Linear Dynamic Controllers Evolved by Genetic Regulatory Network Based Artificial Cells

Navid Rahbari Asr, Vahid Johari Majd, Majid Hassan Zadeh Shojai, and Mehdi Behnam

Abstract— **In this paper a linear representation of a synthetic genetic regulatory network (GRN) model is derived and it is used for evolving linear dynamic controllers for nonlinear systems. A case study is considered in which running the genetic algorithm on the elements of the system matrix of a linear controller is unable to evolve and reach the control ends, while running the genetic algorithm on the genes of an artificial cell with linear regulatory networks evolves and a linear controller is achieved. This justifies the computational burden imposed on computations due to GRN dynamics as GRN representation increases the evolvability of the controller.**

***Index Terms*—Genetic regulatory networks, linear artificial cells, linear dynamic controller, genetic algorithms, evolvability.**

I. Introduction

Biological cells' basic control structure is governed by Genetic Regulatory Networks (GRNs). These networks consist of genes, proteins and metabolites that interact with each other and control the cell's behavior [1], [2]. Modeling these structures has great value from two different aspects. On one hand it can help researchers to understand the underlying interconnection of cell functions and ables them to investigate the cell dynamics within cheap, in silico experiments rather than expensive in vitro experiments. On the other hand as GRNs show properties such as adaptability and robustness, artificial systems modeling genetic regulatory networks can be helpful for engineers to design GRN inspired controllers that show the same properties. In this usage they might have a potential to be as handy as some of the previously developed nature inspired tools like Neural Networks, Genetic Algorithms and Fuzzy logic.

Thus far many researches have been done in modeling GRNs and using them to evolve controllers [3]-[9]. In our research first a linear model of GRN is introduced which is inspired by the BioSys and XBioSys models by neglecting nonlinearities such as saturation and protein diffusions from membrane to environment and vice versa [6],[10]. Section II is dedicated to introducing the artificial cell model. In Section III the application of the artificial cell model as a controller is explained and then the linear state space equations of the model are derived. Further in Section IV a case study is investigated for evolving linear dynamic controllers for which the simple genetic algorithm on the elements of the system matrix of the controller fails to evolve while the linear artificial cell successfully evolves and reaches a linear controller. This shows the potentiality of the linear artificial cell in representing an evolvable formulation of linear dynamic controllers.

Manuscript received September 9, 2012, revised November 7, 2012.

The authors are with the Department of Electrical Engineering, Tarbiat Modares University, Tehran, Iran (e-mail: navidrahbariasr@gmail.com, majd@modares.ac.ir, majid_ee82@yahoo.com, m.behnam@modares.ac.ir).

II. Artificial GRN Model

In this section a model of the synthetic cell is explained which is an inspiration of the BioSys and XbioSys model in [6], [10]. The model differs with BioSys model in neglecting nonlinearity sources such as saturation and diffusion.

A. Genome and Genes:

All the proteins are arrays of size PD (Protein Dimension). The genome is a bit string of size GS (Genome Size) .A promoter is a string of size PR with a fixed structure which defines the starting of a gene. Right after the promoter, the regulatory part of the gene begins. The first $2\times RD$ (Regulatory Dimension) bits of the regulatory part define how many enhancer and inhibitor sites the gene possesses. Let's show them with n_A and n_I respectively. After this part of the regulatory site, the first $n_I \times PD$ bits define inhibitory sites and the first $n_E \times PD$ bits after inhibitory sites define enhancer sites. After the regulatory site, a set of bits define the number of proteins the gene expresses (np). The number of these bits is fixed and equals to a universal constant GD (Gene Dimension). After theses, a sequence of bits defines the proteins to be expressed by the gene plus their default expression values. The protein defining bits are of size PD and expression value defining bits are of size ED (Expression dimension). All the bits in the genome after a gene ends are meaningless unless they are located after a promoter string which would further define another gene.

