Adaptive Behavior

http://adb.sagepub.com/

Designing adaptive humanoid robots through the FARSA open-source framework

Gianluca Massera, Tomassino Ferrauto, Onofrio Gigliotta and Stefano Nolfi

Adaptive Behavior 2014 22: 255 originally published online 14 July 2014
DOI: 10.1177/1059712314536909

The online version of this article can be found at:
http://adb.sagepub.com/content/22/4/255

Published by:

SAGE

http://www.sagepublications.com

On behalf of:

ISAB

International Society of Adaptive Behavior

Original Paper

Adaptive Behavior

Adaptive Behavior
2014, Vol. 22(4) 255–265

DOI: 10.1177/1059712314536909
adb.sagepub.com
SAGE

Designing adaptive humanoid robots through the FARSA open-source framework

Gianluca Massera[1], Tomassino Ferrauto[1], Onofrio Gigliotta[2] and Stefano Nolfi[1]

Abstract

We introduce FARSA, an open-source Framework for Autonomous Robotics Simulation and Analysis, that allows us to easily set up and carry on adaptive experiments involving complex robot/environmental models. Moreover, we show how a simulated iCub robot can be trained, through an evolutionary algorithm, to display reaching and integrated reaching and grasping behaviours. The results demonstrate how the use of an implicit selection criterion, estimating the extent to which the robot is able to produce the expected outcome without specifying the manner through which the action should be realized, is sufficient to develop the required capabilities despite the complexity of the robot and of the task.

Keywords

Evolutionary robotics, embodied cognition, open software, simulation framework

1 Introduction

Adaptive behaviour models focus on the study of how embodied agents develop their capabilities autonomously while interacting with their physical and (eventually) social environment. For many years, these studies have been confined to relatively simple agents and tasks. Recent research, however, demonstrated how this method can be extended to studies that involve agents with complex morphologies and rich sensory–motor systems mastering relatively hard tasks (Baranes & Oudeyer, 2013; Massera, Tuci, Ferrauto, & Nolfi, 2010; Reil & Husbands, 2002; Rolf, Steil, & Gienger, 2010; Savastano & Nolfi, 2012; Tuci, Massera, & Nolfi, 2010; Yamashita & Tani, 2008). From a modelling point of view complexity does not represent a value in itself. We fully bound the Occam's razor argument that claims that given two explanations of the data, all other things being equal, the simpler explanation is preferable. After all, one of the key contribution of adaptive behaviour research consists in the demonstration of how complex abilities can emerge from the interactions between relatively simple agents and the environment. On the other hand, the modelization of a given phenomenon necessarily require the inclusion of the characteristics that constitute key aspects of the targeted objective of study. In some cases, therefore, the use of complex agents and/or tasks is necessary. For example, the modelization of the morphological characteristics and of the articulated structure of the human arm constitutes a prerequisite for modelling human object manipulation skills. Likewise, the use of agents provided with rich sensory systems constitutes a necessary prerequisite for modelling sensory integration and fusion.

From a methodological perspective, however, the need to build rather complex models for tackling these research issues currently represents a barrier that might significantly slow down research progress in this area. In this paper we introduced FARSA, an open source software tool that allows to easily set up and carry out adaptive experiments based on the iCub humanoid robot (Metta, Sandini, Vernon, Natale, & Nori, 2008; Sandini, Metta, & Vernon, 2004) as well as on other robotic platforms. FARSA does not only provide a simulator, since it consists of a set of integrated libraries: a robot/environmental simulator, a sensor and actuator library, a controller library, and an adaptive library. Moreover, it comes with a rich graphical interface that facilitates the visualization and analysis

[1]Institute of Cognitive Sciences and Technologies, CNR, Rome, Italy
[2]NAC Laboratory, Department of Humanities, University of Naples Federico II, Naples, Italy

Corresponding author:
Onofrio Gigliotta, NAC Laboratory, Department of Humanities, University of Naples Federico II, 1 via Porta di Massa, Naples 80133 Italy.
Email: onofrio.gigliotta@unina.it

of the characteristics of the model and of the behavioural and cognitive processes originating from the agent/environmental interaction. For these reasons we believe that it can contribute to boost adaptive behaviour research addressing the acquisition of multiple skills and the development complex capabilities.

We then illustrate a series of experiments in which an iCub robot (Metta et al., 2008; Sandini et al., 2004) is trained through an evolutionary algorithm for the ability to display integrated reaching and grasping capabilities. The results obtained in these experiments demonstrate how the use of an implicit selection criterion, estimating the extent to which the robot is able to produce the expected outcome of the actions, is sufficient to develop the required capabilities despite the complexity of the robot, of the robot's sensory–motor system, and of the task. These experiments have been realized through the use of FARSA and constitute two of the exemplificative examples provided with the tools. Therefore, they can be easily be replicated and varied by the reader.

In the next section we introduce FARSA. In Section 3 we describe the relation of our experiment on integrated reaching and grasping to the state of the art. In Section 4 and 5 we describe our experiments and results. Finally in Section 6 we draw our conclusions.

2 FARSA

FARSA (see http://laral.istc.cnr.it/farsa/) is an open-source tool designed to carry on experimental research in embodied cognitive science and adaptive behaviour.

It combines in a single framework the following features:

- It is open-source, so it can be freely modified, used, and extended by the research community.
- It is constituted by a series of integrated libraries that allow it to easily design the different components of an embodied model (i.e. the agents' body and sensory–motor system, the agents' control systems, and the ecological niche in which the agents operate) and that allow to simulate accurately and efficiently the interactions between the agent and the environment.
- It comes with a rich graphical interface that facilitates the visualization and analysis of the elements forming the embodied model and of the behavioural and cognitive processes originating from the agent/environment interactions.
- It is based on a highly modular software architecture that enables a progressive expansion of the tool features and simplifies the implementation of new experiments and of new software components.
- It is multi-platform, i.e. it can be compiled and used on Linux, Windows, and Mac OS X operating systems.
- It comes with a set of exemplificative experiments and with a synthetic but comprehensive documentation that should enable users to quickly master the tool usage.

