A Developmental Roadmap for Learning by Imitation in Robots

Manuel Lopes, *Member, IEEE*, and José Santos-Victor, *Member, IEEE*

***Abstract*—In this paper, we present a strategy whereby a robot acquires the capability to learn by imitation following a developmental pathway consisting on three levels: 1) sensory–motor coordination; 2) world interaction; and 3) imitation. With these stages, the system is able to learn tasks by imitating human demonstrators. We describe results of the different developmental stages, involving perceptual and motor skills, implemented in our humanoid robot, Baltazar. At each stage, the system's attention is drawn toward different entities: its own body and, later on, objects and people. Our main contributions are the general architecture and the implementation of all the necessary modules until imitation capabilities are eventually acquired by the robot. Also, several other contributions are made at each level: learning of sensory–motor maps for redundant robots, a novel method for learning how to grasp objects, and a framework for learning task description from observation for program-level imitation. Finally, vision is used extensively as the sole sensing modality (sometimes in a simplified setting) avoiding the need for special data-acquisition hardware.**

***Index Terms*—Development, humanoid robots, imitation.**

I. Introduction

A GRAND challenge for robotics is establishing "friendly" and social interaction between robots and humans. Due to the diversity of actions/tasks to be performed and the range of possible interactions with objects and humans, it would be impractical (if not impossible) to explicitly preprogram a robot with such capabilities. Instead, such systems must be able to learn by themselves what tasks to execute and how they should be performed, which requires sophisticated motor, perceptual, and cognitive skills.

To address these challenges, we adopt two fundamental metaphors: 1) learning by imitation [1] as a powerful means to teach a complex humanoid (social) robot and 2) a developmental approach to balance the system complexity at the various levels of functional performance [2]–[4].

Manuscript received December 1, 2005; revised May 23, 2006. Work in Section II-B was done in collaboration with Nicolas Mansard and François Chaumette. This work was supported in part by the European Union Project (IST-004370) RobotCub and in part by the Fundação para a Ciência e a Tecnologia (ISR/IST plurianual funding) through the POS_Conhecimento Program that includes FEDER funds. This paper was recommended by Guest Editor Y. Demiris.

M. Lopes is with the Institute of Systems and Robotics, 1049-001 Lisbon, Portugal.

J. Santos-Victor is with the Department of Electrical and Computer Engineering, Instituto Superior Técnico, 1049-001 Lisbon, Portugal, and also with the Institute of Systems and Robotics, 1049-001 Lisbon, Portugal.

Digital Object Identifier 10.1109/TSMCB.2006.886949

A. Imitation

Learning by imitation is likely to become the primary form of teaching such social cognitive robots [1]. A very intuitive way to program a robot is to demonstrate the task to be performed. The system would learn how to solve similar tasks by looking at a human performance, avoiding the need for supervised learning, or trial-and-error rehearsals. This skill transfer has three major difficulties: 1) how to gather task-relevant information; 2) how to convert the data that is valid for a human to a different robot body; and 3) how to infer the important parts of the demonstration (e.g., "understand" the task).

Several approaches have been adopted to gather the information for imitation. An exoskeleton was used in [5] to capture kinematic data. The work presented in [6] relies on markers to get visual features for hand detection and grasping, in the context of imitation and modeling of the Mirror neurons. In our case, all the data are acquired solely with vision making it a more user-friendly system. Because of this, in some cases, the solutions were guided by the vision-problem considerations. Imitation and skill transfer between systems with different bodies (body correspondence) were first explicitly addressed in [7] using an algebraic formulation (bodies with different skills were considered). For the case of a humanoid robot, adaptation of the trajectories is used to guarantee the correct balance during task execution [8]. We address this problem in a very different way. Instead of trying to infer the complete state of the demonstrator, sensory–motor maps (SMMs) can give the information about the actions that give a certain perception, and so, the imitator will work on the perception space.

One of the first works in imitation was proposed in [9], a system able to learn how to imitate an assembly task by extracting a hierarchical description of the task. The problem of inferring the important parts of the task was addressed in [10] by casting it into an optimization framework. In a chesslike world, several metrics are studied, where the state or the action is considered [11]. In our case, the task interpretation is guided by the visual-processing restrictions. If someone is interacting with objects, there are many occlusions and ambiguous postures, making it very difficult to detect what action is being performed. Therefore, we rely on the visible effects on the objects by using a multiobject tracker that describes tasks by detecting points where the world changes state.

Even if imitation can allow a robot to learn a large variety of tasks, it is clear that it requires the existence of sophisticated motor, perceptual, and cognitive capabilities. Building such complex skills can become an overwhelming task in itself.

For learning one particular skill, many other systems need to be present and their interconnections properly established.

B. Development

How is it possible to deal with such complexity? In living beings, ontogenetic development from conception until adulthood is guided by a genetic program and the particular environment where it is embedded. The program is responsible to guide learning from the simplest things to the most complex ones. All the physical and cognitive capabilities will have to be develop from the interaction with the world and other people.

During the first months of life, infants have limited visual and motor capabilities. Both systems evolve side by side, with the visual system feeding information to "calibrate" hand/arm movements, and arm movements providing stimuli to train and improve visual acuity. "Several reflexes enable a good development of head and body control. During the last four months and the first four months postnatally, reflexive movement is such a dominant form of movement that the human being has been labeled a 'reflex machine.' By nourishing and protecting, the primitive reflexes are critical for human survival. The postural reflexes are believed to be basic to more complex, voluntary movement of later infancy" [12].

As one example, the "sucking reflex" enables a sucking action when there is a lip stimulation. It is easy to understand that without this reflex babies would not be able to eat or to learn how to eat.

For the case of the head–eye system, voluntary control appears very early. Some reflexive movements are evident from birth (head-righting reflex [12]), but voluntary control becomes apparent only at the end of the first month. A five-month-old child already shows good control. This control of the head will enable the tuning of the vision system to start looking at (and understanding) the environment. In [13], there is a discussion about the significance of neonate's arm movements. Usually, these motions are considered as unintentional, purposeless, or reflexive. Some experiments were done where a newborn could see its arm in three different situations: only the arm they were facing, only the opposite arm on a video monitor, or neither arm. Some small forces were applied to pull their wrists. The babies opposed the perturbing force so as to keep an arm up and moving normally, but only when they could see the arm either directly or on the video monitor. The experiments indicate that newborns can purposely control their arms in the face of external forces and that the development of visual control of the arm movement is underway soon after birth.

For object grasping, there are two very distinct phases [14]. In phase 1, there is a simultaneous reaching and grasping, the reach is visually initiated, and the grasp is also visually controlled. In phase 2, there is a differentiation between reaching and grasping, the initiation and guidance of reaching is visually controlled and the grasp becomes tactile controlled. It is interesting to see that different modalities are used in different phases, from visual control of grasp to tactile control.

As we have seen, the biological development ensures that a living being can survive (with some help from its progenitors) and mature. These observations suggested the developmental approach to robotics [2]–[4]. This developmental perspective aims at overcoming the complexity problem by learning and properly integrating many perceptual, motor, or cognitive skills incrementally and overtime.

The robot should "start" with a minimal subset of core capabilities (as newborns do) [15] to bootstrap learning mechanisms. Then, the system would progressively acquire new skills through self-experimentation, interaction with the environment and humans, and integrate all the learning methods internally. As proposed by [2], the main principles/requirements for a developmental machine can be summarized in seven points: 1) Environmental openness; 2) high-dimensional sensors; 3) completeness in using sensory information; 4) online processing; 5) incremental processing; 6) perform while learning; and 7) scale up to muddy tasks.

Development can be divided in three main axes: learning; structure; and complexity. Learning describes the most common mechanism, where a task solution improves with experience. When a newborn evolves from grasping an object with a ballistic motion to a visually controlled motion, we see that two behaviors exist and that one was built on top, or with information from, the other. This is a development in structure, where existing mechanisms elicit the development of new ones. In this iterative process, higher level mechanisms provide information for improving the lower level mechanisms. In this paper, this is the main form of development used, although each mechanism continues to learn and improve its efficiency with experience. The other axis of development is that of complexity, where the same mechanism improves its efficiency by means of a more complex controller or an increase in perception capabilities, e.g., resolution in vision or control. For example, the stereo-acuity of newborn increases until adult acuity is reached at around 24 months [16].

Some examples of robotic system using development in each axis are already present [2], [17], [18]. A developmental approach is used in [19] for a robot that successively acquire vergence, saccade, and vestibular control, as well as head–arm coordination. A system where a binocular head is controlled by a neural network whose input and output resolution is improved with time is presented in [20]. The work in [21] describes a robot that develops artificial emotions by interacting with people acting as caretakers. The approach takes advantage of the social interactions for constraining learning.

C. Our Approach

The development of imitation capabilities requires an appropriate definition of the sequence of learning steps to reach that goal, as well as adequate performance evaluation methods to decide when to switch to higher developmental levels. In other words, it is important to define the overall hierarchy of developmental stages and the skills that must be acquired at each level. Table I shows the structure we adopt for the main developmental stages that the robot goes through: 1) learning about the self; 2) learning about objects and the world; and 3) learning about others and imitation.

