Multi-Phase Genetic Programming: A Case Study in Sumo Maneuver Evolution

Jiming Liu Shiwu Zhang

Abstract

In this paper, we describe a new evolutionary computation approach, called *multi-phase genetic programming* (MPGP). The special features of this approach lie in its *variable-granularity representations* of chromosomes and their corresponding genetic operations. In the paper, we provide an overview of the MPGP approach as well as details on how the sumo maneuver evolution experiments are carried out and how the MPGP-based case study differs from others.

Index Terms

Multi-phase genetic programming (MPGP), sumo maneuver evolution, adaptive behavior.

I. Introduction

As has been demonstrated in laboratory as well as in real-world tasks, evolutionary computation can offer an effective means for solving optimization problems of various natures or for organizing reactive agent behaviors given different environmental conditions [1], [2]. In order to improve the performance of evolutionary computation, specific techniques have been developed by researchers catering to the specific complexities of problems on hand.

In this paper, we will present a new evolutionary computation approach, called *multi-phase genetic programming* (MPGP). The special features of this approach lie in its variable-granularity representations of chromosomes and their genetic operations in light of the observed evolutionary progress. Specifically, we will demonstrate the MPGP approach in an autonomous robot task where

Dr. Jiming Liu is with the Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong. *E-mail: jiming@comp.hkbu.edu.hk*

Mr. Shiwu Zhang is with the Department of of Precision Machinery & Precision Instrumentation, University of Science and Technology of China.

a sumo robot learns how to execute strategic pushing *maneuvers* in response to several opponent postures.

A. Practical Implications

An evolutionary algorithm often involves large populations of individuals as well as many generations of genetic operations. Thus, evaluating potential robot performance in the real world can be costly and time-consuming. Floreano and Mondada utilized *Khepera* to validate a neural network system evolved using a genetic algorithm, in which all populations were tested in the real world [3]. Floreano and Mondada's work showed that emergent navigation and obstacle avoidance behaviors can readily be obtained. However, the evaluations in their work were somewhat time-consuming; the evaluation of one generation, on average, took about 39 minutes.

Some of the earlier evolutionary robotic studies also used simulated robots to evaluate the performance of potential actions [4], [5], [6], [7], [8], [9]. In simulations, the external inputs to a robot and the reaction of the robot to the environment are often simplified. They may not accurately reflect the real world. Therefore, in some cases, the results obtained with simulated robots may not work on physical robots. In order to become more realistic, elaborated design efforts, such as detailed robot parameters, are taken into consideration. As Reynolds pointed out [10], [11], simulated robots may sometimes involve a robust noise-tolerant controller. As a result, the efficiency of simulations may be compromised.

Other studies have used a mixed approach: First, it evaluates individuals and evolves an adaptive control system in a simulated world. Next, it selects the best evolved individuals after some generations. Then, it passes them onto a physical mobile robot, such as *Khepera*, in a real environment for validation [12], [13], [14].

It can be noted that there is a clear need for the *right balance* between computational efficiency

and result validity in an evolutionary robotic system. In other words, it is important to combine the virtues of evolution in a simulated world and validation in the real world.

Unlike the above approaches, our present work adopts an on-line evaluation method that incorporates both evaluations in the simulated world and validations with a physical robot. *In particular, the robot sumo maneuvers are evaluated first in simulations, and the best maneuver evolved at each step is then executed and validated on a physical robot. This approach differs from the above mixed approach in that it immediately executes and evaluates the best individual during the performance of the robot. As will be shown in several sumo maneuver evolution experiments later, specialized maneuvers with respect to some possible changes in the shape and size of an opponent can readily emerge from our proposed on-line MPGP learning sessions. Also, our results show that both the efficiency and the effectiveness of evolution can be achieved.*

B. Organization of the Paper

The remainder of the paper is organized as follows: Section II states the problems and related issues to be addressed in our work. Section III provides the motivation for our work and an overview of the MPGP approach. Section IV concentrates on the sumo maneuver learning problem in order to illustrate the formulation and application of the MPGP approach. Section V describes several experimental case studies that validate the performance of evolving adaptive behavior strategies for a sumo robot based on the MPGP approach. It also discusses the features of the MPGP approach and compares the performance of the MPGP and GP approaches. Section VI compares the formulation as well as the operations of the MPGP approach with those adopted in the related studies. Section VII concludes the paper by summarizing the major results of our work and pointing out the direction for future extensions.

II. Problem Statements

This work is concerned with the design and development of an on-line evolutionary robotic system that incorporates evolution based on simulated robot-environment interaction and real-world validation with the best individual from each generation of the evolution. Central to this on-line behavior learning system is a multi-phase genetic programming (MPGP) approach that is aimed at enabling the robot to gradually acquire sumo maneuvers. It should be pointed out that our present work differs from Nordin and Banzhaf's on-line genetic programming for evolving behavior to control a miniature robot in real time [15], [16]. As in our work, the evolved individual is a sequence of basic behaviors, instead of machine codes as in Nordin and Banzhaf's work.

In order to achieve our objective, there are several specific issues that we need to deal with in the development of the MPGP-based system; namely,

1. how to exactly represent a robot maneuver?
2. how to model robot-environment interaction?
3. how to evaluate all the individuals in a population?
4. how to evaluate the best individual from each generation on a physical robot?
5. how to reuse the best individual for the future generations?

In the following sections, we will attempt to provide answers to the above questions.

