Self-adaptive multi-robot construction using gene regulatory networks

Honglian Guo, Yan Meng, Yaochu Jin

2009

Self-Adaptive Multi-Robot Construction using Gene Regulatory Networks

Hongliang Guo, Yan Meng, and Yaochu Jin

Abstract— **Biological organisms have evolved to perform and survive in a world characterized by rapid changes, high uncertainty, infinite richness, and limited availability of information. Gene regulatory networks (GRNs) are models of genes and gene interactions at the expression level. In this paper, inspired by the biological organisms and GRNs models, a distributed multi-robot self-construction method is proposed. By using this method, a multi-robot system can construct themselves to different predefined shape, and reorganize themselves adaptively under dynamic environments. Various case studies have been conducted in the simulation, and the simulation results show the efficiency and convergence of the proposed method.**

I. Introduction

MULTI-ROBOT systems have drawn considerable attentions to both industry and academia in the last two decades, since it can be used to fulfill tasks that are quite difficult or even unfeasible to be accomplished by a single robot, especially in the presence of uncertainties, incomplete information, distributed control, and asynchronous computation.

In a multi-robot system (MRS), robots can work together more efficiently, robustly, and reliably in those tasks inherently distributed in space, time, or functionality. However, these advantages are at the cost of an increase in system complexity, especially the control and communication complexity [12]. It is often not hard to implement a rudimentary centralized controller that accomplishes the complex tasks using multi-robot systems [11], but achieving optimal performance under unknown dynamic environments can be very challenging. Therefore, distributed methods are more attractive compared to the centralized control methods due to their robustness, flexibility, and adaptivity.

However, designing a distributed self-adaptive multi-robot system is not a trivial task. Nolfi and Floreano [18] claim that, since the individual behavior is the emerging result of the interaction between agents and environment, it is difficult to predict which behavior is resulted from a given set of rules, and which are the rules behind an observed behavior. Similar difficulties are present in the decomposition of the organized behaviors of the whole system into interactions among individual behaviors of the system components. Thus, it is difficult to predict, given a set of individual behaviors, which behavior at the system level will emerge, and it is also difficult to decompose the emergence of a desired global behavior into simple interaction among individuals.

The challenging issues in MRSs also involve the realization of basic behaviors, such as dynamic task allocation, robot coordination, and team reasoning, etc. [28]. Furthermore, MAS is usually bedeviled by the dimensionality when facing large-scale problems. The computation complexity often grows exponentially with the number of agents in multi-agent systems [12]. As an emerging field, multi-agent systems aim at providing both principles for the construction of complex systems involving multiple agents and mechanisms for coordination of independent agent's behaviors [24].

Recently, swarm intelligence has attracted extensive attentions to tackle the scalability issue for multi-agent systems while maintaining system robustness and individual simplicity. Swarm intelligence is an innovative computational and behavioral metaphor for solving problems in a distributed way inspired from the behavior of social insects swarming, flocking, herding, and shoaling phenomena in vertebrates. The social insect colonies are able to build sophisticated structures and regulate the activities of millions of individuals by endowing each individual with simple rules based on local perception. More and more researchers have been applying these swarm intelligence based bio-inspired approaches to solve MRS problems [3, 8, 15, 16, 20, 27].

On the other hand, biological organisms have evolved to perform and survive in a world characterized by rapid changes, high uncertainty, indefinite richness, and limited availability of information [19]. Gene regulatory networks (GRNs) are models of genes and gene interactions at the expression level. It is a collection of DNA segments in a cell which interact with each other indirectly through their RNA, protein product, and other chemicals in the cell, therefore governing the rates at which genes in the network are transcribed into mRNA. GRNs play a central role in understanding natural evolution and development [1]. To this end, various models of GRNs have been suggested [2, 4, 7, 14, 23].

Shen et al. [21] proposed a digital hormone model (DHM) as a bio-inspired distributed control method for robot swarms and self-organization. Essentially, they applied Turing's reaction-diffusion model [26] to describe the interactions between the hormones. The DHM integrated dynamic

H. Guo is a graduate student in the Department of Electrical and Computer Engineering, Stevens Institute of Technology, NJ 07030, USA. (E-mail: hguo@stevens.edu).

Y. Meng is with Department of Electrical and Computer Engineering, Stevens Institute of Technology, NJ 07030, USA (Phone: (201) 216-5496, Email: yan.meng@stevens.edu)..

Y. Jin is with the Honda Research Institute Europe, Carl-Legien-Str. 30, 63073 Offenbach, Germany. (Email: yaochu.jin@honda-ri.de).

network, topological stochastic action selection, and distributed control by hormone reaction-diffusion. More recently, Taylor [25] proposed a gene regulatory network inspired real-time controller for a group of underwater robots. Then a genetic algorithm (GA) was applied to evolve the controller for a simple clustering task.

Inspired by these two algorithms, in this paper, we propose a distributed, GRN-like real-time controller for multi-robot construction. The major advantages of our proposed method compared to the above two methods [21, 25] are: (1) the system's global information, such as the shape function, can be embedded into the GRN dynamics directly; (2) the dynamics of the GRN-like model can automatically drive the robots to their target positions while avoiding collision between the robots and obstacles inside the environment.