For each stage in this developmental pathway, we present the set of skills acquired by the system that are then available

TABLE I
DEVELOPMENTAL PATHWAY FOR THE PERCEPTUAL AND MOTOR CAPABILITIES (IN ITALIC, THE MODULES LEARNED BY THE ROBOT; IN BOLD, OUR MAIN CONTRIBUTIONS)

Time line	Perceptual/Motor Capabilities
sensory-motor coordination	eye vergence chaotic movements ***smm*** in redundant robots
world interaction	near-space mapping object affordances learning *uncalibrated visual control of grasp*
imitation	**task interpretation** **view-point transformation** detection of other's actions **imitation of goal directed actions** **imitation of gestures** **imitation metrics** **body correspondence**

for the next level. We do not claim any distinction between innate versus learned behavior in biological systems ("the nature versus nurture" question). Instead, we discuss all the modules necessary to be present before the system can develop to the next level. The sequence of learning stages is biologically inspired, but the specific division and implementation was a pragmatic option for having a real robotic implementation. Even for artificial systems, several mechanisms are almost the same across different levels. It is important to note that this division does not oblige the levels to be independent. Even when learning a module at a higher level, it is possible and desirable to continue to adapt lower level modules.

In the first developmental level, sensory–motor coordination, the robot acquires very simple and, yet, crucial capabilities: vergence control; object foveation; and perception–action coordination. By executing random arm movements in a self-exploratory mode, it begins to coordinate head and arm configurations by creating an arm–head map. This map is accurate enough to allow reaching for objects in easy positions. It also recognizes its own hand and is able to relate the image of the hand with the corresponding motor actions. Humanoid robots always have redundant degrees of freedom. Although this increases the number of solutions for the same problem, it makes more difficult to learn relations between variables because different action can give the same result. Special attention is given to this problem by providing algorithms that work under redundant conditions while exploiting all advantages of the multiple options of control.

In the second developmental stage, world interaction, the robot builds a map of the surrounding area (object positions and identification), studies objects, their properties, and how are they used by others. Driven by attentional cues, the robot engages in more challenging grasping tasks, for which the previously learned arm–head map is not sufficiently accurate. For that reason, a novel method for visually controlled grasping is presented, which improves over time and ensures the necessary robustness. Special care is taken about the redundancy present in these complex robotic systems. This grasping capability allows to recognize similar gestures performed by others.

At the final developmental stage, the presence of a demonstrator will elicit imitation behaviors. Human gestures will be imitated, by mimicking exactly the same motions, using the learned maps. Higher level tasks, i.e., interacting with objects, will be imitated by decomposing the observed actions and then replicate them in an abstracted way, i.e., if an object is grasped, we do not take into account the way it was grasped. For this purpose, the system must be able to decompose the observed action into the relevant key elementary actions that must be executed for performing a task.

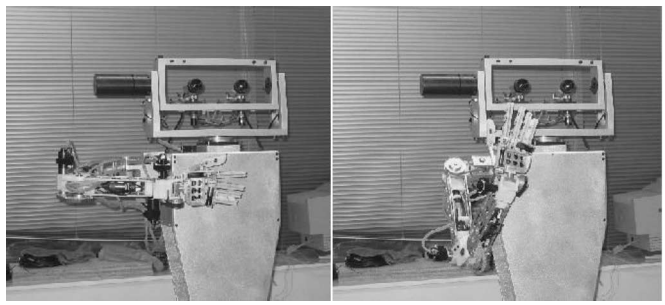

Fig. 1. Sensory–motor coordination learned by self-exploration. The redundancy of robots make it harder to associate perception to actions.

This roadmap [22] is implemented in the humanoid robot Baltazar, as shown in Fig. 1 and described in [23]. The remaining of this document presents each level of this developmental architecture, as presented in Table I. At each level, the main principles guiding development and the developed behaviors are presented. At the end, the conclusions and future work are presented.

D. Contributions

Our contributions are multifold. At a general level, we propose a developmental architecture for imitation, including the definition of the necessary skills at each stage, that allows to imitate gesture and goal-directed actions. We proposed and tested implementations of the various modules of this architecture, at the various levels, with a real robot. No special or intrusive hardware was used by the demonstrator, because the robot acquires all the relevant information through vision.

Specific contributions include methods for learning different types of SMMs with redundant robots and a methodology for abstracting a task description from observation. Different types of imitation behaviors that make use of appropriate imitation metrics for each situation were demonstrated.

II. SENSORY–MOTOR COORDINATION

The goal of sensory–motor coordination is to build associations or maps between perception and action. SMMs can be interpreted in terms of forward/inverse kinematics of robotic manipulators augmented with the sensor model/geometry. With SMMs, robots can predict the changes in the world that result from a given action (forward model), or which actions to take to change the world in a predefined manner (backward model).

The entire imitation architecture proposed in [24] is based on the extensive use of pairs of coupled SMMs. In their model, the basic structure is a forward–backward model capable of prediction and/or reconstruction. The backward model is not a simple inverse kinematics, but it includes a controller for solving a set of tasks. Their systems also deal with problem of limited computation by controlling the attention.

A variety of SMMs can be defined according to the used sensing/actuation modalities and structure of the input/output data. Motor commands can be joint torques, velocities, or positions. Sensor signals (percepts) can be shapes extracted from vision, sounds, or proprioception about the body-state (tactile) or motor actuation. Depending of the sensing modalities involved, we can refer to visuo–motor, auditory–motor, or motor–motor maps.

A different type of SMM must be used for different goals, such as: 1) predicting the image (or the image transformation) resulting from the robot moving the arm to a certain posture; 2) computing the motor command to drive the arm toward a specified appearance; or 3) calculating the head motion required to bring the hand to the camera's field of view.

SMMs can be determined analytically provided that accurate calibration information exists and that it remains fixed over time. The alternative approach that we adopt in this paper consists of estimating SMMs directly from sensory and motor data. First, (perception, action) pairs are collected by having the system operating and (automatic) observing the consequences of its own motor actions. Then, a learning method is used to estimate the model.

The "calibration" from automatic observation process follows the general developmental guidelines, as the system creates its own excitation actions which, in turn, allows it to gather enough information to coordinate its own body. With this approach, there is no need to assume a fixed prior model of the system before experimenting with the real robot. Since the very beginning of this learning process, the system is able to start solving tasks in a limited way. Then, as time goes by, solutions for certain tasks can be improved by exploiting the availability of more data. This process of learning by means of self-exploration is frequently used in this paper.

Humanoid robotic often have more degrees of freedom than those strictly necessary to accomplish a certain task. For example, Fig. 1 shows several positions of a humanoid robot, where the wrist position is always the same but the posture of the arm changes. In terms of SMMs, this redundancy translates into the fact that several different inputs yield the same output observation. As a consequence, backward (inverse) models are not well defined, since multiple solutions exist. Commonly used algorithms will thus fail to learn the inverse model, because the dataset is incoherent.

We propose an approach to learn inverse SMMs in redundant systems by partitioning task-relevant and task-redundant degrees of freedom. We avoid the usual strategy of "freezing" the redundant degrees of freedom. Instead, we can solve several tasks simultaneously or meet additional criteria.

In order to address all the problems described and noting that different skills need different strategies, we classified our sensory–motor-coordination algorithm in three types. The different types of SMMs according to the nature of the mapped information:

- Static versus Incremental: An SMM can describe as a static relation between the input and output, or it can relate input variations to output variations. The static version is useful for positioning (open-loop control), while the incremental map is necessary for closed-loop control.
- Full versus Partial: In a full map, we consider that the output completely determines the input, meaning that the task is nonredundant. If there is some degree of redundancy (DOR), either in the actuation or in the task itself, the number of admissible solutions will be infinite. In such a case, the map determines the input only partially and an extra optimization process is needed to identify a unique solution.
- Geometric versus Radiometric: In most cases, the SMMs we have discussed describe the geometry of observation and actuation. However, in some cases, we can consider radiometric maps that describe the visual appearance of an object (e.g., the hand) in addition to its coordinates in the field of view. Refer to [25] for an example.

In the rest of this section, we will focus on the problem of defining and estimating SMMs for redundant robots in two cases. The first case is a static visuo–motor map between the arm joints and the wrist position in the image. Then, we detail an incremental (differential) visuo–motor map used for servoing tasks as, for instance, when grasping an object.

A. Static Maps in Redundant Robots

In this section, we show how to define a SMM that explicitly takes the DOR into consideration, thus allowing the completion of several simultaneous tasks [26].

Let us define an SMM that maps a vector of control variables $(n,\ r)$ to a vector of image point features $\mathcal{I}$, where n is a minimum set of degrees of freedom that spans the full output space and r is a set of redundant degrees of freedom. Note that there are several admissible partitions of the input space in redundant/non-redundant degrees of freedom. It is also possible to find the redundancy automatically, by analyzing the correlation matrix for the Jacobian estimation [27]. The forward model that predicts the image configuration of the robot given a set of motor commands can be written as

$$\mathcal{I} = f(n, r).$$

We are often more interested in obtaining the inverse map and to compute the motor commands that drive the robot to a desired image configuration $\mathcal{I}$. If there were an inverse mapping $(n, r) = f^{-1}(\mathcal{I})$, this problem could be solved in a straightforward manner. However, as the dimension of the input space is larger than that of the output space, many input combinations generate the same image-point features and $f(n, r)$ cannot be inverted.

To put the problem in another perspective, finding the robot joint angles that move the arm to a desired image configuration $\mathcal{I}$ becomes an ill-posed problem when the arm has redundant degrees of freedom [28], because multiple solutions exist.

One approach to solve ill-posed problems [29], [30] consists in using additional constraints that restrain the set of admissible solutions to guarantee a unique solution. In our case, this corresponds to recast the original problem to that of moving the robot to a desired image position $\mathcal{I}^*$ while minimizing some auxiliary criterion $c(n, r)$.

