A Bio-inspired Agent-based System For Controlling Robot Behaviour

Y. Yao*†, G. Baele*† and Y. Van de Peer*†
*Department of Plant Systems Biology, VIB, B-9052 Ghent, Belgium
†Bioinformatics and Evolutionary Genomics, Department of Molecular Genetics,
Ghent University, B-9052 Ghent, Belgium

***Abstract*—In this paper, we present an agent-based system to control a single robot's behaviour. We present an artificial genome structure, based on gene regulatory networks, in which several regions can be distinguished such as promoter regions, indicator genes, transcription factor binding sites, regulatory genes and expressed genes. We use agent-based modeling (ABM) to simulate a bio-inspired system based on the artificial genome, with the ultimate goal of providing phenotypic information for a simulated robot. We show that the presence of a feedback loop in the agent-based system, along with the corresponding agent replacements, is essential to allow the robot to perform its tasks.**

***Index Terms*—Bio-inspired Robot,Agent-based System,Gene Regulatory Networks(GRN)**

I. Introduction

The field of evolutionary dynamic optimization deals with the application of evolutionary algorithms to dynamic optimisation problems (DOP)[1]. In these problems, the environment changes frequently or is completely unknown and the optimization methods need to adapt their proposed aim to time-dependent contexts. Previous research has seen different approaches to address such problems, such as a bio-inspired agent-based framework, which can be adopted to a highly changing environment with good scalability and flexibility [16]. Other research has shown that degeneracy of solution representation will improve the robustness and adaptiveness of dynamic optimisation, aside from mentioning that degeneracy is the main feature of bio-system in natural evolution [9]. In short, these studies show that evolutionary algorithms (EA) have the possibility to solve the DOP by introducing bio-inspired principles. Further, the flexible agent-based structure is inherently suitable to implement such bio-inspired system. In this paper, we present a layered structure with a dynamic feedback loop which are inherent properties of gene regulatory networks (GRNs), in order to develop a practical framework for achieving a self-adaptive robot in a simulated scenario. By mimicing the principles and features of GRN in our framework, we expect the agent-based system to efficiently deal with a highly changeable or unknown environment.

II. Agent-based Modelling

Agent-based modeling (ABM), also termed individual-based modeling (IBM), is a relatively new approach to modeling systems comprised of autonomous, interacting agents. Computational advances have led to a growing number of agent-based applications in a variety of fields. For example, agent-based models have been used to simulate the electric power market designed to investigate market restructuring and deregulation and to understand implications of a competitive market on electricity prices, availability, and reliability [13]. Agent-based models have also been used to study biological tissue patterning events, which have implications for both physiological and pathological function, arise from a cascade of complex processes and rely on interactions between cells, genomic information, and intra-cellular signaling [15].

The benefits of agent-based modeling (ABM) over other modeling techniques are threefold [3]: (1) ABM captures emergent phenomena; (2) ABM provides a natural description of a system; and (3) ABM is flexible. Essential to ABM is its ability to capture emergent phenomena, which result from the interactions of individual entities. By definition, they cannot be reduced to the system's parts: the whole is more than the sum of its parts because of the interactions between those parts. An emergent phenomenon can have properties that are decoupled from the properties of the part. ABM is, by its very nature, the canonical approach to modeling emergent phenomena: in ABM, one models and simulates the behavior of the system's constituent units (the agents) and their interactions, capturing emergence from the bottom up when the simulation is run. Further, ABM provides a natural description of a system. Finally, the flexibility of ABM can be observed along multiple dimensions. For example, it is easy to add more agents to an agent-based model. ABM also provides a natural framework for tuning the complexity of the agents: behavior, degree of rationality, ability to learn and evolve, and rules of interactions [3].

Different modelers of agent-based systems have differing opinions on what constitutes an agent (see [10] for an overview). Macal and North [10] consider agents to have (amongst others) the following characteristics, which we adopt in this paper:

- An agent is identifiable, a discrete individual with a set of characteristics and rules governing its behaviors and decision-making capability.
- Agents have protocols for interaction with other agents, such as communication protocols, and the capability to respond to the environment.
- An agent is flexible, and has the ability to learn and adapt its behaviors over time based on experience.

978-1-61284-060-4/11/$26.00 ©2011 IEEE

III. Gene Regulation

Gene regulation is a process in which a cell determines which genes it will express and when. Regulation of genes is a popular topic of interest, with respect to determining how the process works and what happens when it goes wrong. One of the easiest ways to illustrate gene regulation is to talk about gene regulation in humans. Every cell in the human body contains a complete copy of that person's DNA, with tens of thousands of potentially viable genes. Obviously, all of these genes cannot be expressed at once. Hence, cells must decide which genes to turn on and which genes to turn off. For example, a skin cell turns on the genes which make it a skin cell, while a bone cell would leave these genes turned off. Neither of these cells would need the genes which allow a cell to differentiate into a neuron, so these genes would be left off as well.

