2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Family Bootstrapping: a Genetic Transfer Learning Approach for Onsetting the Evolution for a Set of Related Robotic Tasks

Amiram Moshaiov[1] and Amir Tal[1]

Abstract— **Studies on the bootstrap problem in evolutionary robotics help lifting the barrier from the way to evolve robots for complex tasks. It remains an open question, though, how to reduce the need for designer knowledge when devising a bootstrapping approach for any particular complex task. Transfer learning may help reducing this need and support the evolution of solutions to complex tasks, through task relatedness. Relying on the commonalities of similar tasks, we introduce a new concept of Family Bootstrapping (FB). FB refers to the creation of biased ancestors that are expected to onset the evolution of "a family" of solutions not just for one task, but for a set of related robot tasks. A general FB paradigm is outlined and the unique potential of the proposed concept is discussed. To highlight the validity of the FB concept, a simple demonstration case, concerning the evolution of neuro-controllers for a set of robot navigation tasks, is provided. The paper is concluded with some suggestions for future research.**

I. INTRODUCTION

ONE of the major goals of Evolutionary Robotics (ER) is to obtain powerful and imaginative systems for the control of autonomous agents (robots) while minimizing the need for human intervention in the design process. Research studies on ER have demonstrated the ability to evolve robots from random solutions, and in particular robot control systems, that are successful in dealing with relatively complex tasks. Yet, most of such studies have been confined to academic demonstrations. To make ER an attractive design approach, several key problems have to be addressed. According to Doncieux et al., [1], one of the remaining challenges of ER is how to scale up to complex behaviors. It is suggested there that this issue is tightly linked to fitness landscapes and to the exploration abilities. One aspect of these issues is the bootstrap problem, which is the focus of this study. Designers using ER are commonly faced with the need to define the fitness function(s) for the considered robotic task. This is often a challenge that becomes increasingly difficult with higher complexity of the task. The aspiration is to reduce the need for designers' knowledge by defining the fitness function as close as possible to a statement on the ultimate task [2]. For example, when trying ER methods to develop controllers for robot soccer players it would be desired to define the fitness function based just on counting the goals.

Using such a high-level fitness definition commonly results with the bootstrap problem [2] (e.g., [3]). In such a case there is a lack of sufficient selective pressure to initiate the evolution process. The bootstrap problem is considered a major stumbling block in the route to achieving more complex and useful robot agents by way of evolution [2].

When reviewing the current methods to overcome the bootstrap problem in ER, it becomes apparent that none is fully automated. As described in section II-A, existing bootstrapping approaches require the engineers to separately analyze each complex mission based on their knowledge of the mission under consideration and of the existing evolutionary tools. Such approaches generally require, for each robotic mission, specific expertise and allocated resources. The aim of the proposed FB method is to deviate from such a paradigm by setting the stage for a non-task-specific methodology.

Inspired by ideas from the machine learning approach of Transfer Learning (TL), which is described in section II-B, a non-task specific bootstrapping method is hereby proposed. It concerns pre-evolving robotic solutions to a common (solvable) source-task. These solutions are expected to be valid for the creation of initial populations to onset the evolution of solutions for related target-tasks. Hence they could be considered as common ancestors. We term the proposed approach Family Bootstrapping (FB). In the following, the term Family-of-Tasks (FoT) refers to the set of related target-tasks that can be solved by separated evolution processes, where each starts from the common ancestors. In addition, the term Family-of-Solutions (FoS) refers to the associated solutions, and the common ancestors are termed family ancestors.

The considered target-tasks of the FoT are taken as tasks that are not solvable by evolution from a random population. Namely, each such target-task suffers from the bootstrap problem. The underlining assumption is that family ancestors exist when the target-tasks of the FoT have sufficient commonalities. It is hypothesized that if an evolutionary process will be started from some sub-set of the family ancestors it will be successful with respect to any task of the FoT. Moreover, it is conceivable that each such evolution process can be successful even if a high-level fitness function is used. Namely, once the current target-task is known to belong to a particular family, the designers will hopefully be left with the simple problem of devising a high-level fitness function, which in principle should not require any special knowledge on the task.

[1]The authors are with the Iby and Aladar Faculty of Engineering, Tel Aviv University, Tel Aviv, Israel (Tel. 972-3-6407098; e-mail: moshaiov@eng.tau.ac.il).

