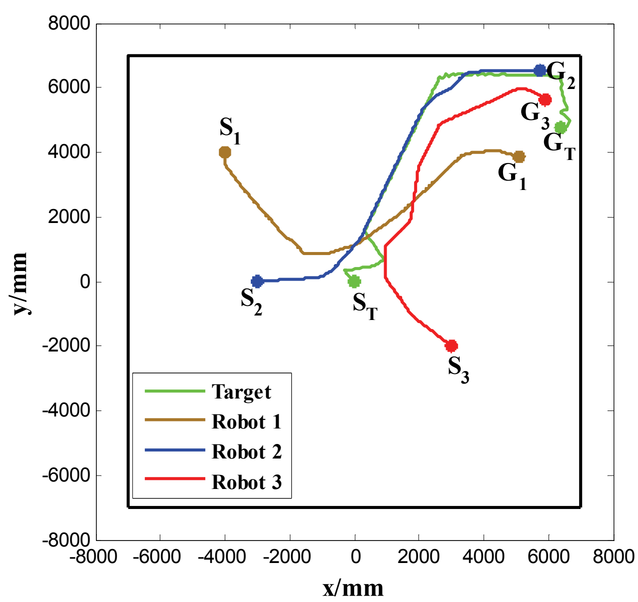


Fig. 5. The trajectories of the robots in simulation 3

In simulation 2, all three robots can observe the target at the beginning. The trajectories of the robots are presented in Figure 4. In simulation 3, the robots 2 and 3 observe the target at the beginning and robot 1 follows robot 2. The corresponding trajectories of the robots and their states are given in Figures 5 and 6, respectively.
The simulation 4 considers the still target with an obstacle between the target and the robots, as shown in Figures 7 and 8.