The paper is organized as follows. Section II introduces the biological background that inspired this work. We propose the GRN inspired distributed control approach for a multi-robot construction in Section III. To evaluate the proposed method, several case studies of a multi-robot construction are presented in Section IV. Conclusion and future work are discussed in Section V.

II. Biological background

First, let us introduce the basic bio-chemical processes that are involved in gene regulation. When a gene is expressed, information stored in an organism's genome is transcribed and translated into proteins. Some of these proteins are transcription factors that can regulate the expression of their own or other genes. Thus, these proteins are under regulatory control, resulting in complex networks of interacting genes. These gene regulatory networks control a number of important cellular processes including responding to the environment, regulating the cell cycle and guiding the development of an organism.

It is very challenging to gain a thorough understanding of the emergence of complex patterns of behavior from the interactions between genes in a regulatory network. A large number of different GRN models have been suggested. Ordinary differential equations (ODEs) have been used to model the reaction kinetics of regulatory systems with a long history. Biological processes are highly complex, and usually simplification assumptions are needed for most mathematical models of GRNs. The first assumption is that the control of gene expression resides in the regulation of gene transcription. The second is that genes are expressed and proteins produced at a continuous rate [5]. The major advantage using ODE models is that their more detailed representation of regulatory interactions provides a more accurate representation of the physical system under investigation. Furthermore, a large number of dynamical systems theory can provide tools for model analysis. In this section, we will introduce a single-cell GRN model, then a multi-cell GRN model.

A. A Single Cell GRN Model

In a simple single-cell non-spatial biological model currently used in biological systems, a cell consists of one genome and several types of proteins. Genome may consist of several genes that interact with each other through their produced proteins. Each gene has a regulatory region and a structural region. The regulatory region specifies the specific protein that inhibits or activates its expression, while the structural region describes the protein that is produced when the gene is expressed. Here, when a gene is expressed, it means that its expression value is over a certain threshold. Fig. 1 provides an example GRN with 2 genes, where product of gene 1 regulates the expression of gene 2, and the product of gene 2 regulates the expression of gene 1.

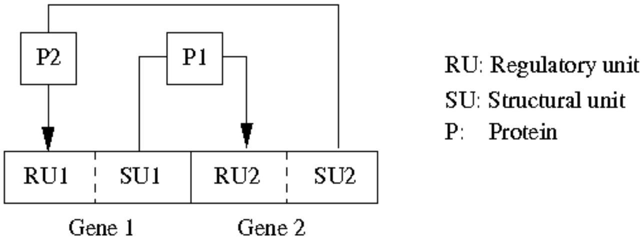


Fig. 1. Illustration of a GRN of a single cell with 2 genes.

According to the central dogma of biology, the expression of a gene with auto-regulation can be described by the following differential equations [9]:

$$\frac{dg_i}{dt} = -\gamma_g g_i + \alpha_g f(p_i) \tag{1}$$

$$\frac{dp_i}{dt} = -\gamma_p p_i + \alpha_P g_i \tag{2}$$

where g_i is the expression level (measured by the concentration of its RNA product) of gene i and p_i is the concentration of protein i. γ_g and γ_P are the decay rate of RNA and protein concentration, respectively. α_g and α_p are the synthesis rate of RNA and protein concentration, respectively. $f(x)$ is a sigmoid function, which can be defined as:

$$f(x) = \frac{\beta}{\theta^n + x^n} \tag{3}$$

where β is the activation coefficient, θ is the threshold, n is known as the Hill coefficient.

B. A Multi-cell GRN Model

In a multi-cell organism, it is necessary to model the intercellular communication. In addition to the internal dynamics of the cell, we should also consider external factors such as protein gradients and physical interactions between cells into the GRNs model. Turing [26] proposed one of the earliest models for pattern formation, where a pair of coupled reaction-diffusion equations was proposed to describe a system consisting of two morphogens. As two morphogens diffuse across a spatial field and react with one another, a variety of patterns emerge depending on parameter values. The gradients of protein concentrations across cells are a critical feature in embryonic development. The reaction-diffusion equations have been widely used in mathematical biology to study pattern formation in development [6, 10, 13].

Salazar-Ciudad et al. [22] proposed a GRN model with reaction-diffusion mechanism as follows:

$$\frac{dx_{ij}}{dt} = f_j(\mathbf{x}_i, \mathbf{u}) - \gamma_i x_{ij} + D_j \nabla^2 x_{ij}, \quad 1 \le i \le n, 1 \le j \le m \qquad (4)$$

where x_{ij} is the concentration of gene product *j* in cell *i*. The first term specifies the production of x_{ij}, the second term is its degradation, and the last term specifies the diffusion component at diffusion rate D_j. f_j is a nonlinear update function of gene product *j*, which is usually defined as a sigmoid function as $f(x) = \frac{1}{1+e^x}$. $\mathbf{u}$ is the vector of external input signals. γ_i is the degradation rate of product *i*. *n* is the number of gene products, and *m* is the number of cells.

III. The Distributed GRN Inspired Control Approach

The objective of the multi-robot construction is to deploy multiple robots uniformly on a predefined two dimensional shape, for example: a circle or a square, through a distributed control approach. Each robot only knows its local information without any global observer.