The proposed FB approach is based on the two-step approach to bootstrapping [3]. According to the two step approach, the evolution for the target task is separated from the evolution to auxiliary (source) tasks. While the approach of [3] sets the foundations to the current study, it deals only with one target task, whereas the current work deals with FoT. The current paper presents FB as a general bootstrapping paradigm, and provides a proof-of–concept to the suggested FB approach.

The perceived long-term main advantage of the proposed FB approach is that it opens the way to building database of common ancestors to various families of target-tasks by using common source-tasks. It is imaginable that when given a target-task it would be possible to automatically classify it to a related source-task. This will allow extracting the associated ancestors from the database. Such a methodology could support the use of ER by non-expert designers. Conceivably, the FB approach will help tackling complex tasks, and will reduce the computational efforts needed to achieve solutions to such tasks.

In section II, two main topics are briefly reviewed as deemed relevant to the understanding of the proposed method. These include: a) bootstrapping in ER and b) transfer learning (as related to reinforcement learning). Next, in section III, the proposed new concept of FB is introduced. It is followed by section IV where a proof-of-concept by demonstration is provided for a family of navigation tasks. The conclusions of this study are provided in section V.

II. BACKGROUND

A. Bootstrapping in Evolutionary Robotics

The bootstrap problem in ER concerns the lack of sufficient selective pressure at the onset of evolution. It is encountered when attempting to evolve robot controllers for a relatively complex task from a low-level initial population (usually from random individuals) using high-level fitness functions. Such a high-level function lacks sufficient details with respect to the task; namely, it does not support the onset of evolution from the low-level population. The use of a high-level fitness function that is decided upon with a minimal designers' knowledge is considered to be preferred [2]. In a picturesque way this preferred scheme resembles a wish to go to a far-away target, without the cost of a map and a compass or a guide that can direct us to landmarks that we should meet on our way. While wishing for minimum intervention by the designers, overcoming the bootstrap problem usually involve the *a-priori* use of designers' knowledge.

The basic idea behind all bootstrapping methods, which have been devised to overcome the aforementioned problem, is that in the absence of a sufficient "fitness gradient" it has to be created by some means. Independently, Nelson et al. [*2*] and Mouret and Doncieux [4] suggested categories of the available schemes. Mouret & Doncieux describe the attempts to overcome the bootstrap problem as different schemes of incremental evolution [4]. They divided the existing bootstrapping methods into four categories including: staged evolution, environmental complexification, fitness shaping and behavioural decomposition.

Traditional approaches for bootstrapping, with one or more stages, use at each stage either single-objective or an aggregated objective function to measure fitness. Hence, all the traditional methods require a single-objective evolutionary algorithm. With the availability of Multi Objective Evolutionary Algorithms (MOEAs), e.g., [5], which aim to find an approximation to Pareto-optimal solutions, new ideas became tractable for overcoming the aforementioned problem. Mouret and Doncieux proposed a one-step multi-objective evolution method to solve the bootstrap problem [6], [4]. A similar multi-objective approach, which involves two-steps rather than one, has been recently suggested by Israel and Moshaiov [3]. At a first glance the method of [3] may appear to constitute only a slight change from that of [4]. The following aims to highlight the significance of the difference between the aforementioned two methods.

In the one-step approach of Mouret and Doncieux, auxiliary-objectives (bootstrapping-objectives) as well as the ultimate task-objective are used to define a multi-objective optimization problem, in which no preference is given to any of these objectives. In their study they evolved robots that seek a set of light sources in a complex order, which involves a bootstrap problem. To overcome the problem, MOEA was used to simultaneously optimize both the "ultimate goal" (time to seek the final light switch) and a few simpler bootstrapping-objectives corresponding to sub-tasks. It is argued in [4] that the elimination of the need to arrange the objectives in some pre-defined order allows for a more general approach as compared with the incremental one.

In contrast, Israel and Moshaiov, [3], separated the process into two-steps. In the first step a multi-objective evolution process is carried out using only auxiliary-objectives, whereas in the second step the evolution is carried out as a single-objective optimization with the "ultimate goal" of the actual task. The second-step starts with an initial population consisting of a selected set of non-dominated solutions, which are obtained from the first stage. This set is considered as ancestors for the desired solutions. In [3] the two-step approach was demonstrated and compared with the bootstrapping approach of [7], for the co-evolution of soccer-like players, which is known to suffer from the bootstrap problem.