Other related tools include: Webots™ (Michel, 2004), USARSim (Carpin, Lewis, Wang, Balakirsky, & Scrapper, 2007), Gazebo (Koenig & Howard, 2004), ARGOS (Pinciroli et al., 2012), and LpzRobots (Der & Martius, 2012).

In the following sub-sections we briefly review the characteristics of its main components.

2.1 The robots/environment simulator library

The robots/environment simulator (worldsim) is a library that allows the simulation of robots and the environment in which they operate. The library supports both individual robot simulation and collective experiments in which several robots are placed in the same environment. The physical and dynamical aspects of the robots and of the robot/environment interactions can be simulated accurately by using a 3D dynamics physics simulator or by using a faster but simplified kinematic engine. For what concern the dynamics simulation, FARSA relies on the Newton Game Dynamics engine (Jerez & Suero, 2004) that enables accurate and fast simulations. The underlying dynamic engine has been encapsulated so as to enable the inclusion of alternative engines.

Currently, FARSA supports the following robotic platforms: the Khepera (Mondada, Franzi, & Ienne, 1993), the e-Puck (Mondada et al., 2009), the marXbot (Bonani et al., 2010), and the iCub (Sandini et al., 2004). These robots have been designed by assembling a series of building blocks (physical elements, sensors, and motorized joints) that users can re-use to implement alternative, not yet supported, robots.

In the case of the iCub, the simulator is based on the YARP (Metta, Fitzpatrick, & Natale, 2006) middleware library (the same command used to read the robot's sensors and control the robot's motor can be used to work with the simulated or real robot). This strongly facilitates the possibility to port results from simulation to reality and the possibility to integrate into FARSA projects the software modules available from the iCub software repository (http://wiki.icub.org/iCub_documentation).

With respect to the iCub simulator developed by Tikhanoff et al. (2008), the simulation library included in FARSA presents a series of advantages: it strictly conforms to the real kinematic joint structure of the robot, it allows to simulate multiple robots, it includes both a dynamic and kinematic engine, and it provides an enhanced visualization tool.

2.2 The sensor and motor library

FARSA also includes a library of ready-to-use sensors and actuators. In some cases, sensors and actuators include software routines that pre-elaborate sensory or motor information (e.g. to reduce its dimensionality) and/or integrate different kinds of sensory–motor information (as in the case of actuators that set the torque to be produced by a joint motor on the basis of the current and desired position of the controlled joint).

Wheeled robots are provided with infrared, ground, traction force, linear vision, and communication sensors, among others. Moreover, they are provided with wheels, grippers, LEDs, and communication actuators.

The iCub robot is provided with proprioceptors that measure the current angular position of the robot's joints, tactile sensors, and vision sensors among others and with actuators that control all the available DOFs. The state of the robot's sensors and actuators, as well as the state of selected variables of the robot's control system, can be graphically visualized while the robot interacts with the environment. This provides a useful analysis and debugging tool.

2.3 The controller libraries

These libraries enable the user to design, modify, and visualize the robot's control system. Currently FARSA includes two libraries that support the design of neurocontrollers. Users willing to use other architectures or formalisms can integrate into FARSA alternative libraries.

Evonet is an easy-to-use library that enables users to graphically design, modify, and visualize the architecture of the robot's neural controller as well as the properties of the neurons and of the connection weights. The library supports logistic, leaky integrator, and threshold neurons. NNFW is an alternative object-oriented library that provides a larger variety of neuron types and output functions (Gaussian, winner-take-all, ramp, periodic, etc.) and supports the use of a radial basis function neural network.

Thanks to the integration between the controller and the sensory and motor libraries, the sensory and motor layer of the neural controller is automatically generated on the basis of the selected sensors and actuators. Moreover, the update of the sensory neurons and the update of the actuators on the basis of the state of the motor neurons is handled automatically.

Finally, the graphic viewer of the robot's controller also enables users to lesion and/or to manually manipulate the state of the sensors, internal, and motor neurons in order to analyse the relationship between the state of the controller and the behaviour that originates from the robot/environmental interaction.

2.4 The adaptation libraries

These libraries enable the user to subject a robot or a population of robots to an adapting process (i.e. to a evolutionary and/or learning process during which the characteristics of the robots are varied and variations are selected so as to improve the abilities of the robots to cope with a given task/environment).

The adaptation libraries that are currently available support the use of evolutionary algorithms (including steady state, truncation selection, and Pareto-front algorithms), supervised learning algorithms (i.e. back-propagation), and unsupervised learning algorithms (i.e. Hebbian learning). The evolutionary algorithms are parallelized at the level of the individual's evaluation and can therefore run significantly faster in multi-core machines and computer clusters.

In the case of evolutionary and supervised algorithm, the variation in performance during the adaptation can be monitored and analysed in the associated graphics renderer.

2.5 Usability and speed

FARSA is well documented, easy to use, and provided with a rich graphical interface that facilitates monitoring and debugging. The inclusion of exemplificative experiments (including the two experiments described in this paper) enables easy replication and a variety of interesting case studies.

A large spectrum of experiments can be configured and varied through parameters. More specifically, the type of robotic platform, the sensors and actuators of the robot, the characteristics of the neural controller, and the type and the characteristics of the adaptive process can be set and varied easily through the graphical interface or through a text editor. The realization of experiments that involve non-parametric variations (i.e. that require a new type of fitness function or a new type of sensor) require writing C++ extensions. This task, however, is facilitated by the fact that experiments are defined as plugins, i.e. relatively short programs that can be compiled separately from FARSA and loaded at runtime. Plugins can also be used to implement larger software extensions (e.g. new learning algorithms or new graphics widgets).

