Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2011
August 29-31, 2011, Washington, DC, USA

DETC2011-48833

A BEHAVIOR BASED APPROACH TO CELLULAR SELF-ORGANIZING SYSTEMS DESIGN

Chang Chen
Dept. of Aerospace & Mechanical Engineering
University of Southern California
Los Angeles, California, USA
changche@usc.edu

Yan Jin*
Dept. of Aerospace & Mechanical Engineering
University of Southern California
Los Angeles, California, USA
yjin@usc.edu
**Corresponding Author*

ABSTRACT

Multi-agent systems (MAS) have been considered a potential solution for developing adaptive systems. The design of MAS however is difficult because the global effect emerges from local actions and interactions that can be hard to specify and control. In order to achieve high level resilience and robustness of MAS and retain the capability of specifying desired global effects, we propose a *cellular self-organizing* (CSO) system framework and a biologically inspired *behavior based design approach* (BDA) and a *field based regulative control* mechanism (FBR). The BDA approach links global functional requirements with the local behavior design of a CSO system. FBR is a real-time, dynamical, distributed mechanism that regulates the emergence process for CSOs to self-organize and self-reconfigure in complex operation environments. BDA and FBR together extend the system adaptability without imposing global control over local agents. This paper describes the models of CSO, BDA and FBR and demonstrates their effectiveness by presenting simulation based case studies in which CSO agents explore an unknown environment and move an object to designated locations.

Keywords: Bio-inspired approach, self-organization, self-reconfiguration, distributed control, field based regulation

1 INTRODUCTION

It has been well recognized that human engineered systems are becoming more and more complex. A major issue with developing complex engineered systems is that the sheer size of the system and interdependencies among the system components create *uncertainty* and *unknowns* to the engineers, leading to high level system risks. Furthermore, new systems are more likely required to function in unpredictably *changing environments,* where unpredictable situations may happen. Dealing with either or both of these problems requires future human engineered systems to be *adaptive* such that they can *robustly* redesign and rebuild themselves in response to task and environmental changes and *resiliently* self-repair and reconfigure as partial system failures happen.

Our observations of system biology (Kitano 2002), self-configurable systems (Subramanian and Katz 2000) and component-based design (Kopetz 1998) have led us to taking a MAS approach to adaptability by devising capable mechanical cells and their interaction mechanisms and then letting them "design and configure themselves" *bottom-up* in a distributed fashion based on their perceived operation situations. We call this a *Cellular and Self-Organizing* (CSO) approach to building complex and adaptive systems with each mechanical cell being an agent of MAS. It is fully understood that the CSO approach will not be able to compete with the traditional methods in a short term. However, the paradigm shift from functionally distinct *component-based* to *cell-based* and from *top-down* to *bottom-up* promises an alternative future for developing complex engineered systems.

Both *multi-agent* systems and *self-organizing* systems have recently been highlighted in many engineering fields, such as computer science, industrial engineering, and material science. Much research has been done to investigate the properties and benefits of those systems, and the ways to build such systems. One critical research question that has yet to be fully explored is: *How can a designer connect the design of local interactions of agents to the desired system level properties and functions*? Answers to this question are needed for us to design *self-organizing mechanisms* to achieve desired system functions with high level robustness and resilience.

To address the abovementioned research question, we propose a *behavior based design approach* (BDA) for designing agents of a CSO system and a *field based behavior regulation* (FBR) mechanism, a bio-inspired behavioral control mechanism, as a basis for agents to interact with each other and with their environment. In this approach, agents' local actions and interactions can be designed as behaviors based on the functional requirements of the system under given conditions. The agent behaviors (i.e., all possible behaviors or actions) as design information are homogeneously stored in every individual agent. An agent's runtime behavior at any given time is a result of FBR based control that is determined by the "field position" of the agent.

This approach allows a designer to design a multi-agent mechanical system from given functional requirements, and the resulting system will be able to possess adaptability for those situations that are not predicted by the designer. All agent behaviors are decided locally by agents themselves in a similar way as biological cells behave in natural systems. Our BDA requires no uniquely designed individual agents for specialized tasks. They are homogeneous agents. The *agent differentiation* is achieved locally through FBR control.

It is worth mentioning that the concept of agent differentiation is a key distinction of our CSO approach as compared to conventional modular or component-based approaches. While the behavior of the modules and components is determined at design time and does not change during system operation, the behavior of our mechanical cells is determined at runtime through FBR regulation based on a predefined set of possible behaviors. This approach mimics cellular differentiation from stem cells and permits dynamical system redesign and reconfiguration, leading to higher levels of system adaptability.

In the rest of this paper, we first review the related work in Section 2 and then introduce our behavior based design approach (BDA) in Section 3. In Section 4, we present a BDA based CSO design approach and in Section 5 we demonstrate a simulation based case study of using BDA for multi-agent system design. Section 6 draws conclusions of our current work and point to future research directions.

2 RELATED WORK

Much research has been done to investigate multi-agent and self-organizing systems and to develop methods to design such systems. Self-organization and emergent behavior as two major features of such systems have been popular research topics in the research field of complex systems (Bojinov, Casal, & Hogg, 2000; Butler, Kota, Rus, & Tomita, 2001; Fukuda & Kawauchi, 1990; Neumann, 1966; Weisbuch, 1991; Wolfram, 2002; Zouein, 2009). Self-organization is the large scale organization through the limited local interactions of the constituent components. Emergence represents the concept of the patterns, often unpredictable ones, which are exhibited in the large scale organization. With non-self-similar agents, Holland and Gell-Mann(1992, 1994) extended the research to non-homogeneous system and pointed out the non-linearity between local and global which becomes the biggest challenge of such systems. To further address the problem, the Game of Life (Garden,1970) and more Cellular Automata based fractals have been explored (Wolfram, 2002). More recent work on understanding and modeling complex adaptive systems can be found on Santa Fe Institute's website (Santa Fe, 2010).

In the field of engineering design, design for adaptability and design of reconfigurable systems have been investigated in the past decade. Martin and Ishii (2002) proposed a design for variety (DFV) approach that allows quick reconfiguration of products but mainly aims to reduce time to market by addressing generational product variation. Indices have been developed for generational variance to help designers reduce the development time of future evolutionary products (Martin and Ishii, 2002). In addition to developing design methods for reconfigurable systems, various reconfigurable robotics have been developed mostly by computer scientists. Fukuda and Nakagawa (1988) developed a dynamically reconfigurable robotic system known as DRRS. Unsal and Khosla (2000) focused on creating very simplistic i-Cube systems (with cubes being able to attached to each other) in order to investigate whether they can fully realize the full potential of this class of systems. PolyBot has gone through several updates over the years (Yim, 1993, 1994; Casal and Yim, 1999; Yim et al., 2000, 2002) but acquired notoriety by being the first robot that “demonstrated sequentially two topologically distinct locomotion modes by self configuration. SuperBot (Shen et al., 2006) is composed of a series of homogeneous modules each of which has three joints and three points of connection. Control of SuperBot is naturally inspired and achieved through what the authors describe as the “hormone” control algorithm (Shen et al., 2000a, 2000b, 2002; Salemi et al., 2001).

Bio-mimetic design methods allow designers to identify appropriate natural systems or mechanisms from which to draw design inspirations. The idea of using DNA and genes to capture genotype of systems is not new. Inspired by the nature’s evolution process, genetic algorithm (GA) (Goldberg, 1989) and genetic programming (GP) have been established to model problems using bit string (GA) or functional tree (GP) genes and to solve problems by evolving the best solution(s) from a population through reproduction, mutation, recombination, natural selection and survival of fitness. This approach has been taken to solve various engineering problems including design optimization, configuration design, and design automation (Maher, 2001; Koza, 1992; Fogel et al, 1996; Parmee, 1997; Bentley, 1999; Bonnie and Malaga, 2000; Lee et al 2001; Koza et al, 1999; Vajna and Clement, 2002; Fan et al, 2003). In addition to direct encoding where genotype codes map to the phenotypes directly, recently researchers have explored indirect coding method, called computational embryogeny (Kumar and Bentley, 2000), to evolve rules that build or develop corresponding phenotypes (Yogev and Antonsson, 2007). Although these computational methods have been successfully applied to solve optimization problems with specific fitness functions, effectively integrating the methods into our proposed CSO systems design and development framework is a key challenge.