We build a cost function $\mathcal{K}$ with two terms: one weighting the error in the position of the end effector (data fitness) and another one corresponding to the weights on the control (regularization term)

$$\mathcal{K}(\mathcal{I}^*, n, r) = \lambda \|\mathcal{I} - \mathcal{I}^*\|^2 + c(n, r). \tag{1}$$

This cost function expresses our willingness to accept some error in the position, if another task can be solved at the same time, e.g., involving the control costs. Examples of control cost criteria c can be "Comfort" (e.g., distance to joint limits), "Energy minimization" (e.g., the position with lower momentum), or "Minimum motion" (i.e., minimum total motion from current to desired position, posture control, among others). The regularized solution can be found by minimizing the cost defined in (1), as follows:

$$(\hat{n}, \hat{r}) = \arg\min_{n,r} \left(\lambda \|\mathcal{I} - \mathcal{I}^*\|^2 + c(n, r)\right) \tag{2}$$

where $\mathcal{I}$ can be computed with the forward model $\mathcal{I} = f(n, r)$. Similar to [31], this formula integrates two terms: One describing the task and another the posture.

There are two important observations to this formulation. First, the optimization is done with respect to all degrees of freedom, which translates into a significant computational cost. Second, the DORs are not treated as such, since they undergo exactly the same process as the nonredundant degrees of freedom.

The consequence of this approach is that the extra degrees of freedom are frozen from the beginning and can no longer be used for a different purpose during execution. In a way, redundancy is lost. Instead, our approach keeps the redundant degrees of freedom available for solving additional tasks online. In essence, we split the problem in two steps. First, we define a "Minimal Order Sensory–Motor Map" $g(\mathcal{I}, r)$ that relates n and $(\mathcal{I}, r)$

$$n = g(\mathcal{I}, r). \tag{3}$$

By taking the DORs as input (independent variables) instead of the output signals, the problem of computing the nonredundant degrees of freedom is now well posed. The DORs r are left unconstrained and can be fixed during runtime, when a secondary task or optimization criterion is specified. Second, the DORs are determined as the solution of a new optimization problem with cost function $\mathcal{L}$

$$\hat{r} = \arg\min_{r} \mathcal{L}(\mathcal{I}^*, r) \tag{4}$$

with the optimization done with a gradient-descent method

$$r_{t+1} = r_t - \alpha \nabla_r l(\mathcal{I}^*, r).$$

In contrast with the previous case (1), this optimization (4) is done with respect to the DORs only. The complexity is thus substantially lower and lends itself to be used as an online process. In general, the solutions in the two cases are not the same, because different local minima could be reached, and the criteria are slightly different. In the first case, both criteria are optimized simultaneously while, in the second case, the posture is optimized after the task criteria.

In a perfectly calibrated setting, this approach guarantees zero-prediction error, because the minimum-order SMM allows us to determine the values of n corresponding to the exact image position for the selected redundant degrees of freedom. This solution tends to the first (regularized) problem when λ becomes large. If the minimum-order SMM is not exact, then it will introduce some error in the final image configuration.

For clarity, we summarize the final algorithm.

Step 1) Select the desired image configuration $\mathcal{I}^*$.
Step 2) Select an initial value for the DORs r and compute n using (3).
Step 3) Select the secondary task-optimization criterion.
Step 4) Solve the optimization of (4) for r and use $g(\cdot)$ to compute n.
Step 5) Move the arm to the obtained solution (n, r).
Step 6) Observe $\mathcal{I}$ and possibly adjust the function $g(\mathcal{I}, n)$.
Step 7) If extra precision is needed, go to Step 4).

There are several important differences in our approach when compared to other methods based on visual servoing. The minimum-order SMM in (3) can be used to determine the final values of the robot joints that correspond to the desired configuration. Then, the introduction of a secondary task-criterion (e.g., comfort, energy) leads to an optimization problem as per (4). The solutions to this optimization process are then used to drive the robot, without requiring any visual feedback. Only if extra precision is needed should visual feedback be used.

In the following example, the secondary goal consists of a comfort criterion, where we would like to keep the joint angles as close as possible to their central position (maximizing the distance to joint limits):

$$\begin{aligned}\mathcal{L}(\mathcal{I}^*, r) &= \|n - n_c\|^2 + \|r - r_c\|^2 \\ &= \|g(\mathcal{I}^*, r) - n_c\|^2 + \|r - r_c\|^2\end{aligned} \tag{5}$$

where r_c and n_c stand for the central positions of the corresponding articular joints. Differentiating this cost function yields

$$\nabla_r \mathcal{L}(\mathcal{I}^*, r) = 2\left(\frac{\partial g(\mathcal{I}^*, r)}{\partial r}\left(g(\mathcal{I}^*, r) - n_c\right) + (r - r_c)\right).$$

We have seen how to partition the redundant and nonredundant degrees of freedom to build a minimum-order SMM, $g(\mathcal{I}, r)$ that allows for the computation of the nonredundant degrees of freedom leaving the DORs unconstrained. To learn this map, we use the locally weighted projection regression method [32]. This method is linear with the number of samples, and every new sample can be added easily. As the method is not capable of extrapolating, the workspace must be well covered in the training set.

We have conducted experiments to assess the quality of the proposed method. We first performed arm movements during which the head tracked the robot hand to estimate the head–arm SMM with the previous algorithm. The head position corresponds to pan, tilt, and eye vergence while four DOFs were considered for the arm. Tests were made by considering

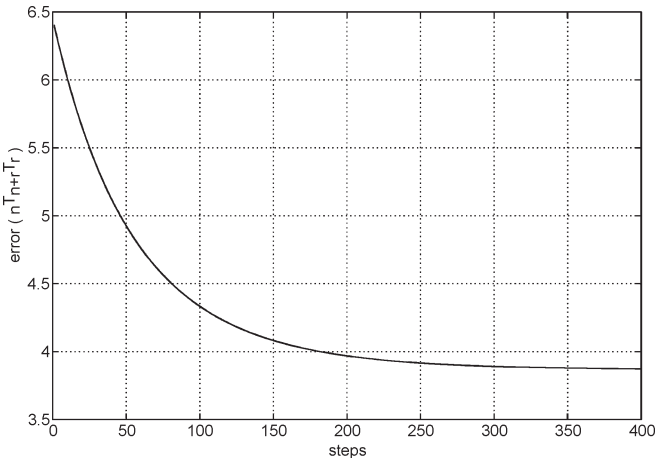

Fig. 2. Convergence rate as a function of the optimization step. The final error is in the order of magnitude of 0.03 rad for a motion range of 0.5 rad.

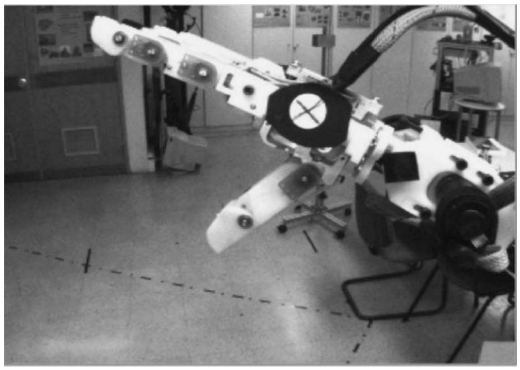

Fig. 3. Robot view of its own arm, hand, and the target being tracked.

a target hand position in the image plane, while comfort was chosen as secondary criterion. The quality can be judged by the distance between the final position and the goal, as well as by the gain in the comfort criterion. It is worth stressing that the optimization process relies on the estimated minimum-order SMM, as described before. Fig. 2 presents the evolution of the cost function $\mathcal{L}$ for each iteration of Step 4) of the method. For this case, the maximum motion amplitude for one joint was 0.5 rad. The final error in the image was ≈0.03 rad, mainly due to elasticity in the robot joints and the approximation errors of the map. Fig. 3 shows the robot's view of the hand. Due to the redundancy in the arm, it is possible to fixate the target while changing the arm posture.

The main conclusion is that the map quality is good enough to guarantee that the hand is always in the image, although not necessarily in the fovea. Thus, it enables the system to reach and, in special cases, grasp objects. The acquisition of this skill provides the required motivation for object grasping to develop in subsequent developmental phases.

B. Dynamic Maps in Redundant Robots

The previous map allows a robot to move the arm to a desired position, without considering the trajectory followed to reach that position and without visual feedback. A visual-feedback loop is necessary, if the final position is not reached with enough precision, or if the goal is to follow a trajectory and not a position. For this, we present an incremental SMM in the context of visual-servoing tasks [33].

One could obtain an incremental SMM directly by differentiating the static maps described in the previous section. This process is too sensitive to noise due to the function-approximation method used to estimate the map. Alternatively, we could repeat the procedure followed in the previous section while using incremental motor and visual data as the SMM input–outputs. However, the time necessary to explore all the input space and provide a good representation is way too large, and the number of parameters to estimate is prohibitive.

Here, we follow a different approach to improve the convergence time and facilitate the use of these maps in closed loop control. We approximate the maps by (locally) linear functions, which can be directly used in visual-servoing tasks. In this control method, we relate image features velocities $\Delta\mathbf{y}$ with motor velocities $\Delta\theta$ by the following relation $\Delta\mathbf{y} = J(\theta)\Delta\theta$, where J is the robot Jacobian.

In this setting, it is possible to consider the redundancy of the manipulator explicitly. The implementation of this incremental and partial visuo–motor maps can be made by resorting to the redundancy formalism [34]. The idea is to decompose a complex task as a sequence of redundant subtasks such that each new subtask does not disturb the previous ones [35]. Using this formalism, a control law is computed to keep a given priority or order of sequencing of the various subtasks. This control law can be implemented for various kinds of closed-loop control, provided that the objective can be written as a task function [34]. Under this formalism, the redundancy is exploited by having several tasks simultaneously performed the main task and subtasks that can define posture, obstacle avoidance, or others.