FARSA is optimized and parallelized so to reduce as much as possible the time required to carry on computationally expensive experiments. The simulation speed clearly depends on the complexity of the robotic platform and of the robot/environment interactions. In the case of the experiments described below, the simulation of the robot/environmental interaction on a standard single processor (Quad-Core AMD Opteron™ Processor 2374 HE at 2.2 GHz) under Linux runs 116 times and 2.6 times faster than real time (in the case of the experiments reported in Sections 4 and 5, respectively). Moreover the simulation of the evolutionary process on a multi-thread cluster runs approximately 666 times and 19 times faster on two quad-core processors using 8 threads (the same type of processor and operating system as above, experiment reported in Sections 4 and 5, respectively).

3 Relation to the state of the art

Reaching and grasping capabilities can be developed through trial-and-error and/or supervised learning methods (Barto, 2003). In trial-and-error methods, the motor capability is acquired without the help of an explicit teacher or trainer, and the adaptive process is driven by intrinsic feedback. Examples of intrinsic feedback are the kinesthetic and tactile sensations experienced when an object has been successfully grasped or the sight of a ball entering inside the net after a kicking action. In supervised training methods, instead, the intrinsic feedback is augmented with extrinsic information provided by the teacher. This information might consist of the sequence of sensory states experienced by the robot while its arm is driven by a caretaker toward a target object to be reached (in kinesthetic teaching methods, see for example Yamashita and Tani (2008)) or by the demonstration performed by the teacher of the action that should be performed by the robot (in learning by demonstration methods, see for example Miyamoto and Kawato (1998)). Most of the research in the field of artificial intelligence and adaptive behaviour focus on the latter paradigm. In this paper, instead, we will focus on trial-and-error methods relying on intrinsic feedback (e.g. the robot's capability to perceive whether or not and eventually to what extent a reaching and/or grasping action has been successfully carried out). Previous attempts to study how robots can develop reaching or grasping capabilities through trial-and-error methods include experiments with non-redundant systems provided with two actuated DOFs (Berthier, Rosenstein, & Barto, 2005; Schlesinger, Parisi, & Langer, 2000) or experiments in which the robots were provided with significant built-in competences (Oztop, Bradley, & Arbib, 2004). More specifically, Schlesinger et al. (2000) studied the development of reaching behaviour in a simulated agent provided with a 2-dimensional arm with two actuated DOFs, a bi-dimensional vision system with one actuated DOF, and a tactile sensor located on the final portion of the arm. The robot's neural network controller received as input the angular state of the arm joints, the state of the tactile sensor, and the visual information extracted from the camera and control the two DOFs of the arm and one DOF of the visual system. The neural network controller was trained through an evolutionary method (Nolfi & Floreano, 2000) on the basis of a performance criterion calculated by computing the average number of time steps spent by the robot touching the target object. Oztop et al. (2004) studied the development of grasping behaviour in a simulated robot provided with an arm and hand with 19 actuated DOFs. The reaching behaviour was pre-programmed in the robot on the basis of the Jacobian transpose method (Sciavicco & Siciliano, 2004). Learning was thus confined to the mapping of a series of sensorily extracted object affordances into a series of grasping parameters able to shape the pre-existing reaching capability into an effective grasping behaviour. The neural network controller was trained through a reinforcement learning algorithm (Sutton & Barto, 1998) and received positive reward for the trials producing successful or nearly successful grasps and negative reward for trials leading to unstable grasps or no object contact. Berthier et al. (2005) studied the development of a reaching behaviour in a simulated robot provided with an arm with 2 controlled DOFs on the shoulder (flexion–extension and adduction–abduction). The robot's neural network controller received as input the current state and velocity of the two joints and produced as output the intensity of the torque to be applied by two muscle-like actuators. The network was trained through a reinforcement learning algorithm (Sutton & Barto, 1998) by providing to the robot positive and negative rewards when the hand of the robot approached or moved away from the target, respectively. The experiments described in this paper, instead, concern the study of how a highly redundant humanoid robot can develop reaching and grasping capabilities from scratch or on-top of simple reflex-like competences. The relation between the experiments presented in this paper and our own previous related work (Massera, Cangelosi, & Nolfi, 2007; Massera et al., 2010; Savastano & Nolfi, 2012, 2013) will be discussed below. Although the first phase of reaching and grasping development in children are clearly characterized by a trial-and-error learning process (Oztop et al., 2004), the objective of this paper is not to model human learning but rather to demonstrate how apparently complex behavioural capabilities can be successfully acquired through a simple trial-and-error adaptive process that do not require specification of the manner through which the target actions should be realized.

4 Reaching

In this section we describe how a simulated iCub robot can acquire the capability to reach with its left arm any arbitrary target position in its peripersonal space by controlling six actuated joints (two joints of the iCub's torso and four joints of the iCub's left arm). The connection weights of the robots' neural controller are adapted through an evolutionary method for the ability to minimize the average distance between the left hand of the robot and the target location averaged over