III. Multi-Phase Genetic Programming (MPGP)

Genetic programming (GP) applies genetic manipulations to the functions and operators of a control program or other representations of a controller in an autonomous system. It was firstly proposed by John Koza in 1992 [17]. Generally speaking, in GP, the entities to be evolved consist of a function set and a terminal set. The function set often corresponds to the set of ordinary arithmetic functions and conditional operators, whereas the terminal set corresponds to the set of

variables and constants. For detailed background on GP, readers are referred to [17].

GP and their variations can readily be implemented to model and acquire a robot's adaptive behavior in an unknown environment. As illustrated by an experimental example given in [17], GP has been used to evolve a subsumption architecture for simulated robots capable of performing reactive wall-following and box-moving behaviors in an environment.

A. Motivation

In commonly-used GP formulations, there is no fixed structure for evolved objects. As a result, the space for searching an optimal program can become enormously large. Another shortcoming that may limit real-world GP applications is that the performance of GP can sometimes be undermined during the evolution of a program as the genetic manipulations are carried out at a *predefined granularity*, *i.e.*, there is no differentiation among the function or terminal sets as far as their complexities are concerned. We believe that in order to make GP most effective, the granularity of function and terminal sets should match the granularity of task constraints. That is, the right GP granularity should match the right problem characteristics (the *characteristics* may be reflected in the sensory measurement of a robot).

In order to speed up the evolutionary process in GP and, at the same time, to enable the dynamic adaptation of GP granularity, we have developed a multi-phase genetic programming (MPGP) approach. In the formulation of MPGP, the evolutionary process varies its granularity in light of the characteristics of a problem on hand. *The granularity of MPGP is adjusted by changing the granularity of function and terminal sets through different phases.*

Figure 1 provides a schematic diagram illustrating the basic ideas of MPGP. As illustrated, MPGP splits a single function in its search space into *different sub-functions* based on the evolutionary progress, which reflects the effectiveness of the present MPGP phase with respect to the

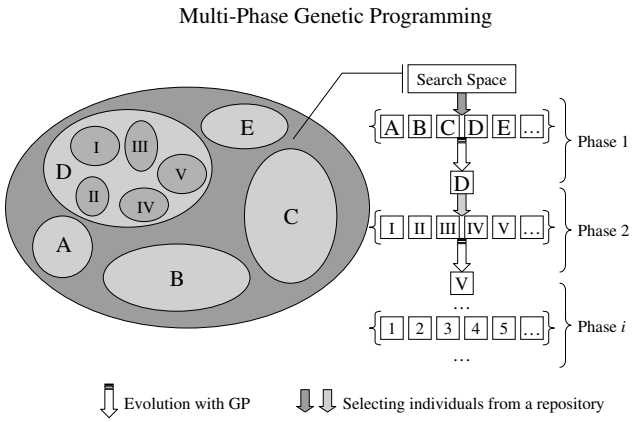

Fig. 1. A schematic illustration of multi-phase genetic programming (MPGP). As shown in the figure, the search space of MPGP varies from phase to phase. One function can be decomposed into multiple sub-functions in order to increase the granularity of MPGP.

characteristics of a problem. On the other hand, different functions can also be merged into a single lower-granularity function. In this way, the population structure of MPGP can have varying granularities.

During an evolutionary process, the lower-granularity individuals (such as A, B, C, D, and E in Figure 1) are evolved and selected first. As the lower-granularity (top-level) individuals are selected, the higher-granularity individuals (such as I, II, III, IV, and V in Figure 1), corresponding to an evolved lower-granularity individual (such as D in Figure 1), will be genetically operated. This process of granularity adjustment lasts until an appropriate level of granularity in the individuals is acquired. The virtue of this approach is that the evolutionary progress can be guaranteed, which means the MPGP approach is a more greedy search approach. The approach is well suited to the on-line evolution of agent behaviors due to its greedy characteristics.

So far, we have described the general objectives and design of the MPGP approach. In order to take a more in-depth look at its implementation and performance, in the following section, we will present a case study of MPGP in on-line sumo maneuver evolution.

IV. Sumo Maneuver Evolution

In this case study of MPGP, we are interested in evolving effective maneuvers for a sumo robot with respect to certain performance requirements. This problem is in essence a problem of genetic programming that is aimed at synthesizing strategic maneuvers for the robot.

In our implementation, the evolutionary process is operated in two different phases *by adopting two action granularities, general and specialized, respectively*. During the first phase, the values of units from a terminal set are fixed or limited within a scope. Thus, the fitter functions are evolved first and saved for the next phase evolution. During the second phase, the terminal set is evolved with the functions that are limited in the fitter function set as evolved from the previous phase.

In this section, we will describe the task, the sumo robot, the MPGP formulation of learning, and the detailed algorithm for our case study.

A. The Sumo Task

The task addressed in this paper is as follows: A sumo contest is to be played in a closed rectangular environment of size $M \times N$. One robot player of palm size ($PM \times PN$) is required to perform necessary sumo maneuvers that can effectively push its opponent out of the contest arena. Here, by necessary sumo maneuvers it is meant successful eye-body-coordinated motions against opponents of varying postures and weights.

For the sake of maneuver evolution, the player is allowed to communicate with an offboard controller responsible for passing advice to the player in order to improve its performance. Suppose that the opponent to this player is capable of showing different standing postures as well as different degrees of resistance (as if different weights) in an attempt to undermine the strength of the player.

