Intrinsically Motivated Exploration for Developmental and Active Sensorimotor Learning

Pierre-Yves Oudeyer, Adrien Baranes and Frédéric Kaplan[1]

Abstract— Intrinsic motivation is a central mechanism that guides spontaneous exploration and learning in humans. It fosters incremental and progressive sensorimotor and cognitive development by pushing exploration of activities of intermediate complexity given the current state of capabilities. This chapter presents and studies two computational intrinsic motivation systems that share similarities with human intrinsic motivation systems, IAC and R-IAC, that aim at self-organizing and efficiently guiding exploration for sensorimotor learning in robots. IAC was initially introduced to model the qualitative formation of developmental motor stages of increasing complexity, as shown in the Playground Experiment which we will outline. In this chapter, we argue that IAC and other intrinsically motivated learning heuristics could also be viewed as active learning algorithms that are particularly suited for learning forward models in unprepared sensorimotor spaces with large unlearnable subspaces. Then, we introduce a novel formulation of IAC, called R-IAC, and show that its performances as an intrinsically motivated active learning algorithm are far superior to IAC in a complex sensorimotor space where only a small subspace is "interesting", i.e. neither unlearnable nor trivial. We also show results in which the learnt forward model is reused in a control scheme. Finally, an open-source accompanying software containing these algorithms as well as tools to reproduce all the experiments in simulation presented in this paper is made publicly available.

Index Terms— active learning, intrinsically motivated learning, exploration, developmental robotics, artificial curiosity, sensorimotor learning.

1.1 Intrinsically Motivated Exploration and Learning

Developmental robotics approaches are studying mechanisms that may allow a robot to continuously discover and learn new skills in unknown environments and in a life-long time scale [1], [2]. A main aspect is the fact that the set of these skills and their functions are at least partially unknown to the engineer who conceive the robot initially, and are also task-independent. Indeed, a desirable feature is that robots should be capable of exploring and developing various kinds of

[1] Pierre-Yves Oudeyer and Adrien Baranes are with INRIA, France (http://flowers.inria.fr), and Frédéric Kaplan is with CRAFT-EPFL, Switzerland. Material presented in this chapter is based on several previous publications of the authors (in particular [27, 61]).

skills that they may re-use later on for tasks that they did not foresee. This is what happens in human children, and this is also why developmental robotics shall import concepts and mechanisms from human developmental psychology.

1.1.1 The problem of exploration in open-ended learning

Like children, the "freedom" that is given to developmental robots to learn an open set of skills also poses a very important problem: as soon as the set of motors and sensors is rich enough, the set of potential skills become extremely large and complicated. This means that on the one hand, it is impossible to try to learn all skills that may potentially be learnt because there is not enough time to physically practice all of them. Furthermore, there are many skills or goals that the child/robot could imagine but never be actually learnable, because they are either too difficult or just not possible (for example, trying to learn to control the weather by producing gestures is hopeless). This kind of problem is not at all typical of the existing work in machine learning, where usually the "space" and the associated "skills" to be learnt and explored are well-prepared by a human engineer. For example, when learning hand-eye coordination in robots, the right input and output spaces (e.g. arm joint parameters and visual position of the hand) are typically provided as well as the fact that hand-eye coordination is an interesting skill to learn. But a developmental robot is not supposed to be provided with the right subspaces of its rich sensorimotor space and with their association with appropriate skills: it would for example have to discover that arm joint parameters and visual position of the hand are related in the context of a certain skill (which we call hand-eye coordination but which it has to conceptualize by itself) and in the middle of a complex flow of values in a richer set of sensations and actions.