In addition to being useful for cell differentiation, gene regulation is also valuable for cell function. As a cell moves through its life, it has different needs and functions, which can be addressed with the use of gene regulation to determine which genes are expressed and when. Likewise, cells can adapt to environmental changes such as an injury which requires repair by activating new genes. For the cell, gene regulation can be accomplished in a number of different ways, with one of the most common simply being regulation of the rate at which RNA transcription occurs. Genes can also be deactivated by changing the structure of the DNA in an individual cell to turn them off or on.

Unicellular organisms also utilize gene regulation to regulate their functions and activity. These organisms must be able to adapt genetic material quickly to adjust to changing circumstances and new environments. Failure to do so will cause not only death of the cell, but death of the organism itself. Gene regulation allows such organisms to do things which will allow them to fit into hostile and extreme environments and to adapt to changes such as the introduction of antibiotics into their environment.

IV. Gene Regulatory Networks

A gene regulatory network or genetic regulatory network (GRN) is a collection of DNA segments in a cell which interact with each other (indirectly through their RNA and protein expression products) and with other substances in the cell, thereby governing the rates at which genes in the network are transcribed into messenger RNA (mRNA). In general, each mRNA molecule goes on to make a specific protein (or set of proteins). In some cases this protein will be structural, and will accumulate at the cell-wall or within the cell to give it particular structural properties. In other cases the protein will be an enzyme; a micro-machine that catalyses a certain reaction, such as the breakdown of a food source or toxin. Some proteins though serve only to activate other genes, and these are the transcription factors that are the main players in regulatory networks. By binding to the promoter region at the start of other genes they turn them on, initiating the production of another protein, and so on. Some transcription factors are inhibitory.

Gene regulation (see section III) is an umbrella term for molecular processes that conduct the cellular control of the functional product of a gene, which may be an RNA or a protein. A general assumption is that the amount of gene product represents how active a gene is. The gene expression underlies several control mechanisms, whereas the regulation of the transcription machinery constitutes the most important gene-regulatory mechanism. Regulators of transcription are mainly proteins, called transcription factors (TFs). However, the overall gene regulation is much more complex and includes processes such as transcript degradation, translational control, and post-translational modification of proteins. Thereby, apart from proteins, also other molecules like RNAs and metabolites participate in a regulatory manner. Finally, the genes, regulators, and the regulatory connections between them form a gene regulatory network.

In single-celled organisms regulatory networks respond to the external environment, optimizing the cell at a given time for survival in this environment. In multicellular animals the same principle has been put in the service of gene cascades that control body-shape. Each time a cell divides, this leads to two cells which, although they contain the same genome, can differ in which genes are turned on and making proteins. Sometimes a 'self-sustaining feedback loop' ensures that a cell maintains its identity and passes it on.

A. Genetic Encoding

The goal for our research experiment is to develop a bio-inspired artificial genome, based on current knowledge and modeling approaches for gene regulatory networks. We have been inspired by the model proposed by Torsten Reil [14] to study gene interactions in order to develop an approach fit for use in the field of evolutionary robotics. Reil [14] presented a biologically plausible framework for studying gene interactions and gene activity over time, where genes regulate each other by binding to regulatory sequences.

At the core of Reil's model is an artificial genome, which consists of a string of digits (similar to DNA sequences which are essentially a string of bases) which is randomly created. Genes are not pre-specified but identified in the genome after creation. The genome is searched for occurrences of the sequence '0101', and on encountering one, the following N digits are defined as a gene. In other words, the concept of a standard promoter ('0101') is used to define genes in the genome, similar to the equivalent in eukaryotic genomes of the so-called TATA box (a succession of 'TA' nucleotide pairs), which is the major component of every gene's promoter (allowing RNA polymerase to bind for transcription).

The product of a gene is yielded by a simple transformation of the gene sequence, in that each digit is simply incremented by 1, so that the gene sequence '221133' becomes a '332244' protein (if the alphabet that makes up a string of digits has to be limited to a fixed amount of digits, the protein might as well be '332200'). For each existing gene product, all matching

regulatory sequences in the genome are identified and stored. For example, for the gene '221133', these would be all occurrences of the sequence '332244' (or alternatively '332200'), each of which controls the gene immediately following it.

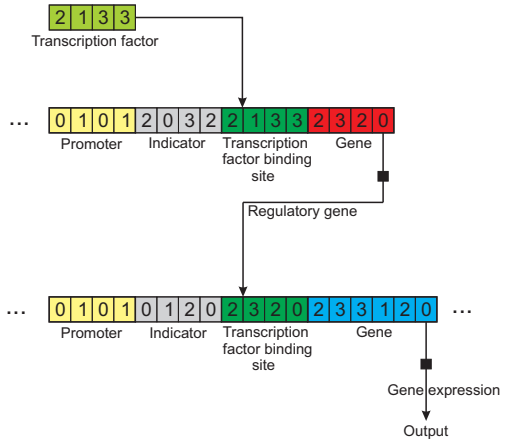

Fig. 1: Our proposed genetic encoding approach, based upon the approach by Reil [14].