Our previous work on CSO generated a design DNA concept and associated system formation mechanisms (Zouein, 2008, Jin et al, 2010). This research extends the previous research by first expanding the design DNA from a static specification to a dynamic and probabilistic representation and then introducing a new field based control mechanism to utilize the potentials of such systems for increased robustness and resilience. In addition, while most current approaches for MAS design requires agents have a global unique identifiers for cooperation and some methods such as DHM (digital hormone model) require explicit local interactions, our behavior based approach allows agents to respond to the field of the task environment spontaneously and interact with other agents only implicitly, rather than deliberately, as a result of their actions in the task field.

3 A BEHAVIOR BASED DESIGN APPROACH

Our *behavior based design approach* described in this paper is different from the existing approaches to multi-agent systems development. In particular, this approach focuses on devising common behaviors in individual agents and facilitating agent *behavior differentiation* based on a field based regulation (FBR) mechanism, mimicking the morphogen based cellular differentiation found in biological systems. This behavior based approach contrasts with the conventional *structure based design* approach because in our proposed CSO systems, structures emerge from behavioral self-organizing processes. The advantage of this behavior based approach is its capability of dealing with “unknowns”, e.g., unpredictable environment changes and new functional requirements, by spontaneously responding to new circumstances through behavioral self-organizing. From a system design perspective, the conventional design approach designs "structures" that generate needed "behaviors" in order to achieve desired "functions" (i.e., structure->behavior->function), while our proposed approach allows multiple agents to self-organize their "behaviors" which will together create "structures" of the system and then achieve "functions" of the system (i.e., behavior->structure->function).

As mentioned above, a CSO system achieves its functions through emergence. Therefore, designing CSO systems means to design individual agent's behaviors that can lead to global emergence that is functional as expected. In the following, we first present the model of CSO systems and then introduce the method for behavior design.

3.1 Concepts and Models

A mechanical Cell (*mCell*) is the basic element or unit of a mechanical CSO system:

Definition 1 (Mechanical Cell): mCell = {Cu, S, A, B};

where Cu: control unit; $S = \{s_1, s_2, ...\}$*: Sensors/Sensor information;* $A = \{a_1, a_2, ...\}$*: Actuators/Actions; B: designed behavior, or design information (see definition 4 below).*

Almost all existing cellular systems, such as Superbot (Shen et all 2001) and Miche (Gilpin et al 2005), can be modeled using this definition. *mCell* is the smallest structural and functional unit of a CSO system. Although for a CSO system design, either homogeneous with identical *mCells* or heterogeneous with different *mCells*, the appearance or the structure of its *mCells* may be different. A *mCell* should be able to sense the environment around it and process material, energy and/or information as their actions. We make following assumptions about *mCells*:

Assumption 1 (Cellular Capability): A *mCell* has the ability to execute predefined programs, sensing the world around it, process sensory information and the incoming communication, and decide on its action and interaction with others.

Assumption 2 (Cellular Limitation): mCells have limited sensors, limited range for each sensor, limited communication range with others, and limited number of possible actions.

Definition 2 (State): $State = \{S_C, A_C\}$

where $S_C \subset S$ *and* $A_C \subset A$ *are currently sensor information and actions, respectively.*

State is used to represent the situation which the current *mCell* is in. It is the combination of the current sensor information *Sc* and current actions *Ac*.

Definition 3 (Behavior): $b = \{S_E, A_E\} \rightarrow A_N$

where $S_E \subset S$ *and* $A_E \subset A$ *are existing sensor information and actions, respectively; and* $A_N \subset A$ *are next step actions.*

A behavior *b* is the designed action for given situations or states. The *Cu* of the *mCell* should be able to judge the situation and make decisions on next actions. The design information (like a complete drawing in a conventional system design) of a CSO system is the fully developed behaviors for each *mCell*.

Definition 4 (Behaviors of System): $BoS = \{B_1, B_2, ..., B_n\}$*;*

where $B_1, B_2, ..., B_n$ *are behavior sets of each mCell in the system.*

The design information for such a system is the set of all the behaviors which local *mCells* should follow; also this *BoS* is supposed to be designed by a designer or designers. If all *mCells* share the same behavior set *B*, then we have a homogeneous CSO system. Otherwise, the CSO system is said to be heterogeneous.

From the above four definitions, one may see that the concept of *mCell* is similar to that of biological cell. A biological cell serves its purpose by the production of proteins which parallels a *mCell* producing local actions; the biological cell can only process the signals that the receptor on the membrane can catch, similar to *mCells*. Furthermore, all biological cells hold a full "design information" stored in DNA. Similarly, *mCells* hold the same design information through a designDNA, or *dDNA*, which is captured by the associated behavior set.

Given the above definitions, there are two problems remaining for designing CSO systems using this model, 1) how to generate or define behaviors *B* for each *mCell*, and 2) how to device “rules” so that *mCells* can self-organize themselves to achieved assigned functional requirements.

Definition 5 (Functional Requirement): $FR_i = \{S_i, A_i\}$

where S_i*, and* A_i *form a specific state or situation.*

There two reasons why the *functional requirement* holds similar construct of *state* described above. First, this representation allows us to specific “desired states” of the system. These desired states can be *goal states* or *transient states* that a designer deems to be necessary. Second, using the state construct to represent functional requirements allows *mCells* to recognize whether the function is achieved by examining combinations their sensor information and actions.

It is worth mentioning that our definition of *functional requirement*, or *function*, of using both sensory information S_i and action A_i is more general than the conventional *function* definition that uses only action A_i. When $S_i = \emptyset$ our definition is consistent with the conventional one. Our general representation allows designers to specific circumstances (i.e., sensory information) in addition to actions, leading to more precise functional specification.

3.2 Behavior Based Design in CSO Systems

From the above definitions, it can be seen that to design a CSO system for a set of given functional requirements is to define behaviors of all *mCells* in the system, as indicated below.

$$\begin{array}{l} FR_1 = \{S_1, A_1\} \\ FR_2 = \{S_2, A_2\} \\ FR_3 = \{S_3, A_3\} \longrightarrow BoS = \{B_1, B_2, ..., B_m\} \\ .... \\ FR_n = \{S_n, A_n\} \end{array} \qquad (1)$$

In a conventional design approach, the leaf functions *FR1* through *FRn* are derived through a specific "function decomposition" process, e.g., systematic design (Pahl and Beitz 1976) or axiomatic design (Suh, 1990) in which the higher level functions are fulfilled by the lower level ones. Assuming a complete decomposition is carried out by following the axiomatic design (Suh, 1990), we will have following design results.

$$\begin{array}{l} FR_1 = \{S_1, A_1\} \rightarrow b1 \\ FR_2 = \{S_2, A_2\} \rightarrow b2 \\ FR_3 = \{S_3, A_3\} \rightarrow b3 \quad \longrightarrow \quad BoS = \{\{b_1\}, \{b_2\}, ..., \{b_n\}\} \\ .... \\ FR_n = \{S_n, A_n\} \rightarrow bn \end{array} \qquad (2)$$

Equation (2) indicates that each *mCell i* (out of *n mCells*) will have a unique behavior *bi* as its behavior set, meaning that each *mCell* behaves as a specific function component. In this paper, we call this approach of design *top-down function based* or simply *top-down*. The advantage of this design approach is that the resulting system is functionally highly efficient with no possible "waste of functions." However, there are at least two cases this *top-down* approach may lead to system failure. The first has to do with *system robustness*: if the operation environment requires functions that are not foreseeable at the design time, then the system will not be able to achieve the required but unforeseeable functions and fail. Second has to do with *system resilience*: if any of the functional components fails, the system will fail.