Figure 2 shows a sumo contest arena with two mobile robots. The physical robot can execute

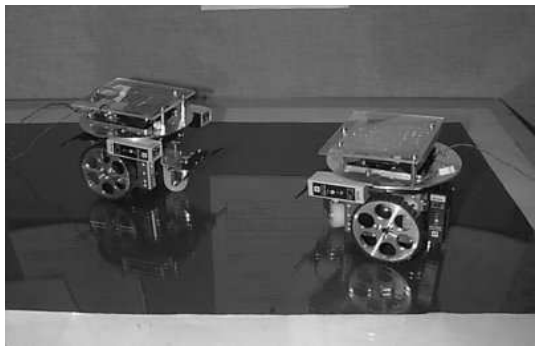

Fig. 2. A rectangular sumo contest arena.

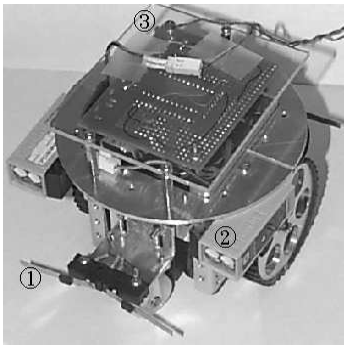

Fig. 3. The anatomy of the Junior robot. Junior is equipped with two limit-switch arms (circle 1), two infra-red eyes(circle 2), an onboard actuator controller with encoder-feedback, a memory board, and a communication device (circle 3).

a maneuver as evolved by an offboard controller, and measure and communicate the performance of the maneuver back to the controller. The evolution and selection of maneuvers by the controller are based on the above-mentioned multi-phase genetic programming (MPGP) approach.

A.1 The Sumo Robot and its Controller

The sumo player, named Junior, is a physical micro-mobile robot. Its objectives are to follow the maneuver commands from an offboard controller and to try out various strategic maneuvers against its opponent. Junior can readily utilize its two limit-switch arms, two infra-red eyes, an onboard actuator controller with encoder-feedback, a memory board, and a communication device, as illustrated in Figure 3. This offers Junior a number of basic capabilities. For instance, the arms can effectively push and at the same time sense its opponent. The two infra-red sensor-based

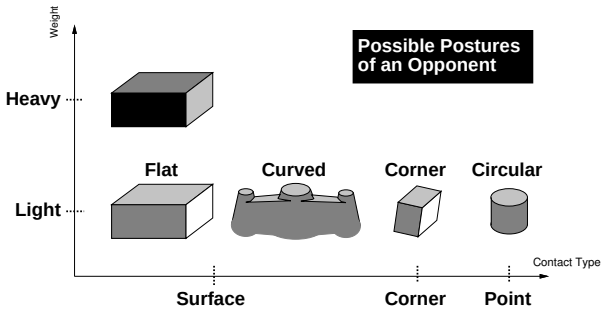

Fig. 4. The possible standing postures and degrees of resistance that an opponent may generate. They are simulated using dummy players of different engagement contact types and/or weights.

eyes are capable of detecting the presence of the opponent. The onboard actuator controller with encoder-feedback enables Junior to perform a specific maneuver with its two arms and two wheels (located right below its eyes). The communication infrastructure serves as a channel of information between Junior and an offboard controller. In other words, what Junior sees and feels will be communicated through such a channel back to the offboard controller for maneuver evolution. At the same time, the maneuver selected by the offboard controller will also be sent to Junior.

The offboard controller is responsible for passing just-in-time maneuver commands to Junior. It starts with a set of general motion maneuvers that Junior can perform, constantly evaluates the performance of Junior, and gradually corrects Junior's maneuvers in order to achieve a better performance. In so doing, the controller applies the MPGP approach to find the best-performing *general* maneuvers prior to moving onto the *specialized* ones.

A.2 The Opponents

In order to readily demonstrate as well as evaluate the effectiveness of maneuver evolution, in this case study, we will use *dummy players* as opponents. The dummy players will have different sizes and shapes, which correspond to opponents holding different standing postures and different degrees of resistance (as if different weights) in an attempt to undermine the strength of the robot player.

Figure 4 presents the dummy players of different postures, *i.e.*, flat, curved, corner, and circular postures, respectively. More specifically, as shown in the figure, the vertical axis corresponds to the weight of an opponent, whereas the horizontal axis corresponds to three types of engagement contact, *i.e.*, surface, corner, and point contact. The weight difference of an opponent is considered only in the case of a flat posture.

In a surface contact, it is relatively easy for Junior to see its opponent with both-eye sensing (i.e., with both front eyes), and at the same time, engage the opponent with both arms. On the other hand, in the case of a corner contact, it is rather difficult for Junior to focus with both-eye sensing and to engage with both arms due to the small contact area of its opponent. What makes the situation even more difficult is that, once Junior pushes its opponent with one arm, the opponent may move in any directions. Sometimes, a corner contact may transition to an edge contact.

Finally, a point contact is similar to a corner contact, except that in the point contact case, it becomes impossible for Junior to have both-eye sensing or both-arm engagement.

B. MPGP-Based Sumo Maneuver Evolution

In this section, we will provide the detailed MPGP formulation and algorithm for the task of sumo maneuver evolution.

B.1 General vs. Specialized Actions

In our case study, we represent the actions of a sumo robot with two different granularities. The optimal maneuvers at a right granularity for a given opponent posture will be evolved using the MPGP approach.