Analogous to biology systems, in our GRN inspired control model, as shown in Fig. 1, it is assumed that each robot corresponds to a single cell. Within each cell's genome, there are two genes, one for x-position and one for y-position in a 2D environment. Each gene can produce a certain protein. Each protein can provide the following three functions: (1) To regulate the expression of the gene that produced it (i.e. auto regulation); (2) To adjust the robots' behaviors; (3) To be able to diffuse proteins to its neighbors to prevent collision from each other.

Inspired by equations (1)-(4), the system dynamics of the GRN for multi-robot construction are defined as:

$$\begin{aligned} \frac{dg_{x,i}}{dt} &= -a\mathbf{z}_{x,i} + p_{x,i} \\ \frac{dg_{y,i}}{dt} &= -a\mathbf{z}_{y,i} + p_{y,i} \end{aligned} \qquad (5)$$

$$\begin{aligned} \frac{dp_{i,x}}{dt} &= -cp_{i,x} + kf(\mathbf{z}_{i,x}) + bD_{i,x} \\ \frac{dp_{i,y}}{dt} &= -cp_{i,y} + kf(\mathbf{z}_{i,y}) + bD_{i,y} \end{aligned} \qquad (6)$$

where $g_{i,x}$ and $g_{i,y}$ are the expression levels of the *i*th robot's gene for x-position and y-position, respectively. $p_{i,x}$ and $p_{i,y}$ are the concentration of the *i*th robot's proteins for x-position gene and y-position gene, respectively.

In order to embed the predefined 2D shape, which is the global information, into the dynamic equations, we define $f(\mathbf{z}_i)$ as the following sigmoid functions:

$$\begin{aligned} f(\mathbf{z}_{i,x}) &= \frac{1-e^{-\mathbf{z}_{i,x}}}{1+e^{-\mathbf{z}_{i,x}}} \\ f(\mathbf{z}_{i,y}) &= \frac{1-e^{-\mathbf{z}_{i,y}}}{1+e^{-\mathbf{z}_{i,y}}} \end{aligned} \qquad (7)$$

where $\mathbf{z}_{i,x}$ and $\mathbf{z}_{i,y}$ are the gradients along x-axis and y-axis, respectively, of a predesigned function h at the robot's current gene expression level, which are defined as:

$$\mathbf{z}_{i,x} = \frac{\partial h}{\partial g_{i,x}}, \quad \mathbf{z}_{i,y} = \frac{\partial h}{\partial g_{i,y}} \qquad (8)$$

where the predesigned function h is the function of the desired shape where robots are supposed to be deployed uniformly. We can also treat function *h* as the predefined gradient for cell migration. To facilitate the generation of the desired dynamics, we defined h as the square of the desired shape function. For example, if we want to deploy the robots onto a unit circle. The shape function is defined as

$$s(g_{i,x}, g_{i,y}) = g_{i,x}^2 + g_{i,y}^2 - 1 = 0 \qquad (9)$$

Then function h can be defined as

$$h = (g_{i,x}^2 + g_{i,y}^2 - 1)^2. \qquad (10)$$

D_i is defined as the protein diffusion which aims at keeping the robot away from its neighbors. The size of neighborhood varies according to different shapes and different number of robots. In the example of a circular shape, the neighborhood size can be defined as $\frac{2\pi r}{N}$, where *r* is the radius of the circle, and *N* is the total number of robots which are expected to deploy on the circle.

When a robot detects its neighbor, it will receive the protein emitted from that neighbor so that it would keep itself away from that neighbor to avoid collision. After summing all the neighbors' diffused protein together, we have

$$D_{i,x} = \sum_{j=1}^{N_i} D_{i,x}^j, \quad D_{i,y} = \sum_{j=1}^{N_i} D_{i,y}^j \qquad (11)$$

where N_i denotes the number of its neighbors, and $D_{i,x}^j$ and $D_{i,y}^j$ are the diffusions along x-axis and y-axis, respectively, on robot *i* emitted from the neighbor robot *j*, which is defined as

$$D_{i,x}^j = \frac{|g_{i,x} - g_{j,x}|}{\sqrt{(g_{i,x} - g_{j,x})^2 + (g_{i,y} - g_{j,y})^2}} \qquad (12)$$

$$D_{i,y}^j = \frac{|g_{i,y} - g_{j,y}|}{\sqrt{(g_{i,x} - g_{j,x})^2 + (g_{i,y} - g_{j,y})^2}} \qquad (13)$$

Where the directions of and $D_{i,x}^{j}$ and $D_{i,y}^{j}$ are defined as from robot *j* to robot *i* along x-axis and y-axis, respectively.

Initially, the robots are located randomly in a 2D space. By following the dynamic equations defined in Equations (5) and (6), eventually multiple robots can be deployed uniformly on the predefined shape automatically. In other words, the system can be stabilized to an equilibrium state defined by the shape. Essentially, the shape information is the global information, which can be elegantly embedded into the dynamics of each individual robot through function $f(\mathbf{z}_i)$.