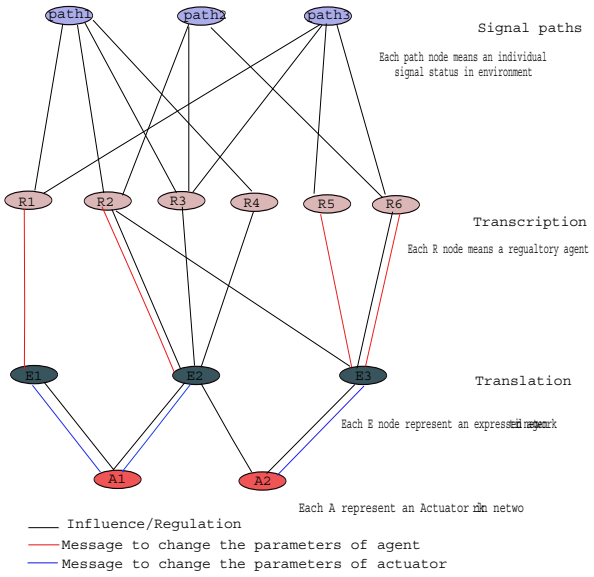

Fig. 2: The different layers in our developed agent-based system: the signal path layer, transcription layer and translation layer.

We have adapted the approach of Reil [14], as can be seen in Figure 1, with the following modifications. Between the promoter sequence and the actual gene, we have introduced an indicator gene and a transcription factor binding site, to mimic biological reality more closely. The encoding of the indicator gene is aimed at providing information concerning the type of gene that is being considered, i.e. either a regulatory or an expressed gene, which has an effect on the type of agent that has the handle the stored genetic information (see Section IV-D). The transcription factor binding site is used in the signal path layer (see Section IV-B) to specify which gene corresponds to a particular combination of sensor values. The other mechanisms of our genetic encoding are identical to the approach of Reil [14].

B. A Layered Approach

In order to simplify the complexity of our agent-based modelling approach, we distinguish three different layers in our agent-based system, which are shown in Figure 2. Separating the agent-based system into different layers results in agents that only have connections with the other agents in the same layer (not shown). This way, changes to a particular agent in a given (sub)network will only affect that same (sub)network. Further, interactions between agents in different layers can then be replaced by interactions between different layers, thereby simplifying the agent-based system's design.

The following layers can be distinguished in our agent-based system:

- Signal path layer: the main functions of this layer are receiving the sensor values and transforming this sensory information to usable values/signals for the agent-based system. This layer can be regarded as a hardware abstraction layer, so that different types of robots (providing sensor inputs of different magnitudes) can be used in the agent-based model. This layer is hence the bottom layer which is connected to the actual sensor of the robots. Each signal path consists of a combination of sensory inputs, determined by a corresponding gene in the artificial genome, and is allowed to evolve through time. The loss of certain sensors (for example due to damaged hardware) or other kinds of unexpected changes in the sensory inputs will be taken care of by this layer.
- Transcription layer: this layer contains the agent-based equivalent of a gene regulatory network (GRN) in that it consists of a network of regulatory agents (see section IV-D). This layer is the middle layer in our layered approach and does not directly interface with any hardware components of the robots. The main tasks of this layer is the optimization / emergence of the robot's behavioral / movement patterns, with the biological counterpart being the production of novel functions and patterns for the actuators.
- Translation layer: this layer is the top layer in our system and interacts with the actuators of the robot. The translation layer consists of expressed agents (see section IV-D) that take their information from the network of regulatory agents in the transcription layer. This layer then provides values for the actuators of the robots (in this case, the wheels of the robots). The main task in this layer is finding suitable strategies for each actuator in order to provide an adequate output pattern for the robot.

The essential aim of our agent-based system is to optimize its structure to adapt with a changing environment. In order to efficiently do so, we adopt a layered approach in our agent-based system. While a layered approach has the same components and relationships as a non-layered approach (not shown), the layered structure has a better flexibility for coping with changes than the non-layered structure, as it allows the

components in each layer to be adapted independently from one another, without affecting the whole network at once. This opens up various possible approaches to deal with the information retrieved from the feedback loop, as it allows for each layer to react to this information in a specific way.

C. Runtime Fitness In The Feedback Loop

In our experiment, we adopt an implicit fitness evaluation on the feedback loop and use that feedback to adapt the agents during runtime. This kind of feedback minimizes the direct reference to the patterns of robot behaviour. The reason for using such an evaluation is because the output of single agent does not directly determine the behaviour of the robot. Indeed, only through the cooperation of multiple agents can each agent's behaviour affect the robot's behaviour. To evolve every agent, the system can not reward any agent's behaviour without its context. On the other hand, the pattern of robot behaviour also doesn't depend on any single agent.