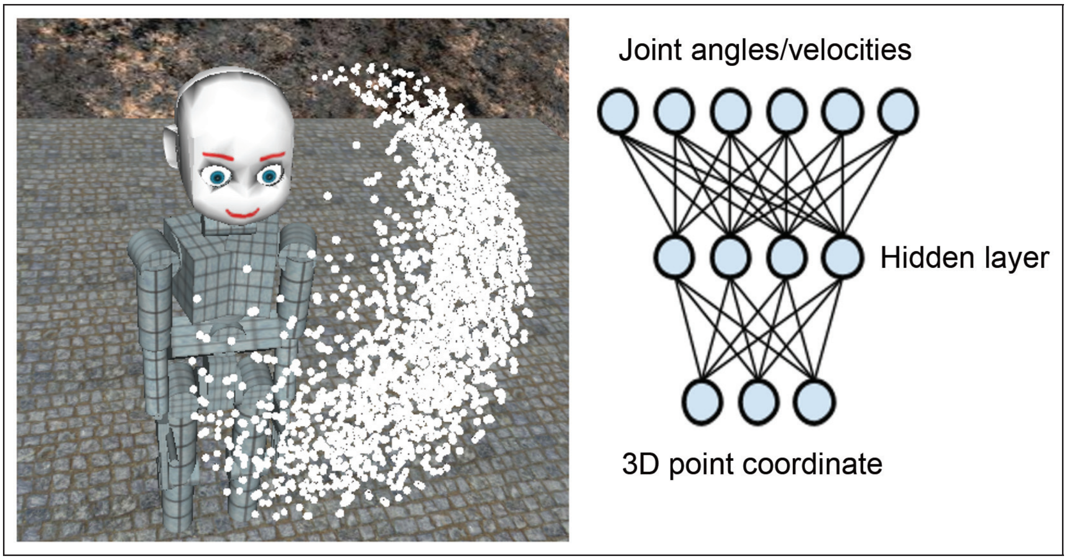

Figure 1. (*Left*) The simulated iCub. The white points shows all the possible target positions (see text for an explanation). (*Right*) The architecture of the neural controller. The lower, intermediate, and upper layer indicate the sensory, internal, and motor neurons. Lines represents connections from the lower to the upper layer.

several trials in which the robot has to reach different target positions.

4.1 Method

The robot's neural controller (Figure 1) is provided with three sensory neurons that encode the position of the target object in Cartesian coordinates, four internal neurons, and six motor neurons that control the desired angular position or velocities (depending on experimental setup, see below) of the six actuated DOFs (i.e. the rotation and the extension–flexion of the torso; the extension–flexion, the abduction–adduction and the supination–pronation of the arm; the extension–flexion of the forearm). The sensory neurons are fully connected to internal neurons that are fully connected to motor neurons. To verify the role of the sensory feedback during the robot/environment interactions we ran two sets of experiments. In **steady-encoding** experiments the sensory neurons encode the offset of the current target position along the X, Y, and Z axes, normalized in the range [0,1], with respect to centre of the iCub body, and the motor neurons encode the desired angles for the final posture of the arm by using a linear mapping (actual torques are set through a PID controller). The offset between the desired and the target angular positions are then used to set the velocity of the joint motors on the basis of a simple proportional controller. In the **unsteady-encoding** experiments, instead, the three sensory neurons encode the offset of the current target position along the X, Y, and Z axes, normalized in the range [0,1], with respect to the centre of the left palm, and the motor neurons encode directly the velocity of the joint motors. In the latter case the robot can use the perceptual feedback of its own actions to refine its behaviour while it interacts with the environment. In the former case, instead, the sensory state does not change while the robot moves and consequently the robot cannot exploit the sensorial effects of its own actions. Moreover, the sensorial information perceived in the **unsteady-encoding** correlate directly with the extent to which the robot has successfully carried out its action.

The output of internal and motor neurons was computed accordingly the following equation:

$$o_i = \sigma\left(\sum_{j=0}^{N} x_j w_{ji}\right) \tag{1}$$

where σ is the standard logistic function: $1/(1 + e^{-x})$, x_j is the output of the jth presynaptic neuron and w_{ji} is the synaptic weight from the jth presynaptic to ith postsynaptic neuron. The update rate of the state of the sensors, of the neural controller, of the actuators, of the robot and of the environment is 25 Hz. The characteristics of the robot and of the architecture of the robots' neural network are kept fixed. The strength of the connection weights are adapted by using an evolutionary method (Nolfi & Floreano, 2000). The initial population consists of 20 randomly generated genotypes, which encode the 46 free parameters of 20 corresponding neural controllers. Each gene is constituted by 8 bits that encode a corresponding floating point value in the range $[-5.0, +5.0]$. During each generation, each individual is allowed to produce an offspring (i.e. a genotype identical to that of the parent with 5% of its bit randomly mutated). The 20 parent and the 20 offspring

Table 1. Angular positions selected uniformly within the joints limits used to generate a representative set of all possible reachable positions.

Joint	Limits
Torso rotation	$[-25, 0, +25]$
Arm abduction–adduction	$[32.16, 64.32, 96.48, 128.64]$
Torso extension–flexion	$[-2.5, 5, 12.5]$
Arm supination–pronation	$[-13.6, 9.8, 33.2, 56.6]$
Arm extension–flexion	$[-74.4, -53.3, -32.2, -11.1]$
Forearm extension–flexion	$[33.2, 51.4, 69.6, 87.8]$

individuals are evaluated. The genotypes of offspring individuals that outperform parents are used to replace the genotypes of the worst parents. The genotypes of the remaining offspring are discarded. The reproduction, evaluation, and selection process is repeated for 5000 generations. Each individual is evaluated for 400 trials, each lasting up to 17.5 s, during which it should reach 400 corresponding target positions extracted from a set of 2304 reachable positions. To get a subset of points as equidistributed as possible, we use the crowding distance (Deb, Agrawal, Pratap, & Meyarivan, 2000) to sort all reachable positions on the basis of their Cartesian coordinates. The 2304 reachable points have been calculated by storing the position that the centre of the robot's left palm assumed when the six actuated joints were moved to all possible combination of states within the values indicated in Table 1. Furthermore, to favour the selection of individuals that are able to generalize their abilities to any possible reachable position, the target position experienced during each trial was randomly chosen within a spherical area with a diameter of 2 cm centred around the current extracted reachable position.