B. Dynamic of the Cell

When a gene is expressed it releases proteins to the cytoplasm. There are different kinds of proteins in the cytoplasm with different concentrations. Some amount of proteins diffuse to out of the cell (environment), some amount decay and some amount act as transcriptional factors and bind to regulatory sites of the genes and increase or decrease their rate of expression. Assume that there are N_p different kinds of proteins in the cytoplasm $p_1, p_2, ..., p_{N_p}$ with concentrations $c_1, c_2, ..., c_{N_p}$ and there are N_g genes $g_1, g_2, ..., g_{N_g}$ that interact with these proteins. We define following terms:

$$g_i(j) = \text{the jth protein produced by gene i} \tag{1}$$

$$n_p(i) = \text{number of proteins in the expression region of gene i} \tag{2}$$

Assume that function f_{ij} determines how much of protein j sticks to regulatory sites of gene i .Inhibitor and enhancer signals for gene i caused by protein j would be computed as relations (1) and (2) in which the function $dmatch$ computes the degree of match between two proteins which is the Hamming distance of the two sets of bits. It is the number of identical bits of two proteins divided by the size of the protein. Terms $in_i(k)$ and $en_i(k)$ are the k th site of inhibitor and enhancer sections of gene i respectively. Number of inhibitor and enhancer sites of gene i are shown by n_{I_i} and n_{A_i} .

$$in_{ij} = \frac{1}{n_{Ii}} \sum_{k=1}^{n_{Ii}} f_{ij}(C_j) \times \alpha \times e^{\beta \times dmatch\ (P_j, in_i(k))} \tag{1}$$

$$en_{ij} = \frac{1}{n_{Ej}} \sum_{k=1}^{n_{Ej}} f_{ij}(C_j) \times \alpha \times e^{\beta \times dmatch\ (P_j, in_i(k))} \tag{2}$$

The total inhibitor and enhancer signals for a single gene i are computed as relations (3) and (4):

$$en_i = \frac{1}{N_p} \sum_{j=1}^{N_p} en_{ij} \tag{3}$$

$$in_i = \frac{1}{N_p} \sum_{j=1}^{N_p} in_{ij} \tag{4}$$

The expression rate of j th protein of gene i at time $t+1$, $re_{ij}(t+1)$, is computed by relation (5) .

$$re_{ij}(t+1) = re_{ij}(t) + en_i(t) - in_i(t) \tag{5}$$

We define ξ_{imj} as (6) .

$$\xi_{imj} = \begin{cases} 1 & g_i(m) = j \\ 0 & otherwise \end{cases} \tag{6}$$

Protein j 's concentration in cytoplasm at time $t+1$ would be computed by relation (7) .

$$C_j(t+1) = \psi \times C_j(t) + \lambda \sum_{i=1}^{N_g} \sum_{m=1}^{np(i)} \xi_{imj} \times re_{im}(t+1) + inp_j(t) - outp_j(t) \tag{7}$$

In the above relation, terms ψ and λ are universal constants determining decay rate and expression effect respectively. The amounts of the protein which goes out of the cell and enters the cell at instant t are shown by $outp_j(t)$ and $inp_j(t)$ respectively.

III. Applying the Artificial Cell as a Controller

In the previous section we explained the artificial cell model and its dynamics. In this section we want to show how this model can be used as a controller. First of all the inputs and outputs of the controller should be determined.

A. *Inputs and Outputs of the Cellular Controller*

All the information passes through cellular environment by the means of proteins. Thus the inputs and outputs of the cellular controller must be proteins too. To adapt a cellular network as a controller, at first the input and output proteins should be attributed. Moreover a predefined method should be chosen to transform the proteins to system signals and vice versa. The necessary information from the system enters to the cellular environment by means of special input proteins. This causes the cell to start developing. The proteins and their concentration in the cellular environment change due to cellular development. Then special output proteins are transformed to system appropriate signals and in this way the system receives signals from the controller. The following figure depicts the process.

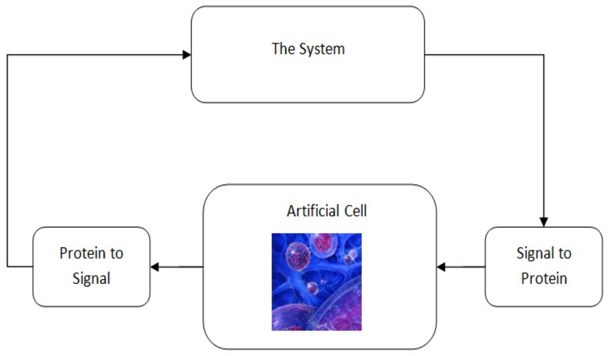

Fig. 1. Artificial cell as a controller.