In our research, we propose a behavior based design approach to facilitate the design of CSO adaptive systems. In this approach, we do not require a designer be able to foresee all required functions. Furthermore, *mCells* may hold partial or fully redundant behaviors so that if some *mCells* fail others can replace them and perform their functions. Although top-down functional decomposition can still be carried out but the final set of functional requirements does not need to be complete since the completeness does not exist in the unpredictable situations. The fundamental idea behind this behavior based approach is that by introducing *mCells* with multiple and redundant behaviors and letting them self-organized, it is expected that the emergence of these self-organizing *mCells* will yield system functions that are needed for the changing circumstances. From a design perspective, there are two issues that must be addressed to complete this approach. One is that we need to know what constitute a "sufficiently compete" set of behaviors of a *mCell* for a given task domain, and the second is how one can design guidance for *mCells* to self-organize properly. The first issue will be addressed separately in other papers. In the following, we explore the issue of emergence and the ways to guide it.

3.3 Traditional Systems Behavior

Traditional engineered systems operate based on strictly pre-specified behaviors of all the components involved. Furthermore, the possible interactions between the components are also restricted. The complete specification is the guarantee of proper system functions and the complete conformance is required for system realization.

In order to show the behavior of conventional systems and compare it with our proposed CSO systems, in the following discussion we assume that the behaviors of all functional component can be "freely" and "linearly" combined to form system behaviors.

Let's consider the mobility of an automobile. To make it simple, we assume that, as a physical system, an automobile is composed of a steering system and a drivetrain system, i.e.,

Auto = {SteeringSystem, Drivetrain}.

Let's use *beh* to denote "behavior" and assume:

Beh(Auto) = Move

Beh (SteeringSystem) = Steer = {left, right, ahead}

Beh (Drivetrain) = Drive = {forward, back, stop}

Then we have:

*Beh (Auto)= Beh(SteeringSystem)*Beh(Drivetrain)*

Or

Beh (Auto)=
{left, right, ahead }{ forward, back, stop}*

The operator "*" can be defined based on the dependency of different behaviors. For the automobile example, the total possible overall system behaviors are calculated as 3 x 3 = 9. We can write these possible combinations in a matrix form as indicated below in Equation (3).

$$Beh(Auto) = [L, R, A]^T \times [F, B, S] = \begin{bmatrix} LF & LB & LS \\ RF & RB & RS \\ AF & AB & AS \end{bmatrix} \qquad (3)$$

3.3.1 Internal Physical Constraints

The matrix above shows all the possible behaviors of an automobile system. *L, B, A* stands for steering *left*, *right* and *ahead*, respectively*;* and *F, B, S* for *forward*, *backward*, and *stop*, respectively. In many system designs, it is often the case that not all possible behavioral combinations are allowed for *internal physical* reasons. In this circumstance, a dependency matrix may be applied. For example, if *left-stop* and *right-stop* are not allowed, one can introduce the following dependency matrix:

$$Dep(steer, drive) = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{4}$$

Combining (3) and (4), we have:

$$\begin{aligned} &Beh(Auto) \\ &\quad = Dep(steer, drive) \times [L, R, A]^T \times [F, B, S] \\ &\quad = \begin{bmatrix} LF & LB & 0 \\ RF & RB & 0 \\ AF & AB & AS \end{bmatrix} \end{aligned} \tag{5}$$

Equation (5) shows that the constraining dependencies between different behaviors of components limit the behavior of the overall system. When more than two components are involved, the dependency matrices can become significantly large and complicated.

In traditional engineering design, it is assumed that the designer has a *complete* map of all the constraining dependency matrices. Although this might be the case for simple design cases, the increasing level of complexity of recent and future engineered systems threatens this “taken for granted” wisdom: we may not have a complete, not event sufficiently partial, understanding of the matrices. The result of this lack of understanding can be loss of functions or catastrophes. Assuming the complete map is not obtainable, the research questions will be how one can make the system, or the components of the system, to deal with these dependencies by themselves. We will come back to this question later.

3.3.2 External Environmental Impact

In addition to *internal physical* constraints, the *external environmental* may also limit the behaviors of functional components and hence restrict the system behaviors. We call such constraints *environmental impact* or *EI*.

For our automobile example, given the environmental impact *EI (auto),* the system behavior will be constrained as indicated below.

$$Beh\ (Auto) = Beh(Auto)*EI(Auto) \tag{6}$$

This same Equation (6) can also be applied to functional components. For the SteeringSystem, assuming the environment is an ally, there is way to move either *left* or *right*, then moving *ahead* will be the only possibility, as indicated below in Equation (7).

$$\begin{array}{l} EI_{ally}(steer) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \\ \qquad\qquad\qquad\ \ L \quad R \quad A \end{array} \tag{7}$$

Furthermore, assuming the “destination” is at the end of the ally, then for Drivetrain, the environmental impact can be captured by the following matrix:

$$\begin{array}{l} EI_{dest}(drive) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \\ \qquad\qquad\qquad\ \ F \quad B \quad S \end{array} \tag{8}$$

The overall EI of the automobile is calculated through the similar way of Functionality:

$$\begin{aligned} EI_{ally}(move) &= EI_{ally}(steer) \times EI_{dest}(drive) \\ &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \times \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \end{aligned} \tag{9}$$

Now we have the overall behavior of the system under the influence of the environment is to *move ahead (A)* and *forward (F)*, as indicated below.

$$\begin{aligned} Beh(Auto) &= Beh(Auto) \times EI_{ally}(Auto) \\ &= \begin{bmatrix} LF & LB & LS \\ RF & RB & RS \\ AF & AB & AS \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ AF & 0 & 0 \end{bmatrix} \end{aligned} \tag{10}$$

If there is an obstacle in the environment, then we may have the following system behavior: system fails to act.

.

$$\begin{aligned} Beh(Auto) &= Beh(Auto) \times EI_{ally}(Auto) \times EI_{obs}(auto) \\ &= \begin{bmatrix} LF & LB & LS \\ RF & RB & RS \\ AF & AB & AS \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{aligned} \tag{11}$$

3.3.3 Implications

Limited variability and little adaptability: In traditional system design, the intention is to limit the behaviors of functional component only the ones that is intended. Any extra behavior is considered either as a waste or as a risk. The results of such design are the systems that have limited variability and little adaptability.

Unintended behaviors and their dependencies: Although only intention unintended behaviors are not welcome, for complex engineered system, when the number and the level of sophistication of components increase, unintended component behaviors become unavoidable. Furthermore, unintended dependencies may not be fully counted as indicated in Equation (4) and (5). When the ratio of unintended behaviors and dependency increase as system complexity levels, it may become difficult to keep the system within the bound of intended behaviors, leading to failures or accidents.

Environmental impact: Environmental impact can either guide the system behavior, as shown in Equation (10), or hinder it, as in Equation (11). Without the full knowledge of how environment impact may be, a designer can hardly device intended behaviors to deal with unknown environments. Recent "death" of Mars rover "Spirit" is an example of this case. Fixed and fully intended system behaviors made it impossible for "Spirit" to work around the "new seen before" situations.

Little inherent system "intelligence": It may not be fair to ask for "intelligent" behavior of mechanical systems unless a computer program is devised to do so. It is however fair to say that the mechanical systems composed by conventional approaches are inherently inadequate for dealing with unforeseeable, either endogenous or exogenous, changing situations. It is enlightening to see how biological systems, and even some chemical systems, may "smartly" live through unpredictables through their capabilities resulted from emergence.