One may claim that the one-step approach includes the non-dominated solutions of the two-step method. This is due to the no objective preference approach, which means that in the obtained front of the one-step method there are also solutions that counts only for the auxiliary-objectives. Hence, it can be argued that in principle the two-step approach cannot be considered advantageous. However, it should be noted that the focus is different. The advantage of the two-step approach of [3], over the one-step approach of [4], is rooted in the different focus of the two methods. While the later is task oriented, the two-step approach employs the aforementioned separation to set the stage for the proposed FB method, which is a non-task-specific approach to bootstrapping.

In spite of the existence of various bootstrapping methods, the bootstrap problem should still be considered a major stumbling block in the route to achieving more complex and useful robots by way of evolution. Existing bootstrapping approaches still require, in most cases, some a-priori knowledge to design the proper means to overcome the problem. With existing methods, each complex mission under consideration should usually be separately analyzed based on the designers' knowledge about the mission. This includes the careful selection of the fitness function(s). It is conceivable that novice designers may fail to accomplish the task without becoming well acquainted with the available bootstrapping methods and the existing evolutionary search algorithms. Above all, they should have an insight with respect to the complexity of the task under consideration. The aim of the proposed FB method is to deviate from such a paradigm by setting the stage for a non-task-specific approach. As argued in the introduction the suggested methodology could eventually lead to supporting the use of ER, for complex tasks, by non-expert designers.

B. Transfer Learning

Transfer Learning (TL) is the name given in machine learning to a group of methodologies involving the improvement in the learning of one task by previously learning a related task [8], [9]. TL is inspired by the way humans learn and accomplish complex tasks. One of the key aspects of human learning is the fact that they face a stream of learning problems over their entire lifetime. For example, when a person learns a skill as complex as baking a cake using a cook-book, multiple learning experiences come into play. These include acquiring the basic motor skills, learning to recognize objects, and acquiring language comprehension while being an infant, learning how to read while during childhood, and so on.

TL methods are constantly being studied with respect to various machine learning methods, such as supervised and unsupervised learning [8]. TL has only lately been studied with reinforcement learning approaches [9], mostly in problems involving discrete states and actions. ER techniques are often considered to be a close variant of reinforcement learning techniques, as both select for improved solutions based on some sort of reward [10]. Applying reinforcement learning to robotics is challenging, since that such methods commonly deal with discrete worlds of actions and states, while robotics deals mostly with continuous states and actions. A profound difference between ER and RL is the central use, in the former case, of evolutionary computations to achieve the improved solutions. Such differences have to be accounted for if TL methods that were developed in the context of RL are to be adapted for ER applications. While saying that, it should be pointed out that similarly to the traditional use of random populations to start the evolution in ER, in the case of classical RL the exploration starts from scratch. In both cases this may cause a problem and in the former, such problems have been addressed by TL methods (e.g., [12]).

The idea of using TL approaches for robotics is not new (e.g., [13], [14]). Yet, one may suggest that research on using TL in the context of ER has only begun. Doncieux has recently reported, in [11], on a TL experiment in ER in which a robot first learns to mobilize objects towards a goal from scratch and then transfers its knowledge to a door opening task. The above claim on the infancy of using TL in ER could be questioned. In fact, while not referring to the notion of transfer of knowledge, most techniques to overcome the bootstrap problem might be considered as using some sort of knowledge transfer. The proposed FB is no exception. The common ancestors for the FoS are obtained by using a source-task, and the FoT are in fact multiple target-tasks. As such, FB could be viewed as a special type of a TL method.

A major element of TL is tasks relatedness [8], [9]. Studies on TL have shown that the positive effect of TL is linked to the closeness, or relatedness of a given source-target pair. Tasks which were considered related to one another made the knowledge transferring more successful [8]. This relatedness, however, is neither very easily defined nor easily measured. This is due to the many features by which a typical learning task may be defined. The goal of the current study is to provide FB with a proof-of-concept by demonstration. Hence, in this study the focus is neither on formalizing the relations among the multiple target-tasks of the FoT, nor on formalizing the relation of the FoT to the common source-task.

III. FAMILY BOOTSTRAPPING

As mentioned in the background, a two-stage bootstrapping approach was suggested and demonstrated in [3]. Restricting the first-stage to evolving solutions based just on the auxiliary-objectives, sets the foundations for the proposed FB methodology. The idea behind the aforementioned restriction is that the population obtained by the first- stage is not biased towards any particular target-task. Hence, if several target-tasks share similar sub-tasks or have other types of commonalities, which are considered when defining the objectives for the first-stage, then it is conceivable that the population obtained in that stage would be useful for bootstrapping the evolution of solutions for all the target-tasks. Although pointing at such an option, this hypothesis was neither discussed nor tested by Israel and Moshaiov in [3].