B. *Evolution of the Controller*

Once the input and output signals are attributed to specific proteins, the artificial cell evolves to meet the control problem goals which are represented in an objective function. The genetic algorithm creates different populations of genomes and passes them to the objective function which defines the control problem. The objective function returns a value which determines how well the artificial cell could control the system. Genetic algorithm chooses the best cells of a population as parents and generates the new population. This process is continued until a proper artificial cell is found.

C. *Representing the Artificial Cell as a Linear Controller*

We assume that the proteins which act as input to the controller enter directly to the cytoplasm of the cell and the output signals are directly and linearly computed from the cytoplasm proteins. The states of these cells are concentrations of the proteins and their expression rates by genes. Due to the fact that the external signals are directly affected to the cytoplasm of the cell and output signals are directly resulted by cytoplasm proteins:

$$outp_j(t) = 0 \quad for\ all\ proteins \tag{8}$$

Thus and according to (7) :

$$C_j(t+1)=\psi\times C_j(t)+\lambda\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\xi_{imj}\times re_{im}(t+1)+inp_j(t) \tag{9}$$

Due to (5) we have relation (10) .

$$C_j(t+1)=\psi\times C_j(t)+\lambda\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\xi_{imj}\times\{re_{im}(t)+en_i(t)-in_i(t)\}+inp \tag{10}$$

By applying (3) and (4) we have relation (11) .

$$\begin{aligned}C_j(t+1)&=\psi\times C_j(t)\\&+\lambda\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\xi_{imj}\times\{re_{im}(t)+\frac{1}{N_p}\sum_{l=1}^{N_p}en_{il}(t)-\frac{1}{N_p}\sum_{l=1}^{N_p}in_{il}(t)\}\\&+inp_j(t)\end{aligned} \tag{11}$$

Further by using relations (1) and (2) , relation (12) is reached.

$$\begin{aligned}&C_j(t+1)=\psi\times C_j(t)+\lambda\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\xi_{imj}\{re_{im}(t)\}+\\&\lambda\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\{\frac{1}{N_p}\sum_{l=1}^{N_p}\frac{1}{n_{Ei}}\sum_{k=1}^{n_{Ei}}f_{il}(C_l(t))\times\alpha\times e^{\beta\times dmatch\ (P_l,en_i(k))}\\&-\frac{1}{N_p}\sum_{l=1}^{N_p}\frac{1}{n_{Ii}}\sum_{k=1}^{n_{Ii}}f_{il}(C_l(t))\times\alpha\times e^{\beta\times dmatch\ (P_l,in_i(k))}\}\\&+inp_j(t)\end{aligned} \tag{12}$$

By taking $f_{il}(C_i)$ a linear function of C_i , and organizing the relation (12), we have relation (13).

$$\begin{aligned}&C_j(t+1)=\psi\times C_j(t)+\lambda\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\xi_{imj}\times re_{im}(t)\\&\frac{\lambda}{N_P}\sum_{l=1}^{N_p}\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\xi_{imj}\alpha'_{il}\{\frac{1}{n_{Ei}}\sum_{k=1}^{n_{Ei}}e^{\beta\times dmatch\ (P_l,en_i(k))}\\&-\frac{1}{n_{Ii}}\sum_{k=1}^{n_{Ii}}e^{\beta\times dmatch\ (P_l,in_i(k))}\}C_l(t)\\&+inp_i(t)\end{aligned} \tag{13}$$

By defining η_{jl} as relation (14) we reach relation (15) .

$$\begin{aligned}\eta_{jl}=&\frac{\lambda}{N_P}\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\xi_{imj}\alpha'_{il}\{\frac{1}{n_{Ei}}\sum_{k=1}^{n_{Ei}}e^{\beta\times dmatch\ (P_l,en_i(k))}\\&-\frac{1}{n_{Ii}}\sum_{k=1}^{n_{Ii}}e^{\beta\times dmatch\ (P_l,in_i(k))}\}\end{aligned} \tag{14}$$

$$\begin{aligned}&C_j(t+1)=\psi\times C_j(t)+\lambda\sum_{i=1}^{N_g}\sum_{m=1}^{np(i)}\xi_{imj}\{re_{im}(t)\}\\&+\sum_{i=1}^{N}\eta_{jl}C_l(t)+inp_i(t)\end{aligned} \tag{15}$$