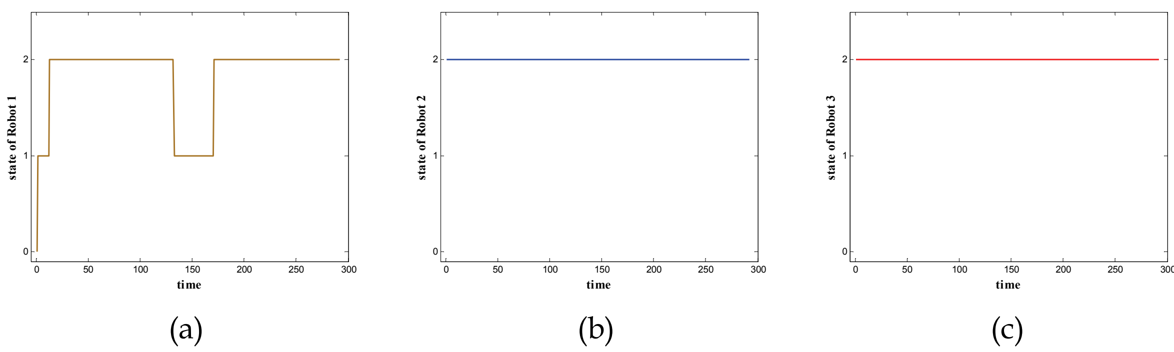


Fig. 6. The states of the robots in simulation 3

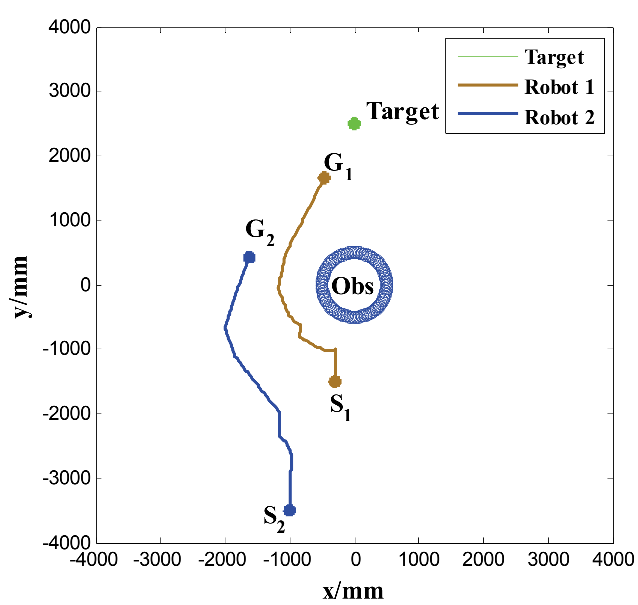
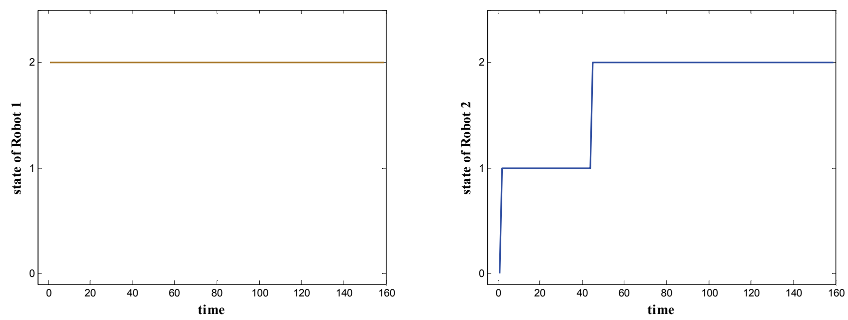


Fig. 7. The trajectories of the robots in simulation 4



(a) (b)

Fig. 8. The states of the robots in simulation 4

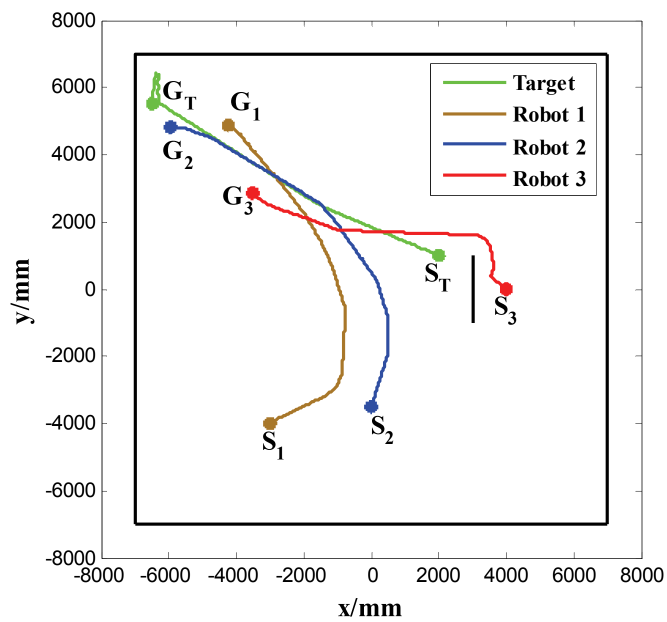
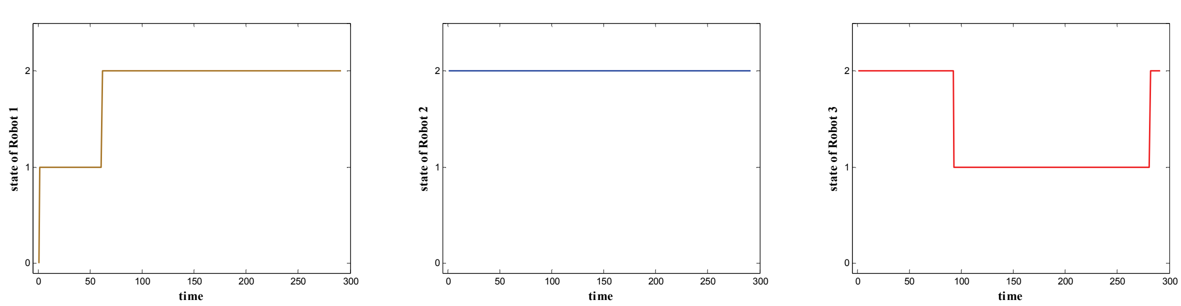


Fig. 9. The trajectories of the robots in simulation 5



(a) (b) (c)

Fig. 10. The states of the robots in simulation 5

In simulations 5 shown in Figures 9 and 10, initially, the robots 2 and 3 observe the target. After robot 1 observes the target, all the robots pursue the target. For robot 3, because of the obstruction of the obstacle, it loses the target and its state is from 2 to 1. Finally, the hunting task is completed.