1.1.2 Intrinsic motivations

Developmental robots, like humans, have a sharp need for mechanisms that may drive and self-organize the exploration of new skills, as well as identify and organize useful sub-spaces in its complex sensorimotor experiences. Psychologists have identified two broad families of guidance mechanisms which drive exploration in children:

1) **Social learning**, which exists in different forms such as stimulus enhancement, emulation, imitation or demonstration, and which many groups try to implement in robots [e.g. 3,4,5,6,7,8,9,10,11,12,13,14];

2) **Internal guiding mechanisms**, also studied by many robotics research groups (e.g. see [15,16,17,18,19,20]) and in particular intrinsic motivation, responsible of spontaneous exploration and curiosity in humans, which is the mechanisms underlying the algorithms presented in this paper.

Intrinsic motivations are mechanisms that guide curiosity-driven exploration, that were initially studied in psychology [21]-[23] and are now also being approached in neuroscience [24]-[26]. Machine learning and robotics researchers have proposed that such mechanism might be crucial for self-organizing developmental trajectories as well as for guiding the learning of general and reusable skills in machines and robots [27,28]. A large diversity or approaches for operationalizing intrinsic motivation have been presented in the literature [e.g. 29,30,31,32,33,34,28,27,35], and see [27] for a general overview. Several experiments have been conducted in real-world robotic setups, such as in [27,36,34] where an intrinsic motivation system was shown to allow for the progressive discovery of skills of increasing complexity, such as in the Playground Experiment that we will present in section 4. In these experiments, the focus was on the study of how developmental stages could self-organize into a developmental trajectory of increasing complexity without a direct pre-specification of these stages and their number. As we will explain in section 4, this can lead to stimulating models of the self-organization of structured developmental trajectories with both universal tendencies and diversity as observed in humans [60]. Furthermore, in this chapter, we argue that such intrinsic motivation systems can be used as efficient active learning algorithms. With this view, we present a novel system, called **R-IAC**, which improves **IAC** over a number of features. Through several experiments, we will show that it can be used as an efficient active learning algorithm to learn forward and inverse models in complex unprepared sensorimotor spaces with unlearnable subspaces.

1.2 IAC and R-IAC for Intrinsically Motivated Active Learning

1.2.1 Developmental Active Learning

In **IAC**, intrinsic motivation is implemented as a heuristics which pushes a robot to explore sensorimotor activities for which learning progress is maximal, i.e. subregions of the sensorimotor space where the predictions of the learnt forward model improve fastest [27]. Thus, this mechanism regulates actively the growth of complexity in sensorimotor exploration, and can be conceptualized as a **developmental active learning** algorithm. This heuristics shares properties with statistical techniques in optimal experiment design (e.g. [37]) where exploration is driven by expected information gain, as well as with attention and motivation mechanisms proposed in the developmental psychology literature (e.g. [22], [38], or see [23] for a review) where it has been proposed that exploration is preferentially focused on activities of intermediate difficulty or novelty [39,40], but differs significantly from many active learning heuristics in machine learning in which exploration is directed towards regions where the learnt model is maximally uncertain or where predictions are maximally wrong (e.g. [41, 42], see [27] for a review). As argued in [27], developmental robots are typically faced with large sensorimotor spaces which cannot be entirely learnt (because of time limits among other reasons) and/or in which subregions are not learnable (potentially because it is too complicated for the learner, or because there are no correlations between the input and output variables, see examples in the experiment section and in [27]). In these sensorimotor spaces, exploring zones of maximal uncertainty or unpredictability is bound to be an inefficient strategy since it would direct exploration towards subspaces in which there are no learnable correlations, while a heuristics based on learning progress allows to avoid unlearnable parts as well as to focus exploration on zones of gradually increasing complexity.

In [27, 34], experiments such as the Playground Experiment described in section 4 showed how **IAC** can allow an AIBO robot, equipped with a set of parameterized motor primitives (in a 5 DOF motor space), as well as a set of perceptual primitives (in a 3 DOF perceptual space), to self-organize a developmental trajectory in which a variety of affordances uses of the motor primitives where learnt in spite of not having been specified initially. In [36], a slightly modified version of **IAC** allowed an AIBO robot, equipped with parameterized central pattern generators (CPG's) in a 24 DOF motor space and 3 DOF perceptual space, to learn a variety of locomotion skills. Yet, these previous results focused on qualitative properties of the self-organized developmental trajectories, and **IAC** was not optimized for efficient active learning per se.