In this first version, the development is focused on the agent-based system so we only have a quite simple implicit fitness function here. The fitness will consider four variables. The first variable is the movement distance between time steps (represented as d), representing the efficiency of the two actuators. The second variable is the exploration range of the robot (represented as e), representing the robot's general performance. The third variable is the replacement ratio of agents at every time step (represented as r), which represents how quickly the inner environment changes with respect to a single agent. The fourth variable is the current total number of agents, which gives information on the current complexity of the system (represented as n). To an agent, the better the efficiency of movement, the broader the search range, the smaller the replacement ratio and the simpler the system complexity, the higher the chance for that agent to accumulate a high fitness score. The following formula shows how we deal with the fitness in our current system (i represents the current time step, p the weight parameters and T the time):

$$Fitness_i = \frac{d_i}{d_{avg}} \times p_d + (\frac{e_i/T_i}{e_{i-1}/T_{i-1}} + (\frac{n_{i-1}}{n_i} - 1)) \times p_e - \frac{r_i}{n_i} \times p_r$$

As can be seen from the formula above, fitness in our simulation reflects a comparative situation rather than an absolute value. That means the fitness score will be compared with its previous record at first and then the ratio will be regarded as the final fitness. If there is no comparable record, the fitness level will be kept neutral. The final fitness is calculated by the following formulas:

$$Fitness_{avg} = \frac{\sum Fitness_{0-i}}{i}$$

$$Fitness_{final} = \frac{Fitness_i}{Fitness_{avg}}.$$

D. Agent Simulation

Here we present the different types of agents that make up our agent-based model.

1) Genome Agent: A genome agent reads the relevant genes from the artificial genome. This artificial (fixed length) genome consists of a sequence of randomly generated nucleotides (A, C, G, T). The main functionalities of the genome agent are:

- Reading the artificial genome input file, if such a file is provided.
- If an artificial genome is not provided, generate a random genome.
- Look up a target gene in the genome when a binding site is provided by the environment agent.
- Altering a particular region of the artificial genome according to environmental conditions provided by the environment agent (for example in order to enhance gene expression).
- Mutating the artificial genome through random point mutations.
- Provide a new (and possibly altered) copy of the artificial genome at the end of a robot's lifetime.

2) Environment Agent: An environment agent reads the sensor inputs and establishes combinations of sensor values in the robot. In other words, a single environment agent does not correspond to a single sensor input (or a transformation thereof), but to a combination (unweighted sum) of the different sensor inputs. The resulting gathered sensory information of the environment agent is then used as a binding site during the scan of the genome in search of a fitting gene, one for each combination of sensory inputs. When such a gene is found, the indicator region of that gene is scanned to check whether that gene is a regulatory gene or an expressed gene, for which a fitting agent (i.e. either a regulatory or an expressed agent) is then created by the environment agent. The main functionalities of the environment agent are:

- Update (i.e. read) the sensory input values of the robot at every time step.
- Every time step, the environment agent evaluates the status of the robot (i.e. calculates the different combinations of the sensory inputs) and calculates feedback on the robot's performance.
- If the sensory information of the robot has changed (sufficiently), the environment agent will detect different regions of the genome than in the previous cycle and create new agents (either regulatory or expressed). The old agents (of the previous cycle) are then considered at the end of their lifetime and are removed from the system.
- Optimizing the combinations of the sensory inputs. The environment agent will calculate the adaptation value for each combination at each time step and will then try to optimize those combinations. The general procedure of this approach is as follows. This agent evaluates each sensory input combination by its adaptation value (this value is based on the feedback, the number of agents and the importance of the output for the particular combination) and checks if the combination can be adapted to the current environment. The environment

agent will delete those combination that have a lower adaptation value than a given threshold and produce a new sensory input combination by randomly reading a special-purpose gene.
- Monitoring the gene regulatory network and ensuring the interactions of agents will not exceed the limitation. The environment agent will check the number of agents to avoid that too many of them are created. The environment agent will also delete those combinations of agents for which a corresponding gene cannot be found or for which the output value is too low/high.

3) Regulatory Agent: A regulatory agent regulates, for example enhances or represses, a particular gene and must hence retrieve the gene it expresses. The main functionalities of the regulatory agent are:

- Finding the regulatory gene from the genome based on its corresponding signals. If there is an agent which shares the same gene in the system, the concentration degree of this existing agent will be increased. Otherwise, a new agent will be created and the new one will read the target gene to initialize itself.
- Identifying the target gene and sending an instruction to the genome agent to change the gene status (repress or enhance). When agents have been created, the position of the target gene on the genome will be recorded by its agent. All agents will have a limited life cycle in the agent-based system. When the agent dies, every agent will modify the adaptation value of its target gene according to its own feedback. This way, an agent that performs well will enhance the adaptation value of its target gene. The higher the adaptation value of a gene, the more chance this gene has to be read from the genome at a future occasion.
- Evaluating its own importance and adaptation in the system's interactions. The agent needs to evaluate its adaptive status in the system and needs to know the influence of its outputs. If the agent has more outputs than others, it will be regarded as more important in interaction and it will also has more responsibility with respect to the feedback. The adaptive status for an agent indicates how good the performance of the robot is when the agent is active. The adaptive status of an agent will be used to affect the adaptation value of its corresponding gene.
- Reading the artificial genome in order to find the gene, either regulatory or expressed, it regulates.
- Upon retrieval of the expressed gene, creating the expressed agent.