First, we define a set of general actions, *i.e.*, forward move, backward move, left turn, right turn, and stop, that can readily be executed by Junior through controlling the angular displacements and directions of its wheels. These general actions are regarded as low-granularity actions. Next,

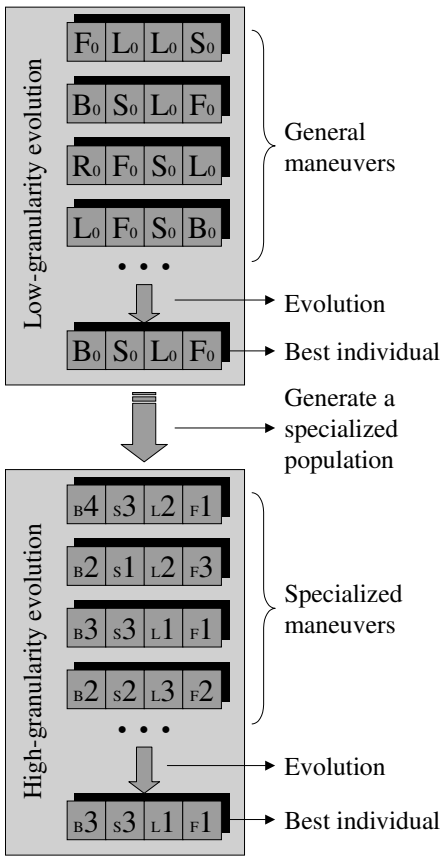

Fig. 5. An illustration of a two-phase MPGP. Here we assume that the length of an action sequence l is 4. The larger font letters or numbers in a chromosome indicates the general or specialized actions to be evolved.

we add some subtle movements into each general motion and thus obtain a set of high-granularity actions. The high-granularity actions are more specialized and refined than the low-granularity ones, and can be used by Junior to create specialized actions. The general and specialized actions are summarized in Table I.

In Table I, M_d denotes the step size of a moderate forward action or a moderate backward action, whereas M_s denotes the discrepancy between a fast or slow action and a moderate action. T_a denotes the angular displacement of turning left or right. It should be pointed out that the turning center is not fixed in each situation. Specifically, when Junior executes the specialized actions of LSRF and LSRB, the turning center will be at its left wheel. When Junior executes the

TABLE I
THE SET OF GENERAL AND SPECIALIZED ACTIONS THAT CAN BE EXECUTED BY A SUMO ROBOT.

General Action (low-granularity)	Symbol in MPGP	Specialized Action (high-granularity)	Symbol in MPGP	Displacement	Turning Angle
F: forward move	F_0	FF: fast forward	F_1	$M_d + M_s$	0
		MF: moderate forward	F_2	M_d	0
		SF: slow forward	F_3	$M_d - M_s$	0
B: backward move	B_0	FB: fast backward	B_1	$-M_d - M_s$	0
		MB: moderate backward	B_2	$-M_d$	0
		SB: slow backward	B_3	$-M_d + M_s$	0
L: left turn	L_0	LSRF: l-wheel stall r-wheel forward	L_1	0	T_a
		LBRS: l-wheel backward r-wheel stall	L_2	0	T_a
		LBRF: l-wheel backward r-wheel forward	L_3	0	T_a
R: right turn	R_0	LFRS: l-wheel forward r-wheel stall	R_1	0	T_a
		LSRB: r-wheel backward l-wheel stall	R_2	0	T_a
		LFRB: l-wheel forward r-wheel backward	R_3	0	T_a
S: stop	S_0	S: stop	$S_{(1\ 2\ 3)}$	0	0

specialized actions of LFRS and LBRS, the turning center will be at its right wheel. And, when Junior executes the specialized actions of LFRB and LBRF, the turning center will be the robot's geometric center.

B.2 Maneuver and Chromosome Representations

In the formulation of sumo maneuver evolution, we represent a sequence of l actions as a single *maneuver*. Based on the preceding section, we can readily define the *function and terminal sets* to be used in the process of MPGP, as given in Table II.

Figure 5 presents an illustrative diagram showing a two-phase MPGP formulation. In the figure, the action types, F, B, L, R, and S, can be viewed as functions, whereas the action degrees, 0, 1, 2, and 3, can be viewed as terminals.

TABLE II
THE FUNCTION AND TERMINAL SETS AS USED IN THE MPGP CASE STUDY.

Function Set	Terminal Set
F	0
B	1
L	2
R	3
S	

B.3 A Two-Phase MPGP Case Study

In the MPGP formulation, we split the evolutionary process into two distinct phases[1]. In one *phase*, general maneuver evolution is concerned only with general function evolution, *i.e.*, to select a sequence of general actions. In another *phase*, specialized maneuver evolution deals with sub-function evolution, *i.e.*, to select a sequence of specialized actions. The objective of this phase is to derive fine-tuned maneuvers. The representations of chromosomes for general and specialized maneuvers are illustrated, respectively, in Figure 5.

B.4 Fitness Function

The evaluation of individuals in a population is a crucial step in both GP and MPGP. In MPGP, the fitness of a candidate chromosome corresponds to the performance of a maneuver specified in that chromosome. The performance is estimated according to the effectiveness of Junior with respect to its opponent in executing the maneuver. Specifically, the fitness function f of a chromosome C with l number of actions X_i is composed of several sumo performance requirements. Generally speaking, the fitness function measures how Junior positions itself and acts with respect to its opponent. A good maneuver means that Junior can constantly face and see the opponent with both-eye sensing and try to actively engage the opponent with its arms. The fulfillment of these requirements can be measured by using the following fitness function:

[1] This is why this evolutionary approach is called multi-phase genetic programming (MPGP).

$$f = \sum_{i=1}^{l} w_{ai} \times \left(\sum_{j=1}^{n} \alpha^j(X_i) + \sum_{k=1}^{m} (\beta^k(X_i) + \gamma^k(X_i)) \right) \quad (1)$$

where $\alpha^j(X_i)$, $\beta^k(X_i)$, and $\gamma^k(X_i)$ will return 1, if Junior finds the opponent with its eye j, its arm k holds onto the opponent, and the arm is in contact with the opponent during action X_i, respectively. Otherwise, they will return 0. The measurements of $\alpha^j(X_i)$, $\beta^k(X_i)$, and $\gamma^k(X_i)$ can readily be determined by Junior via its sensors. In addition, l denotes the number of actions in a maneuver (which is set to 4 in our experiments). n denotes the number of eyes. m denotes the number of arms. w_{ai} denotes a positive weight for action X_i.