In the proposed FB approach there are two evolutionary stages. In the first-stage the evolution is performed to solve a *Common Source-Task*. The obtained solutions are termed *Family Ancestors*. These are used to initiate a population for evolving solutions to any member of a set of associated target-tasks. That set is termed *Family-of-Tasks (FoT)*, and the union of the solutions to the tasks of the FoT is termed Family-of-Solutions (FoS). A schematic presentation of the FB procedure is depicted in figure 1.

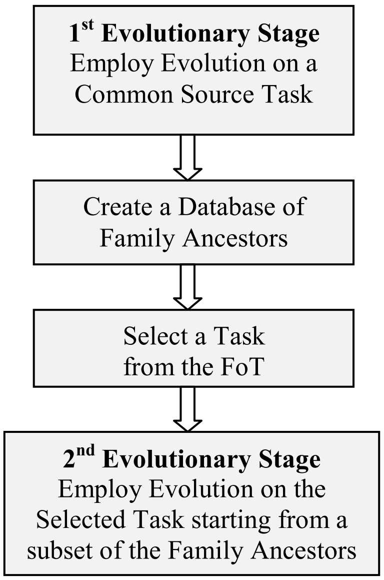


Figure 1. The FB Paradigm

It is noted that FB does not have to follow the evolutionary approach of [3]. In fact the multi-objective first-stage of [3] can be replaced by incremental evolution, and in some cases even with a single-objective evolution that might solve the common source-task. In addition, in contrast to [3], the current presentation of the FB procedure does not restrict the second-stage to a single-objective evolution. In the current demonstration we try for the first-stage two alternatives, one with a multi-objective evolution and the other with a single-objective approach. For the second-stage we use an ultimate single-objective.

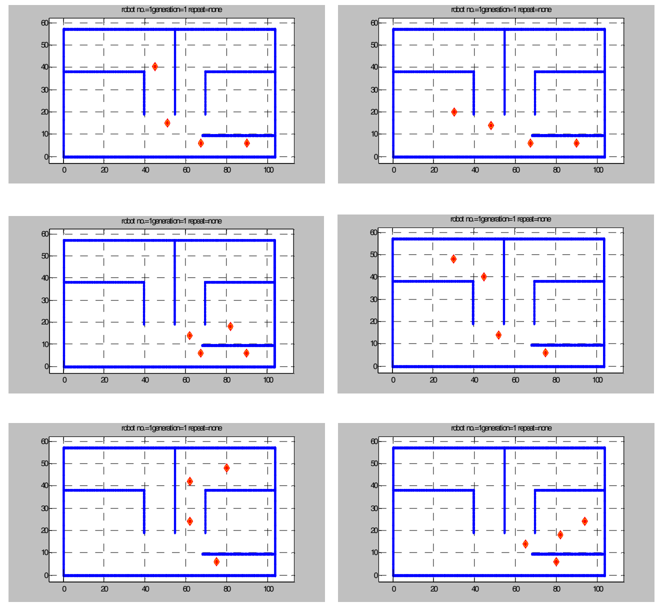
IV. DEMONSTRATION

A. Family of Target-tasks

A simulated robot is operating in an arena with rooms and corridors (a maze). In the maze there are lights at different locations. It is noted that the robot does not have a model of the arena and doesn't know the location of the lights or the walls in advance; however, it can sense the walls and also the lights when they are on. The robot switches off a light when reaching it. The details of the robot, the neuro-controllers, and the sensors, are given in the appendix.

Adapting the bootstrapping example of [4], a family of six navigational target-tasks is presented in figure 2. The six target-tasks are denoted as follows. The case at the top-left side is denoted as T1, the one below T1 is T3, and the last in the left side is T5. The others, at the right column, are: at the top is T2, next is T4, and the one at the right bottom corner is T6.

Figure 2. Lights' Settings for Six Target-tasks



In this simulation study, the aim is to separately evolve simulated robot neuro-controllers for each of the target-tasks. All of the proposed T1-T6 target-tasks share the same simulated robotic arena in terms of the layout of the rooms and corridors. For each target-task there is a particular setting of light buttons within the arena. Namely, they differ in their locations from one target-task to the other as depicted by the marks in figure 2. The starting point for the robot, which is common to all the six cases, is at the right bottom corner and the robot faces towards the left. It should be noted that, in contrast to what may be wrongly perceived from the figure, in the target-tasks the lights are not on at once as further explained below.