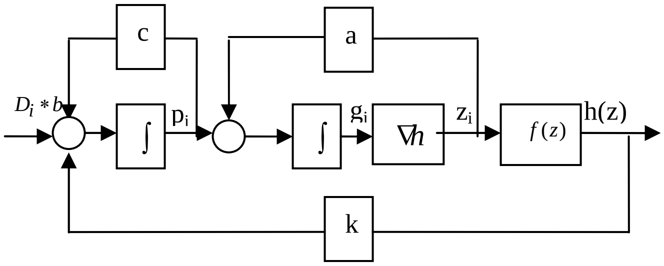


Fig. 2. The diagram of the GRN model.

The diagram of the GRN model defined in Equations (5) and (6) is provided in Fig.5 to facilitate the understanding of the system. From Fig. 2, we can see that other robots' impact on a particular robot is excised through the diffusion of proteins which is implemented by D_i. The robot itself has two state vectors: gene expression g_i and protein concentration p_i. These two variables regulate each other via positive or negative feedback. Through these coherent links, we can achieve the goal of deploying robots to form a certain shape.

In general, each robot always has to balance two motivations: one motivation is to approach to the predefined shape as provided inside the gene expression, and the other one is try to prevent collision from each other as implemented via protein diffusion. Sometimes these two forces may drive the robots to somewhere which is not optimal from a global observer. Since this is a truly distributed control system, there exists no global controller. Therefore, each robot has to make movement decisions only based on its local information. Since it is difficult to achieve an optimal balance, from this point of view, as long as these sub-optimal behaviors are good enough for multi-robot systems, we can say our objective is achieved.

IV. Simulation and Results

To evaluate the reliability and the efficiency of the proposed method, several case studies of a multi-robot construction are studied. We implement all the case studies using MATLAB.

A. Parameter Adjustment

In equation (5) and (6), there are four parameters, i.e., *a*, *c*, *k*, and *b*, which need to be set up for the simulations. Tuning the parameters for the algorithm can speed up the system convergence and assure the system stability.

Initially, we randomly select different combinations of four parameters, and implement the multi-agent construction experiments. Through rough observations, we find out that *a* and *k* shall be equal. Otherwise the system cannot converge. Therefore, in the following experiments, we always set *a*=*k*. By setting up different values to *a* and *k* while keeping other parameters same, we find out that the bigger the value is, more force toward the predefined shape. We also find out that the system convergence time is very sensitive to the values of parameter *c*.

First, we setup parameters as $k=100, b=10, a=100$, and only change the value of *c* for 3-robot construction. The results are listed at the first two columns in Table I. It can be seen that when c=10, the robots can achieve the fastest convergence to their final states. Then, we change parameter *b* while keeping other parameters as $c=10, k=100, a=100$ for 5-robot construction. The results are listed at the third and forth column. It can be seen that when *b*=10, the system can reach the fastest convergence. Last, we only change values of *k (=a)*, while keeping other parameters as *c*=10 and *b*=10 for 5-robot construction. The results are listed on the last two columns in Table I. It can be seen that when k=a=20, the system can achieve the fastest convergence.

TABLE I. Convergence speed (measured by number of iterations) for different parameter setups

c	*Nc*	*b*	*Nc*	*k/a*	*Nc*
50	35605	100	1069	100	537
20	7329	50	689	50	445
15	1053	20	581	25	393
12	697	15	529	20	361
10	589	10	517	18	669
5	661	1	681	15	6033
1	1053	0.1	3789	10	6121

Note: *Nc* is the number of iterations needed to converge to final state.

The next question is how these parameters affect the robot trajectories? Due to page limitations, we only show the results of parameter *c* while keeping others constant. We still use 3 robots to implement our algorithm with different *c*, and draw the trajectories of robots from their initial positions to final position. The simulation results are shown in Fig. 3. From Fig. 3, we can see that when *c*=50, robots only need to travel a short way to converge to the desired shape. While from Table I, we can see that robots need to take quite a long time to finish their short trajectories. In other words, robots are moving very slowly. When *c*=1, robots travels fast, however, their trajectories oscillate very frequently and they spend quite a lot time to converge to the desired shape. When *c*=10, the robots behave somewhere in between. This indicates that it is necessary to tune *c* properly to achieve a good balance between the speed of convergence (time to reach the final position) and the length of the trajectory. From the balance point of view, *c*=10 is a good choice. Similar procedure can be implemented for other parameters as well.

It is a very tedious procedure to adjust these parameters to achieve optimal system performance. Therefore, we plan to apply a genetic algorithm (GA) to automatically tune these parameters, which will be reported elsewhere.

B. Case Study 1: To deploy multiple robots to a unit circle

For this case study, the simulation parameters of equations (5) and (6) are set up as follows: a = 1, c = 1 and k = 1, and b=1.

In this case, we aim onto deploy a number of initially randomly distributed robots to a unit circle. That is, a circle with center at (0, 0) and radius of 1. During the deployment, the robots would try to approach to the unit circle and meanwhile to avoid colliding with each other.