Relation (15) is the update rule for protein concentrations in cytoplasm. A similar procedure leads to relation (16) .

$$\begin{aligned}re_{im}(t+1)=re_{im}(t)+&\sum_{l=1}^{N_p}\alpha'_{il}\{\frac{1}{n_{Ei}}\sum_{k=1}^{n_{Ei}}e^{\beta\times dmatch\ (P_l,en_i(k))}\\-&\frac{1}{n_{Ii}}\sum_{k=1}^{n_{Ii}}e^{\beta\times dmatch\ (P_l,in_i(k))}\}C_l(t)\end{aligned} \tag{16}$$

By defining χ_{il} as (17) we reach (18) which is the update rule for protein expression rates.

$$\chi_{il}=\alpha'_{il}\{\frac{1}{n_{Ei}}\sum_{k=1}^{n_{Ei}}e^{\beta\times dmatch\ (P_l,en_i(k))}-\frac{1}{n_{Ii}}\sum_{k=1}^{n_{Ii}}e^{\beta\times dmatch\ (P_l,in_i(k))}\} \tag{17}$$

$$re_{im}(t+1)=re_{im}(t)+\sum_{l=1}^{N_p}\chi_{il}C_l(t) \tag{18}$$

State space equations of the artificial cell are given by relations (15) and (18). The changing parameters due to evolution are η_{il} and χ_{il} .So once the genome of the proper artificial cell is obtained via evolution, the η_{il} and χ_{il} s are computed and the linear controller's system matrix is achieved.

D. The Application of Cell Based Method to Get Linear Controllers

Obviously when the system under control is linear, using the cell based method to obtain a linear controller would not be suitable as there are efficient linear, direct methods to achieve desired linear controllers for linear systems. The method shows its value when it is applied to nonlinear complex systems for which there isn't a straight forward method to obtain linear controllers and methods differ from case to case.

A natural question that arises is why to impose the computational burden of the artificial cell to evolve a linear controller while the controller may be evolved using a simple genetic algorithm ran on its parameters, i.e. if our target is to evolve a controller with state equations of type (19) why not run the genetic algorithm on the elements of matrix A directly and eliminating the computational complexity associated with the artificial cell dynamics. (Note that matrices B and C are defined initially for the problem as we have attributed how inputs affect protein concentrations and how output of the controller is determined from the concentrations of proteins.).The answer to this question lies in what is called the representation problem and evolvability in Genetic Algorithm methods. In biological terms evolvability is an organism's capacity to generate heritable phenotypic variation [11]. In the case study section we show that a simple genetic algorithm on the parameters of the system matrix of the controller is unable to evolve and reach an answer, while the artificial cell representation evolves and reaches a suitable controller that meets the problem goals.

$$\begin{aligned}X_{k+1}&=AX_k+BY_k\\U_k&=CX_k\end{aligned} \tag{19}$$

IV. Case Study

In this part we want to develop a linear dynamic controller for a robot so that it can move in its path and avoid hitting any obstacle or walls during its travel. The robot is a simple two dimensional entity that can move in one of four directions forward, backward, up and down or a mixed of them in each time unit. The obstacles are vertical lines of arbitrary size scattered in robot's path. Sensors with limited ranges are placed on the robot to diagnose the obstacles and walls. An extra sensor is placed to sense the passing of the robot from an obstacle. The inputs of the controller are distances from the nearest ahead obstacle and upper and down walls. Outputs of the controller are four actuating signals for the robot. Each signal moves the robot in one of the four directions in an amount commensurate with the intensity of the signal. Fig. 2 shows the field with obstacles.