For each target-task the maze contains a "hidden" trail of a sequence of four lights and their associated buttons. Only one light is on at any given time. A robot in the maze can initially sense only the first light of the sequence. By stepping on such an active light it is switched-off and the next light in the sequence is turned-on, and so forth. This means that finding the last light in the sequence, which is the goal of the target-task, can be achieved only when all the preceding lights are reached and stepped upon in the given order.

The objective as defined here is to find and turn on the last (fourth) light within the shortest time possible. This means that the robot must find in the right order all the lights and optimize its route from one light to the other. During the evolutionary process, robots that do not reach the final light by the end of an allotted time are stopped and given a fitness score of 1 (which is the worst possible). Although commonly phrased for all the target-tasks, the above objective differs from one target task to the other. This is due to the location of the fourth light and the "trail" to reach, which vary from one task to the other. Namely, for the T1 case the last light is in the top-left narrow room, whereas in T2 the last light is in the large room below, and so on.

The described objective of "minimizing the time to light the fourth light" is extremely hard to achieve when starting the evolution from random controllers. In fact, as tested by us (not shown here), each of the proposed target tasks suffers from the bootstrap problem.

B. Source-task

For the source-task, the arena, the robot and its starting point are the same as for the target-tasks. However both the set-up of lights and the objectives are different from those of the target-tasks. Moreover, in contrast to the target-tasks, there is no sequence of switching the lights; all lights are on at the start of the source-task. Figure 3 depicts the arena and lights for this case.

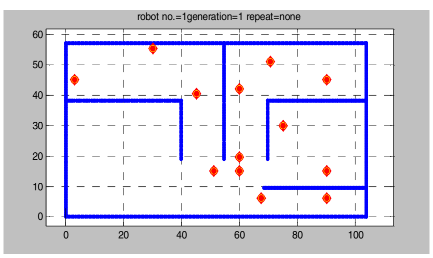


Figure 3. Arrangement of lights for the source-task

The source task follows the task which is presented in [15]. Once a light is "stepped upon" it is turned-off, and becomes undetectable to the robot's sensors. The robot is given 200 seconds for the mission; this is more than enough time for a robot following a path connecting all lights to step on them all.

In our demonstration-study two different types of optimization processes are used in the first evolutionary stage of the FB procedure (for the common source-task). The first type is an Evolutionary Single-objective Optimization (ESO) and the second is an Evolutionary Multi-objective Optimization (EMO). For finding optimal solutions for the common source-task using the EMO process, two objective functions marked by F_1 and F_2 are employed as used in [15]. In the ESO case only F_2 is used.

F_1 is based on [16] and is defined as follows:

$$F_1 = \frac{\sum_{i=1}^{final\ step} f_1}{max\ step};\ f_1 = V_i\left(1-\sqrt{\Delta v}\right)_i\left(1-I\right)_i \qquad (1)$$

$$0 \le V \le 1$$
$$0 \le \Delta v \le 1$$
$$0 \le I \le 1$$

Where:

- V is the absolute sum value of the 2 rotational wheel speeds. V is high when the robot is moving straight and fast.
- Δv is the absolute difference value between the two rotational wheel speeds. $1-\sqrt{\Delta v}$ is high when the robot is not making any turns.
- I is the normalized activation value of the sensor with the highest value. $1-I$ is high when the robot does not sense any obstacle.

F_1 is calculated as an average of the accumulated temporary step performances marked as f_1. The sum is taken from the initial step to the final step of the robot. We note that during the evolution some neuro-controllers do not manage to complete the maximum allowable number of steps (marked max step). The purpose of F_1 is to achieve fast and straight motions while avoiding obstacles. In this case there is no specific destination.

F_2 is defined as follows:

$$F_2 = \frac{\sum_{i=1}^{final\ step} f_2}{max\ step};\ f_2 = \begin{cases} H & robot\ reaches\ target \\ 1/\left(1+d\right) & else \end{cases} \qquad (2)$$

Where:

- H is a score that the robot gets for reaching a light. Here it is set to 50.
- d is the distance from the robot to the closest light

Similar to F_1, F_2 is calculated as an average over the accumulated temporary step performances marked as f_2. The purpose of F_2 is to achieve a controller that reaches as many lights as possible with no concern for obstacle avoidance.

Based on [15] it is expected that these objectives are contradicting as related to the current arena and light setup.

C. ESO and EMO Results for the Source-task

Table 1 summarizes the details for the evolutionary runs. The details are with respect to the use of NSGA-II. For the ESO case a NSGA-II was used with one objective fixed.