4) Expressed Agent: An expressed agent translates the encoded information of its underlying gene to an actuator. The main functionalities of the expressed agent are:

- Binding with the corresponding actuators and output the value encoded in its underlying gene to an actuator.
- Identifying the target expressed gene and sending the instruction to genome agent to change the gene status (repress or enhance). The genome agent will change the binding site of the expressed gene to increase or decrease the probability that it will be read at a future occasion.
- Evaluating its own importance and adaptation in the system's interactions. The importance (score) of an expressed agent is the sum of its actuators multiplied by the concentration degree of this agent. The adaptation status of an agent is used to evaluate its performance. For an agent that performs well, the agent-based system will increase its concentration degree and life time. For an agent that does not perform well, there will be a tendency for deletion of this particular agent.
- When multiple expressed agents correspond to a single robot actuator, the different values in the expressed genes need to be aggregated into one output signal for the actuator.

E. Agent And Pathway Replacment

During runtime all agents and simulated pathways interact with their inner environment and connect to form an emerged dynamic structure. The creation of new agents is triggered by the constant input stimulations from the environemnt, while the replacements are always conducted by the inner feedback loop. A single agent replacement will be built up due to a particular signal combination from the environment. Signal pathways are initialized from the genome and are sensitive to its inner environment. Environmental changes could lead to a potential pathway being activated or the activated pathway being repressed. Agents also have their own lifetime and concentration value, something the pathways don't have. Even if the calculated fitness from the feedback loop is good, an agent will still be replaced when it has run out of lifetime or when its concentration value is too low. When this happens, the replacement will be regarded as successful with respect to the individual agent and the built up gene of that agent will be enhanced in the genome. In other words, the gene will become more competitive to be read in a certain input range. The concentration value of an agent will hence increase or decrease corresponding to fitness.

V. SIMULATIONS

All experiments were performed in the Player/Stage simulation environment [8]. Classes necessary to read, store and manipulate robot genomes were written in C++ for cooperation with the programming code in Player/Stage. The simulation map consists of a rectangular area with several obstacles and corners where the robot can become cornered or stuck. The fitness of an individual robot is a function of the amount of the environment the robot is able to explore (the more, the better). Inherent to this requirement for a high fitness value is the ability to perform obstacle avoidance. Simply calculating a measure to perform obstacle avoidance has the property that robots that simply turn in circles can also be regarded as 'avoiding obstacles', while it is not performing a useful task. The combination with map exploration fixes this problem.

In our experiments, we have tested the developed agent-based system using a randomly generated population consisting of 50 robots. In other words, a (different) random genome was generated automatically for each of the 50 individual robots. The 50 robots were tested individually by placing each of them in a separate copy of the environment. While the starting position within the simulation map was set to be identical for each robot, the starting orientation of the robot was randomly selected to avoid that robots perform well simply because they are oriented in a direction with no obstacles.

We have used simulated e-puck robots in our experiments [11]. These robots have eight infrared (IR) proximity sensors (similar to the Khepera robots [12]) placed around the body which can be used to measure the closeness of obstacles and two stepper motors, controlling the movement of the two wheels.

VI. Experiments And Results

A. Resolving Collisions And Repetitive Motion

The simulations performed using our developed agent-based system, based upon a large genome inspired by gene regulatory networks, show evidence of self-adaptive abilities (i.e. the ability to adapt itself depending on occurring problematic situations). These situations can be considered to be getting stuck in a corner, not being able to move away from a wall or even just undesired behaviour in terms of achieving a decent fitness level. In order to do this, the agent-based system was provided with a feedback loop to signal potential problems with the current environment of the robot. This feedback loop ensures that the robot can adapt to new situations or environments up to a certain level. This also means that as long as the feedback loop does not signal any problems to the agent-based system, the robot's behaviour remains unchanged. An example of this type of behaviour can be seen in Figure 3.

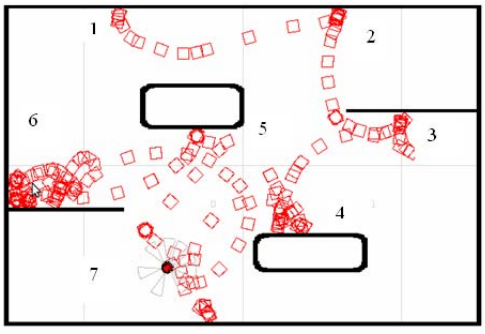

Fig. 3: Due to the agent-based system, the robot is able to resolve difficult situations. Starting from a fixed position near the center of the area (close to position 5), the robot encounters a series of problems, such as being stuck against a wall (positions 3 and 4) and being stuck in a corner (position 6).