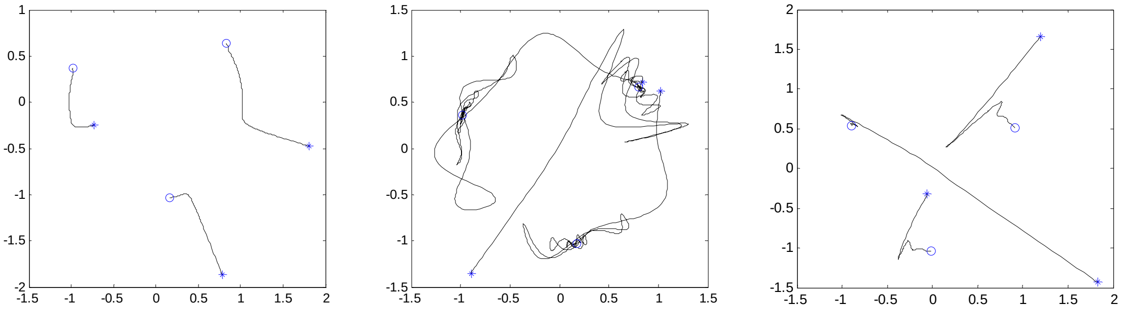
Three groups of robots have been used to evaluate the proposed distributed control algorithms, which contain 5 robots, 10 robots and 20 robots. Due to the page limitation, we only show the simulation results of 20 robots in Fig. 4. The simulation results of 5 and 10 robots are similar to Fig. 4. Initially, robots are randomly located in the environment, as shown in Fig. 4(a). Then robots are trying to converge to the unit circle while avoiding each other, as shown in Fig. 4(b). After certain rounds of simulation, they are uniformly deployed on the circle, as shown in Fig 4(c). From the above simulation results, we can see that any number of randomly initiated robots can ultimately converge to the desired shape uniformly.

Fig. 5 shows the trajectories of robots moving from the initial state to the final state for all three cases with 5, 10, and 20 robots. It can be seen from Fig. 5 that the trajectories of the robots are obviously not optimized. Sometimes some robots have to travel a long journey from their initial position to their final position on the circle. The underlying reason for these scenarios mainly depends on the parameter setup, as we mentioned in parameter adjustment section. When *c*=1, robots move fast but with oscillated trajectories.

In addition, to evaluate our algorithm statistically, Table II lists the mean errors of the final state of the robots compared to the expected final position to the predefined shape. It can be seen that the mean errors are very small. In other words, the proposed GRN inspired dynamic models for multi-robot systems is efficient and feasible.

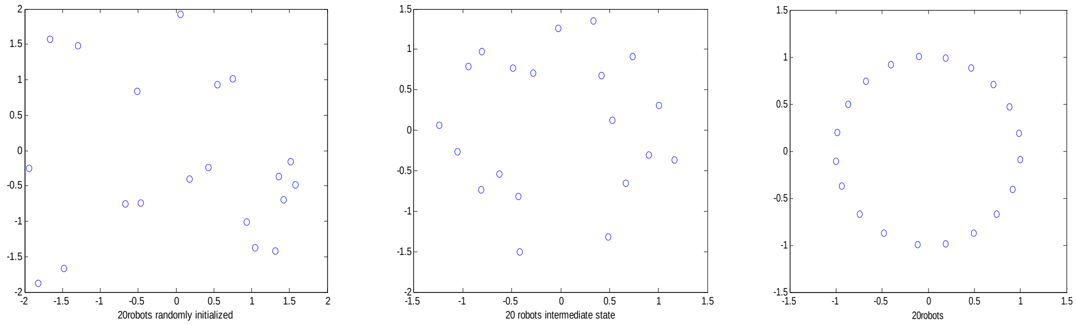
TABLE II
NUMBER OF ROBOTS VS MEAN DIFFERENCE IN CIRCLE DEPLOYMENT

Number of robots in the environment	Mean difference to the expected shape
5 robots	0.0800
10 robots	0.0401
20 robots	0.0301



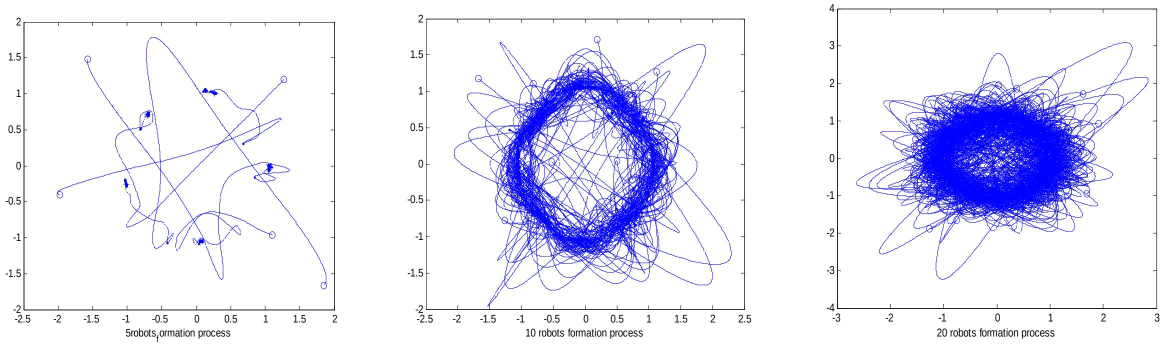
(a) *c*=50; (b) *c*=1; (c) *c*=10.

Fig. 3. Trajectories of 3 robot with different value of parameter *c*.



(a) Initial state of robots; (b) Intermediate state of robots; (c) Final state of robots.

Fig. 4. 20 robots deployed uniformly on a unit circle.



(a)5 robots; (b)10 robots; (c) 20 robots.