TABLE 1
SUMMARY OF RUN PARAMETERS

Parameter Description	Par. value
1. Robot & Environment	
Wheel diameter	1 cm
Body diameter	5.5 cm
Tar. Sens. Range	100 cm
Obs. Sens. Range	7.5 cm
Tar. Sens. F.O.V	30 deg
Obs. Sens. F.O.V	6 deg
Max. speed	0.5 cm/ sec
No. of targets (Total)	13
2. Neural Network	
Input layer	16
Hidden layer	3
Output layer	2
Bias term	0
Activation functions	Sigmoid
Activation slope term	1 Encoded
3. Evolutionary Process Details	
Sim. Time step	5 sec
Tot. time	1000 sec
Population size	56
Terminal generation	300
Type of EA	NSGA-II
Encoding	Direct
Chromosome	55 real-val.
Crossover type	SBX
Crossover rate	1
Mutation type	Polynomial
Mutation rate	1/55
4. Evolutionary Runs	
No. of repeats	10

Figure 4 shows a typical result for the best controller, which is obtained with ESO.

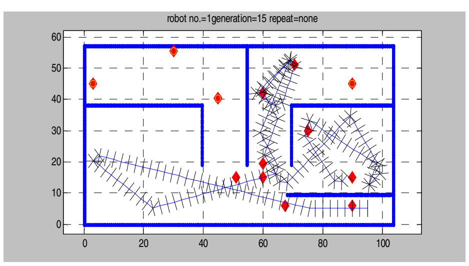


Fig. 4 Typical Best Path in the ESO runs

Figures 5 and 6 show such paths, respectively, for the best F1 and F2 controllers, of the EMO case.

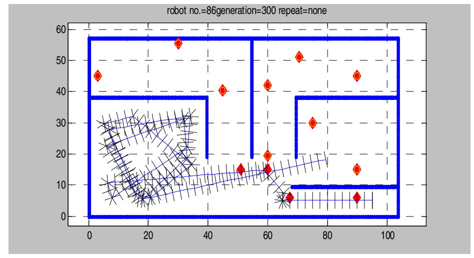


Figure 5 Typical Best F1 Path in the EMO runs

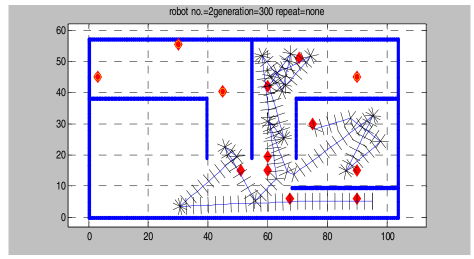


Figure 6 Typical Best F2 Path in the EMO runs

For the EMO process, ten repeats are run. Then the controllers belonging to the approximated Pareto-fronts of the last generation from all the repeated runs are merged to form one union-set. The union is then sorted to obtain the non-dominated set from all the runs. In the current study the later set contained 86 individuals. These serve as the family ancestry for the experiments in the second evolutionary stage, which are based on the EMO first stage.

Similarly, for the ESO process ten runs are also conducted. The union of the individuals, of the last generations from all the runs, is sorted for the best 86 ones. These are used as the family ancestry for the experiments in the second evolutionary stage, which are based on the ESO first stage.

D. Results for the Target-tasks

Robots were evolved for each of the six target-tasks using at each run the ancestry obtained from either the ESO source-task or the EMO source-task. Each evolutionary run was stopped at 500 generation (rather than 300). The size of the population is 86 (as opposed to 56). Ten runs were carried out for each target-task. Other parameters used are the same as those in the source-task case (as detailed in table 1). The obtained paths are shown in figure 7. The cases shown in figure 7 are for the target tasks T3, T4, and T6 as described in figure 2. The two top cases in figure 7 are for T3. The following two middle cases are for T4, and the last two at the bottom are for T6.