Figure 3 illustrates how the robot smoothly adapts its behaviour during the runtime of the simulation. When the robot hits an obstacle, it will adapt its behaviour slightly in order to solve the problem. Should small changes in the robot's behaviour not be sufficient to resolve the current problem, additional changes to its behaviour will be made by the agent-based system until the robot is able to free itself. The more problematic the situation, the longer it will take for the robot to resolve the problem. This is an important aspect of the agent-based system used here, as in other approaches (see e.g. [2] for a neural network approach). The emergence of optimal robot behaviour is obtained by removing robots that do not perform well from the population, across a large number of generations. This is a process which may end up taking a huge amount of time, even though bio-inspired approaches have been proposed to facilitate this process, i.e. to make the population of robot reach adequate fitness level at a faster pace. For example, Calabretta et al. have published a series of papers on the advantages of modelling gene duplications on the performance of a robot population [4], [5], [6], [7]. A comparison between an approach with feedback enabled (such as our agent-based system) and an approach without such a feedback loop (for example, a simple artificial neural network) can be seen in Figure 4.

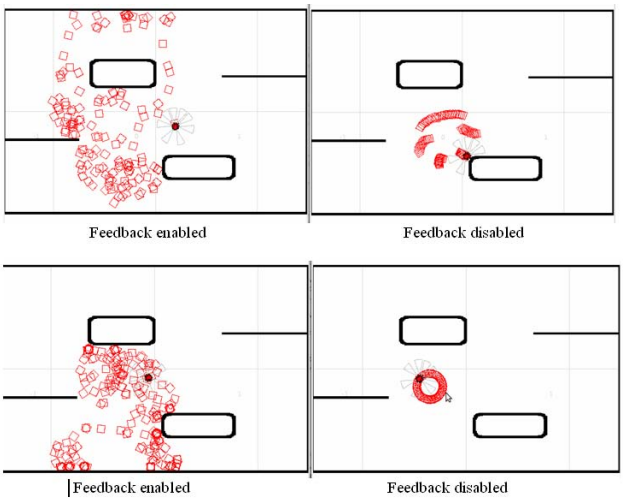

Fig. 4: The feedback loop incorporated into the agent-based system also allows the robot to detect when it performs repetitive movements, such as just turning in circles, which may occur in the absence of such a feedback loop.

The way in which each robot's decision and sensing mechanisms works is encoded in its genome and is essentially the responsability of the environment agent in the agent-based system. Hence, when the robot gets stuck, the feeback loop informs the appropriate environment agent of this, after which the current sensory information combination will be removed and a new sensory input combination will be proposed to the system by reading and decoding different genes from the genome.

B. Genome-dependent Behavior

Since the agent-based system relies upon an artificial genome to create its various components, different artificial genomes will lead to different types of behaviour (and hence

a difference in performance) for different robots. Figure 5 illustrates this diversity in the robot population.

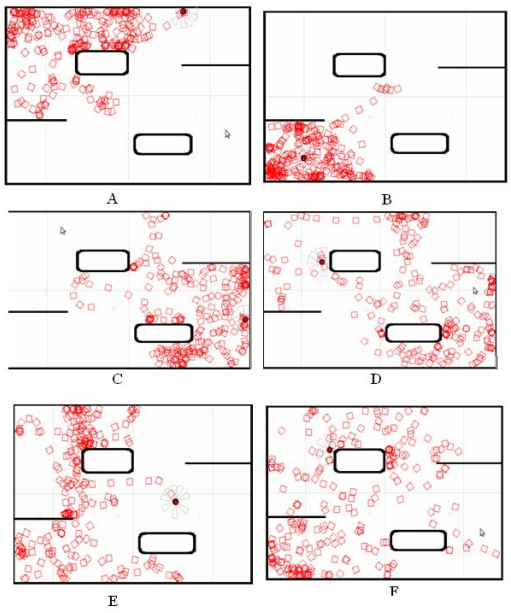

Fig. 5: Depending on the genome of the robot, different behaviour can be observed, with the feedback loop making sure that the robot does not get permanently stuck. More complicated regions of the area (e.g. with only one possible escape direction) show a more dense trail as it takes longer for the robot to resolve the situation.