Using the above definition, the fitness of each candidate chromosome can be evaluated once its corresponding maneuver is performed.

In our present case study, Junior will not empirically test all the candidate maneuvers encoded in the population. Instead, we will utilize a computational model of player-opponent interaction to predicate the performance of such maneuvers. This computational model is aimed at capturing the effect of a maneuver similar to that of a physical contact pushing situation, and hence reduce the time required for the evaluation.

C. The Evolutionary Algorithm for MPGP-Based Sumo Maneuver Evolution

During the process of MPGP-based sumo maneuver evolution, the population at each evolutionary step consists of two parts: the *seed population* and the randomly produced population. The best candidate maneuver will be selected by an offboard controller and sent to Junior for execution. After Junior executes the selected sumo maneuver, the actual performance of such a maneuver will be evaluated by Junior using its sensors. This feedback will then be communicated to the controller. Thereafter, the controller will keep this maneuver along with its fitness in the seed population for further maneuver evolution.

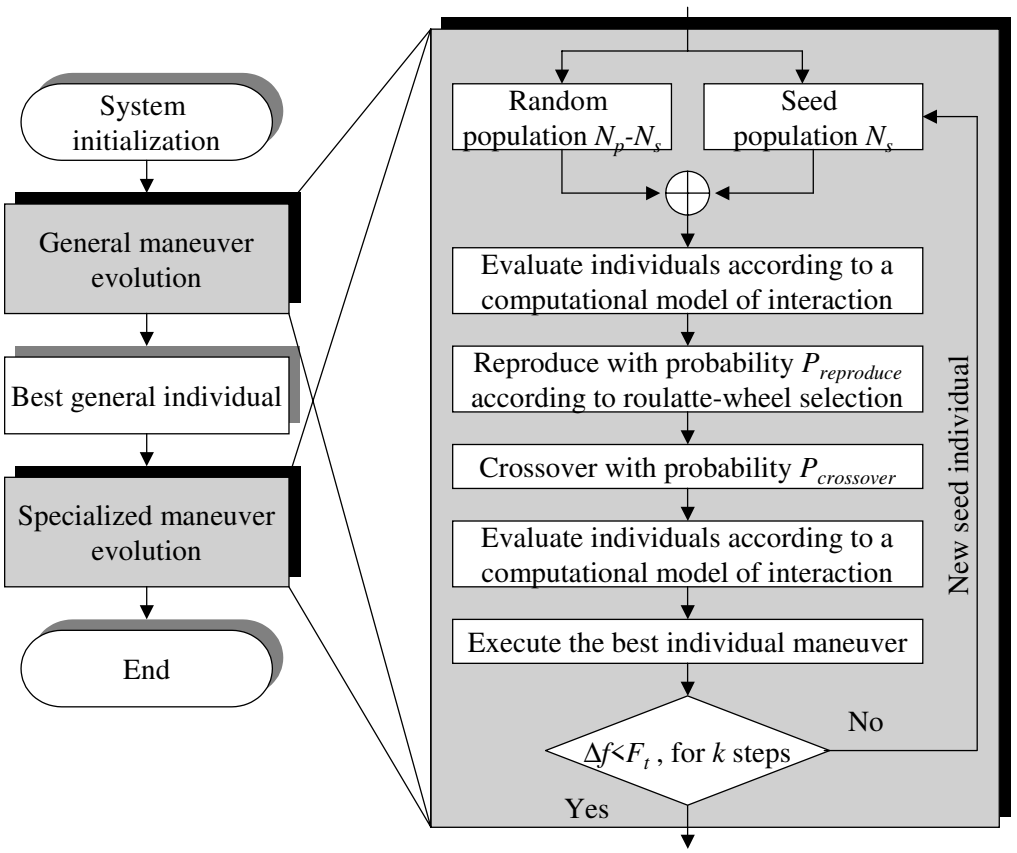

Fig. 6. An outline of the algorithm for the MPGP case study in sumo maneuver learning. Here the conditions are different in different phases. In the general maneuver evolution phase, if there is no large fitness improvement for k steps, the system will switch to a higher-granularity phase in order to acquire finer actions. While in the specialized maneuver evolution, the condition will become whether or not the opponent is outside the arena or the number of generations exceeds a predefined threshold.

The seed population will be updated according to the fitness values of maneuvers and the life span of each evolution step. That is, both the lowest-fitness and the oldest maneuver will be eliminated. The rule for updating the seed population of size N_s can be stated as follows:

$$i \Rightarrow \text{eliminated} \qquad \text{if } f_s^i = \min(f_s^j | f_s^j = y_s^j \cdot f^j,\ \forall j \in N_s) \tag{2}$$

where y_s^j denotes an aging factor of individual j in the seed population.