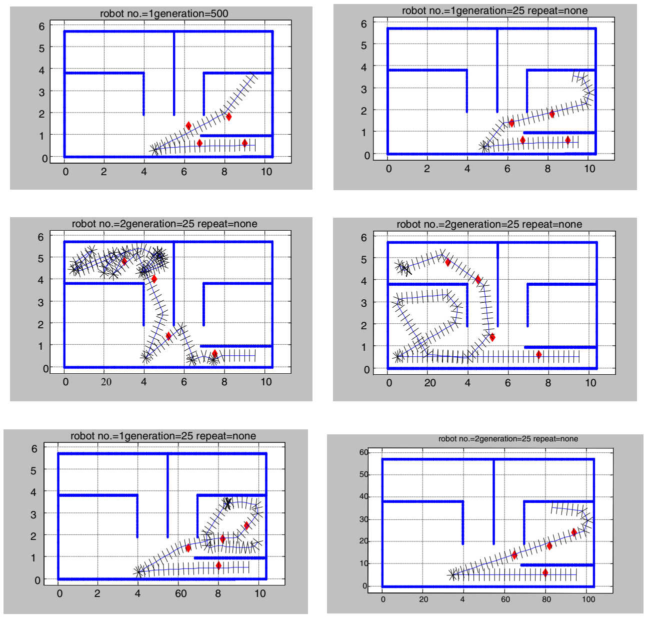


Fig. 7. Typical Paths of Successful Robots
Left: using EMO ancestry
Right: using ESO ancestry

Table 2 provides a summary of the results from both the ESO and EMO cases.

Table 2: Summary of results
(captions are described in the text below)

	T1	T2	T3	T4	T5	T6
Using ESO ancestry						
P	15	7	16	14	0	22
M	1	1	1	1	----	1
B	0.48	0.30	0.20	0.60	----	0.26
Using EMO ancestry						
P	23	40	79	58	14.5	60
M	1	1	0.36	0.40	1	0.39
B	0.15	0.14	0.20	0.38	0.29	0.25

The six columns of the table refer to the six target-tasks as marked by T1 to T6. These symbols stand for the target-tasks as presented in the description of figure 2. The rows are marked as follows. P marks the percentage of robots out of the population solving the target-task. M and B mark, respectively, the median and best times required for the solution (normalized with respect to the total time allotted).

E. Discussion

The FB approach is successfully demonstrated here for both types of evolutionary processes at the first evolutionary stage of FB, namely for both ESO and EMO. The demonstration involves robot navigation tasks.

As seen from table 2, using ancestry from the EMO source-task clearly outperforms using ESO ancestry. The EMO ancestors were able to handle all the targeted tasks of the FoT used here. In contrast, the ESO ancestry failed to accomplish T5 (reaching the light inside the right-top narrow room). Moreover, the EMO ancestors beat the ESO ones, in almost all cases of the FoT, in terms of the measured performances as depicted in the table. A possible explanation to the superiority of EMO is the effect of the diversity which is encouraged in the EMO method.

The results shown here were obtained using NSGA-II (see [5]). Using this kind of MOEA for the evolution of neuro-controllers is not optimal, as discussed in [17]. Hence the results here might be improved when using a more appropriate algorithm. However, for the purpose of highlighting the FB concept validity, NSGA-II seems to have been sufficient.

The successful demonstration of the FB paradigm, which is provided here, is limited in scope. Moreover, the issue of relatedness, among the target-tasks of the FoT and between the FoT and the common source-task, is kept at the intuitive-abstractive level. These limitations should certainly be acknowledged. Nevertheless, the results are encouraging and they provide a proof-of-concept by demonstration to the proposed FB paradigm.

V. Conclusions

A new approach to the ER problem of bootstrapping is presented. It concerns a transfer learning approach to bootstrapping the evolution of solutions to a set of related target-tasks.

To demonstrate the suggested FB approach, a family of six robot navigation tasks is presented and studied. Each of the targeted ER tasks suffers from the bootstrap problem. The family is shown to be bootstrapped using a common source task, which is much simpler than the target-tasks. In addition to demonstrating the proposed concept, this paper compares two approaches. It is found that while either an ESO or an EMO approach can be employed, for the common source-task, the later considerably outperforms the former.

To improve the understanding on the mechanisms of FB it is suggested to investigate the relations among members of the FoT and the relation between the FoT and the common source-task. Future studies should explore how such relationships influence the existence of bootstrapping. Moreover, future investigation should also examine the correlation between closeness of the tasks and the performances of the evolution process such as the convergence performance in the second evolutionary stage of the FB paradigm.

Appendix: Additional Details of the Simulation

Robot and Sensors The simulated robot used is a simplified model of Khepera [16]. It is a round, two-dimensional rover running on two wheels, one on each side. Robot diameter is 5.5 cm and the wheel diameter is 1 cm. Each wheel is actuated directly from one of two output signals of the neural network controller (details below). The robot can sense the nearest light to it and the nearest wall point to it with two sets of peripheral sensors: light sensors (for the light buttons) and IR sensors (for the walls). The sensing ranges and fields of view for both types of sensors are provided in table 1 of the paper. No noise was used in the simulations. The simulator was written in Matlab as a part of this study.