Fig. 5. Robots' movement trajectories. (a) 5 robots; (b) 10 robots; (c) 20 robots.

C. Case Study 2: To deploy multiple robots to a unit square

In this case, we plan to deploy a number of initially randomly distributed robots to a unit square. The unit square is defined as follows: lower-left point at (-0.5,-0.5) and the upper-right point at (0.5, 0.5). It is a little bit tricky to define the shape function s of the unit square compared to the unit circle. We first set up a circle with the center at (0, 0) and radius of $\sqrt{\frac{1}{2}}$. Therefore, the shape function of a circle can be defined as: $s_1(g_{i,x}, g_{i,y}) = g_{i,x}^2 + g_{i,y}^2 - \frac{1}{2} = 0$ and $f_1 = (s_1(g_{i,x}, g_{i,y}))^2$. Then, through the first function, we can deploy the robots to the specific circle first, and then define $s_2(x, y)$ according to the following rules:

$$s_2(g_{i,x}, g_{i,y}) = \begin{cases} (g_{i,x} + 1/2) \\ (g_{i,x} - 1/2) \\ (g_{i,y} + 1/2) \\ (g_{i,y} - 1/2) \end{cases}$$

$$\text{if} \begin{array}{lcl} g_{i,x} \le 0 & & -1/2 \le g_{i,y} \le 1/2 \\ g_{i,x} \ge 0 & \text{and} & -1/2 \le g_{i,y} \le 1/2 \\ g_{i,y} \le 0 & & -1/2 \le g_{i,x} \le 1/2 \\ g_{i,y} \ge 0 & & -1/2 \le g_{i,x} \le 1/2 \end{array} \tag{14}$$

$$f_2 = (s_2(g_{i,x}, g_{i,y}))^2 \tag{15}$$

Note that it is impossible that a robot's position satisfies both of the two aforementioned conditions, since after the first phase deployment, the robots are on the circle and will satisfy only one condition as depicted in Fig. 6.

We have tested two groups of robots to deploy them to the unit square with 8 robots and 12 robots, respectively. Fig. 7 shows the simulation results of deploying 12 robots to a unit circle in the environment. The simulation results for 8 robots are similar. Initially, robots are randomly located in the environment, as shown in Fig.7(a). Then robots are trying to converge to the unit square while avoiding each other, as shown in Fig. 7(b). After enough iterations of simulation, they are uniformly deployed on the square, as shown in Fig.7(c). Table III lists the statistics regarding the mean errors of the final state of the robots and the expected final position to the predefined shape. From it, we can see that the mean errors are very small which indicates that our robots under the genetic control can converge to the desired shape.

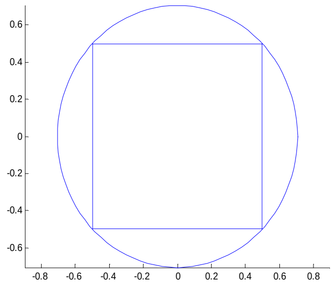


Fig 6: The relationship between the circle and the expected square

TABLE III
NUMBER OF ROBOTS VS MEAN DIFFERENCE IN SQUARE DEPLOYMENT

Number of robots in the environment	Mean difference to the expected shape
8 robots	0.0640
12 robots	0.0317

D. Case Study 3: Self-reorganization

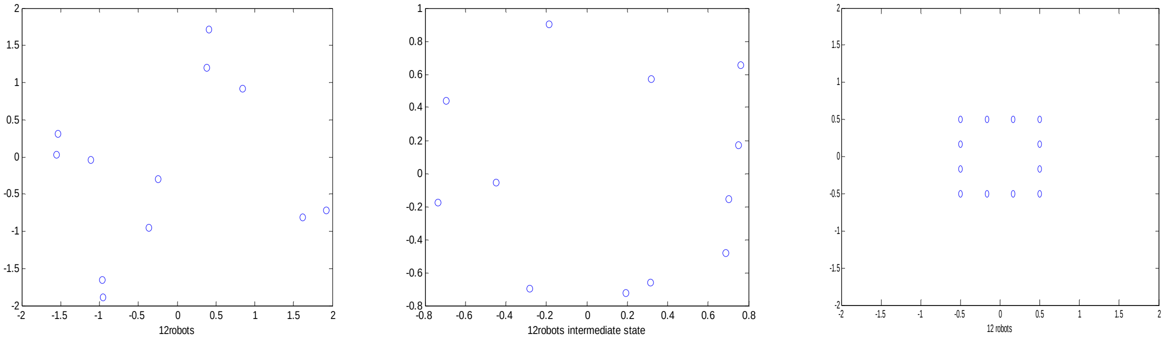
From previous two case studies, we can see that the proposed GRN-based control algorithm can automatically drive multiple robots to a predefined shape. In this case study, we would like to evaluate the system's capability of self-reorganization. More specifically, when new robots join the team, can the original team reorganize themselves to incorporate the newcomers?

For this case study, the simulation parameters of equations (5) and (6) are set up as follows: *a* = *k* =100, *c* =50, and *b* = 10. The main reason to set *c*=50 instead of *c*=10 here is because we want to show the robot trajectories during the self-organization procedure. The robot trajectories with c=50 are much clearer with weaker oscillations, which would make the figure more easy to read, although the convergence speed is relatively slow. Fig. 8 provides the trajectories of robots' initial shape formation as well as self-reorganization to incorporate the new comers. Fig. 8 demonstrates the self-reorganization capability of our proposed approach for various sizes of multi-robot systems.