Figure 6 presents an outline of the algorithm (*i.e.*, the reproduction, crossover, and selection

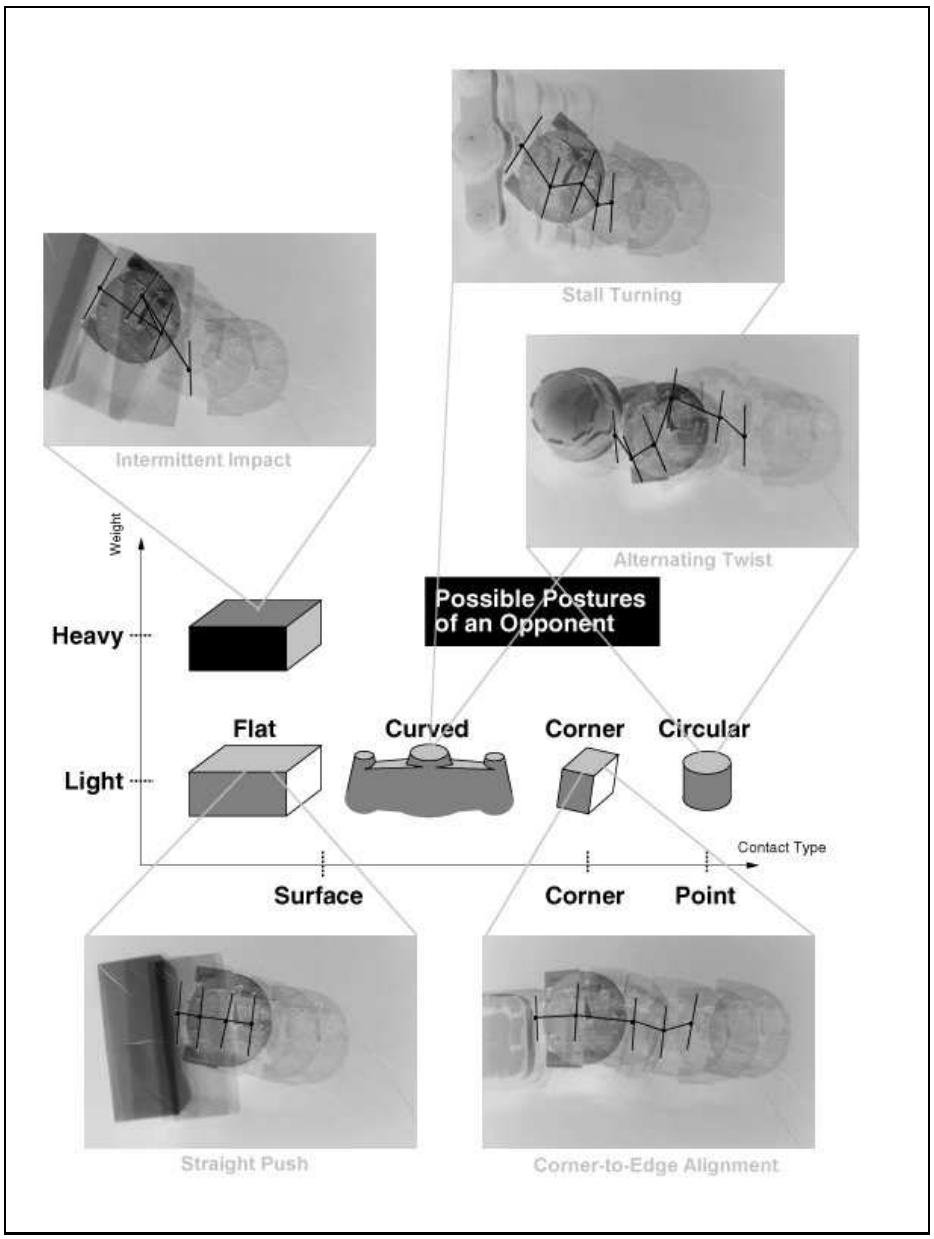

Fig. 7. Traced *trajectories* for the five specialized maneuvers evolved in the presence of five different opponent postures, respectively. In each of the five overlaid maneuver pictures, the equal-length line segments show the front of Junior at different action steps while performing a specialized maneuver. The captured maneuvers are: (i) *straight push* for lightweight flat postures, (ii) *intermittent impact* for heavyweight flat postures, (iii) *stall turning* for lightweight curved postures, (iv) *corner-to-edge alignment* for lightweight corner postures, and (v) *alternating twist* for lightweight circular postures.

of a population) as used in our MPGP case study.

V. Performance and Discussions

Having presented the basic principles of MPGP and its formulation in sumo maneuver evolution, the following sections describe the experimental validation of the approach with the sumo case study.

A. Validation of Sumo Maneuver Evolution

A.1 The Dummy Opponents

In our experiments, we use five dummy objects, simulating five different types of opponent postures and/or weights. As illustrated in Figure 4, the five dummies used represent the opponents with (i) lightweight flat posture, (ii) heavyweight flat posture, (iii) lightweight curved posture, (iv) lightweight corner posture, and (v) lightweight circular posture, respectively. The reasons why we decide to use dummies, instead of a physical robot, are:

1. We can readily observe the performance of sumo maneuver evolution under a specific posture or weight condition. This is rather difficult to achieve with a physical robot of a fixed configuration.
2. For the purpose of serving as only a domain of exploration and validation, this case study addresses learning how to push an opponent with a possible sumo posture, instead of how to counter-attack an opponent with adaptive strategies for a full-blown wrestling content.
3. If we use a physical opponent robot with a fixed strategy, our present minimalist hardware of the sumo robot will not be able to pick up reliable information using its infra-red and limit-switch sensors. Also, the mechanical composition of the robot, such as the positions of its limit-switches and two wheels, does not support any effective physical contact.