The fitness is calculated on the basis of the following equation:

$$F = \sum_{t=1}^{400} e^{\left(-\frac{\|\mathrm{palmPos}-\mathrm{targetPos}\|}{0.04}\right)} \qquad (2)$$

Where t is the trial, and $\|$ palmPos $-$ targetPos $\|$ is the Euclidean distance in meters between the centre of the robot's left palm and the centre of the target location measured at the end of each trial. To verify whether an incremental adaptive process can lead to better performance, we ran an additional set of experiments referred to below as incremental. More specifically, to simulate the condition on which the problem is initially simplified and become progressively harder as soon as the skills of the individuals improve, they are rewarded with the maximum fitness during trials in which the palmPos $-$ targetPos distance is below a threshold . This threshold is initially set to 5 cm at generation 0 and it is progressively reduced by 20% after a

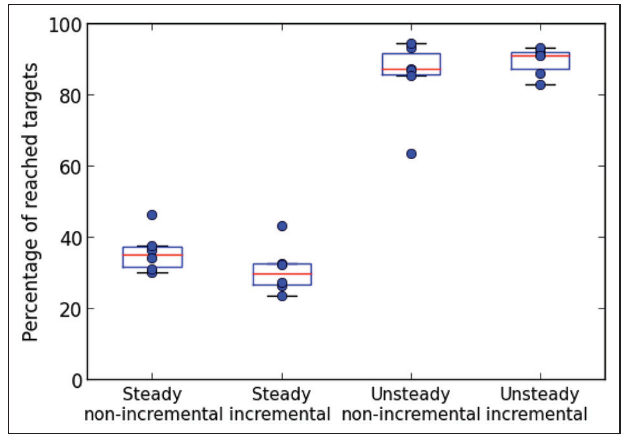

Figure 2. Percentage of target locations reached with an accuracy of at least 5 cm (i.e. with a distance between the centre of the palm and the centre of the target location less or equal to 5 cm). The four box plots show the distribution of performance of the six best individuals each from an independent run with random initial conditions.

generation in which the average fitness of all individuals is greater than 0.6, and it is definitely set to 0.0 when it becomes lower than 1 cm. For sake of comparison, consider that the height of the iCub is about 1 m.

4.2 Results

The combination of the steady versus unsteady encoding and incremental versus non-incremental adaptive process leads to four sets of experiments. For each experiments six replications starting from different randomly generated populations were run. Evolved individuals were then post-evaluated on the entire set of 2304 reachable target locations by calculating percentage of target location reached with an accuracy of at least 5 cm.

By analysing the performance of the best evolved individuals in the four experimental conditions (see Figure 2), we can see how the individuals evolved in the unsteady condition significantly outperform those evolved in the steady condition. This results confirms that the possibility to exploit the sensory feedbacks caused by the robot's actions and/or the availability of information that strongly correlates with the extent to which the robot successfully accomplishes the current action strongly facilitates the development of the effective solutions.

The comparison of the performance obtained in the incremental versus non-incremental experimental conditions does not reveal significant differences.

Overall the analysis of the results in the best experimental conditions indicates how the adapted individuals can reach close to optimal performance. This is a remarkable result given the simplicity of the neural controller and given that some of the targets located in peripheral areas of the robot's peripersonal space are hard to reach due to the limits and constraints that affect the robot's movements in these regions.

For instructions on how to replicate this experiment with FARSA and on how to analyze the evolved solutions, see http://laral.istc.cnr.it/res/reach.

5 Reaching and grasping

In this section we describe how a simulated iCub robot can develop integrated reaching and grasping capabilities that enable it to reach a ball located in varying positions over a table, grasp it, handle it, and elevate it. Beside the difficulties concerning the need to control an articulated arm with many DOFs (Bernstein, 1967), this represents a rather challenging task since it requires interaction with physical objects (including a sphere that can easily roll away from the robot's peripersonal space) and integration of three interdependent behaviours (reaching, grasping, and lifting).

5.1 The method

In the case of this experiment, the robot's controller includes a richer set of sensors and actuators, a larger neural network, and a greater number of parameters to be varied during the adaptive process. Adapting individuals are provided with an hand-coded neural circuit that produce a simple reflex behaviour consisting in turning the robot head toward red objects.

The sensory system (Figure 3(b), bottom layer) includes two neurons that encode the offset between the sphere and the hand over the visual plane (*dx*, *dy*, by visual plane we means the two dimensional image perceived by the robot's camera), four neurons that encode the current angular position of the pitch and yaw DOFs of the neck (n0, n1) and of the torso (t0, t1), and nine sensory neurons that binarily encode whether the five tactile sensors located on the fingertips (Rf1, Rf2, Rf3, Rf4, Rf5) and the four tactile sensors located on the palm (Rp1, Rp2, Rp3, Rp4) are stimulated.

The motor system (Figure 3(b), top layer) includes two motor neurons that control the desired angular position of pitch and yaw DOFs of the torso (T0 and T1), seven motor neurons that control the desired angular position of the seven corresponding DOFs of the right arm and wrist (RA0, RA1, RA2, RA3, RA4, RA5 and RA6) and a right-hand motor (RF0) that controls the desired angular position of all joints of the hand (the fingers' abduction–adduction is kept fixed). This means that all fingers extend/flex together.

The neural network is also provided with seven internal neurons that receive connections from all sensory neurons and project connection to all motor neurons (Figure 3(b), intermediate layer). These neurons are leaky integrators, that is their activation at a given time step depends on both the input at that time step and on the activation at the previous time step. The output of the *i*th internal neuron is computed as follows:

$$o_{i,t} = \alpha_i o_{i,t-1} + (1-\alpha_i)\sigma\left(\sum_{j=0}^{N} x_{j,t} w_{ji}\right) \qquad (3)$$

where $o_{i,t}$ is the output of the *i*th internal neuron at time step t, α_i is a time integrator parameter that determines how much the output at the current time step depends on the output at the previous time step, $\sigma(z)$ is the standard logistic function as before, $x_{j,t}$ is the output of the *j*th presynaptic neuron at time step t and w_{ji} is the synaptic weight from the *j*th presynaptic to *i*th postsynaptic neuron. The update rate of the state of the sensors, of the neural controller, of the actuators, of the robot and of the environment is 25 Hz (50 ms per step).

The reflex behaviour is realized by a neural circuit (Figure 3(a)) with two sensory neurons, that encode the average offsets of red pixels over the vertical and horizontal axis of the visual field, which are directly connected to two motor neurons, that control the angular position of the neck (N0, N1). The four weights and the two biases of the motor neurons are set manually. The other 226 parameters are adapted.