Here, we present a novel formulation of **IAC**, called **Robust-IAC (R-IAC)**, and show that it can efficiently allow a robot to learn actively, fast and correctly forward and inverse kinematic models in an unprepared sensorimotor space. As we will explain, **R-IAC** introduces four main advances compared to **IAC**:

- **Probabilistic action selection**: instead of choosing actions to explore the zone of maximal learning progress at a given moment in time (except in the random action selection mode), R-IAC explores actions on sensorimotor subregions probabilistically chosen based on their individual learning progress;
- **Multi-resolution monitoring of learning progress**: in R-IAC, when sensorimotor regions are split into subregions, parent regions are kept and one continues to monitor learning progress in them, and they continue to be eligible regions for action selection. As a consequence, learning progress is monitored simultaneously at various regions scales, as opposed to IAC where it was monitored only in child regions and thus at increasing small scales;
- **A new region splitting mechanism** that is based on the direct optimization of learning progress dissimilarity among regions;
- **The introduction of a third exploration mode** hybridizing learning progress heuristics with more classic heuristics based on the exploration of zones of maximal unpredictability;

1.2.2 Prediction Machine and Analysis of Error Rate

We consider a robot as a system with motor/actions channels $\mathbf{M}$ and sensory/state channels **S.** $\mathbf{M}$ and $\mathbf{S}$ can be low-level such as torque motor values or touch sensor values, or higher level such as a "go forward one meter" motor command or "face detected" visual sensor". Furthermore, $\mathbf{S}$ can correspond to internal sensors measuring the internal state of the robot or encoding past values of the sensors. Real valued action/motor parameters are represented as a vector $\mathbf{M(t)}$, and sensors, as $\mathbf{S(t)}$, at a time t. $\mathbf{SM(t)}$ represents a sensorimotor context, i.e. the concatenation of both motors and sensors vectors.

We also consider a Prediction Machine **PM** (Fig. 1), as a system based on a learning algorithm (neural networks, KNN, etc.), which is able to create a forward model of a sensorimotor space based on learning examples collected through self-determined sensorimotor experiments. Experiments are defined as series of actions, and consideration of sensations detected after actions are performed. An experiment is represented by the set $\mathbf{(SM(t), S(t+1))}$**,** and denotes the sensory/state consequence **S(t+1)** that is observed when actions encoded in **M(t)** are performed in the sensory/state context **S(t).** This set is called a "**learning exemplar**". After each trial, the prediction machine **PM** gets this data and incrementally updates the forward model that it is encoding, i.e. the robot incrementally increases its knowledge of the sensorimotor space. In this update process, **PM** is able to compare, for a given context $\mathbf{SM}(\boldsymbol{t})$, differences between predicted sensations $\tilde{\mathbf{S}}(\boldsymbol{t}+\mathbf{1})$ (estimated using the created model), and real consequences $\mathbf{S}(\boldsymbol{t}+\mathbf{1})$. It is then able to produce a measure of error $\boldsymbol{e}(\boldsymbol{t}+\mathbf{1})$**,** which represents the quality of the model for sensorimotor context $\mathbf{SM}(\boldsymbol{t})$. This is summarized in figure 1.

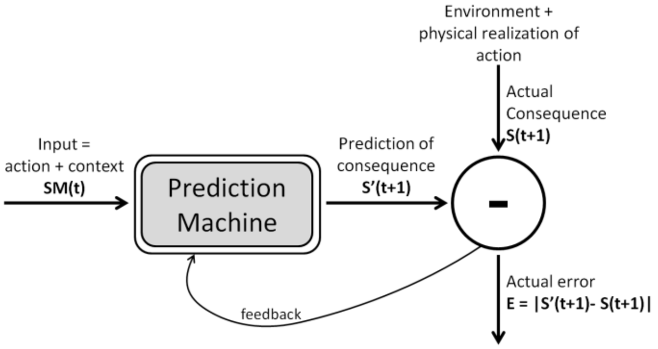


Fig. 1. The prediction learning machine (e.g. a neural network, an SVM, or Gaussian process regression based algorithm)