3.4 Emergence of CSO systems Behaviors

Given the above definitions and discussion, it becomes straightforward to analyze the behavioral space of CSO systems and to evaluate the potentials that this huge behavioral space may bring. Assuming a CSO system, CSO1 is composed of two *mCells*, mCell1 and mCell2, and they each possess identical set of behaviors. We have:

$$\begin{aligned} Beh(CSO1) &= Beh(mCell1) \times Beh(mCell2) \\ &= [b_1, b_2, ..., b_p]^T \times [b_1, b_2, ..., b_p] \\ &= \begin{bmatrix} b_1 b_1 & b_1 b_2 & ... & b_1 b_p \\ b_2 b_1 & b_2 b_2 & ... & b_2 b_p \\ : & : & ... & : \\ b_p b_1 & b_p b_2 & ... & b_p b_p \end{bmatrix} \end{aligned} \quad (12)$$

In our research, we assume *mCells* are multifunctional in the sense that the p in Equation (12) can be large. For a CSO system having n *mCells*, the possible system behavior space can be as large as n^n. When each behavior potentially has multiple parameters, the potential behaviors space can be unimaginably bigger than n^n, providing a fertile ground for *emergence* at system level.

In our current research, we explore CSO systems with homogeneous *mCells*. Following the stem cell analogy, we consider that the initial homogeneous *mCells* with multiple behavioral capabilities will, during the process of emergence, differentiate and find their "specialty" behaviors during the period of task execution. We expect that this self-organized emergence may create functional blocks consisting of multiple *mCells*, as organs forming in biological systems. Once a task is accomplished, or the environment changes, the functional blocks may dissolve by themselves and the *mCells* will continue to renew their differentiation and form new functional blocks. The high level of self-organizing and redundancy ensures the robustness and resilience of the system, i.e.,

- The enormous size of the potential behaviors resulted from the cellular formation of the system provides functional basis for "unforeseeable" functional requirements, increasing the system robustness, and
- The redundancy of *mCells* together with the large number of *mCells* makes the role of single *mCell* insignificant during the emergence of the system behaviors. Failures with a single *mCell* can be dealt with by other similar *mCells*, leading to high level system resilience.

From a design perspective, developing CSO systems is a difficult task. As much as we attempt to understand how biological systems develop their *emergence*, we face enormous challenge in developing such fruitful *emergence* in our engineered systems. In our research instead of "free emergence" we target "guided emergence" by providing rules for *mCells* to self-organize and for desired system behavior to emerge. Two fundamental issues must be addressed. First relates to design information representation. We have introduced a design DNA or dDNA based representation scheme to capture CSO system information at the cellular level (Jin et al, 2008; Geroge et al 2011). The second issue has to do with devising mechanisms to guide self-organizing. In the following, we introduce a field driven approach to regulating *mCells*' behaviors in order to induce system level emergence.

3.5 Cellular Differentiation and Field Based Behavior Regulation

3.5.1 Local Behavioral Cellular Differentiation

In the biological world, the function of an organism is realized by a collection of different types of cells working together. While all stem cells possess the same DNA information and have identical properties and structures, the developmental process allows stem cells to differentiate into different cell types by responding to specific chemical signals. Although differentiated cells still hold the same DNA, the biological regulation (i.e., gene regulation) enable them express different "portion" of the DNA "string", leading them to producing different proteins. The distribution of the chemical signals, also called *morphogene*, controls the biological regulation hence the shape and organ formation. Without cell differentiation, there will be no advanced biological systems existing in today's world.

In our CSO systems, our *mCells* need the similar differentiation capability in order to collectively become a functional system. Instead of producing different proteins, our differentiated *mCells* produce different actions. Instead of being triggered by chemical signals, our *mCells* differentiation must be triggered by the *functional requirements* and *environmental constraints.*

3.5.2 Field driven Behavior Regulation (FBR)

In biological systems, the distribution of morphogene creates a "chemical field" that triggers cellular differentiation. Depending on the "location" of a cell in the "chemical field", the cell will produce the protein which is specific to that "location". In our CSO framework, we extend the "chemical field" into more general "fields" and introduce a "field driven behavior regulation" or FBR for guiding cellular behaviors and building CSO systems.

For a CSO system, the sensory capabilities of its *mCells* are pre-defined and given. In this case, whenever a task (defined by its FRs) and an operation environment (may or may not be fully known) are given, we can define a *task field* which captures the external world to a *mCell* encompassing both task requirements and environmental conditions. We have:

Definition 6 (Task Field): tField := {FR, Env, S }

where, FR: global function requirements;
Env: environmental constraints;
S: sensory information of a mCell;

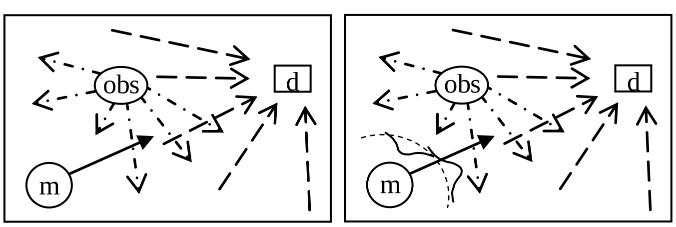
Figure 1(a) shows a simple example of *tField.* A *mCell m* is moving to its destination *d* with the potential of encountering an obstacle *obs* in a two dimensional space. In this case, the destination *d* can be considered as an attractor that creates an *attraction field,* capturing the task requirements; and in the similar way, the obstacle, *obs*, creates a *repelling field*, characterizing the operation environment. It can be seen from Figure 1(a) that the task field *tField* serves as a "complete" context for a *mCell* to operate.

Since *mCell* differentiation is about behavior distribution, an *mCell* must be able to determine its *behavior field*, or *bField* for short, based on the given *task field*. We use FBR_{FD} to denote the transformation from a *task field* into a *behavior field* and introduce the following definition:

Definition 7 (Behavior Field): $bField = FBR_{FD}(tField)$

where, FBR_{FD}*: FBR operator for field transformation;*
bField: behavior field;
tField: task field.

Figure 1(b) shows a simple example of *bField.* A *mCell m* is moving in the *task field* caused by the destination *d's attraction field* and the obstacle *obs*' *repelling field*. Based on some given FBR_{FD}, the *mCell* creates a *bField* around itself.



(a) *m* moves to *d* in *tField* (b) *m* moves to *d* in *bField*

Figure 1: An Example of Task Field and Behavior Field

There can be different ways to represent the concept of *bField*. One may associate "rewards", "risks", or "times" with different "locations" for a *mCell*. The "locations" can be defined as real 2- or 3-dimention spaces or n-dimension virtual spaces depending on the task domain and *mCell* properties. In our current research, we associate a *mCell*'s "behavior distributions" with its surrounding "locations", and we further call this "behavior distribution" *behavior profile*, or *bProfile*. Therefore, we introduce the following definition.

Definition 8 (Behavior Profile):
$bProfile = FBR_{FD}\ (tField, B)$

where: $bProfile := \{(b_1, p_1), ..., (b_n, p_n)\}$; & $[b_i \epsilon B,\ 0 \le p_i \le 1,\ 1 \le i \le n]$ *indicates (behavior, probability) pairs for a mCell to choose its actions, and n is the total number of possible behaviors that the mCell can perform.*
tField: task field
B: mCell's behavior set

Figure 2 illustrates how FBR_{FD} works in a CSO system: for a given task in a certain environment, the Function Requirements (FRs) and Environment Locally, a *mCells* has different sensor information and different current actions, if the *state* of the current situation meets the FRs, just keep doing the current actions. But when the current state is different from FRs, the decision needs to be made for the system to reduce the difference between the future State and FRs by applying FBR_{FD}.