A.2 Experimentation

We have conducted five sets of experiments, each of which validates the performance of MPGP-based sumo maneuver evolution against one of the five dummies. All experiments start with the same initial positions and orientations for both the dummies and the sumo robot. The experiments end as soon as the number of generations in sumo maneuver evolution exceeds the threshold of 50. After each generation (*i.e.*, a maneuver execution), the relative positions and orientations of the dummy and the robot are reset.

A.3 Resulting Performance and its Significance

Figure 7 presents the performance of a sumo learning robot. Specifically, it records the evolved specialized maneuvers of the robot, overlaid on the schematic diagram of five opponent types as given in Figure 4. Each of the overlaid pictures in Figure 7 shows a sequence of actions in a respective maneuver actually performed, in which the equal-length line segments (some are overlapped) indicate the position and orientation of the robot front at consecutive action steps, and the line that connects the centers of those orientation line segments indicates the actual trajectory of the robot in performing a specialized maneuver.

As may be noted from the recorded trajectories, the MPGP approach selects a *straight push* maneuver for Junior to deal with an opponent of a lightweight flat posture. When the posture of an opponent remains the same but its resistance against Junior is large, the specialized maneuver to use will be a series of back and forth movements, as this turns out to be the most effective way to move a heavyweight opponent. This maneuver is named *intermittent impact*.

The remaining three postures, *i.e.*, lightweight curved posture, lightweight corner posture, and lightweight circular posture, have several interesting characteristics, such as:

1. The two limit-switch arms may not be able to simultaneously hold onto the body of such

an opponent due to its geometrical curvature.

2. The two infra-red sensors may not be able to focus on the opponent due to either its new position after an action or its small area of body exposure.
3. The direction and displacement of the opponent becomes very uncertain. This can be caused by inconsistent surface or point contacts between Junior and the opponent or by the light weight of the opponent.

In comparison, the opponents with flat postures have relatively simpler dynamics once being pushed and are easier to sense and engage with Junior's infra-red and limit-switch sensors.

When dealing with curved, corner, and circular postures, the MPGP approach is able to generate three specialized maneuvers, namely, *stall turning*, *corner-to-edge alignment*, and *alternating twist*, respectively. In *stall turning*, Junior makes a small forward movement and then switches to sharp turn. This maneuver can in effect push the opponent by working on the middle convex part of the opponent's body. In *corner-to-edge alignment*, the specialized maneuver is composed of forward movements, during which the contact between Junior and the opponent is effectively transitioned from a corner contact to an edge contact, even though Junior may not be able to see the opponent with both of its infra-red sensors. Finally in *alternating twist*, Junior moves alternately in two side directions, which in effect creates a large *virtual contact covering area*, given the small size and dynamic uncertainty of the opponent.

The significance of the above results can be viewed in the following ways:

1. It is feasible to empower the robot with maneuvers, instead of individual actions, in order to effectively push an opponent of a certain shape or weight. For instance, an intermittent impact composed of repeated back and forth movements is quite effective in pushing a relatively heavier opponent. And, a forward movement coupled with some degrees of sudden turning performs well in pushing an opponent with a curved or circular posture.

2. A sumo robot with a very simple, minimalist hardware configuration can still achieve reasonably sophisticated maneuvers, having employed the maneuver representation and manipulation schemes as described in the preceding sections. As can be noted, the robot that we have used in the experiments contains only four external sensors (excluding two wheel encoders), *i.e.*, two infra-red sensors and two limit-switches, and two actuators, *i.e.*, one for the right wheel and another for the left wheel.
3. Each of the specialized maneuvers is uniquely adapted to a specific type of postures.

In order to have a better understanding of the strengths and weaknesses of the MPGP approach, let us now examine the specific MPGP performance in the maneuver evolution experiments, and at the same time, highlight the factors that could affect the performance.

VI. Comparison with Related Work

Floreano, Urzelai, and Mondada studied an evolutionary approach to the on-line self-organizing control of mobile robotic systems [18], [19], [20]. In their approach, they utilized a discrete-time neural network to evolve and control a physical robot. The differences between their approach and ours can be summarized as follows:

1. They adopted a genetic algorithm to evolve the weights, or rules for defining the weights, of neural networks. This approach is commonly used in evolutionary robotics. The input to neural networks is connected to a robot's sensors, whereas the output unit is used to control a robot's wheels. In our approach, we do not predefine a neuro-controller system, instead the evolved maneuvers are sent directly to control a robot's behavior.
2. Floreano *et al.*'s work adopted an evolutionary process similar to that of Nordin's, where all individuals were evaluated on a physical robot. Although the environmental conditions for the robot may be different, adaptive behaviors, such as navigation and obstacle avoidance,

can be acquired after a number of generations.

In our approach, the individuals of the same generation share the same simulated environment. Only will the best individual selected from the simulated environment be validated on a physical robot, and used to update the behavioral repository of the robot, from which the population, *i.e.*, potential maneuvers, is in turn selected. Thus, our approach can be relatively less time-consuming than Floreano *et al.*'s, *as it does not physically validate all individuals but only the best one from the computational model based evaluations.*

Another related approach is Harvey's Species Adaptation Genetic Algorithm (SAGA) [9], [21], [22]. The objective of SAGA was to simulate a long-term evolutionary process in adaptive species. In order to achieve this objective, two issues were addressed: (1) changing the length of a genotype during the evolutionary process, and (2) converging a population while keeping the genotype length constant. The SAGA approach adjusts mutation rates in order to make the evolutionary process converged and to escape from local optima.

In our approach, the populations are also grouped into different 'species', according to different action granularities. The advantage of our approach is that it is relatively *easier to implement*, as it does not require an explicit mechanism for genotype length or mutation rate adaptation.