At the beginning of each trial the sphere is placed in a random position inside one of four square areas with a side of 4 cm (Figure 4). The first two of these areas are located in front of the iCub at a distance of 25 cm and 35 cm, the other two are located 10 cm on the left and on the right side and at a distance of 30 cm. Each trial lasts 300 time steps (i.e. 15 s) plus 10 additional time steps during which the plane is removed to verify whether or not the ball is held by the robot.

The fitness is computed on the basis of the following equations:

$$F_t = 0.3D_t + 0.2T_t + 0.2O_tC_t + 0.3Q_t + G_t \qquad (4)$$

$$F = \sum_{t=1}^{N} F_t \qquad (5)$$

Where F is the overall fitness of the individual, F_t is the fitness at trial t, N is the number of trials and D_t, T_t, O_t, C_t, Q_t, and G_t are fitness components, ranging from 0 to 1, that reward the individuals for bringing their hand near the object (D_t),touching the object with the palm (T_t), opening the fingers far from the object (O_t), closing the finger near the object (C_t), closing the finger around the object (Q_t), holding and elevating the object (G_t). These fitness components have been introduced to increase the individuals' evolvability (i.e. the probability that random variations might lead to performance improvements) and to channel the adaptive process toward the acquisition of abilities that constitute a prerequisite for the development of the required

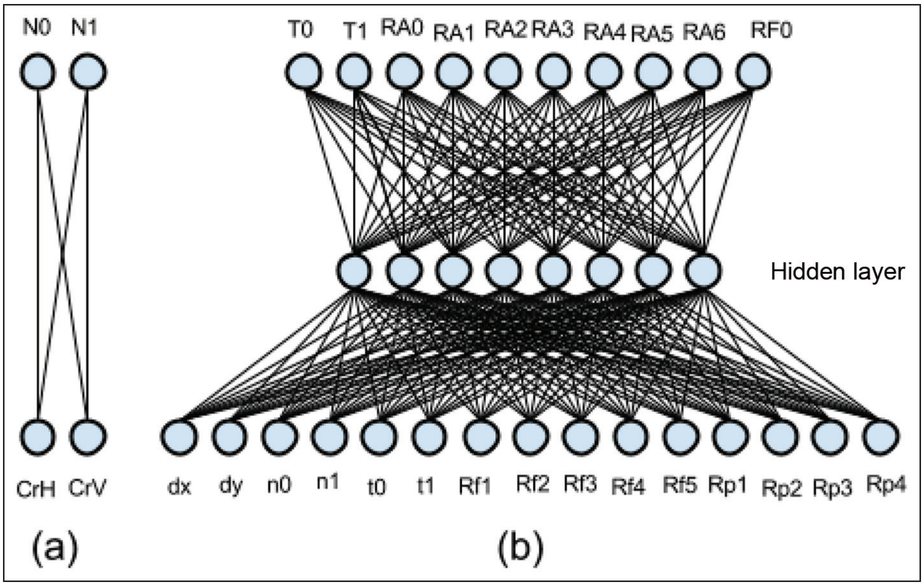

Figure 3. The architecture of robot's neural controller. The lower, intermediate, and upper layer represent the sensory, internal, and motor neurons, respectively. Lines represents connections from the lower to the upper layer. The connection weights and biases and of the neural circuit shown in (a) are manually set and fixed. All other connection weights and biases are adapted.

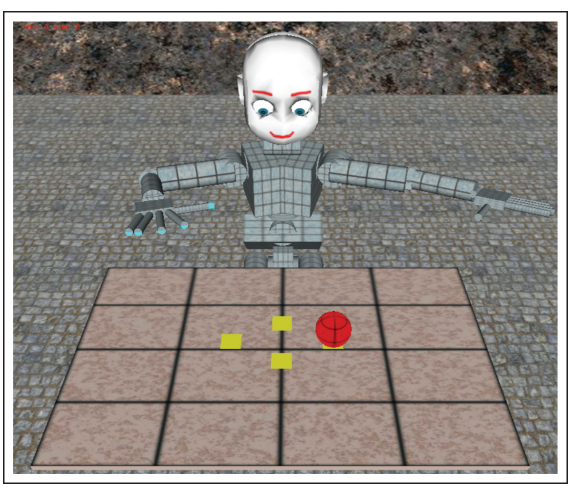

Figure 4. The experimental setup. The robot is shown in the posture set at the beginning of each trial. The left arm of the robot is not moved. The yellow squares on the table show the areas where the object can be located.

capabilities (we will come back to this issue in the discussion section). The used components and their parameters have been chosen on the basis of our intuition and have not be optimized on the basis of a trial and error approach. They are computed on the basis of the following equations (the subscript indicating the dependency on the trial has been removed for clarity):

$$D = e^{-5d} \tag{6}$$

$$T = \min\left(\frac{n}{10}, 1\right) \tag{7}$$

$$O = \frac{1}{n_0} \sum_{s=1}^{n_0} E_s \tag{8}$$

$$C = \begin{cases} \frac{1}{300-n_c} \sum_{s=n_c}^{300} 1 - E_s, & \text{if } n_c \neq 300 \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

$$Q = \min\left(\frac{f}{4}, 1\right) \tag{10}$$

$$G = \begin{cases} 0, & \text{if } o_z \leq -0.1 \\ 0.5 + \frac{o_z + 0.1}{0.2} 0.5, & \text{if } -0.1 < o_z < 0.1 \\ 1, & \text{if } o_z \geq 0.1 \end{cases} \tag{11}$$

where d is the distance between the centre of the palm and the surface of the object at the end of the trial; n is the number of steps in which the palm of the robot touched the object during the current trial; n_0 and n_c are the steps at which palm enters in contact with the object for the first and for the fifth time, respectively, or 300 when the conditions are never satisfied; E_s is the extension of the fingers at step s; f is the maximum number of fingers that entered in contact with the object concurrently during the trial; o_z is the displacement along the vertical axis of the object centre (0 means the object is exactly on the table).

To support the evolution of robust behaviours while minimizing the simulation costs, the number of trials is initially set to 4 and is then increased to 8, 12, 16, 20, 24, and 28 as soon as an evolving individual successfully grasps and holds the objects during 50%, 60%, 70%, 80%, 90% and 100% of the trials. Five replications of the experiment lasting 2000 generations were run. All other parameters were identical to that described in Section 3.