4. Conclusion

In this work, the controller based on SNN is applied to the hunting task for the multi-robot system. By using time-to-first-spike coding and winner-take-all strategy, the controller with 12 direction-sensitive modules and four motor neurons can make the robots coordinate with each other to finish the hunting task, as demonstrated in the simulations. In this work, the function $f(\bullet)$ is the same for all the modules, which, in fact, may be different for different modules. The will be considered in the future work.

5. Acknowledgement

This work is supported in part by the National Natural Science Foundation of China under Grants 60805038, 60725309, and the National High Technology Research and Development Program of China (863 Program) under Grant 2009AA043901-1.

6. References

Burgsteiner, H.; Kroll, M.; Leopold A. & Steinbauer G. (2005). Movement prediction from real-world images using a liquid state machine, *Proceedings of Lecture Notes In Computer Science, Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence*, pp. 121-130, Bari, Italy

Floreano, D. & Mattiussi, C. (2001). Evolution of spiking neural controllers for autonomous vision-based robots, In: *Evolution Robotics. From Intelligent Robotics to Artificial Life*, Lecture Notes in Computer Science, Vol. 2217, pp. 38-61, Springer, ISBN 978-3-540-42737-7, Berlin / Heidelberg

Hagras, H.; Cornish, A.P.; Colley, M.; Callaghan, V. & Clarke, G. (2004). Evolving spiking neural network controllers for autonomous robots, *IEEE International Conference on Robotics and Automation*, pp. 4620-4626

Joshi, P. & Maass, W. (2005). Movement generation with circuits of spiking neurons, *Neural Computation*, Vol. 17, No. 8, pp. 1715-1738

Kasabov, N. (2010). To spike or not to spikes: probabilistic spiking neuron models, *Neural Networks*, Vol.23, No.1, pp. 16-19

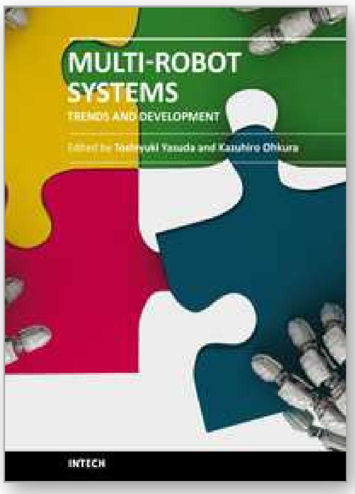
Kiselev, M.V. (2009). Self-organized spiking neural network recognizing phase/frequency correlations, *Proceedings of International Joint Conference on Neural Networks*, pp. 1633-1639, Altanta, Georgia, USA

Maass, W. & Bishop, C. M. (1999). *Pulsed Neural Networks*, MIT-Press, Cambridge, MA

Qu, H.; Yang, S.X.; Willms, A.R. & Yi, Z. (2009). Real-time robot path planning based on a modified pulse-coupled neural network model, *IEEE Transaction on Neural Networks*, Vol. 20, No. 11, pp.1724-1739

Roggen, D.; Hofmann, S.; Thoma, Y. & Floreano, D. (2003). Hardware spiking neural network with run-time reconfigurable connectivity in an autonomous robot, *NASA/DOD Conference on Evolvable Hardware*, pp. 189-198

Kempter, R.; Gerstner, W. & Van Hemmen, J.L. (1999). Hebbian learning and spiking neurons, *Physical Review E*, Vol. 59, No. 4, pp. 4498-4514



Multi-Robot Systems, Trends and Development

Edited by Dr Toshiyuki Yasuda

ISBN 978-953-307-425-2
Hard cover, 586 pages
Publisher InTech
Published online 30, January, 2011
Published in print edition January, 2011

This book is a collection of 29 excellent works and comprised of three sections: task oriented approach, bio inspired approach, and modeling/design. In the first section, applications on formation, localization/mapping, and planning are introduced. The second section is on behavior-based approach by means of artificial intelligence techniques. The last section includes research articles on development of architectures and control systems.