In Figure 5 it can clearly be seen that there are various locations in the area which are difficult for the robots to explore, such as the top-left corner (situation A), the bottom-left corner (situation B; same goes for the top-right corner) and the bottom-right corner (situation C). In none of these situations does the robot remain stuck however, although it is apparent that the bottom-left corner is the most difficult part of the area to escape from, hence the large amount of time that the robot spends there. Situations D and E in Figure 5 show robot behaviour for two artificial genomes that allow the robot to explore large portions of the area, with little time being spent stuck in a corner or against a wall. Finally, situation F in Figure 5 shows robot behaviour for an artificial genome that allows the robot to explore the entire area, visiting all the difficult to reach (and escape) areas of the map. A given agent is able to autonomously select better-suited genes in order to achieve its goals. For example, if the behaviour of a given robot in the current environment is rewarded, the corresponding agents responsible for the robots behaviour will be automatically rewarded as well and the gene(s) that set those agents will also be slightly enhanced. Such enhancements can accumulate on the genome and eventually render those genes easier to read by agents, leading to next-generation agents that will tend to select better-suited genes to achieve the robots goal(s).

C. Agent Dynamics And Robot's Performance

The agents in our agent-based system can be replaced by new ones during runtime. Such replacements may result in a positive influence on the robot's behaviour in terms of a robot's ability to adapt to its environment. In other words, a change in environment can mean the transition from a problem-free environment (when the robot does not encounter any obstacles or other difficulties) to a problematic environment (i.e. being stuck in a corner or against a wall). Agent replacements will hence occur most in difficult environments, driven by the data in the feedback loop, while there is no need for such replacements in problem-free environments. This way, each robot possesses a self-adaptive ability when confronting a new environment. As an example, we show in Figures 6 and 7 an example of a robot's behaviour (i.e. its phenotype) and the corresponding agent dynamics within the agent-based system (i.e. its genotype). Figure 6 shows the movement trail of the robot in the simulation map (with numbers indicating the different situations), whereas Figure 7 shows the corresponding agent dynamics during those time steps. As can be seen from Figure 6, the robot encounters four difficult situations during its runtime (i.e. at situations 2, 3, 4 and 5), which results in a temporary halt in the robot's task to explore the simulation map.

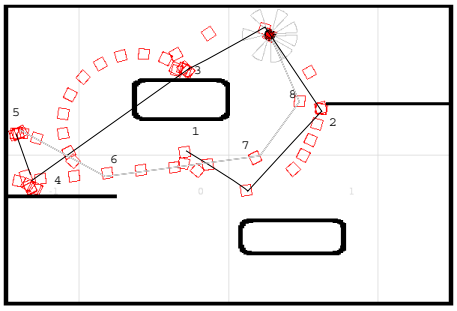

Fig. 6: The movement trail of the robot during 200 time steps (i.e. input/output cycles). Numbers 1 through 8 indicate (in order) the various situations the robot can be found in.

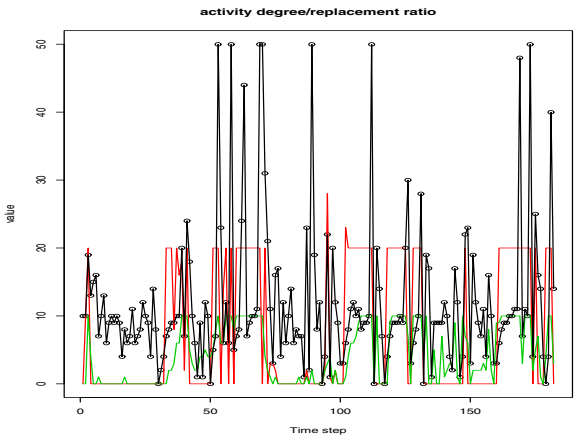

Fig. 7: The replacement rates of agent and signal pathways are correlated with the robot's activity degree, which is the main measurement of fitness in this simulation. The black curve represents the activity degree, while the green curve represents the signal pathway replacment rate and the red curve represents the agent replacement rate.

These problematic situations can be resolved through agent and signal pathway replacements in order to change the robot's current behaviour, which will allow the robot to overcome the problem rather than to remain stuck. This will result in an increased performance of the robot, as otherwise the robot would remain stuck (see Figure 6). For example, the difficult situations the robot has to overcome occur after 50 time steps (situation 2 in Figure 6), 90 time steps (situation 3 in Figure 6), 120 time steps (situation 4 in Figure 6) and 145 time steps (situation 5 in Figure 6). Figure 7 shows a clear increase in agent and signal pathway replacement rate corresponding to these reported time steps. In other words, in order to resolve the fact that the robot is stuck in a difficult environment, agents and/or signal pathways are replaced in order to equip the robot with a suitable behavioral pattern, fit to the changed environment (i.e. corner or obstacle). Indeed, when a robot gets stuck, its "activity degree" (i.e. its ability to explore the area) drops to very low levels, indicating the need for a different behaviour in order to be able to continue performing its task adequately. Following this drop, both the replacement rates of agents and signal pathways increase, after which the activity degree of the robot is seen to increase again.