Table 2. Percentage of trials in which the best evolved robot of each replication successfully grasps and hold the ball during a post-evaluation test conducted for 100 trials.

Replication	1	2	3	4	5
Success rate (%)	77	74	69	66	50

5.2 Results

By analysing the obtained results we observed that in all replications of the experiment the evolved individuals display an ability to reach, grasp, and hold spherical objects located in varying positions (Table 2). In the case of the best replication of the experiment, the best individual displays a rather robust capability that allows it to successfully carry on the task in 77% of the trials. This represents a remarkable result in consideration of the rigidity of the robot body and of the difficulties of physically interacting with spherical objects that can easily roll away from the peripersonal space of the robot. The obtained solutions also represent progress with respect to the previous studies carried by some of the authors (see Massera et al. (2007)), in which the individuals were able to successfully accomplish a similar task but showed limited generalization capabilities with respect to variations of the object positions.

The visual inspection of the behavioural solutions displayed by these individuals (see http://laral.istc.cnr.it/res/reach-grasp/) can allow us to appreciate the importance played by the integration between the required elementary behaviours (i.e. reaching, grasping, and lifting) and by the way in which they are combined over time. Indeed, the way in which the best evolved individuals reach the object by bending the torso toward the table and by carefully pressing the ball over the table so as to block it, while the fingers are wrapped around the object, clearly demonstrate the importance of the fact that the reaching and the grasping abilities have been co-evolved to serve a common function.

Overall this demonstrates the potential advantages of acquiring the required elementary behavioural capacities through an adaptive process and of using methods that enable the co-development of multiple capacities. More specifically, for what concerns the experiments illustrated above, this suggests that the introduction of a fitness component that rewards the development of the required elementary capabilities and of components that reward the ability to appropriately combine and integrate the acquired elementary capabilities might be crucial for the development of general and effective solutions.

For instructions on how to replicate this experiment with FARSA and on how to analyse the evolved solutions see http://laral.istc.cnr.it/res/reach-grasp/.

6 Discussion and conclusion

The possibility to design adaptive agents able to develop their behavioural skills autonomously, while they interact with the physical and social environment in which they are situated, represented one of the most fascinating scientific landmarks of the end of the last century. Whether and how such methods can enable the synthesis of robots able to acquire complex behavioural and cognitive skills and able to progressively expand their behavioural and cognitive repertoire still represents an open question.

To achieve this challenging objective agents need to be able to first develop elementary capabilities and then more complex skills by recombining and integrating previously developed skills. However, the way in which this can be achieved still represents an open question.

A possible approach postulates a modular organization of the agents' control system in which different modules support the acquisition and production of the elementary capabilities and in which the elementary modules/capabilities are then combined to produce more complex skills (Mataric, 1998; Schaal, 2002; Wolpert & Kawato, 1998). The composition, however, is far from easy to achieve (Nemec & Ude, 2012), often employs very heuristic schemes (Reinhart, Lemme, & Steil, 2012), or needs in itself sophisticated modelling approaches (Kulic, Ott, Lee, Ishikawa, & Nakamura, 2012; Wrede et al., 2012).

An alternative approach postulates that multiple and complex capabilities can be obtained by recombining previously acquired behavioural and cognitive skills that do not necessarily correspond to different parts of the agent's control system (Nolfi, 2009; Yamashita & Tani, 2008). Within this approach, compositionality is seen as a property that arises from the acquisition and integration of multiple skills rather than a consequence of architectural constraints. The question of whether and how this approach can really lead to the progressive acquisition of a rich behavioural and cognitive repertoire, however, still remains to be answered.

In this paper we introduced a software framework that enables researchers to easily perform and analyse adaptive experiments involving relatively complex agents and tasks. We believe that the availability of tools of this type can significantly contribute to boost research in adaptive robotics by enabling the investigation of hard problems and the comparison of alternative models and methods.

Moreover, we reported the result of a series of experiments that demonstrate how a relatively complex humanoid robot provided with a simple non-modular controller can acquire multiple integrated behavioural capabilities. This is achieved through the use of multiple component fitness functions that enhance the evolvability of the system and channel the adaptive process toward promising directions. The question of whether and how this type of approach can scale to larger behavioural repertories constitutes an important research challenge for future research.

Funding

The authors gratefully acknowledge the financial support provided by the Italian National Research Council (CNR) through the EuroBioSAS Programme of the European Science Foundation.

References

Baranes, A., & Oudeyer, P.-Y. (2013). Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, *61*(1), 69–73.

Barto, A. G. (2003). Reinforcement learning in motor control. In M. A. Arbib (Ed.), *Handbook of brain theory and neural networks* (pp. 968–972). Cambridge, MA: MIT Press.

Bernstein, N. (1967). *The co-ordination and regulation of movements*. Oxford, UK: Pergamo.

Berthier, N. E., Rosenstein, M. T., & Barto, A. G. (2005). Approximate optimal control as a model for motor learning. *Psychological Review*, *112*, 329–346.

Bonani, M., Longchamp, V., Magnenat, S., Rtornaz, P., Burnier, D., Roulet, G., …, Mondada, F. (2010). The marXbot, a miniature mobile robot opening new perspectives for the collective-robotic research. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4187–4193). IEEE.

Carpin, S., Lewis, M., Wang, J., Balakirsky, S., & Scrapper, C. (2007). USARSim: A robot simulator for research and education. In *Proceedings of the IEEE international conference on robotics and automation (ICRA)* (pp. 1400–1405). IEEE.

Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Parallel Problem Solving from Nature PPSN VI* (pp. 849–858). Berlin: Springer.

Der, R., & Martius, G. (2012). The LpzRobots Simulator. In R. Der, & G. Martius (Eds.), *The playful machine: theoretical foundation and practical realization of self-organizing robots* (*Vol. 15*) (pp. 293–308). Berlin, Germany: Springer Verlag.