Then, we consider a module able to analyze learning evolutions over time, called Prediction Analysis Machine **PAM**, Fig. 2. In a given subregion $\boldsymbol{R_n}$ of the sensorimotor space (which we will define below), this system monitors the evolution of errors in predictions made my **PM** by computing its derivative, i.e. the learning progress, $\boldsymbol{LP_n} = e_N - e_F$ in this particular region over a sliding time window (see Fig 2). $\boldsymbol{LP_n}$ is then used as a measure of interestingness used in the action selection scheme outlined below. The more a region is characterized by learning progress, the more it is interesting, and the more the system will perform experiments and collect learning exemplars that fall into this region. Of course, as exploration goes on, the learnt forward model becomes better in this region and learning progress might decrease, leading to a decrease in the interestingness of this region.

To precisely represent the learning behavior inside the whole sensorimotor space and differentiate its various evolutions in various subspaces/subregions, different **PAM** modules, each associated to a different subregion R_i of the sensorimotor space, need to be built. Therefore, the learning progress $\boldsymbol{LP_i}$ provided as the output values of each **PAM** becomes representative of the interestingness of the associated region $\boldsymbol{R_i}$**.** Initially, the whole space is considered as one single region $\boldsymbol{R_0}$, associated to one **PAM**, which will be progressively split into subregions with their own **PAM** as we will now describe.

1.2.3 The Split Machine

The Split Machine **SpM** (Fig. 3) possesses the capacity to memorize all the experimented learning exemplars $(\mathbf{SM(t)}, \mathbf{S(t+1)})$**,** and the corresponding errors values $\boldsymbol{e(t+1)}$**.** It is both responsible for identifying the region and **PAM** corresponding to a given **SM(t)**, but also responsible of splitting (or creating in R-IAC where parent regions are kept in use) sub-regions from existing regions.

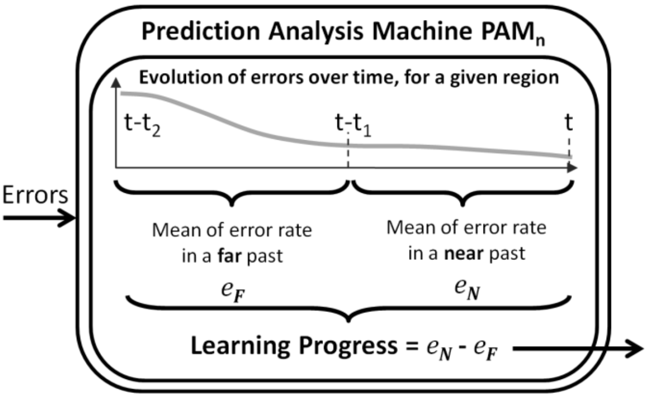

Prediction Analysis Machine PAM$_n$
Evolution of errors over time, for a given region
t-t$_2$
t-t$_1$
t
Errors
Mean of error rate in a **far** past
Mean of error rate in a **near** past
e_F
e_N
Learning Progress = e_N - e_F

Fig. 2. Internal mechanism of the Prediction Analysis Machine $\mathbf{PAM}_{\boldsymbol{n}}$ associated to a given subregion $\boldsymbol{R_n}$ of the sensorimotor space. This module considers errors detected in prediction by the Prediction Machine **PM**, and returns a value representative of the learning progress in the region. Learning progress is the derivative of errors analyzed between a far and a near past in a fixed length sliding window.