1) Redundancy Formalism for the Two Tasks: Let θ be the articular vector of the robot. Let $\mathbf{e_1}$ and $\mathbf{e_2}$ be the two tasks, with Jacobian $\mathbf{J_i} = (\partial\mathbf{e_i})/(\partial\theta)(i = 1, 2)$ defined by

$$\dot{\mathbf{e}}_\mathbf{i} = \frac{\partial\dot{\mathbf{e}}_\mathbf{i}}{\partial\theta}\dot{\theta} = \mathbf{J_i}\dot{\theta}. \tag{6}$$

To control the robot with the articular velocity $\dot{\theta}$, (6) has to be inverted. The general solution (with $i = 1$) is

$$\dot{\theta} = \mathbf{J_1^+}\dot{\mathbf{e}}_\mathbf{1} + \mathbf{P_1}\mathbf{z} \tag{7}$$

where $\mathbf{P_1}$ is the orthogonal projection operator on the null space of $\mathbf{J_1}$ and $\mathbf{J_1^+}$ is the pseudoinverse (or least square inverse) of $\mathbf{J_1}$. Vector $\mathbf{z}$ can be used to apply a secondary command that will not disturb $\mathbf{e_1}$. Here, $\mathbf{z}$ is used to fulfill the task $\mathbf{e_2}$. Introducing (7) in (6) (with $i = 2$) gives

$$\dot{\mathbf{e}}_\mathbf{2} = \mathbf{J_2}\mathbf{J_1^+}\dot{\mathbf{e}}_\mathbf{1} + \mathbf{J_2}\mathbf{P_1}\mathbf{z}. \tag{8}$$

By inverting this last equation and introducing the computed $\mathbf{z}$ in (7), we finally get

$$\dot{\theta} = \mathbf{J_1^+}\dot{\mathbf{e}}_\mathbf{1} + \mathbf{P_1}(\mathbf{J_2}\mathbf{P_1})^+(\dot{\mathbf{e}}_\mathbf{2} - \mathbf{J_2}\mathbf{J_1^+}\dot{\mathbf{e}}_\mathbf{1}). \tag{9}$$

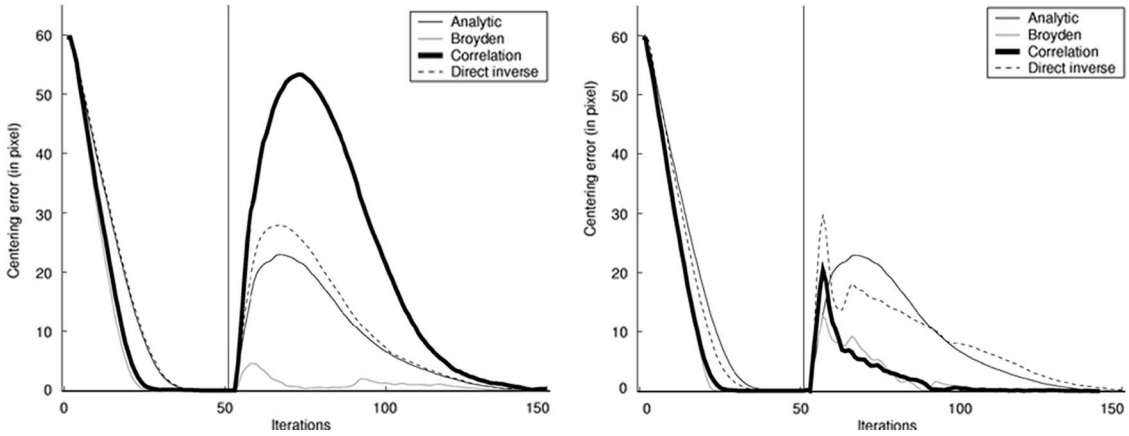

Fig. 4. Temporal evolution of the image error during servoing using (left) offline or (right) online learning methods. The vertical line shows the time instant, where the second task started.

Since $\mathbf{P_1}$ is Hermitian and idempotent (it is a projection operator), (9) can be written

$$\dot{\theta} = \mathbf{J_1^+}\dot{\mathbf{e}}_\mathbf{1} + \widetilde{\mathbf{J_2}}^{+}\widetilde{\dot{\mathbf{e}}_\mathbf{2}} \tag{10}$$

where $\widetilde{\mathbf{J_2}} = \mathbf{J_2P_1}$ is the partial Jacobian of the task $\mathbf{e_2}$, giving the available range for the secondary task to be performed without affecting the first task, and $\widetilde{\dot{\mathbf{e}}_\mathbf{2}} = \dot{\mathbf{e}}_\mathbf{2} - \mathbf{J_2J_1^+}\dot{\mathbf{e}}_\mathbf{1}$ is the secondary task function after subtracting the part $\mathbf{J_2J_1^+}\dot{\mathbf{e}}_\mathbf{1}$ already accomplished by the first task. A very good intuitive explanation of this equation is given in [36].

2) Learning: Although it is possible to evaluate J analytically, we adopted a modeless approach as it allows the system to learn and develop from its own experience. A particularly useful method for online estimation of visual–motor relations is based on the Broyden update rule, well-known from optimization theory [37] and used in real robotic applications with visual control [38], [39]. The image Jacobian is estimated iteratively

$$\hat{J}(t+1) = \hat{J}(t) + \alpha \frac{\left(\Delta\mathbf{y} - \hat{J}(t)\Delta\theta\right)\Delta\theta^T}{\Delta\theta^T\Delta\theta}$$

where $\alpha \in [0,1]$ denotes the Jacobian update rate. To move the system to the desired image position y^*, we apply the following control law:

$$\mathbf{\Delta}\theta = h\left(J^+(\mathbf{y}^* - \mathbf{y})\right)$$

where J^+ represents the pseudoinverse of J and the function $h(\cdot)$ can be chosen to ensure an exponential, linear, or any other type of convergence. It is important to notice that the influence of the estimation process is twofold. On the one hand, errors in the Jacobian estimation will influence the computation of the control law, but visual servoing is known to be robust to such errors. On the other hand, these estimation errors will also impact on the delicate procedure of task decomposition and computation of the projection operators [40].

3) Experiments: In the first experiment, two tasks are considered: centering $(\mathbf{e_g})$ the hand in the image and rotation $(\mathbf{e}_\alpha)$ in the image. Our goal was to test the influence of the Jacobian estimation errors on the task-sequencing approach due to errors introduced in the projection operators. When the Jacobian of the first task is misestimated, the centering task is lost with the activation of the second task. When the error increases, the target moves further away from the image center and possibly leaving the image, if the disturbance is too strong (which results of course in the visual-servoing failure).

Fig. 4 presents the evolution of the error for the first task, comparing offline and online methods along with an analytical estimate of the Jacobian. The vertical line represents the time instant, where the second task was introduced. We can see that because the Jacobian estimation is not perfect the first task is perturbed, i.e., its error is not maintained at zero. Offline learning relies on simple motions of the arm, lasting for approximately 250 iterations. Online learning is carried out at every frame. As for the estimation methods, we compare the proposed Broyden update rule with a standard least square estimate of the Jacobian (correlation method) or its inverse (direct-inverse method).

The first result shows that analytic or offline learning are worse, in terms of perturbation rejection and convergence times. This can be explained by the uncertainty in the parameters used for analytical computations and by the linear approximation of a nonlinear process in the case of the offline method (the linearization is done in a point different from the actual execution). Instead, online-estimation methods lead to much better results, outperforming the results with the analytical Jacobian. Although a large disturbance appears when the second task is added, it is quickly reduced afterward.

The amplitude of the perturbation ranged from 20 to 30 pixels. Broyden and correlation methods were able to eliminate the error after 30 iterations. The maximal perturbation is equivalent to the one obtained with analytic computation, but the duration is much shorter. The task-error convergence is very similar for all methods (it occurs before iteration 50 for task $\mathbf{e_g}$; see Fig. 4). This emphasizes that the reduction of the perturbation is not made at the cost of convergence. The convergence is very robust to Jacobian estimation errors, since the task convergence rates are the same. It is nevertheless not true for the projection-operator estimation, which is very sensitive and requires an accurate estimation.

All online-learning methods succeeded to solve the task. From several experiments, starting from different initial

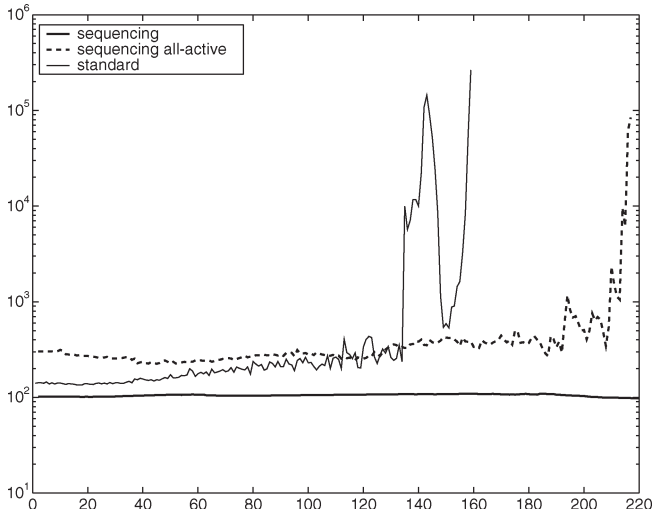

Fig. 5. Condition-number evolution of the estimated interaction matrix during the servo. The matrix is learned from three different trajectories: (a) sequencing as done above, (b) sequencing formalism with all tasks activated simultaneously at the first iteration, and (c) classical visual servoing using a 6-DOF task composed of all the visual features. The matrix learned from a classical servo has a very large condition number. It increases until the servo becomes impossible. The learning under the sequencing procedure provides a properly conditioned matrix.

positions and using different tasks, the correlation method produced better results in sense of perturbation amplitude, perturbation average, and perturbation-correction time when properly tuned. However, it is not as robust to gain tuning as the Broyden approach that could solve the task in all situations with the same parameters settings (note the Broyden performances for offline learning).