Jerez, J., & Suero, A. (2004). Newton game dynamics. Retrieved www.newtondynamics.com

Koenig, N., & Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator. In *Proceedings of IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 2149–2154). IEEE.

Kulic, D., Ott, C., Lee, D., Ishikawa, J., & Nakamura, Y. (2012). Incremental learning of full body motion primitives and their sequencing through human motion observation. *International Journal of Robotics Research*, *31*(3), 330–345.

Massera, G., Cangelosi, A., & Nolfi, S. (2007). Evolution of prehension ability in an anthropomorphic neurorobotic arm. *Frontiers in Neurorobotics*, *1*(4).

Massera, G., Tuci, E., Ferrauto, T., & Nolfi, S. (2010). The facilitatory role of linguistic instructions on developing manipulation skills. *IEEE Compotational Intelligence Magazine*, *5*(3), 33–42.

Mataric, M. J. (1998). Behavior-based robotics as a tool for synthesis of artificial behavior and analysis of natural behavior. *Trends in Cognitive Sciences*, *2*(3), 82–86.

Metta, G., Fitzpatrick, P., & Natale, L. (2006). Yarp: Yet another robot platform. *International Journal of Advanced Robotics Systems, special issue on Software Development and Integration in Robotics*, *3*(1), 43–48.

Metta, G., Sandini, G., Vernon, D., Natale, L., & Nori, F. (2008). The iCub humanoid robot: An open platform for research in embodied cognition. In *Proceedings of the 8th workshop on performance metrics for intelligent systems* (pp. 50–56). New York, USA: ACM.

Michel, O. (2004). Webots: Professional mobile robot simulation. *Journal of Advanced Robotics Systems*, *1*(1), 39–42.

Miyamoto, H., & Kawato, M. (1998). A tennis serve and upswing learning robot based on bi-directional theory. *Neural Networks*, *11*(7–8), 1331–1344.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., …, Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions* (*Vol. 1*, pp. 59–65). IPCB.

Mondada, F., Franzi, E., & Ienne, P. (1993). Mobile robot miniaturisation: A tool for investigation in control algorithms. In *Proceedings of the 3rd international symposium on experimental robotics* (pp. 501–513). Tokyo: Springer.

Nemec, B., & Ude, A. (2012). Action sequencing using dynamic movement primitives. *Robotica*, *30*(5), 837–846.

Nolfi, S. (2009). Behavior and cognition as a complex adaptive system: Insights from robotic experiments. In C. Hooker (Ed.), *Handbook of the philosophy of science*. Amsterdam, The Netherlands: Elsevier.

Nolfi, S., & Floreano, D. (2000). *Evolutionary robotics*. Cambridge, MA: MIT Press.

Oztop, E., Bradley, N. S., & Arbib, M. A. (2004). Infant grasp learning: A computational model. *Experimental Brain Research*, *158*(4), 480–503.

Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., …, Dorigo, M. (2012). ARGoS: A modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence*, *6*(4), 271–295.

Reil, T., & Husbands, P. (2002). Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation*, *6*(2), 159–168.

Reinhart, F., Lemme, A., & Steil, J. J. (2012). Representation and generalization of bi-manual skills from kinesthetic teaching. In *Proceedings of the IEEE-RAS international conference on humanoid robots*. IEEE.

Rolf, M., Steil, J. J., & Gienger, M. (2010). Goal babbling permits direct learning of inverse kinematics. *IEEE Transactions on Autonomous Mental Development*, *2*(3), 216–229.

Sandini, G., Metta, G., & Vernon, D. (2004). Robotcub: An open framework for research in embodied cognition. In *Proceedings of IEEE/RAS international conference on humanoid robots* (pp. 13–32). IEEE.

Savastano, P., & Nolfi, S. (2012). Incremental learning in a 14 DOF simulated iCub robot: Modeling infant reach/grasp development. In T. Prescott, N. Lepora, A. Mura, & P. Verschure (Eds.), *Biomimetic and biohybrid systems* (Vol. 7375, pp. 250–261). Berlin; Heidelberg: Springer.

Savastano, P., & Nolfi, S. (2013). A robotic model of reaching and grasping development. *IEEE Transactions on Autonomous Mental Development*, *5*(4), 326–336.

Schaal, S. (2002). Learning robot control. In M. Arbib (Ed.), *The handbook of brain theory and neural networks* (pp. 983–987). 2nd edition.Cambridge, MA: MIT Press.

Schlesinger, M., Parisi, D., & Langer, J. (2000). Learning to reach by constraining the movement search space. *Developmental Science*, *3*, 67–80.

Sciavicco, L., & Siciliano, B. (2004). *Modelling and control of robot manipulators*. London; New York: Springer.

Sutton, R., & Barto, A. (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Tikhanoff, V., Cangelosi, A., Fitzpatrick, P., Metta, G., Natale, L., & Nori, F. (2008). An open-source simulator for cognitive robotics research: The prototype of the iCub humanoid robot simulator. In *Proceedings of the 8th workshop on performance metrics for intelligent systems* (pp. 57–61). New York, NY: ACM.

Tuci, E., Massera, G., & Nolfi, S. (2010). Active categorical perception of object shapes in a simulated anthropomorphic robotic arm. *IEEE Transactions on Evolutionary Computation*, *14*(6), 885–899.

Wolpert, D. M., & Kawato, M. (1998). Multiple paired forward and inverse models for motor control. *Neural Networks*, *11*(7–8), 1317–1329.

Wrede, B., Rohlfing, K. J., Steil, J. J., Wrede, S., Oudeyer, P. -Y., & Tani, J. (2012). Towards robots with teleological action and language understanding. In *IEEE humanoids, workshop on developmental robotics*. Osaka: IEEE.

Yamashita, Y., & Tani, J. (2008). Emergence of functional hierarchy in a multiple timescale neural network model: A humanoid robot experiment. *PLoS Comput Biol*, *4*(11), e1000220.