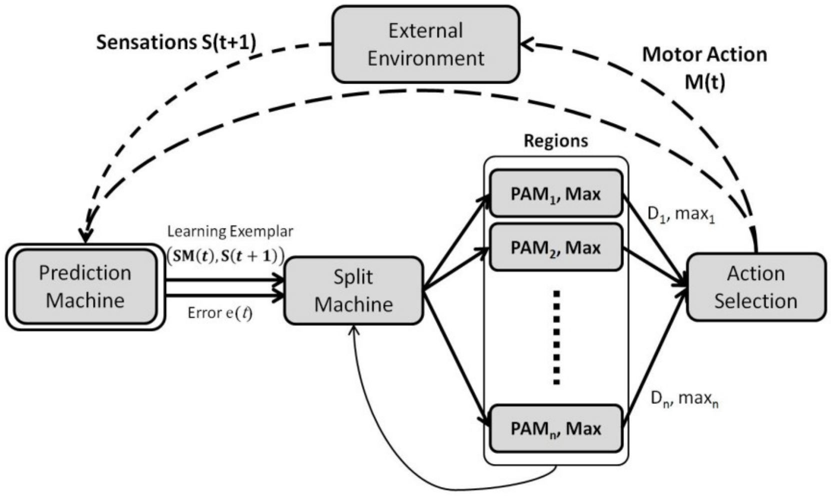


Fig. 3. General architecture of IAC and R-IAC. The prediction Machine is used to create a forward model of the world, and measures the quality of its predictions (errors values). Then, a split machine cuts the sensorimotor space into different regions, whose quality of learning over time is examined by Prediction Analysis Machines. Then, an Action Selection system, is used to choose experiments to perform.

1) Region Implementation

We use a tree representation to store the list of regions as shown in Fig. 4. The main node represents the whole space, and leafs are subspaces. $\mathbf{S(t)}$ and $\mathbf{M(t)}$ are here normalized into $[0;1]^n$. The main region (first node), called R_0, represents the whole sensorimotor space. Each region stores all collected exemplars that it covers. When a region contains more than a fixed number $\mathbf{T_{split}}$ of exemplars, we split it into two ones in **IAC**, or create two new regions in **R-IAC**. Splitting is done with hyperplanes perpendicular to one dimension. An example of split execution is shown in Fig. 4, using a two dimensions input space.

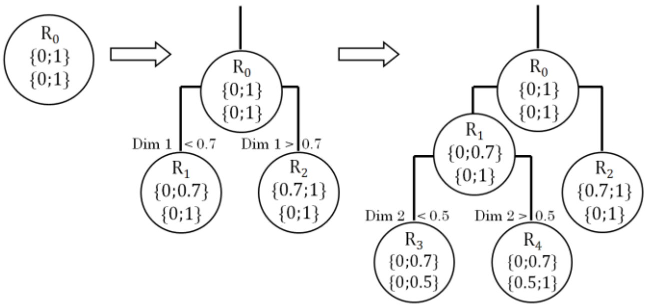


Fig. 4. The sensorimotor space is iteratively and recursively split into sub-spaces, called "regions". Each region R_n is responsible for monitoring the evolution of the error rate in the anticipation of consequences of the robot's actions, if the associated contexts are covered by this region.

2) IAC Split Algorithm

In the **IAC** algorithm, the idea was to find a split such that the two sets of exemplars into the two subregions would minimize the sum of the variances of $\mathbf{S}(\boldsymbol{t}+\mathbf{1})$ components of exemplars of each set, weighted by the number of exemplars of each set. Hence, the split takes place in the middle of zones of maximal change in the function $\boldsymbol{SM}(\boldsymbol{t}) \rightarrow \boldsymbol{S}(\boldsymbol{t}+\mathbf{1})$. Mathematically, we consider $\varphi_n = \left\{ \left(\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t}+\mathbf{1})\right)_i \right\}$ as the set of exemplars possessed by region R_n. Let us denote j a cutting dimension and v_j, an associated cutting value. Then, the split of φ_n into φ_{n+1} and φ_{n+2} is done by choosing j and v_j such that:

(1) All the exemplars $(\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t}+\mathbf{1}))_i$ of φ_{n+1} have a j^{th}component of their $\mathbf{SM}(\boldsymbol{t})$ smaller than v_j