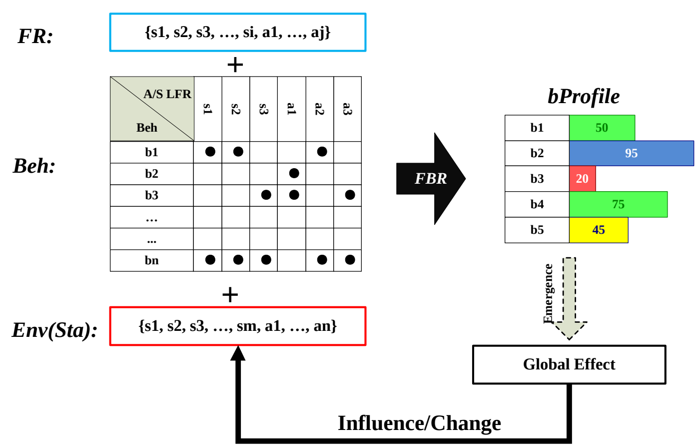


Figure 2. An Illustration Field driven Behavior Regulation (FBR) in CSO Systems

Given a behavior profile, a *mCell* still needs to make a decision to choose a specific behavior or action from the profiled behaviors. In our research, we introduce the second FBR operator called FBR_{DM} for behavior selection:

Definition 9 (Behavior Selection): $b = FBR_{DM}\ (bProfile)$

where: $bProfile := \{(b_1, p_1), ..., (b_n, p_n)\}$; & $[b_i \epsilon B,\ 0 \le p_i \le 1,\ 1 \le i \le n]$.
b: selected behavior $b \epsilon B$.

Our "field driven" approach mimics the natural systems, although we define and apply "information fields" rather than physical fields such as Ph gradient in chemistry and gravity field in physics. In our framework, both tasks (or task requirements) and operation environments are interpreted in terms of *tFields* and transformed by *mCells* into their *behavior fields* or *behavior profiles*. The hope is that the field concept can be used to uniformly represent the world to the *mCells* and therefore allow them to self-organize and emerge as a single system.

For any single *mCell*, it is the goal that the current environment is the same as the functional requirements require (in the right state). It can be said that in our FBR framework, such ideal goals are represented as "attractors" in the *tField*. If the given task is constant but the environment is changing, then the resulting *tField* will have changing "attractors", and the *mCells* will pursue these changing "attractors" as part of their emergent behavior in the changing world. This way, the "guided emergence" problem is translated into the problems of *tField* representation and *FBR* design. In the following section we present computer simulation based case studies to illustrate how our "behavior based design" and "field driven behavior regulation" can be effectively applied to CSO systems.

4 CASE STUDIES AND DISCUSSION

The previous sections introduced our behavior based design approach and discussed the potential of applying field driven behavior regulation mechanisms to facilitate emergence of CSO systems. To investigate how such an approach and mechanism can be applied to CSO systems design, a set of computer simulation based case studies were performed with the intention of addressing the following questions:

- What constitutes the task and behavior fields?
- What is the benefit of using the concept of behavior field?
- How will locally regulated behaviors emerge into desired global effects?
- How will the behavior transformation (FBR_{FD}) and behavior selection (FBR_{DM}) impact the global system behavior?

In the following subsection, we present two case studies. The first case study is designed to investigate the concept of *field* and the second one for demonstrating FBR effectiveness.

4.1 Case Study 1: Single Exploration Cell

The overall task for this case study is for one *mCell* to travel to a given destination in a unknown environment. The two functional requirements are:

FR1 = "move to destination", and
FR2 = "avoid obstacle".

The *mCell* can decide the *direction* of movement, so the two behaviors are:

b_1 = "move to the direction toward destination", and
b_2 = "move away from the direction to obstacle".

We further assume that the obstacles between the *mCell* and the destination can be everywhere with any density and that the *mCell* can always sense the location of the destination and can sense the locations of the obstacles only when they are within a certain range. Given the two functional requirements, the sensor information and current actions, a *mCell* needs to decide which "action", i.e., direction, to take.

4.1.1 Task Field

The task field for this example is composed of the *attraction field* of the destination and the *repelling fields* of various obstacles, and more than one obstacle can exist at any time. We use parameter θ to represent the *attraction field* and β the *repelling field*, as show in Figure 3. Combining the two, we have task field for *mCell m* :

$$tFiled_m = \{\theta; \beta_1, \beta_2, ..., \beta_n\};\ where,\ n = no.\ of\ obstacles$$

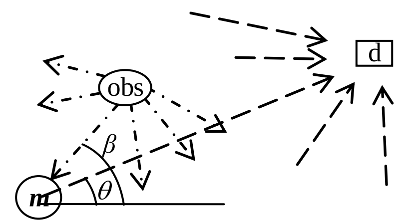


Figure 3: Tasks Field for *mCell m*

4.1.2 Behavior Regulation

As described above, in CSO systems field-driven behavior regulation has two steps, i.e.,

Step1: Transform *tField* into *bFiled* through FBR_{FD}
Step2: Select a specific behavior/action through $FBR_{DM.}$

Behavior field and FBR_{FD}: In this example, the *bField* or *bProfile* determines the likelihood in which a *mCell* is taking its next move into direction α, and the likelihood the *mCell* is avoiding this direction due to the existence of obstacles. The distribution of these two likelihoods around the 360 degree circle around a *mCell* constitutes the *bField* or *bProfile* of the *mCell*. Specifically, for one destination and on obstacle, we introduce the following FBR_{FD}:

$$bField_m(\alpha) = FRB_{FD}(tField_m, B) = \{\alpha, p_\alpha, q_\alpha\}$$
$$= \{\alpha, \frac{1}{\sqrt{2\pi}} e^{-\frac{(\alpha-\theta)^2}{2}}, \frac{1}{\sqrt{2\pi}}\left(1 - e^{-\frac{(\alpha-\beta)^2}{2}}\right)\}$$

where, α*: direction for the next move*
p_α*: probability that direction* α *should be taken*
q_α*: probability that direction* α *should be avoided*

Behavior selection and FBR_{DM}: After the behavior field is established, a *mCell* needs a mechanism for behavior selection. In this case study, we define two types of behavior selections: "select the best" and "select any one good enough", as indicated below.

FBR_{DM-B} = [Select the action with the highest probability in the *bField*]

FBR_{DM-G} = [Select any action, randomly from the actions that has a bigger than threshold probability in the *bField*]

In this case study, we will show how the above mentioned behavior field can be useful and the effective of applying different behavior selection strategies.