A very important point is to note that learning improves the sequencing quality by reducing convergence times and the amplitude of the perturbations. At the same time, the sequencing generates more efficient trajectories for learning. This experiment tests this hypothesis by comparing learning four simpler tasks in sequence against learning four tasks at the same time.

We compared the learning when running the robot under three different control laws. During the first run, task sequencing was used in the same way as in previous experiments. In the second trial, all tasks are active at the same time. In other words, the same formalism is used, but every task is active from the beginning as opposed to starting a new task only after all the previous ones are completed. The last trial consisted on classical visual servoing, using only one single task of full rank. The conditioning number of the full-rank Jacobian matrix was then estimated at each iteration. When a sequencing was used, the Jacobians of all tasks were piled up and the overall conditioning number evaluated.

Fig. 5 shows that for the Broyden method, the condition number of the matrices are much worse for the full task and convergence cannot be attained. This method is thus very reliable for learning partial and incremental maps.

At the end of this stage, the robot has learned how to predict what happens in the visual field when it acts in a particular way. It also learned which action creates a desired perceptual change (the inverse map). Having mastered the control of its own body, the system is going to deal with entities in the world during the next developmental stage.

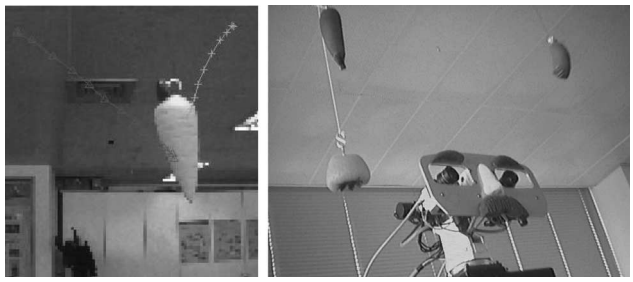

Fig. 6. World-level behaviors. At this level, the system interacts with objects in the world by learning about their properties and their locations to finally grasp them. (Left) Verging on an object. (Right) Mapping object positions in head coordinates.

III. World Interaction

As the robot gains control over its own perceptual and motor capabilities, it gets more and more interested in exploring the surrounding world. This exploratory motivation calls for the development of manipulative capabilities.

Object grasping requires the use of several motor programs: Detect the target position; approach the object (reaching); correct eventual errors with visual feedback; and finally, grasp it. This capability, by allowing interaction with objects, enables the system to learn about object's physical properties but also their affordances. This knowledge can also be used to recognize similar gestures performed by others. At this stage, all the robot can do is to fixate at salient objects and approach them. Saliency is hand-coded, and objects are detected by color segmentation. The developmental path requires the acquisition of the following new skills.

1) near-space mapping;
2) learn to grasp objects;
3) learn object affordances.

This section describes the approach to acquire this new behavior by making use of the previously learned SMMs. These new capabilities permit the robot to move on to the next developmental level, where it gains awareness of others (humans or robots) and the actions they perform.

A. Near-Space (Objects) Mapping

There is neurological evidence of spatial-aware neurons that are activated by motion or objects near the skin [41]. It is also known in developmental psychology that infants became aware of the near and far space very early [42]. The near space contains the touchable objects and the robot's own body.

The head can be moved to look toward the hand using disparity as a feedback signal to control it. Fig. 6 shows Baltazar verging on an object. By this exploratory behavior, we create a map of the localization of objects around the robot—the peripersonal map—through various steps.

1) Find an object in the visual space.
2) Foveate on this object.
3) Memorize the object position in body-centered (proprioceptive) coordinates.

By gazing at an object, the 3-D position of the object becomes defined in proprioceptive coordinates: two angles with the neck position and the distance with the eye vergence angle. Through exploration, the robot thus creates a mental image of the surrounding space. The positions of objects are memorized in terms of proprioceptive coordinates. Fig. 6 presents Baltazar searching and mapping "fruits" around him.

B. Object Grasping—Multistep Approach

Infants start reaching for objects without any visual feedback. The movement is only initiated with vision but not guided throughout the entire action. In case of failure, the movement restarts from the beginning.

At the first stage of development, the estimated arm–head map allows the system to (crudely) move the hand toward an object. Hence, a simple trajectory may put the hand in contact with the object. The problem with this (open loop) approach is the absence of a mechanism for error correction.

The second stage of object reaching relies on visual feedback, coping with the problem of error correction. The static head–arm map is used to move the hand to the object's vicinity. Then, accurate positioning is achieved by visual guidance using the incremental head–arm map. With this phase, it is possible to grasp objects in a reflex-type manner; the hand closing after the touch. The missing capability of visual closed-loop control can be the reason why babies in this phase restart the grasp when it fails instead of correcting it [12]. Our grasping mechanism can be summarized as follows.

1) Move the head in order to have the eyes gazing at the object.
2) Use the head–arm map to move the arm into the image and as close as possible to the object. This phase uses the minimum-order head–arm map from Section II-A.
3) As soon has the hand is detected in the image, start the visual closed loop toward the object. This phase uses the incremental and partial map from Section II-B.
4) Close the hand upon contact with the object.

We made several experiments to access the quality of the resulting algorithm. Our system measures a specific dot in the hand with two cameras giving an image position of the hand, (u_l, v_l) for the left eye and (u_r, v_r) for the right eye. The features are calculated as follows:

$$\mathbf{y} = \begin{bmatrix} \dfrac{u_l + u_r}{2} & \dfrac{v_l + v_r}{2} & u_l - u_r \end{bmatrix}^{\mathrm{T}}.$$

This gives the position and distance information estimation of the hand related to the head. The head was maintained fixed and four arm joints were used. The Jacobian update rate was chosen as $\alpha = 0.1$.

After moving the head, the hand was positioned near the object using the head–arm map. The resulting error corresponds to about 8 cm. The associated image error is corrected in the final phase (visually controlled). Fig. 7 shows the convergence of the grasp sequence shown in Fig. 8 using our proposed algorithm. We can see an almost linear convergence. The use of the open-loop motion made this possible because it moves the hand near the target.

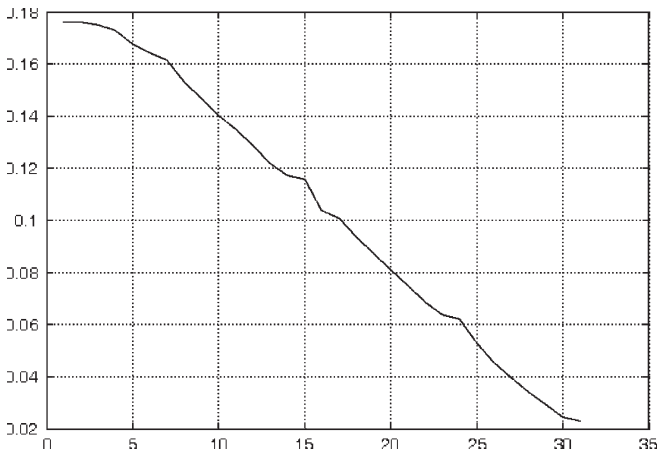

Fig. 7. Convergence of the servoing algorithm for object grasping, plotter as error versus sample.

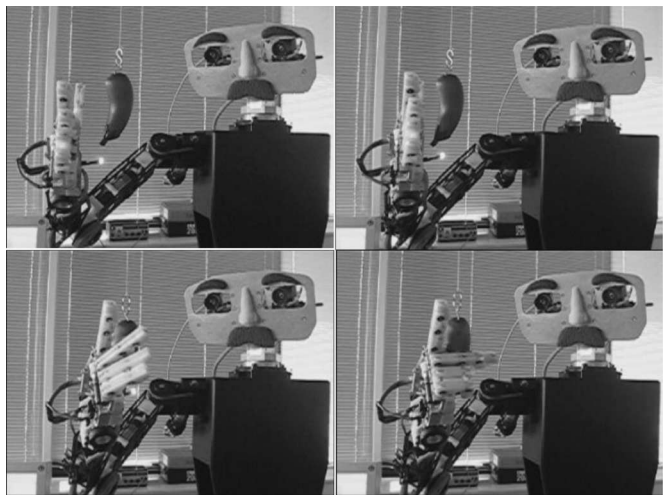

Fig. 8. Several frames in the grasping sequence from the initial position resulting from the head–arm map, the visual-guided part, and finally, the object grasping.

C. Object Affordances

In order to interact with the world, it is necessary to have some knowledge about it. Physical entities have different uses: Some are graspable, some can be combined, some can be eaten, and there are others that move by themselves. Learning about properties of objects is done by observing the way they are acted upon by others, giving information about their affordances [43].

This understanding of the world is becoming more and more important as robots are expected to interact with people in a home setting. The robot can look around and start to learn the identities of things and their properties.