(2) All the exemplars $(\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t}+\mathbf{1}))_i$ of φ_{n+2} have a j^{th}component of their $\mathbf{SM}(\boldsymbol{t})$ greater than v_j

(3) The quantity :

$$Qual(j, v_j) = \\ |\,\varphi_{n+1}|.\,\sigma\left(\left\{\mathbf{S}(\boldsymbol{t}+\mathbf{1}) \,|\, \left(\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t}+\mathbf{1})\right) \in \varphi_{n+1}\right\}\right) \\ + |\,\varphi_{n+2}|.\,\sigma\left(\left\{\mathbf{S}(\boldsymbol{t}+\mathbf{1}) \,|\, \left(\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t}+\mathbf{1})\right) \in \varphi_{n+2}\right\}\right)$$

is **minimal**, where

$$\sigma(\mathrm{S}) = \frac{\sum_{v \in \mathrm{S}} \left\| s - \frac{\sum_{v \in S} v}{|\mathrm{S}|} \right\|^2}{|\mathrm{S}|}$$

where S is a set of vectors, and $|\mathrm{S}|$, its cardinal. Finding the exact optimal split would be computationally too expensive. For this reason, we use the following heuristics for optimization: for each dimension j, we evaluate N_{sp} cutting values

v_j equally spaced between the extrema values of φ_n, thus we evaluate $N_{sp}.|\{j\}|$ splits in total, and the one with minimal $Qual(j, v_j)$ is finally chosen. This computationally cheap heuristics has produced acceptable results in all the experiments we ran so far. It could potentially be improved by allowing region splits cutting multiple dimensions at the same time in conjunction with a Monte-Carlo based sampling of the space of possible splits.

3) R-IAC Split Algorithm

In **R-IAC**, the splitting mechanism is based on comparisons between the learning progress in the two potential child regions. The principal idea is to perform the **separation which maximizes the dissimilarity of learning progress** comparing the two created regions. This leads to the direct detection of areas where the learning progress is maximal, and to separate them from others (see Fig. 5). This contrasts with **IAC** where regions where built independently of the notion of learning progress.

Reusing the notations of the previous section, in **R-IAC** the split of φ_n into φ_{n+1} and φ_{n+2} is done by choosing j and v_j such that:

$$\begin{aligned} Qual(j, v_j) &= \\ (LP_{n+1}(\{\mathbf{e}(\boldsymbol{t}+\mathbf{1}) | (\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t}+\mathbf{1})) &\in \varphi_{n+1}\}) \\ - LP_{n+2}(\{\mathbf{e}(\boldsymbol{t}+\mathbf{1}) | (\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t}+\mathbf{1})) &\in \varphi_{n+2}\}))^2 \end{aligned}$$

is **maximal**, where

$$LP_k(E) = \frac{\sum_{i=1}^{\frac{|E|}{2}} e(i) - \sum_{i=\frac{|E|}{2}}^{|E|} e(i)}{|E|}$$

Where E is a set of errors values $\{e(i)\}$ with errors indexed by their relative order i of encounter (e.g. error *e(9)* corresponds to a prediction made by the robot before another prediction which resulted in *e(10)*: this implies that the order of exemplars collected and associated prediction errors are stored in the system), and $LP_k(E)$is the learning progress of region R_k. The heuristics used to find an approximate maximal split is the same as the one described above for **IAC**.

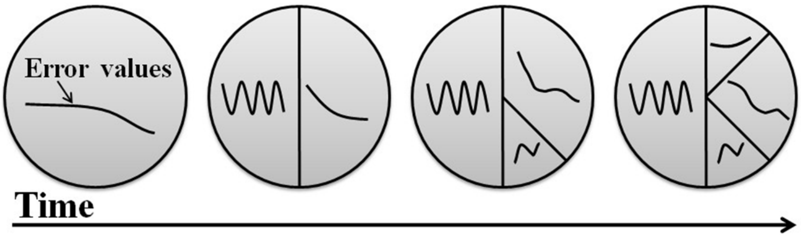


Fig. 5. Evolution of the sensorimotor regions over time. The whole space is progressively subdivided in such a way that the dissimilarity of each sub-region in terms of learning progress is maximal.