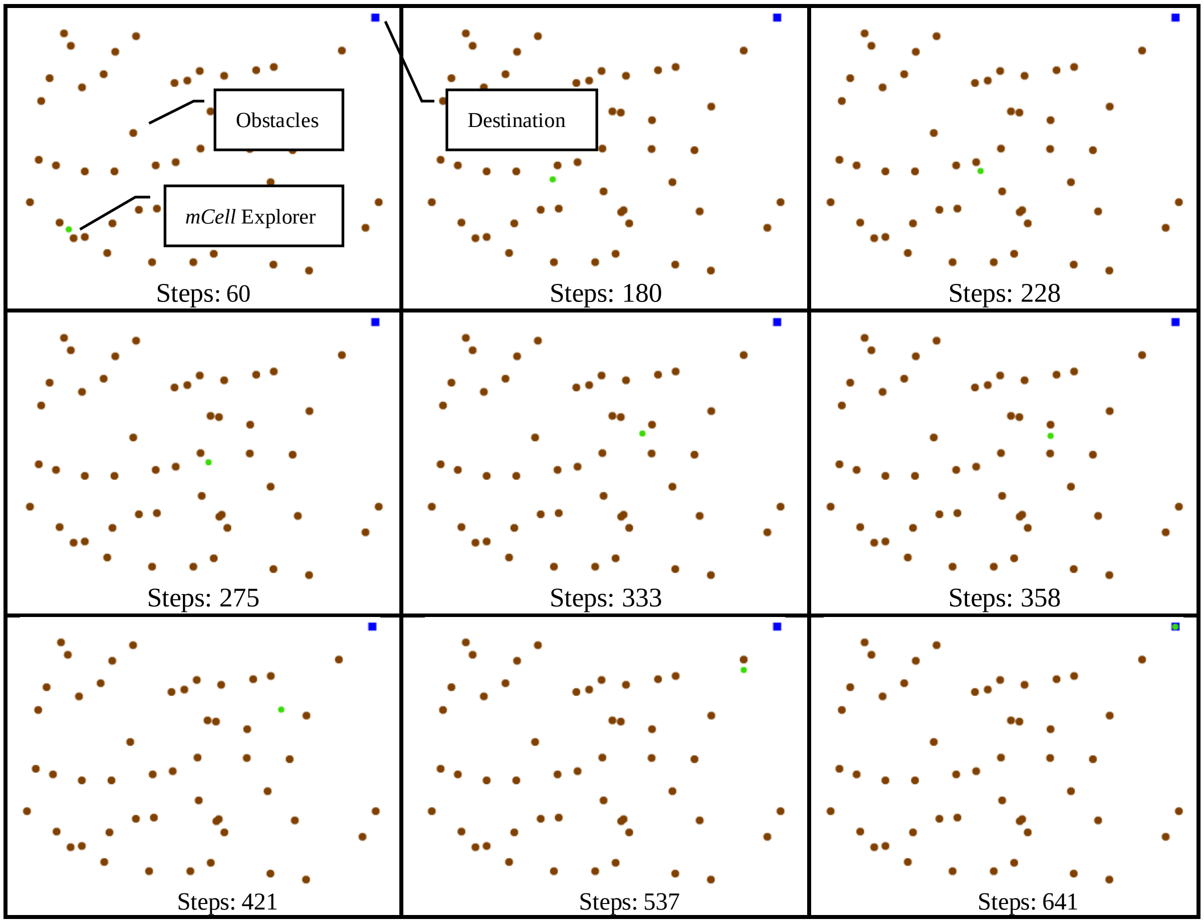


Figure 4: Simulation Results of a Single *mCell* Exploring in a Random Obstacle Field

4.1.3 Simulation Results

Figure 4 shows the time sequence of screen dumps of one of our simulation runs, with time steps indicated at the bottom of each box. As shown in Figure 4, a single explorer *mCell* can travel from a randomly assigned position on the left to a given destination on the upper right. Both the *mCell*'s initial position and the positions of all obstacles are randomly generated for each simulation run.

In this case study, the *mCell* acts solely based on the task assignment (represented as FRs) and its sensory information without memory and planning. The FBR_{FD} constantly transforms the perceived *task field* into local *behavior field*, allowing the *mCell* to "know" what are possible valid behaviors that can be performed at each moment. Furthermore, the FBR_{DM} converts behavior or action potential into specific actions. By splitting the process of FBR into two steps, a designer can make various combinations and find the good ones for his/her task domain.

As one may imagine, when the density of obstacles increase, the *mCell* may be trapped on its way and not be able to reach the destination. Our simulation results verified this statement. To investigate how different FBR strategies may influence the "success rate" of the simulation runs, we examined two "behavior selection" strategies, i.e., $FBR_{DM\text{-}B}$ (select the best) and $FBR_{DM\text{-}G}$ (select from good enough, i.e. top 40%, randomly). We ran 500 test runs for each obstacle density for $FBR_{DM\text{-}B}$ and $FBR_{DM\text{-}G}$, respectively, and calculate the success rate based on the 500 runs. Figure 5 shows the comparison result with 40 to 120 randomly assigned obstacles.

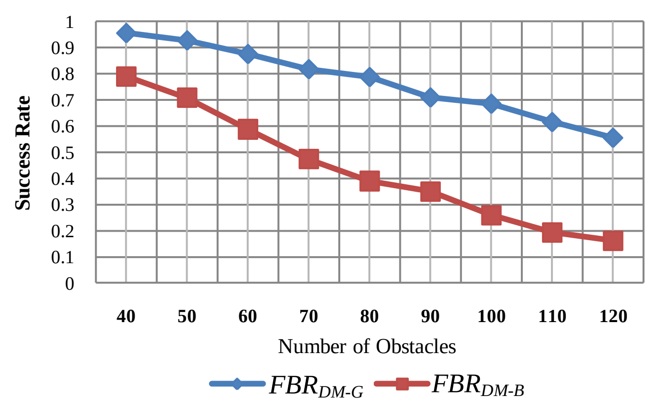


Figure 5. Comparison of "Select the Best" *($FBR_{DM\text{-}B}$)* and "Select from Top 40% Randomly" ($FBR_{DM\text{-}G}$)

Figure 5 shows that overall the "select from good enough randomly" works better than "select the best" and that as the density of obstacles increases the advantage of the former increases. From a CSO system development perspective, the result is interesting in two ways. First it indicates that behavior regulation strategies have profound impact on individual *mCell*'s performance, and secondly the "randomness" seems to bring "intelligence" into the system mechanically.

With the "select the best" strategy, a *mCell* always targets on one single best direction in deciding on their next move. When the obstacle density is low, this strategy can likely produce ideal performance in which both time and energy can be saved. The reason behind is that with limited number of obstacles distributed sparsely, there is close to zero likelihood that the *mCell* may get trapped by its own "best" calculation. When the density of obstacles increases, however, much more likely the "traps" exist in the field, resulting in lower success rate for this strategy.

The "select from top 40% randomly" strategy may not work perfectly in terms of saving time and energy. However, when the environment becomes more unpredictable and unfriendly, the *mCell* can robustly sustain the environmental change and maintain its performance. Thanks to the randomness of behavior selection, the "traps" may be overcome by the *mCell* through internal variability. Only the intrinsic variety of the system (i.e., *mCell* in this case) can concur the variety of the environment (Ashby 1958).

4.2 Case Study 2: CSO Mover System

In the single *mCell* case study, we demonstrated how *tField* can be defined and how *bField* can be generated and *behavior selection* be carried out through field-driven regulation (FBR). To investigate how FBR may impact on the emergence when multiple such *mCells* work together for a single task, we conducted the second case study. In Case Study 2, the task for multiple identical *mCells* is to move an object from a start point to a destination point in a two dimensional unknown environment with all the obstacles randomly distributed in the field in the same way as in Case Study 1. The *mCells* are limited in action: they only push the object from their center to the object's center. At a given time, a *mCell* must decide on which direction to push the object. The overall movement of the object will be the result of the emergent behavior of all the *mCells* pushing the object.

In this case study, all *mCells* can only push from their centers to the object's center with the same force, and the overall movement of the object is the emergence of all *mCells*' relative locations. The behavior of each *mCell* is to choose a “right” location to push the object. The three functional requirements are:

FR1 = “stay close to the object”,
FR2 = “push object to destination”, and
FR3 = “avoid obstacles”.

A *mCell* can choose a relative location to the object, so the three behaviors are:

b_1 = “move to locations as close as possible to the object”,
b_2 = “push the object towards destination”, and
b_3 = “push the object away from obstacles”.

We assume that all the *mCells* have similar setup as the previous case study; they can sense the destination anywhere and they can only sense the obstacles within a certain range.

4.2.1 Task Field

Similar to the previous case study, we also use parameter θ to represent the *attraction field* and β the *repelling field*. In addition to those two, this case study introduces a new *attraction field d* as the relative distance from *mCell* to the Object. The related task field is shown in Figure 6 and besides *mCell* m there are *mCells* i, j and k in dash line. Combining the three, we have task field for *mCell m*:

$tFiled_m = \{d, \theta; \beta_1, \beta_2, ..., \beta_n\}$; *where, n = no. of obstacles*

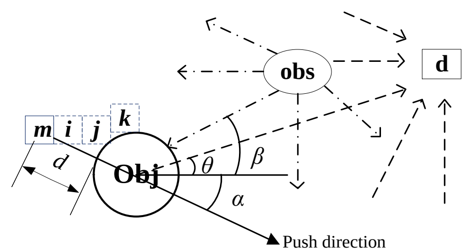


Figure 6: Tasks Field for *mCell m*

4.2.2 Behavior Regulation

The two steps behavior regulation describe in the previous case study is still valid in this case:

Behavior field and FBR_{FD}: In this example, the *bField* or *bProfile* determines the likelihood in which a *mCell* is taking its next move to either stay in the current location to push the object or move to other locations because the relative distance is too far, the pushing direction is towards a collision or the pushing direction is away from destination. The relative location for *mCell* is represented by α and d. For one destination and on obstacle, we introduce the following FBR_{FD}:

$$bField_m(\alpha, d) = FRB_{FD}(tField_m, B)$$
$$= \{\alpha, d, p_d, p_\alpha, q_\alpha\}$$
$$= \{\alpha, d, \frac{1}{\sqrt{2\pi}} e^{-\frac{d^2}{2}}, \frac{1}{\sqrt{2\pi}} e^{-\frac{(\theta-\alpha)^2}{2}}, \frac{1}{\sqrt{2\pi}} \left(1 - e^{-\frac{(\beta-\alpha)^2}{2}}\right)\}$$

where α: the angle corresponding to an arbitrary predefined coordinate
d: the related distance.
p_d: probability that distance d should be taken
p_α: probability that pushing direction α should be taken
q_α: probability that pushing direction α should be avoided

Behavior selection and FBR_{DM}: After the behavior field is established, a *mCell* needs a mechanism for behavior selection. In this case study, we assume that the *mCell* will change their location when the probability is below a threshold instead of choosing the “best” locations.