In our architecture, the observation of objects being grasped is useful in two ways: it suggests how to grasp them and gives possible uses for them. Therefore, it is important to recognize grasping action to learn about objects and to interact with them. Noteworthily, neuroscience suggests that the ability of

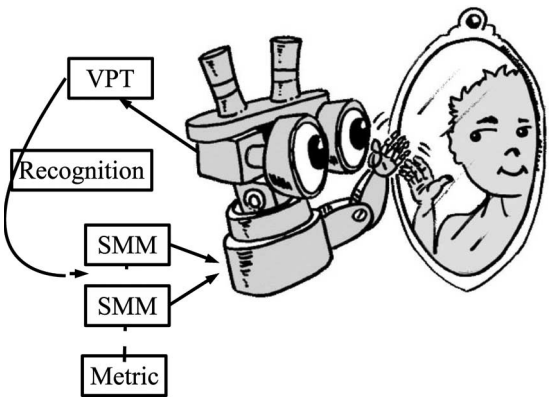

Fig. 9. Imitation architecture. Observed actions are first transformed to an ego frame of reference (VPT), where segmentation and recognition are made. After that, an imitation metric and body correspondence are chosen (by selecting the corresponding SMM). In the end, the imitation is performed.

recognizing someone's gestures is facilitated by the fact that the system knows how to perform those same gestures. Further details of an implementation based on a model for mirror neurons is presented in [25].

IV. Imitation

We present the final development stage, where the system looks at people in the environment to learn by imitation. The imitation process consists of the following steps: 1) observating the demonstrator's actions; 2) view-point transformation (VPT) of the description in the demonstrator's frame alloimage to the imitator's frame egoimage; 3) recognition of observed actions to abstract the observed motion (if necessary); and 4) an SMM generates the adequate actions. A metric is chosen to select among the imitation behaviors, VPTs and SMMs. Fig. 9 presents this process.

A. View-Point Transformation

Understanding events and object's localizations at far distances (i.e., more than the arm can reach) is different from mapping the surrounding space. The frame of reference is no longer one's own body. Instead, a description of the object's positions is made by referencing to another person or environmental cues. Object's position should be coded in terms of allocoordinates, and for this, it is necessary to transform a description from allocoordinates to egocoordinates by means of a VPT.

VPT involves scene understanding and reconstruction. This reconstruction can be very coarse or purely 2-D (2-D VPT). Alternatively, if depth information is required, a 3-D transformation is considered (3-D VPT). A VPT, describing a rigid transformation that aligns the allocentric and egocentric image features, can be written as

$$I_e = \mathcal{P}T\,\mathrm{Rec}(I_a) = \mathrm{VPT}(I_a)$$

where $\mathcal{P}$ is the camera-projection matrix, T is a rigid transformation, and $\mathrm{Rec}(I_a)$ stands for the reconstruction of the demonstrator posture from allocentric-image features. The properties of the reconstruction, transformation, and projection give different properties to the VPT. Extra details of these processes are presented in [25].

B. Imitation Metrics

In this section, we present the metrics used to evaluate and guide imitation. Two different sets of metrics are presented for the cases of action-level and program-level imitation (in the context of an object manipulation task).

1) Action-Level Imitation: Gestures are a very important mean of communication. They are used to wave someone goodbye or to make some warnings like: you are out of time and everything is fine. Although the gesture itself can be produced in a variety of different ways, the meaning is almost always unambiguous and recognition or understanding will be relatively easy. When waving goodbye, the speed or the exact distance between the hand and the head are not critical.

The choice of the metric and the VPT are extremely intertwined. If a metric is defined in 3-D terms, it is not possible to use a VPT that expresses a partial transformation (e.g., 2-D) only. Therefore, in the general imitation architecture, the metric is the first thing to be defined. Then all the rest follows. The following equation gives a general metric used for action-level imitation:

$$\mathrm{im} = \int \left(\mathrm{VPT}(\mathcal{I}_a) - \mathcal{I}_e^{\mathrm{self}}\right) dt$$

where $\mathcal{I}_a$ denotes the image of the demonstrator seen by the imitator (alloimage) and $\mathcal{I}_e^{\mathrm{self}}$ represents the image of the imitator's body as seen by itself (egoimage). Clearly, different properties of the VPT give different imitation behaviors.

With these metrics, the imitator is required to move its body in order to match the position of the demonstrator, as closely as possible. The great advantage of the VPT becomes now very clear. Because of the egorepresentation of the gestures, all the sensory–motor-coordination mechanisms learned in the first development stage can now be used. To imitate according to a given metric, the body is moved through the selection of the appropriate SMM and giving as control reference $\mathcal{I}^* = \mathrm{VPT}(\mathcal{I}_a)$.

2) Program-Level Imitation: A different type of task involves acting on objects, like placing dishes on a table or storing books in a shelf. The key issue in these tasks does not reside on pure gesture imitation, the most important part is the final state (or task goal). The way in which the task is solved, i.e., the posture and the speed is not so relevant. This calls for different metrics than the ones we have seen before. The actions and movements of the demonstrator must be segmented and coded in a way that is meaningful for imitating the task goals and subgoals.

We developed a method consisting in a multiple-object tracking and an action detector. In manipulation tasks, the hand often occludes objects. Grasping and releasing can be very difficult to detect. The fact that the hand is the only active element in the scene provides some implicit information that will help in dealing with the occlusions. We assume that every object can have two movement models: "rest" and "moving." When an object is being moved, it has the same velocity as the hand.

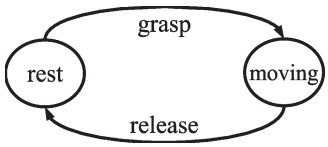

Fig. 10. Object-state transitions used in task segmentation.

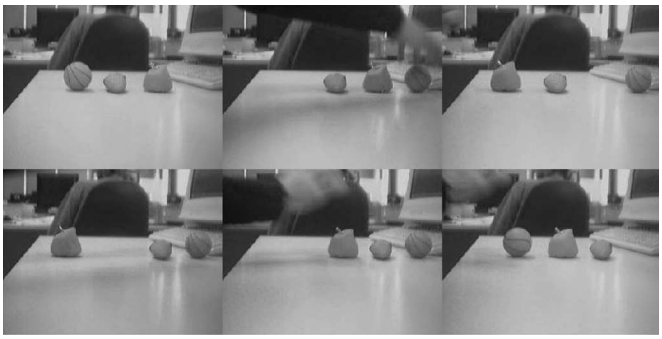

Fig. 11. Task Segmentation. Notice that from the third to the fourth image, there is no difference in the ordering of the object, just their absolute distances. These relevant points were extracted online from a video sequence with 234 frames.

Object grasping is detected in two situations: 1) when it starts to move and 2) when it is occluded by the hand. Detecting object releasing is done by detecting a previously grasped object becoming static while the hand moves away. Using these hypotheses, our algorithm will mark every grasping/releasing point in the trajectories of the objects. Fig. 10 gives a finite-state machine that controls the detection of object's state. The process of task segmentation is illustrated in Fig. 11. If the grasping type is important, the grasping classification method presented in Section III-B could be used.

The task is then codified in a sequence of world states, the transitions between states occur by grasping, or releasing a given object. Each state describes the objects spatial relations (A between B and C, A right of B, or A left of B) and metric positions.

Although this approach cannot be seen as a general framework for goal-directed program-level imitation, it is noteworthy to mention that the goals and subgoals of certain tasks can be abstracted in this way. As a consequence, a rich imitation behavior is achieved following the proposed developmental roadmap. The summary of the algorithm is summarized below.

1) Detect and localize objects around the demonstrator and apply the VPT to map those objects in the observer's coordinate frame.
2) Observe the sequence of task execution.
3) Segment the sequence by detecting interesting points (changes in the tracker state) in time and space.
4) Make a description of the task as a chain of meaningful events such as grasp and release objects.
5) Perform the same task.

V. Experiments

We have implemented the modules discussed in the previous sections to build a system that is able to learn by imitation. We start by describing the approach used for hand tracking before presenting the overall results on imitation, both for the action- and program-level imitation.

Fig. 12. Vision system. (Left) Original image. (Right) Background segmentation of the human figure and hand detection. The rectangular frame is used for the template matching.

A. Vision

1) Vision System: To model the arm position of the demonstrator, we have three steps of segmentation for the background, person, and hand.

During initialization, the background is estimated by modeling the intensity of each pixel as a Gaussian random variable. We need about 100 frames to obtain a good model. After this process, we can estimate the probability of each pixel belonging to the background. In order to increase the robustness of segmentation to illumination variations, we use red-green-blue color representation normalized by the blue channel.

The model of the background is used to determine the areas in the image, where motion has been observed. After detection, the position of the person is estimated by template matching and correlation. The template consists of a rectangle for the body on top of which a second rectangle represents the head. The body–head proportions used were those corresponding to a *fronto*-parallel person at a nominal distance from the cameras. By scaling the template, we can estimate the size of the person and the scale parameter s of the camera model. In addition, if we need to detect if the person is rotated with respect to the camera, we can scale the template independently in each direction, and estimate this rotation by the ratio between the head height and shoulder width. To find the hand in the image, we use a color-segmentation scheme, implemented by a feed-forward neural network with three neurons in the hidden layer. As inputs, we use the hue and saturation channels of hue-saturation-value (HSV) color representation. The training data are obtained by selecting the hand and the background in a sample image. After color classification, a majority morphological operator is used. The hand is identified as the largest blob found, and its position is estimated over time with a Kalman filter. Fig. 12 shows the result of this process.

B. Action-Level Imitation

The first imitation experiments deal with action-level imitation. Here, gestures made by a person should be repeated by the robot. Using the generic architecture of Fig. 9, the robot observes the scene using the person/hand tracking system presented earlier. After choosing the metric, the robot applies the correct VPT, and then, the previously learned SMM gives directly the necessary motor commands.