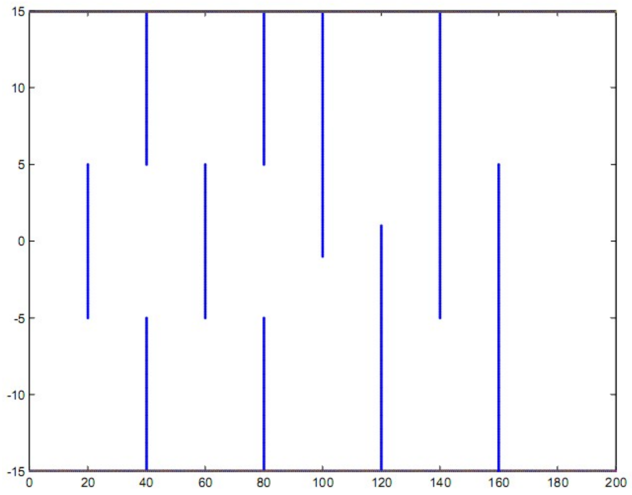
Fig. 2. Field with obstacles.

The robot is initially positioned at point [0, 0] and is wanted to pass through the left of the field to the right of the field without hitting the obstacles. If we take the robot and field as the open loop system and position of the robot and distance from nearest obstacle as the states of the system, the state space equations of this system would be as equations (20) in which x and y stand for horizontal and vertical positions of the robot, s_1 stands for obstacle sensor output and s_2 , s_3 for upper and down wall sensor outputs. The output of the sensor which detects passing of a robot from an obstacle is shown by s_{pass} . It is assumed that obstacle sensor range is 20 units and obstacles farther than 20 units are not recognized by the sensor. The sensor's output is reversely proportional to the distance of the robot with obstacle or walls.

From equations (20) it is seen that nonlinearity of the state space equations appear in the output equations of obstacle, wall and pass obstacle sensors. Our aim is to design a dynamic linear controller that can lead the robot from left to right without hitting obstacles.

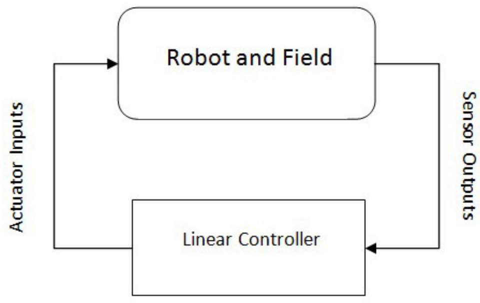

Fig.3. A linear controller for robot and field system

$$
\begin{aligned}
&x_{k+1} = x_k + u_1 - u_2\\
&y_{k+1} = y_k + u_3 - u_4\\
&s_{1k} = \begin{cases}
\dfrac{1}{20-x_k}+0.01 & x_k \le 20 \wedge -5 \le y_k \le 5\\
0.01 & x_k \le 20 \wedge \{y_k \le -5 \vee y_k \ge 5\}\\
0.01 & 20 \le x_k \le 40 \wedge -5 \le y_k \le 5\\
\dfrac{1}{40-x_k}+0.01 & 20 \le x_k \le 40 \wedge \{y_k \le -5 \vee y_k \ge 5\}\\
\dfrac{1}{60-x_k}+0.01 & 40 \le x_k \le 60 \wedge -5 \le y_k \le 5\\
0.01 & 40 \le x_k \le 60 \wedge \{y_k \le -5 \vee y_k \ge 5\}\\
0.01 & 60 \le x_k \le 80 \wedge -5 \le y_k \le 5\\
\dfrac{1}{80-x_k}+0.01 & 60 \le x_k \le 80 \wedge \{y_k \le -5 \vee y_k \ge 5\}\\
\dfrac{1}{100-x_k}+0.01 & 80 \le x_k \le 100 \wedge -5 \le y_k \le 15\\
0.01 & 80 \le x_k \le 100 \wedge -15 \le y_k \le -5\\
\dfrac{1}{120-x_k}+0.01 & 100 \le x_k \le 120 \wedge -15 \le y_k \le 5\\
0.01 & 100 \le x_k \le 120 \wedge 5 \le y_k \le 15\\
0.01 & 120 \le x_k \le 140 \wedge -15 \le y_k \le -5\\
\dfrac{1}{140-x_k}+0.01 & 120 \le x_k \le 140 \wedge -5 \le y_k \le 15\\
\dfrac{1}{160-x_k}+0.01 & 140 \le x_k \le 160 \wedge -15 \le y_k \le 5\\
0.01 & 140 \le x_k \le 160 \wedge 5 \le y_k \le 15
\end{cases}\\
&s_{2k} = \frac{1}{15-y_k}+0.01\\
&s_{3k} = \frac{1}{y_k-15}+0.01\\
&s_{pass_k} = \begin{cases} 1 & robot\ \ passed\ an\ \ obstacle\\ 0 & otherwise \end{cases}
\end{aligned}
\tag{20}
$$