E. Case Study 4: Self-adaption to environment changes

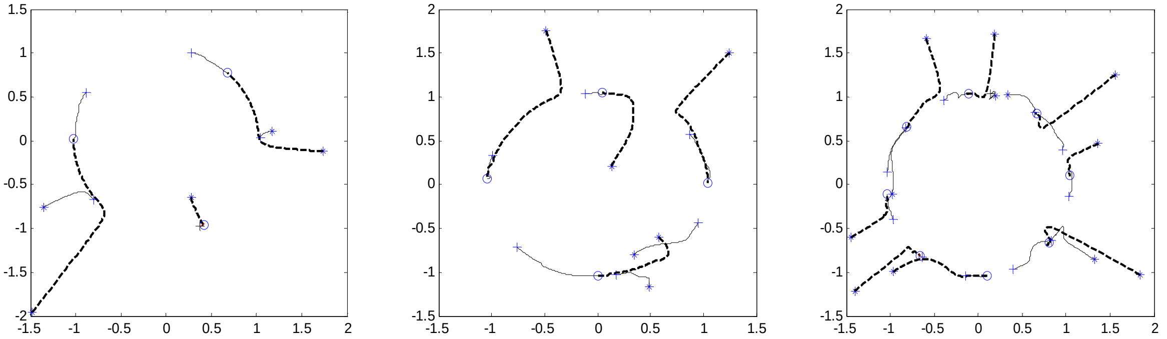
Another interesting test for our algorithm is whether our algorithm can self-adapt to the dynamic changing environment? In the environment, when we setup an obstacle on the perimeter of the desired shape, can robots circumvent the obstacle and continue to form the desiring shape on the rest of the shape using our proposed control method?

For this case study, the simulation parameters of equations (5) and (6) are set up as follows: $a = k = 100$, $c = 50$, and $b = 10$. Fig. 9 shows the simulation results of three different multi-robot systems with one obstacle located on the perimeter of the predefined shape where robots are supposed to be deployed on. Here, the obstacle is modeled as a static robot. The obstacle can emit protein to neighboring robots so that it can influence neighboring robots' behaviors. However, it doesn't receive protein diffusion and it can't move. From Fig. 9, we can see that the multi-robot systems can adapt themselves to the environment change. They are supposed to form a circle, since there is an obstacle, they can circumvent the obstacle and form a circle on the rest of the perimeter.



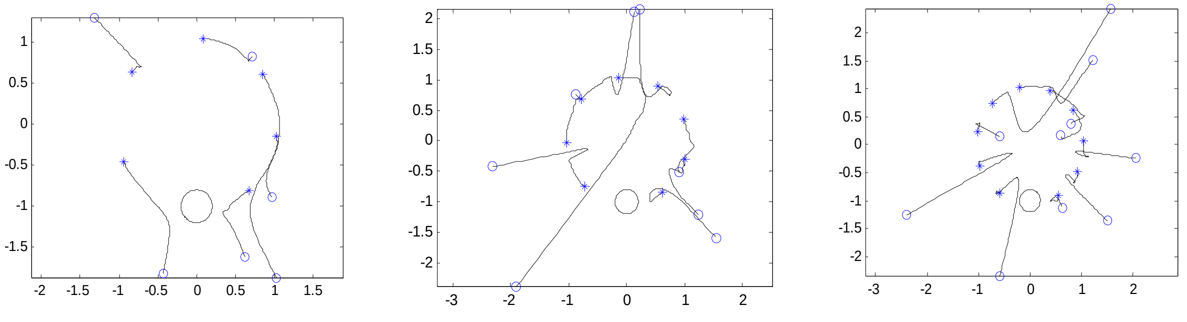
(a) Initial state of robots; (b) Intermediate state of robots; (c) Final state of robots.

Fig. 7: 12 robots deployed uniformly on the unit square



(a) 3 robots with 2 newcomers; (b) 4 robots with 2 newcomers; (c) 8 robots with 4 newcomers.

Fig. 8: Trajectories of multi agents during self-organization with newcomers. The initial positions of the robots are plotted as '*', the intermediate states where the first batch of robots are located are plotted as 'o', and the final states of all the robots are plotted as '+', the dash lines are the initial deployment trajectories of the first batch of robots and the solid lines are the trajectories of all the robots after incorporating newcomers.



(a) 6 robots with one obstacle; (b) 8 robots with one obstacle; (c) 10 robots with one obstacle.

Fig. 9: Trajectories of multi agents in self-adaptive test cases. The big red circle represents one obstacle in the environment. The initial positions of the robots are plotted as 'o', and the final states of robots are plotted as '*'.

V. Conclusion and Future works

In this paper, we have presented a novel GRN inspired distributed control approach for multi-robot construction. Compared to other multi-robot control methods, the major advances of the proposed method are: (1) embedding the global shape information into the dynamic models through a sigmoid function to provide boundaries; (2) truly distributed behaviors of each robot balanced by two different forces: one force to approach to the predefined shape and the other force to avoid collision with other robots. The local interaction among the robots is represented by the reaction-diffusion model inspired from the multi-cellular mechanism of living organisms.