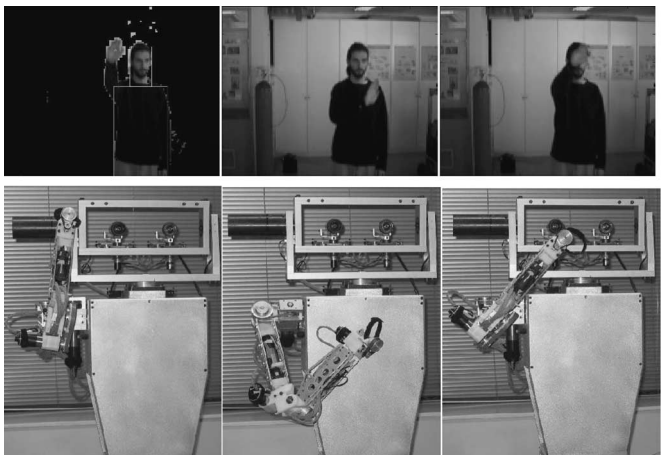

Fig. 13. Robot imitating the hand movements made by a demonstrator. A partial SMM is used in such a way that the elbow position is left unconstrained.

If we assume that the hand movement is constrained to a plane or that the depth changes are small, we can use a VPT that does not take depth information into account to estimate the position of the person. Here, the system succeeds in imitating the hand gesture, but as expected due to the properties of the VPT used, there are differences in the configuration of the elbow, particularly at more extreme positions. Fig. 13 shows the system imitating a tutor in real-time.

C. Program-Level Imitation

The goal of the imitation task illustrated here consists on moving a set of objects, as shown by a demonstrator. It follows the imitation system presented in Section IV-B2.

All the modules developed until this point are essential to replicate the task at hand. Although the way we describe this particular set of tasks could be replaced by possibly more sophisticated processes, the modules would still remain as valid building blocks to perform such a new set of tasks.

Fig. 14 shows an example of the execution of a task, consisting of grasping a set objects and moving them around. To imitate this task, the robot first needs to understand the spatial relations of objects in the vicinity the demonstrator (understand the far space). Then, understanding the near space becomes fundamental to establish correspondence between the demonstrator perspective and its own (self) viewpoint (i.e., the blue object located on the left-hand side of the demonstrator is in front of me). After the observation of the demonstrator's movements, the important task moments must be extracted and temporally segmented. Finally, the learned task is repeated by the robot (Fig. 15), using the imitation architecture and the proposed developmental pathway. The robot places the objects in the same order as the demonstrator. In the final step, the robot assumes that the task subgoal consists in changing the absolute position of one object, since the demonstrator did not affect the objects relative spatial relations. The task interpretation and execution is the following.

1) By moving the head, detect objects A and B (on the right and on the left of the demonstrator);
2) foveate on object A, grasp object A;

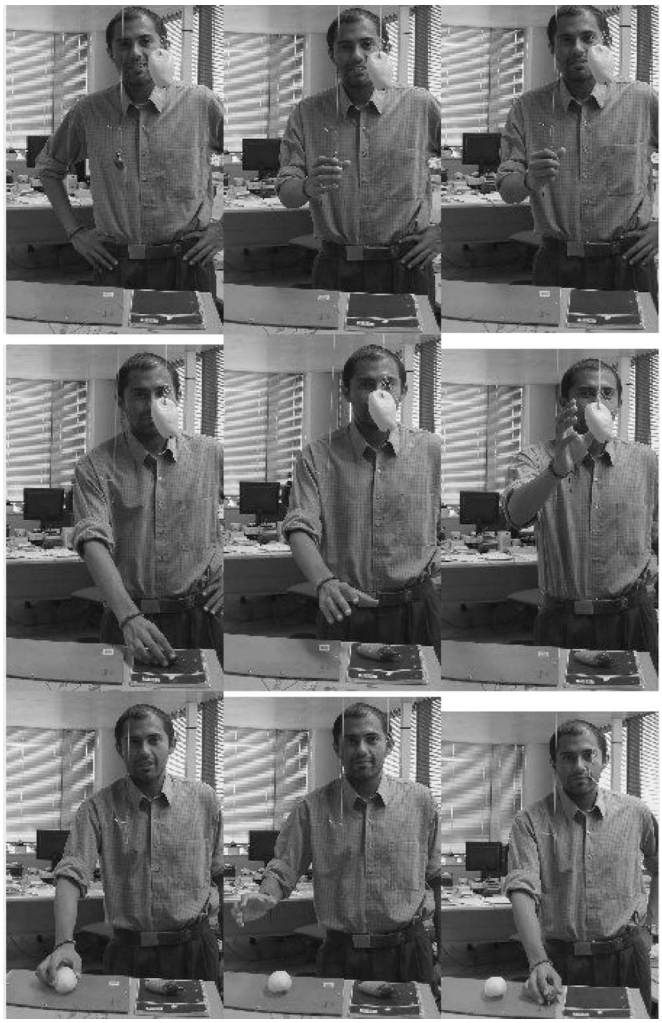

Fig. 14. Several frames of the task demonstration. The person is moving objects from position to position.

3) foveate on position 0, release object A;
4) foveate on object B, grasp object B;
5) foveate on position 1, release object B;
6) foveate on object A, grasp object A;
7) foveate on position 2, release object A.

Note that all positions are restricted to a vertical plane. First, the head foveates the objects of interest. This step facilitates the control, because the target is in the center of the image, but it is also a necessity due to the limited field of view of the robotic head. Then, the grasp action is elicited to finally grasp the objects. To grasp (or release) an object, a static head–arm SMM is used for the initial reaching phase. Then, a visual-servoing loop is used for the final phase of the grasp. Upon contact, the hand closes.

VI. Conclusion/Future Work

We presented a developmental route for creating an humanoid robot[1] that is able to learn by imitation. This route allows the robot to acquire increasingly more sophisticated skills

[1]See http://vislab.isr.ist.utl.pt/baltazar for videos showing some of the experiments in this paper.

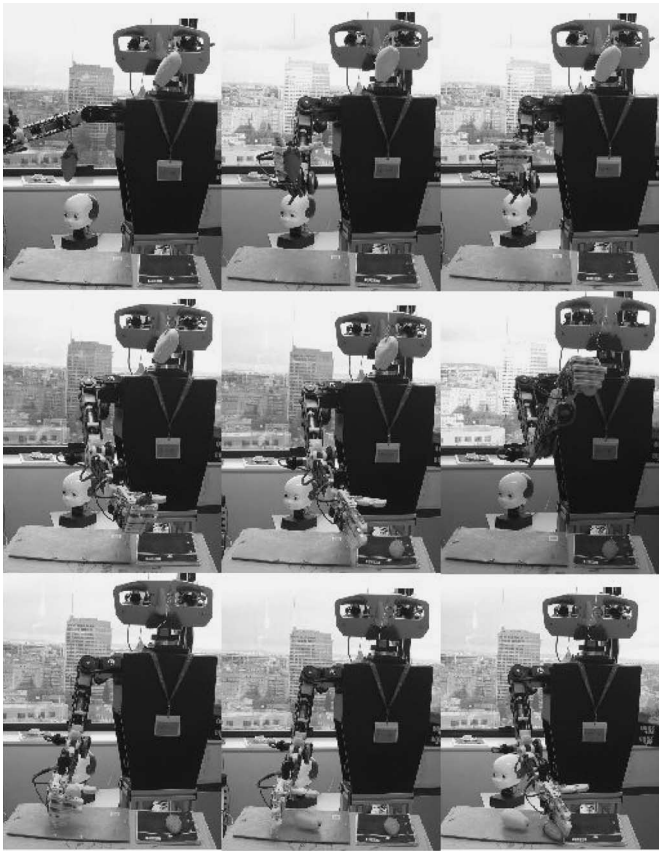

Fig. 15. Repetition of the task by the robot. Two different metrics are being used here: object ordering and metric position. The robot localizes and grasps object using the behaviors learned in previous developmental stages.

by slowly increasing the task complexity. We described results, implemented in a robotic system, of the various developmental stages of the system.

The robot first learns about its own body and surrounding environment; gathering all information by self-exploration. In the end of this stage, the coordination achieved is sufficient to ensure that the hand always remains in the image and that objects can be grasped in simple cases. We propose methods for learning different types of SMMs for redundant robots.

Motivated to further interact with objects, in a second phase, the system develops a closed-loop control behavior capable of precise grasping. The method consists in two phases: an open-loop controller putting the hand close to the object and a closed-loop vision-based controller for precisely touching the object. This method does not need calibration and can be learned online in a very efficient way. It also creates a map of the interesting objects in the surrounding space.

In the final developmental phase, people acting in the environment are the major source of information. The system is able to look at gestures and repeat them. In a much more complex problem, the system is able to see someone interacting with objects and extract an abstract description of this task. Then the system can repeat the task at a later time, relying on all the information learned previously.

Needless to say, several modules we described can be improved in the future. Also, new skills and mechanisms can be incorporated in the system following this developmental perspective. Further improvements of the grasping system will allow the system to explore the properties of objects in a richer manner. When interacting with people, mechanisms for joint attention can be very important from a communication point of view. Finally, the whole issue of learning task descriptions from observation has a lot of room for additional developments.

There are a number of far-reaching open questions for future endeavors. What kind of events should guide or trigger development? When is the system "ready" to go to a next stage? To attempt answering these questions, one option is to explore the role of time, quality, or event-driven processes to guide the behavior and development of the robot during its lifetime and through the interaction with the environment and other agents.