VII. Conclusion

In this paper, we have described a new evolutionary computation approach, called multi-phase genetic programming (MPGP). The special features of this approach lie in its variable-granularity representations of chromosomes and their different-phase genetic operations and evaluations. We have demonstrated the MPGP approach in an autonomous robot task where a sumo robot learns how to execute pushing maneuvers in response to various opponent postures (*e.g.*, specific shape and size of an opponent).

From our experiments, it was shown that the MPGP-based maneuver evolution can effectively lead a sumo robot to execute specialized maneuvers in the presence of uncertainty in the dynamics of its opponent. Furthermore, from the obtained results of comparison with the conventional GP approach, it was shown that the MPGP approach is relatively more efficient, *as it focuses on general (or grouped) solutions (*e.g.*, general maneuvers) first and then moves onto specialized ones (*e.g.*, specialized maneuvers) based on the obtained high-fitness general solutions (*e.g.*, inheriting general sumo expertise).*

As a direct extension of this work, we plan to further investigate the effectiveness of the MPGP approach with more levels of granularity. In addition, we are interested in examining how sophisticated the MPGP-evolved maneuvers can be, given a *minimalist* hardware configuration as in our present case studies, in response to dynamically changing postures.

REFERENCES

[1] J. Liu, *Autonomous Agents and Multi-Agent Systems: Explorations in Learning, Self-Organization, and Adaptive Computation*, World Scientific, 2001.

[2] J. Liu and J. Wu, *Multi-Agent Robotic Systems*, CRC Press, 2001.

[3] D. Floreano and F. Mondada, "Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot," in *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior (SAB-94)*, pp. 421–430. MIT Press, Cambridge, MA, 1994.

[4] I. Harvey, P. Husbands, and D. Cliff, "Issues in evolutionary robotics," in *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior (SAB-94)*, pp. 364–373. MIT Press, Cambridge, MA, 1994.

[5] I. Harvey, "Is there another new factor in evolution?," *Evolutionary Computation*, vol. 4, no. 3, pp. 313–329, 1996.

[6] F. Gomez and R. Miikkulainen, "Incremental evolution of complex general behavior," *Adaptive Behavior*, vol. 5, no. 3-4, pp. 317–342, 1997.

[7] J. Xiao, Z. Michalewicz, L. Zhang, and K. Trojanowski, "Adaptive evolutionary planner/navigator for mobile robots," *IEEE Transaction on Evolutionary Computation*, vol. 1, no. 1, pp. 18–28, 1997.

[8] B. Yamauchi and R. D. Beer, "Integrating reactive, sequential, and learning behavior using dynamical neural networks," in *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior (SAB-94)*, pp. 382–391. MIT Press, Cambridge, MA, 1994.

[9] I. Harvey, "Species adaptation genetic algorithms: A basis for a continuing SAGA," in *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, pp. 346–354. MIT Press, Cambridge, MA, 1992.

[10] C. Reynolds, "An evolved, vision-based model of obstacle avoidance behavior," in *Artificial Life III, Santa Fe Institute Studies in the Sciences of Complexity*, C. Langton, Ed., vol. XVI, pp. 327–346. 1993.

[11] C. Reynolds, "Evolution of corridor following in a noisy world," in *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior (SAB-94)*, pp. 402–410. MIT Press, Cambridge, MA, 1994.

[12] O. Miglino, H. H. Lund, and S. Nolfi, "Evolving mobile robots in simulated and real environments," *Artificial Life*, vol. 2, no. 4, pp. 417–434, 1995.

[13] S. Nolfi and D. Parisi, "Evolving non-trivial behaviors on real robots: an autonomous robot that picks up objects," in *Proceedings of Fourth Congress of Italian Association of Artificial Intelligence*, pp. 243–254. Springer Verlag, 1995.

[14] S. Nolfi, D. Floreano, O. Miglino, and F. Mondada, "How to evolve autonomous robots:

Different approaches in evolutionary robotics," in *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, pp. 190–198. MIT Press, Cambridge, MA, 1994.

[15] P. Nordin and W. Banzhaf, "An on-line method to evolve behavior and to control a miniature robot in real time with genetic programming," *Adaptive Behavior*, vol. 5, no. 2, pp. 107–140, 1997.

[16] P. Nordin and W. Banzhaf, "Genetic programming controlling a miniature robot," in *Working Notes for the AAAI-95 Fall Symposium on Genetic Programming*, E. V. Siegel and J. R. Koza, Eds., Cambridge, MA, 1995, pp. 61–67.

[17] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA, 1992.

[18] D. Floreano and J. Urzelai, "Evolutionary robots with on-line self-organization and behavioral fitness," *Neural Networks*, vol. 13, pp. 431–443, 2000.

[19] D. Floreano and F. Mondada, "Evolution of homing navigation in a real mobile robot," *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, vol. 26, no. 3, pp. 396–407, June 1996.

[20] D. Floreano and F. Mondada, "Evolutionary neurocontrollers for autonomous mobile robots," *Neural Networks*, vol. 11, no. 7-8, pp. 1461–1478, October 1998.

[21] I. Harvey, "Evolutionary robotics and SAGA: the case for hill crawing and tournament selection," *Artificial Life III, Santa Fe Institute Studies in the Sciences of Complexity*, vol. XVI, pp. 299–326, 1993.

[22] I. Harvey, "Artificial evolution: A continuing SAGA," in *Evolutionary Robotics: From Intelligent Robots to Artificial Life, ER2001*, T. Gomi, Ed., vol. 2217, pp. 94–109. Springer-Verlag, 2001.