VII. Discussion

In this paper we present a first version of our agent-based system aimed at controlling robot behaviour. The main benefit of this system is the robot's ability to resolve problematic situations at runtime, in its goal to explore as much of a rectangular area, filled with obstacles, as possible. Currently, the area in which the robot performs its task is static, i.e. does not have any moving obstacles, and the robot's task is not overly complicated. The current set-up is however adequate to test our developed agent-based system as we aim to add more complexity to the robot's tasks and the simulation area as well. For example, it would be more realistic that a robot has a limited lifetime, depending on battery power, a scenario where the robot would not only have to explore a given area but also make sure that it doesn't run out of battery power. This is the subject of ongoing work.

The structure of the artificial genome is currently a drastic simplification of the real-life workings of gene regulatory networks. Hence, the representation by the agent-based system of the gene regulatory networks in the artificial genome are oversimplified as well. The work presented in this paper however serves as a proof of principle and we aim to increase the complexity of our artificial genome to resemble biological reality more closely in future work.

VIII. Conclusions

We have shown the adequate performance of our developed agent-based system, which uses a bio-inspired artificial genome based upon current knowledge on the workings of gene regulatory networks. Robots equipped with our agent-based system are able to find their way out of difficult situations, allowing them to continue performing their task. This is specifically due to the presence of a feedback loop in our agent-based system, which signals potential problems to the system, allowing for a solution to be found. Further, the desired behaviour of the simple task in this paper (exploring a simulation map) is brought about without the need for an evolutionary strategy. However, we expect that more difficult simulation scenarios will require an evolutionary strategy in order to yield adequate results.

Acknowledgment

The "SYMBRION" project is funded by the European Commission within the work programme "Future and Emergent Technologies Proactive" under the grant agreement no. 216342.

References

[1] T. Blackwell, J. Branke, and X. Li. Particle Swarms for Dynamic Optimization Problems. In *Swarm Intelligence*, pages 193-217, 2008.

[2] A. Berlanga, P. Isasi, A. Sanchis, and J. M. Molina. Neural networks robot controller trained with evolution strategies. In *Proc. of the 1999 IEEE Congress on Evolutionary Computation (IEEE CEC-1999)*, pages 413–419. IEEE Press, 1999.

[3] E. Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *PNAS*, 99:7280–7287, 2002.

[4] R. Calabretta, R. Galbiati, S. Nolfi, and D. Parisi. Two is better than one: a diploid genotype for neural networks. *Neural Processing Letters*, 4(3):149–155, 1996.

[5] R. Calabretta, S. Nolfi, and D. Parisi. Investigating the role of diploidy in simulated populations of evolving individuals. In P. Husbands and I. Harvey, editors, *Proc. of the Fourth European Conference on Artificial Life*, Brighton, UK, 1997. The MIT Press.

[6] R. Calabretta, S. Nolfi, D. Parisi, and G. P. Wager. Duplication of modules facilitates the evolution of functional specialization. *Artificial Life*, 6:69–84, 2000.

[7] R. Calabretta, S. Nolfi, D. Parisi, and G.P. Wagner. A case study of the evolution of modularity: towards a bridge between evolutionary biology, artificial life, neuro-and cognitive science. In Ch. Adami, R.K. Belew, H. Kitano, and Ch. Taylor, editors, *Proc. of the sixth international conference on Artificial life*, pages 275–284. The MIT Press, 1998.

[8] B.P. Gerkey, R.T. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proc. of the 11th International Conference on Advanced Robotics (ICAR-2003)*, pages 317–323, 2003.

[9] X. Yao J. M. Whitacre, P. Rohlfshagen and A. Bender. The role of degenerate robustness in the evolvability of multi-agent systems in dynamic environments. In *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature*, 2010.

[10] C. M. Macal and M. J. North. Tutorial on agent-based modeling and simulation. In *Proceedings of the 2005 Winter Simulation Conference*, 2005.

[11] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli. The e-puck, a robot designed for education in engineering. In *Proc. of the 9th Conference on Autonomous Robot Systems and Competitions*, pages 59–65, 2009.

[12] F. Mondada, E. Franzi, and P. Ienne. Mobile robot miniaturization: A tool for investigation in control algorithms. In *Proc. of the Third International Symposium on Simulation on Experimental Robots*, pages 501–513, 1993.

[13] T. North, G. Conzelmann, V. Koritarov, C. Macal, P. Thimmapuram, and T. Veselka. Elaboratories: agentbased modeling of electricity markets. In *American Power Conference, Chicago, IL, Apr. 15-17.*, 2002.

[14] T. Reil. Dynamics of gene expression in an artificial genome - implications for biological and artificial ontogeny. In *Proc. of the 5th European Conference on Advances in Artificial Life*, 1999.

[15] B. C. Thorne, A. M. Bailey, and S. M. Peirce. Combining experiments with multi-cell agent-basedmodeling to study biological tissue patterning. *Briefings in Bioinformatics*, 8(4):245–257, 2007.

[16] K. Yeom. Bio-inspired self-organization for supporting dynamic reconfiguration of modular agents. *Mathematical and Computer Modelling*, 2009.