In the future, we will continue with our research on GRNs inspired multi-robot controller. We will investigate arbitrary shape construction using the evolving GRNs model. We will also investigate the system robustness when one or more robots fail.

References

[1] U. Alon, An Introduction to Systems Biology: Design Principles of Biological Circuits. Chapman & Hall/CRC, July 2006.

[2] H. de jong, "Modeling and simulation of genetic regulatory systems: a literature review." Journal of Computational Biology, vol. 9, no. 1, pp. 67-103, 2002.

[3] M. Dorigo, V. Maniezzo, and A.Colorni, "Ant System: Optimization by a Colony of Cooperating Robots," *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics,* Vol. 26, No. 1, February 1996.

[4] D. Endy , and R. Brent 2001. Modelling cellular behavior. Nature 409, 391-395

[5] N. Geard, Modeling gene regulatory networks: systems biology to complex systems, *ACCS technical report*. 2004.

[6] A, Gierer, Generation of biological patterns and form: some physical, mathematical, and logical aspects. Prog. Biophys. Mol. Biol. 37, 1-47.

[7] J. Hasty, D. McMillen, F. Isaacs., and J.J.Collins 2001. Computational studies of gene regulatory networks: In numero molecular biology. Nat. Rev. Genet. 2, 268-279.

[8] W. Jatmiko, K. Sekiyama, and T. Fukuda, "A PSO-based Mobile Robot for Odor Source Localization in Dynamic Advection-Diffusionwith Obstacles Environment: Theory, Simulation and Measrument", *IEEE Computational Intelligence Magazine*, May 2007. pp.37-51.

[9] Y. Jin and B. Senthoff, Evolving in silico Bistable and Oscillatory Dynamics for Gene Regulatory Network Motifs, *IEEE/CEC Congress on Evolutionary Computation*, pp. 386-391, Hongkong, China, 2008.

[10] S. A. Kauffman, The origins of order: self-organization and selction in Evolution, Oxford University Press, New York, 1993.

[11] B. Khoshnevis and G. A. Bekey, "Centralized sensing and control of multiple mobile robots", Computers in Industrial Engineering, 35(3-4). Pp.503-506. 1998.

[12] E. Klavins, "Communication Complexity of Multi-Robot Systems" *Springer Tracts in Advanced Robotics,* Volume 7/2003 page 275-292.

[13] P. K. Maini, K. J. Painter, and P. C. Nguyen. Spatial pattern formation in chemical and biological systems. J. Chem. Soc., Garaday Trans. 93(20), 3601-3610.

[14] H.H. McAdams , and A.Arkin 1998. Simulation of prokaryotic genetic circuits. Ann. Rev. Biophys. Biomol. Struct. 27, 199-224.

[15] Y. Meng and J. Gan, LIVS: Local interaction via virtual stigmergy coordination in distributed search and collective cleanup, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007), San Diego, CA, USA*.

[16] Y. Meng and J. Gan, A Distributed Swarm Intelligence based Algorithm for a Cooperative Multi-Robot Construction Task, *IEEE Swarm Intelligence Symposium*, 2008. St. Louis, Missouri, Sept. 21-23, 2008.

[17] E. Mjolsness, D. H. Sharp, and J. Reinitz. A connectionist model of development. Journal of Theoretical Biology, 152:429-453, 1991.

[18] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines.* MIT Press, Cambridge, MA, 2000.

[19] R. Pfeifer, M. Lungarella, and F. Iida, Self-organization, embodiment, and biologically inspired robotics, *Science*, Vol. 318, November 2007.

[20] J. Pugh and A. Martinoli, Inspiring and Modeling Multi-Robot Search with Particle Swarm Optimization, *Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 20*07). 2007.

[21] W. Shen, P. Will and A. Galstyan, Hormone-Inspired Self-Organization and Distributed Control of Robotic Swarms, *Autonomous Robots*, 17, pp.93-105, 2004.

[22] I. Salazar-Ciudad, H. Garcia-Fernandez, and R. V. Sole. Gene networks capable of pattern formation: from induction to reaction-diffusion. Journal of Theoretical Biology, 205:587-603, 2000.

[23] P. Smolen, D.A.Baxter and J.H.Byrne 2000. Modeling transcriptional control in gene networks: Methods, recent results, and future directions. Bull. Math. Biol. 62, 247-292.

[24] P. Stone, M. Veloso "Multiagent Systems: A Survey from a Machine Learning Perspective." *Autonomous Robots*, 8(3):345–383, July 2000.

[25] T. Taylor (2004), A Genetic Regulatory Network-Inspired Real-Time controller for a Group of Underwater Robots, *Proceedings of Eighth Conference on Intelligent Autonomous Systems (IAS-8)*, 2004.

[26] A. M. Turing, The chemical basis of morphogenesis, *Philos Trans. R. Soc*. London B, 237. 1952.

[27] J. Werfel, Building blocks for multi-robot construction. *7th International Symposium on Distributed Autonomous Robotic Systems (DARS 2004).* June 23-25, 2004. Toulouse, France.

[28] E. Yang and D. Gu, "Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey". *Technical Report CSM-404*, Department of Computer Science, University of Essex, 2004