Neural network The Neural network Controllers (NC) used are of type feed forward network (FFN). An input layer of 16 nodes is connected to a hidden layer of 3 nodes, which is in turn connected to the output layer of two nodes. Nodes of the hidden layer perform a simple weighted sum of the network inputs, while the output nodes use a sigmoid activation function (no bias term weighted in) and output in the range (-0.5, 0.5). No recurring connections are used. NC weights are directly encoded to a real-valued chromosome along with the slope term for the output layer activation function (same for both output nodes). Simulated binary crossover is used as well as polynomial mutation. Additional information is available in table 1 of this paper.

ACKNOWLEDGMENT

This work was supported by the Dean of the Iby and Aladar Faculty of Engineering, Tel-Aviv University and by the Vice-president for Research of Tel-Aviv University. The anonymous reviewers are acknowledged for their useful comments.

REFERENCES

[1] S. Doncieux, J. B. Mouret, N. Bredeche, and V. Padois, "Evolutionary Robotics: Exploring New Horizons." In: *New Horizons in Evolutionary Robotics: Extended Contributions from the 2009 EvoDeRob Workshop*, *Studies in Computational Intelligence* vol. 341 (pp. 3-25), Springer, 2011.

[2] A. L. Nelson, G. J. Barlow, and L. Doitsidis, "Fitness Functions in Evolutionary Robotics: A Survey and Analysis," *Robotics and Autonomous Systems*, vol. 57, pp. 345-370, 2009.

[3] S. Israel and A. Moshaiov, "Bootstrapping aggregate fitness selection with evolutionary multi-objective optimization," presented at Parallel Problem Solving from Nature - PPSN XII, 2012.

[4] J. B. Mouret and S. Doncieux, "Overcoming the Bootstrap Problem in Evolutionary Robotics Using Behavioral Diversity," presented at IEEE Congress on Evolutionary Computation, 2009.

[5] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation, vol. 6, pp. 182-197, 2002.

[6] J. B. Mouret. and S. Doncieux, "Incremental evolution of animats' behaviors as a multi-objective optimization," presented at International conference on the simulation of adaptive behavior (SAB '08), 2008.

[7] H. E. Óstergaard and H. H. Lund, "Co-evolving complex robot behavior," Evolvable Systems: From Biology to Hardware, pp. 308-319, 2003.

[8] L. Torrey, and J. Shavlik, "Transfer learning." *Handbook of Research on Machine Learning Applications. IGI Global*, *3*, 17-35, 2009.

[9] M. E. Taylor, and P. Stone, " Transfer Learning for Reinforcement Learning Domains: A Survey," *Journal of Machine Learning Research*, vol. 10, 2009.

[10] , D. E. Moriarty, A. C. Schultz, and J. J. Grefenstette, "Evolutionary Algorithms for Reinforcement Learning., J. Artificial Intelligence Res., vol. 11 pp. 241-276, 1999.

[11] S. Doncieux, "Transfer Learning for Direct Policy Search: A Reward Shaping Approach," In: IEEE Int. Conf. Dev. and Learning – ICDL '13, 2013.

[12] M. E. Taylor, P. Stone, and Y. Liu, "Value Functions for RL-Based Behavior Transfer: A Comparative Study," Proc. Of the Twentieth National Conf. on Artificial Intelligence, July 2005.

[13] S. Thrun and T. M. Mitchell, "Lifelong Robot Learning," Robotics and Autonomous Systems, 15:25{46, 1995.

[14] Z. Kira. Inter-Robot Transfer Learning for Perceptual Classification. Proc. of the 9th Int. Conf. on Autonomous Agents and Multiagent Systems: Vol. 1, pages 13-20, 2010.

[15] A. Moshaiov, and M. Zadok, "Evolving counter-propagation neuro-controllers for multi-objective robot navigation." In *Applications of Evolutionary Computation*, Berlin: Springerpp. 589-598, 2013..

[16] F. Mondada and D. Floreano, Evolution and mobile autonomous robotics, *Towards Evolvable Hardware (Lecture Notes in Computer Science)* Vol. 1062, 1996, pp 221-249, 1996.

[17] A. Moshaiov. And O. Abramovich., Is MO-CMA-ES Superior to NSGA-II for the Evolution of Multi-objective Neuro-controllers?, Proc. of CEC 2014.