A. Control Problem Representation as an Optimization Problem

For each controller the number of the obstacles the robot can pass successfully is chosen as the fitness value. For the given configuration the maximum fitness value is eight. So finding a suitable controller is equivalent to finding a controller with maximum fitness value.

B. Genetic Algorithm Directly on System Matrix Elements

The three obstacle and wall sensor outputs are taken to be three states of the controller. The fourth sensor's output is used to reset the state space of the controller to its initial value whenever a robot passes an obstacle. Four more states of the controller are assigned to determine the four actuator inputs of the robot. So the controller has at least seven states. Equations (21) show the controller state space equations.

From equations (21) the unknown is matrix A .We ran genetic algorithm on the elements of matrix A for 100 times with different sizes, each for 500 generations and 50 population size. None of the trials were able to find a matrix A to lead the robot to pass even one obstacle.

C. Genetic Algorithm on the Artificial Cell

The three obstacle and wall sensors are attributed to three protein concentrations of a linear artificial cell. Four more proteins are needed to determine the four actuator signals of the robot. The linear cell model is used for updating proteins. Protein Dimension are taken to be three so that we have $2^3=8$ different proteins. Regulatory dimension, Gene Dimension and Expression Dimension are all taken as three as well. The promoter is taken to be the string 1000.The genome of the cell is chosen to be of 500 bits size and is put to evolution with the fitness function described in section A. The genetic algorithm is run for ten times, each for 500 generations with 50 population size. Mutation generates 20% of the new populations and the rest is formed by cross over. Out of ten runs of the GA, four were able to evolve a controller to lead the robot through all eight obstacles without hitting them. Two were able to pass the robot through seven obstacles, three were able to pass it through five obstacles and the remaining one was able to pass the robot through four obstacles. Fig. 4 shows the pass of the robot controlled by the four successful runs of GA.

$$\begin{bmatrix} x_{1k+1} \\ x_{2k+1} \\ x_{3k+1} \\ x_{4k+1} \\ x_{5k+1} \\ x_{6k+1} \\ x_{7k+1} \\ X_{k+1} \end{bmatrix} = A \begin{bmatrix} x_{1k} \\ x_{2k} \\ x_{3k} \\ x_{4k} \\ x_{5k} \\ x_{6k} \\ x_{7k} \\ X_{k} \end{bmatrix}, x_{ik} = s_{ik} (i = 1,2,3)$$
$$\begin{bmatrix} u_{1k} \\ u_{2k} \\ u_{3k} \\ u_{4k} \end{bmatrix} = \begin{bmatrix} x_{4k} \\ x_{5k} \\ x_{6k} \\ x_{7k} \end{bmatrix} \quad (21)$$