REFERENCES

[1] S. Schaal, "Is imitation learning the route to humanoid robots," *Trends Cogn. Sci.*, vol. 3, no. 6, pp. 233–242, 1999.

[2] J. Weng, "The developmental approach to intelligent robots," in *Proc. AAAI Spring Symp. Series, Integrating Robotic Res.: Taking NextLeap*, Stanford, CA, Mar. 1998.

[3] M. Asada, K. MacDorman, H. Ishiguro, and Y. Kuniyoshi, "Cognitive developmental robotics as a new paradigm for the design of humanoid-robots," *Robot. Autom.*, vol. 37, no. 2/3, pp. 185–193, Nov. 2001.

[4] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini, "Developmental robotics: A survey," *Connect. Sci.*, vol. 15, no. 40, pp. 151–190, Dec. 2003.

[5] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Phil. Trans. R. Soc. London, B Biol. Sci.*, vol. 358, no. 1431, pp. 537–547, 2003.

[6] E. Oztop and M. A. Arbib, "Schema design and implementation of the grasp-related mirror neuron system," *Biol. Cybern.*, vol. 87, no. 2, pp. 116–140, Aug. 2002.

[7] C. L. Nehaniv and K. Dautenhahn, "Like me?—Measures of correspondence and imitation," *Cybern. Syst., Int. J.*, vol. 32, no. 1/2, pp. 11–51, 2001.

[8] S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, and K. Ikeuchi, "Generating whole body motions for a biped humanoid robot from captured humandances," in *Proc. ICRA*, 2003, pp. 3905–3910.

[9] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: Extracting reusable task knowledge from visual observation of human performance," *IEEE Trans. Robot. Autom.*, vol. 10, no. 6, pp. 799–822, Dec. 1994.

[10] A. Billard, Y. Epars, S. Calinon, G. Cheng, and S. Schaal, "Discovering optimal imitation strategies," *Robot. Auton. Syst.*, vol. 47, no. 2/3, pp. 69–77, 2004.

[11] A. Alissandrakis, C. L. Nehaniv, and K. Dautenhahn, "Imitating with Alice: Learning to imitate corresponding actions across dissimilar embodiments," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 32, no. 4, pp. 482–496, Jul. 2002.

[12] V. G. Payne and L. D. Isaacs, *Human Motor Development: A Lifespan Approach*, 4th ed. Mountain View, CA: Mayfield, 1999.

[13] A. van der Meer, F. van der Weel, and D. Lee, "The functional significance of arm movements in neonates," *Science*, vol. 267, no. 5198, pp. 693–695, Feb. 1995.

[14] T. G. R. Bower, *A Primer of Infant Development*. San Francisco, CA: Freeman, 1977.

[15] E. Spelke, "Core knowledge," *Amer. Psychol.*, vol. 55, no. 11, pp. 1233–1243, Nov. 2000.

[16] E. Birch, S. Morale, B. Jeffrey, A. O'Connor, and S. Fawcett, "Measurement of stereoacuity outcomes at ages 1 to 24 months: Randot stereocards," *J. Amer. Assoc. Ped. Ophthalmol. Strabismus*, vol. 9, no. 1, pp. 31–36, Feb. 2005.

[17] Y. Nagai, M. Asada, and K. Hosoda, "A developmental approach accelerates learning of joint attention," in *Proc. Int. Conf. Develop. and Learning*, 2002, p. 277.

[18] A. Arsénio, "Cognitive-developmental learning for a humanoid robot: A caregiver's gift," Ph.D. dissertation, MIT, Cambridge, MA, Sep. 2004.

[19] G. Metta, "Babybot: A study on sensori-motor development," Ph.D. dissertation, Univ. Genova, Genova, Italy, 1999.

[20] P. E. Hotz, G. Gómez, and R. Pfeifer, "Evolving the morphology of a neural network for controlling a foveating retina and its test on a real robot," in *Proc. Artif. Life VIII—8th Int. Conf. Simulation and Synthesis Living Systems*, 2003, vol. 2003, pp. 243–251.
[21] C. Breazeal and J. Velasquez, "Toward teaching a robot 'infant' using emotive communication acts," in *Proc. Simulated Adaptive Behavior Workshop Socially Situated Intell.*, Zurich, Switzerland, 1998, pp. 25–40.
[22] M. Lopes, "A developmental roadmap for learning by imitation in robots," Ph.D. dissertation, Inst. Superior Técnico, Lisboa, Portugal, May 2006. [Online]. Available: vislab.isr.ist.utl.pt
[23] M. Lopes, R. Beira, M. Praça, and J. Santos-Victor, "An anthropomorphic robot torso for imitation: Design and experiments," in *Proc. IEEE ICRA*, Sendai, Japan, 2004, pp. 661–667.
[24] Y. Demiris and B. Khadhouri, "Hierarchical attentive multiple models for execution and recognition (hammer)," *Robot. Auton. Syst.*, vol. 54, no. 5, pp. 361–369, May 2006.
[25] M. Lopes and J. Santos-Victor, "Visual learning by imitation with motor representations," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 3, pp. 438–449, Jun. 2005.
[26] ——, "Learning sensory-motor maps for redundant robots," in *Proc. IEEE/RSJ Int. Conf. IROS*, 2006, pp. 2670–2676.
[27] K. Hosoda and M. Asada, "Advances in robot learning," in *How Does a Robot Find Redundancy by Itself?—A Control Architecuture for Adaptive Multi DOF Robots*, J. Wyatt and J. Demiris, Eds. New York: Springer-Verlag, 2000.
[28] A. Tikhonov and V. Arsenin, *Solution of Ill-Posed Problems*. Washington, DC: Winston, 1977.
[29] T. Poggio, V. Torre, and C. Kock, "Computational vision and regularization theory," *Nature*, vol. 317, no. 6035, pp. 314–319, Sep./Oct. 1985.
[30] M. Bertero, T. Poggio, and V. Torre, "Ill-posed problems in early vision," *Proc. IEEE*, vol. 76, no. 8, pp. 869–889, Aug. 1988.
[31] O. Khatib, O. Brock, K.-S. Chang, D. R. Sentis, L. Sentis, and S. Viji, "Human-centered robotics and interactive haptic simulation," *Int. J. Robot. Res.*, vol. 23, no. 2, pp. 167–178, Feb. 2004.
[32] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: An O(n) algorithm for incrementalreal time learning in high dimensional spaces," in *Proc. ICML*, Stanford, CA, 2000, pp. 288–293.
[33] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *IEEE Trans. Robot. Autom.*, vol. 12, no. 5, pp. 651–670, Oct. 1996.
[34] C. Samson, M. Le Borgne, and B. Espiau, *Robot Control: The Task Function Approach*. Oxford, U.K.: Clarendon, 1991.
[35] N. Mansard and F. Chaumette, "Tasks sequencing for visual servoing," in *Proc. IEEE/RSJ Int. Conf. IROS*, Sendai, Japan, Nov. 2004, pp. 992–997.
[36] P. Baerlocher and R. Boulic, "An inverse kinematic architecture enforcing an arbitrary number of strictpriority levels," *Vis. Comput.*, vol. 6, no. 20, pp. 402–417, 2004.
[37] R. Fletcher, *Practical Methods of Optimization*, 2nd ed. Chichester, U.K.: Wiley, 1987.
[38] K. Hosoda and M. Asada, "Versatile visual servoing without knowledge of true Jacobian," in *Proc. Int. Conf. Intell. Robots and Syst.*, Munchen, Germany, Sep. 1994, pp. 186–193.
[39] M. Jagersand and R. Nelson, "On-line estimation of visual–motor models using active vision," in *Proc. ARPA Image Understanding Workshop*, 1996, pp. 677–682.
[40] N. Mansard, M. Lopes, J. Santos-Victor, and F. Chaummette, "Jacobian learning methods for sequencing visual servoing," in *Proc. IEEE/RSJ Int. Conf. IROS*, 2006, pp. 4284–4290.
[41] G. Rizzolatti, L. Fadiga, L. Fogassi, and V. Gallese, "The space around us," *Science*, vol. 277, no. 5323, pp. 190–191, Jul. 1977.
[42] P. Rochat, N. Goubet, and S. J. Senders, "To reach or not to reach? Perception of body effectivities by young infants," *Infant Child Dev.*, vol. 8, no. 3, pp. 129–148, Sep. 1999.
[43] J. J. Gibson, *The Ecological Approach to Visual Perception*. Boston, MA: Houghton Mifflin, 1979.

Manuel Lopes (M'06) received the Ph.D. degree in electrical and computer engineering in the area of robotics from the Instituto Superior Técnico, Lisbon, Portugal, in 2006.

He is currently a Researcher with the Institute of Systems and Robotics, Lisbon. He has participated in various international research projects in the areas of robotics and cognitive systems. His research interests include robotics, development, computer vision, and human–robot interaction.

José Santos-Victor (M'96) received the Ph.D. degree in electrical and computer engineering from the Instituto Superior Técnico (IST), Lisbon, Portugal, in 1995, in the area of active computer vision and robotics.

He is an Associate Professor at the Department of Electrical and Computer Engineering, IST, and a Researcher at the Institute of Systems and Robotics, Lisbon, where he coordinates researches at the Computer and Robot Vision Laboratory. He is responsible for the participation of IST in various European and national research projects in the areas of computer vision and robotics. His research interests are in the areas of computer and robot vision, particularly in the relationship between visual perception and the control of action in (land, air, and underwater) mobile robots.