FBR_{DM} = [Select any action, randomly from the actions that has a bigger than threshold probability in the *bField*]

In this case study, we will show how the above mentioned behavior field can be useful and effective for not only a single *mCell* case but an emergent system of multiple *mCells*.

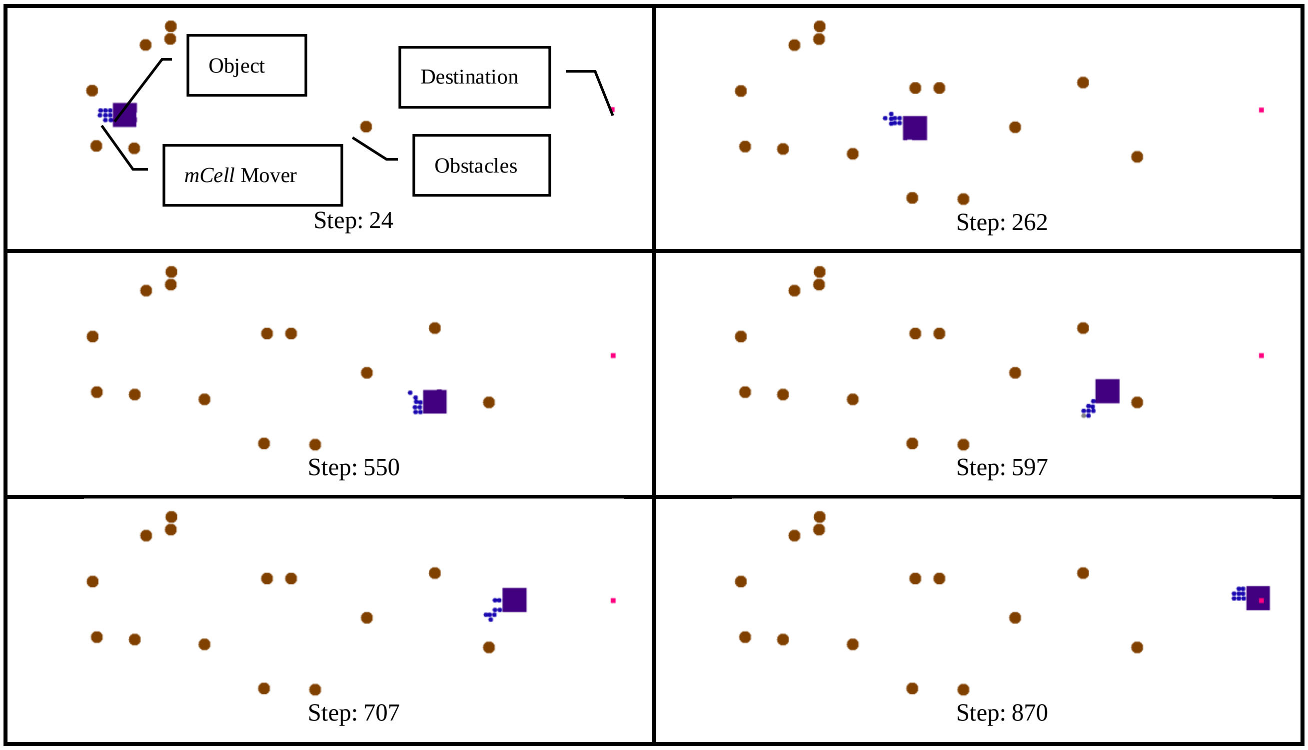


Figure 7. Simulation for Design Case 2, CSO Field Move

4.2.3 Simulation Results

Figure 7 shows the time sequence of simulation screen dumps with time steps indicated at the bottom of each block. All each *mCell* chooses a location to push the blue square Object. Each *mCell* attempts to choose a "highly" recommended zone and move into it when the zone of its current location has the probability below the threshold. There is no explicit communication between the *mCells*, reducing the need for more design efforts. However, the *mCells* interact indirectly by avoiding overlapping with each other. Our simulation results showed that with the setup of this simulation, in almost all simulated test runs, the *mCells* were successful in pushing the square object into its destination.

One advantage of this behavior based design is that the shape of the Object and therefore the shape of the overall system are not predefined and limited in any way. The *mCells* observe the world and decide on their behaviors locally, as the global behavior and result emerge. Based on Kolmogorov complexity measure (Li and Vitanyi 2008), our CSO system of multiple *mCells* can be considered highly complex since the states of each *mCell* changes dynamically without certainty and it takes a rather long description to capture the whole system. However, using FBR makes it possible to regulate *mCells*' behaviors and to lead the emergence process to a productive direction.

Figure 8 illustrates the dynamically changing behavior field (*bField*), and how *mCells* choose their behaviors (i.e., locations) through FBR. As shown in Figure 8 the different current situations introduce two different *bFields*. Dependng on the relative locations of the destination, obstacles and the object, the field changes as shown in color changes. Different colors in Figure 8 correspond to different probabilities, as indicated in the figure. The *mCells* try to choose the "green" or "yellow" zone to occupy. Through the use of field driven behavior regulations (FBR), the system dynamically adapts to its new situations even for the simple designed *mCells* of limited capability (can only push from its center to the object's center). The system can moving the object in an unknown environment by *mCells* using the fields as their dynamic vision of the world. It is conceivable that the *bFields* and FBR concepts can be applied to those task situations where physical fields and chemical fields exist. We plan to expand our application example domains to assess the effectiveness of our field and FBR concepts.

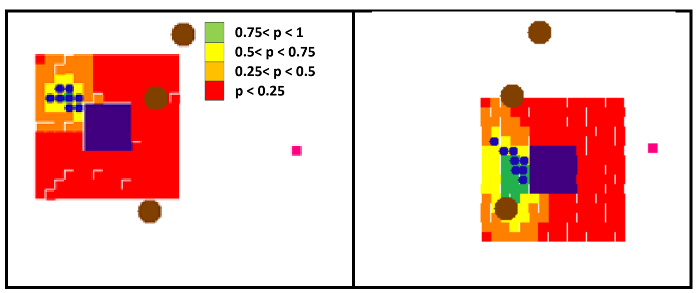


Figure 8: Illustration of the Dynamic *bField* of the CSO Mover in the Simulated Field of Obstacles

In our simulation test runs, we also examined how the system might perform if some *mCells* become inactive. Figure 9 shows the resilience of the overall system when some of the *mCells* become "dead" during the simulation. There are four *mCells* that were deactivated at the step 400, since the system is fully decentralized, deactivated *mCells* had little influence to the rest of the *mCells* in the system. This way, although the system losses its performance due to the loss of *mCells*, it could still successfully accomplish the task of moving the object to its destination, showing the system resilience.

Because CSO systems are decentralized and have redundancies maintained among its *mCells*, they are more resilient than the systems with specified local functional components. When one part of the system fails, other nearby *mCells* can modify its functionality and redistribute their functions. This way, the system can not only adapt to the environmental change but also to the system change.