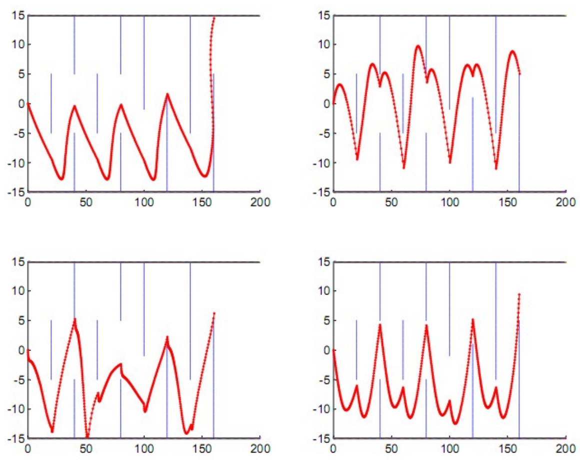

Fig. 4. Four robot traces controlled by four linear artificial cells

V. Conclusion and Future Work

In this paper we have developed a linear state space representation of a synthetic genetic regulatory network model and in a case study we have investigated its application in evolving linear dynamic controllers for nonlinear systems. In the case study the new method for evolving linear controllers shows its superiority over simple genetic algorithms for obtaining linear dynamic controllers. However we believe that the new approach should be studied on different case studies to prove its applicability. So an open area of research is applying the represented model as a linear controller for different nonlinear systems. Another track of research would be the possibility of proving the evolvability of the new method over simple genetic algorithms with analytical tools such as the one introduced in [12]. These two trends constitute our future research directions.

References

[1] E. H. Davidson, "Genomic regulatory systems," in *Development and Evolution*, Academic Press. 2001.
[2] W. Arthur, "The origin of animal body plans," A Study in Evolutionary Developmental Biology, Cambridge, 2000.
[3] T. Reil, "Dynamics of gene expression in an artificial genome – implications for biological and artificial ontogeny," in *Proc. Advances in Artificial Life, 5th European Conference, ECAL'99',* Lecture Notes in Artificial Intelligence, Springer, vol. 1674, pp. 457–466, 1999.
[4] J. Bongard (2002), "Evolving modular genetic regulatory networks," in *Proc. of the Congress on Evolutionary Computation 2002*, IEEE Computer Society, Washington, DC, USA, pp. 1872–1877.
[5] W. Banzhaf, "On the dynamics of an artificial regulatory network," in *Proc. Advances in Artificial Life, 7th European Conference, ECAL'03'*, Lecture Notes in Artificial Intelligence, Springer, vol. 2801, pp. 217–227, 2003.
[6] T. Quick, "Evolving embodied genetic regulatory network-driven control systems," in *Proc. of ECAL 2003*.pp. 266-277.
[7] M. Kirschner and J. Gerhart, "Evovlvability," in *Proc. of the Natural Academic Science,* USA, vol.95, pp. 8420-8427, 1998.
[8] P. Bentley, "Evolving fractal gene regulatory networks for robot control, " Lecture Notes in Computer Science, *Advances in Artificial Life,* Springer Berlin, 2003, pp. 753-762.
[9] P. J. Bentley, "Adaptive fractal gene regulatory networks for robot control," in *Workshop on Regeneration and Learning in Developmental Systems, Genetic and Evolutionary Computation Conference* (GECCO 2004).
[10] J. F. Knabe, C. L. Nehaniv, and M. J. Schilstra, "Regulation of Gene Regulation Smooth Binding with Dynamic Affnity affects Evolvability," in *IEEE Congress on Evolutionary Computation (CEC 2008).Proc. WCCI 2008,* pp. 890-896, IEEE Press, 2008.
[11] Kirschner. M and Gerhart.J , "Evovlvability," in *Proceedings of the Natural Academic Science*, USA, Vol.95, pp. 8420-8427, 1998.
[12] Valiant, L.G. , "Evolvability," in *proceedings of the 32nd International Symposium on Mathematical Foundations of Computer Science*, LNCS Vol4708, Springer-Verlag, 22-43, 2007

Navid Rahbari Asr received his Bachelor of Science in Electrical Engineering from Tabriz University, Tabriz, Iran and his Master of Science in Electrical Engineering from Tarbiat Modares University, Tehran, Iran in 2008 and 2011 respectively. He is currently a PhD student in North Carolina State University. His research interests include control, biologically inspired intelligence and systems biology.

Vahid Johari Majd received his B.Sc. degree in 1989 from the electrical engineering department of the University of Tehran, Iran. He then received his M.Sc. and Ph.D. degrees in the area of Control Theory from the electrical engineering department of the University of Pittsburgh, PA, U.S.A. in 1991 and 1995, respectively. He is currently an associate professor in the control system department of Tarbiat Modares University, Tehran, Iran, and is the director of intelligent control systems laboratory. His areas of interest include: Intelligent identification and control, multi-agent learning, fuzzy control, cooperative control, formation control, robust nonlinear control, fractional order control, and systems biology.