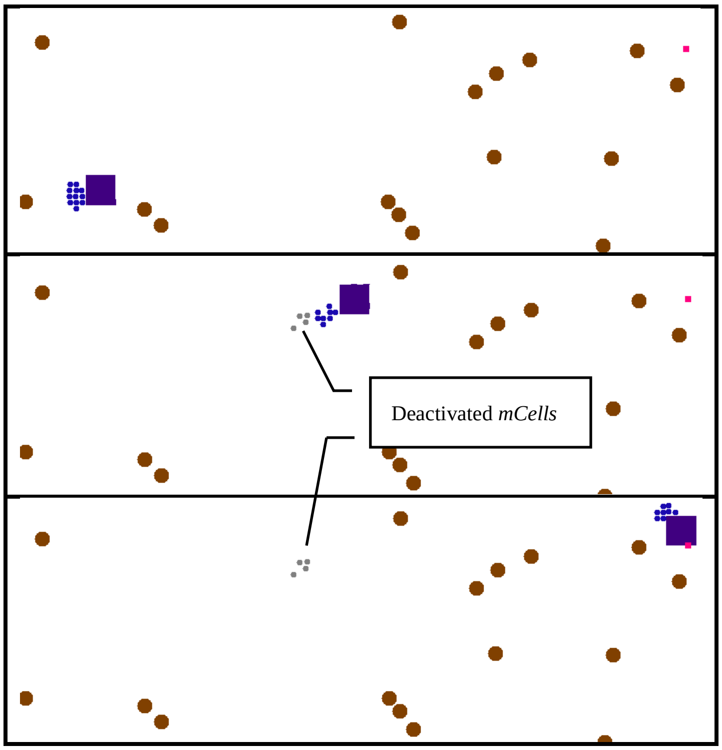


Figure 9: Resilience Test by Deactivating 4 of 12 *mCells* at Step 400

5 CONCLUSIONS

This paper presents a behavior based design approach to multi-agent complex mechanical system development. This approach focuses on individual agent's behaviors instead of structures, maps system functional requirements into agent behavior sets, and devises *field-driven behavior regulartion* mechanisms for agents to self-organize in response to requirement changes, environmental situation changes, and system changes. The behavior based design approach embeds design information into every individual agent in the system, achieving the maximum level of design information redundancy and making it possible for the system to self-organize, self-repair and self-reconfigure for high level robustness and resilience. The case studies and simulation results have shown that our behavior based design approach allows *mCells* to utilize their limited vision to choose the right actions as they perform collectively in a CSO system. The emergence process is controlled and maintained through a field based regulation (FBR) mechanism only at local level, allowing high level adaptability at the system level. The behavior based approach also links the system functional requirements and agent local behaviors, providing a way for mapping global effects and local decision-making process in designing CSO system.

Our current work on this research includes expanding the case study into more sophisticated problem domains, examining trade-offs of having various combinations of *mCells* including heterogeneous ones and between swarm *mCell* structures as we presented in this paper and more structured organizations that require more tight connections, e.g., physical dockings, among *mCells*.

This paper is based on the work supported in part by the National Science Foundation under Grant No. CMMI-0943997. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

6 REFERENCES

[1] Alon, U. (2007) “Network motifs: theory and experimental approaches”, Nature Reviews Genetics Volume 8, 450-461.

[2] Ashby W.R. (1958). “Requisite variety and its implications for the control of complex systems.” *Cybernetica* 1:2, p. 83-99.

[3] Audesirk, G., Audesirk, T., Byers, B.E., (2007). “Biology: Life on Earth with Physiology (8th Ed.).” Benjamin Cummings, San Francisco.

[4] Bongard, J., Zykov, V. and Lipson, H. (2006), “Resilient Machines through Continuous Self-Modeling", *Science*, 314. no. 5802, pp. 1118-1121.

[5] Dickinson, M.H. (1999). “Bionics: Biological Insight into Mechanical Design.” Proc. of the National Academy of Sciences of the USA, 96/25:14208-14209.

[6] George Zouein, Chang Chen, and Yan Jin (2010): Create Adaptive Systems through “DNA” Guided Cellular Formation, *Proceedings of First International Conference on Design Creativity (ICDC2010)*, 2010, Kobe, Japan

[7] Gilpin K, Kotay K, Rus D and Vasilescu I (2008) Miche: modular shape formation by self-disassembly. *International Journal of Robotics Research* 27: 345–372.

[8] Jin, Y., Li, W., Lu, C-Y.S., (2005). “A Hierarchical Co-Evolutionary Approach to Conceptual Design,” Annals of the CIRP, 54/1:155-158.

[9] Jin, Y. G. Zouein, S. Lu, “A Synthetic DNA based Approach to Design of Adaptive Systems”, CIRP Annals – Manufac-turing Technology, Vol. 58/1, pp.153-156, 2009

[10] Kitano, Hiroaki (2002), “Systems Biology: A Brief Overview”, Science 1 March 2002: Vol. 295 no. 5560 pp. 1662-1664

[11] Li, M. and Vitanyi, P. (2008) "An introduction to Kolmogorov complexity and its applications", Springer

[12] Lipson,H., (2007) “Evolutionary Robotics: Emergence of Communication,” *Current Biology*, Volume 17, Issue 9, pp R330-R332.

[13] Rus, D. and Vona, M., (2001). "Crystalline robots: self-reconfiguration with compressible unit modules." *Kluwer Autonomous Robots 10*, pp. 107–124.

[14] Shen W. M.; Will, P., Galstyan, A., Chuong, C. M.; (2004) " Hormone-Inspired Self-Organization and Distributed Control of Robotic Swarms," *Autonomous Robots*, Vol. 17, No. 1., pp. 93-105.

[15] Shen, W. M., Krivokon, M., Chiu, H., Everist, J., Rubenstein, M. and Venkatesh, J., (2006). "Multimode locomotion for reconfigurable robots." *Autonomous Robots*, *20(2)*, pp.165–177.

[16] Shen, W. M., Salemi, B., and Will, P., (2002). "Hormone inspired adaptive communication and distributed control for CONRO self-reconfigurable robots." *IEEE Transactions on Robotics and Automation, Volume 18, Number 5*, pp. 700-713.

[17] Shu, L.H. Lenau, T.A., Hansen, H.N., Alting, L. (2003). "Biomimetics Applied to Centering inMicro-assembly," Annals of theCIRP, 52/1:101-104.

[18] Suh, N. P., (1990). "The principles of design." *Oxford University Press*, New York, NY.

[19] Suh, N. P., (2001). "Axiomatic design." *Oxford University Press*, New York, NY.

[20] Unsal, C., Kilic, H., and Khosla, P., (2001). "A modular self-reconfigurable bipartite robotic system: implementation and motion planning." *Kluwer Autonomous Robots 10*, pp. 23-40.

[21] Vincent, J.F.V., Bogatyreva, O., Bogatyrev, N., (2006). "Biology Doesn't Waste Energy: That's Really Smart." Proceedings of the SPIE, 6168:1-9.

[22] Yim, M., Duff, D. G. and Roufas, K. D., (2000). "PolyBot: a modular reconfigurable robot." *Proceedings of the ICRA IEEE International Conference on Robotics and Automation*, pp. 514-520.

[23] Yu, C. H.; Haller, K.; Ingber, D.; Nagpal, R.; , (2008) "Morpho: A self-deformable modular robot inspired by cellular structure," *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference* , vol., no., pp.3571-3578, 22-26.

[24] Yu, C.-H., Haller, K., Ingber, D. and Nagpal, R. (2008), "Morpho: A self deformable modular robot inspired by cellular structure," in Proc. of Intl. Conf on Intelligent Robots and Systems (IROS 08).

[25] Zouein, G., Jin, Y. (2008) "A biologically inspired DNA-Based Approach to Developing Cellular Adaptive Systems", Ph.D. Thesis, USC, Los Angeles, CA.

[26] Zykov, V., Mytilinaios, E., Adams, B. and Lipson, H. (2005). "Self Reproducing Machines." *Nature*, **435**, pp. 163-164.