1.2.4 Action Selection Machine

We present here an implementation of Action Selection Machine **ASM**. The **ASM** decides of actions $\mathbf{M}(\boldsymbol{t})$ to perform, given a sensory context $\mathbf{S}(\boldsymbol{t})$. (See Fig. 3.). The ASM heuristics is based on a mixture of several **modes**, which differ between **IAC** and **R-IAC**. Both **IAC** and **R-IAC** algorithms share the same global loop in which modes are chosen probabilistically:

<u>Outline of the global loop of IAC and R-IAC algorithms:</u>

- **Action Selection Machine ASM**: given $\mathbf{S(t)}$, execute an action $\mathbf{M}(\boldsymbol{t})$ using the **mode** $(\boldsymbol{n})$ with probability $\boldsymbol{p_n}$and based on data stored in the region tree, with $\boldsymbol{n} \in \{\mathbf{1},\mathbf{2}\}$ for IAC and $\boldsymbol{n} \in \{\mathbf{1},\mathbf{2},\mathbf{3}\}$ for R-IAC;
- **Prediction Machine PM**: Estimate the predicted consequence $\tilde{\boldsymbol{S}}_{\boldsymbol{t+1}}$ using the prediction machine **PM ;**
- **External Environment**: Measure the real consequence $\boldsymbol{S_{t+1}}$
- **Prediction Machine PM**: Compute the error $\boldsymbol{e}(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{abs}(\tilde{\boldsymbol{S}}_{\boldsymbol{t+1}} - \boldsymbol{S_{t+1}})$;
- Update the **prediction machine PM** with $\left(\mathbf{SM(t)}, \mathbf{S(t+1)}\right)$
- **Split Machine SpM:** update the region tree with $\left(\mathbf{SM(t)}, \mathbf{S(t+1)}\right)$ and $\boldsymbol{e}(\boldsymbol{t}+\mathbf{1})$;
- **Prediction Analysis Machine PAM:** update evaluation of learning progress in the regions that cover $\left(\mathbf{SM(t)}, \mathbf{S(t+1)}\right)$

We now present the different exploration modes used by the Action Selection Machine, in **IAC** and **R-IAC** algorithm:

1) Mode 1: Random Babbling Exploration

The **random babbling** mode corresponds to a totally random exploration (random choice of $\mathbf{M(t)}$ with a uniform distribution), which does not consider previous actions and context. This mode appears in both **IAC** and **R-IAC** algorithm, with a probability $\boldsymbol{p_1}$ typically equal to 30%**.**

2) *Mode 2: Learning Progress Maximization Exploration*

This mode, chosen with a probability $\boldsymbol{p_1}$ typically equal to 70%**,** aims to maximize learning progress, but with two different heuristics in **IAC** and **R-IAC**:

IAC: In the **IAC** algorithm, mode *2* action selection is straightforward: among the leaf regions that cover the current state $\mathbf{S(t)}$ (i.e. for which there exists a $\mathbf{M(t)}$ such that $\mathbf{SM(t)}$ is in the region - there are typically many), the leaf region which learning progress is maximal is found, and then a random action within this region is chosen;

R-IAC: In the **R-IAC** algorithm, we take into account the fact that many regions may have close learning progress values, and thus should be selected roughly equally often, by taking a probabilistic approach to region selection. This avoids the problems of a winner take-all strategy when the region splits do not reflect well the underlying learnability structure of the sensorimotor space. Furthermore, instead of focusing on the leaf regions like in **IAC**, **R-IAC** continues to monitor learning progress in node regions and select them if they have more learning progress: thus learning progress is monitored simultaneously at several scales in the sensorimotor space. Let us give more details:

i) Probabilistic approach to region selection

A region R_n is chosen among all eligible regions $R = \{R_i\}$ (i.e. for which there exists a $\mathbf{M(t)}$ such that $\mathbf{SM(t)}$ is in the region) with a probability $\boldsymbol{P_n}$ proportional to its learning progress LP_n, stored in the associated $\boldsymbol{PAM_n}$:

$$\boldsymbol{P_n} = \frac{|LP_n - min(LP_i)|}{\sum_{i=1}^{|R|} |LP_i - min(LP_i)|}$$

j) Multi-resolution monitoring of learning progress

In the **IAC** algorithm, the estimation of learning progress only happens in leaf regions, which are the only eligible regions for action selection. In **R-IAC**, learning progress is monitored in all regions created during the system's life time, which allows us to track learning progress at multiple resolution in the sensorimotor space. This implies that when a new exemplar is available, **R-IAC** updates the evaluation of learning progress in all regions that cover this exemplar (but only if the exemplar was chosen randomly, i.e. not with mode *3* as described below). Because regions are created in a top-down manner and stored in a tree structure which was already used for fast access in IAC, this new heuristics does not bring computational overload and can be implemented efficiently.

In **R-IAC mode** *2*, when a region has been chosen with the probabilistic approach and the multi-resolution scheme, a random action is chosen within this region with a probability $\boldsymbol{p_2}$ typically equal to 60%, (which means this is the dominant mode.

3) Mode 3: Error Maximization Exploration

Mode 3 combines a traditional active learning heuristics with the concept of learning progress: in mode 3, a region is first chosen with the same scheme as in **R-IAC** mode 2. But once this region has been chosen, an action in this region is selected such that the expected error in prediction will be maximal. This is currently implemented through a k-nearest neighbor regression of the function $SM(t) \rightarrow e(t+1)$ which allows finding the point of maximal error, to which is added small random noise (to avoid to query several times exactly the same point). Mode 3 is typically chosen with a probability $\boldsymbol{p_3} = 10\%$ in R-IAC (and does not appear in **IAC**).

1.2.5 Pseudo-code of R-IAC

```
RIAC( $PM, p_1, p_2, p_3, T_{split}, l, \eta, \Gamma, \kappa, \zeta$)

Init
    • Let $R_0$ be the whole space of mathematically possible values of the sensorimotor context SM(t) (typically a hypercube in $\mathbb{R}^d$);
    • Let $LP_0 = 0$ be the learning progress associated to $R_0$ ;
    • Let $Lex_{R_0} = \{\emptyset\}$ (later on in the algorithm, $Lex_{R_k}$ will be the set $\{((\mathbf{SM_i(t)}, \mathbf{S_i(t+1)}), \mathbf{e_i(t+1)}, \omega_i)\}$ where the set of $(\mathbf{SM_i(t)}, \mathbf{S_i(t+1)})$ components is the set of learning examplars collected in $R_k$ , the set of $\mathbf{e_i(t+1)}$ components is the set of associated prediction errors, and $\omega_i$ is an indice whose value indicates the relative order in which each particular learning examplar was collected within $R_k$ );
    • Init the prediction/learning machine PM with an empty set of learning exemplars;

Loop

Let S(t) be the current state;
Let $R = \{R_0, R_1, \ldots, R_n\}$ be the set of subregions $R_l$ of the sensorimotor space such that there exists a M(t) such that SM(t) $\in R_l$ ;
For all $n$, let $LP_n$ be the learning progress associated to $R_n$ ;
```

```
Action Selection

• Select action selection mode mode among mode 1, mode 2 and mode 3 with probabilities $p_1$, $p_2$, $p_3$;
• If mode = mode 1
    o Let $\mathbf{M(t)}$ be a random vector (uniform distribution)
• If mode = mode 2
    o For $l = 0 \dots n$,
      let $\boldsymbol{P}_l = \frac{\left|LP_l - min_{LP_i \in \boldsymbol{R}}(LP_i)\right|}{\sum_{i=1}^{|R|}\left|LP_i - min_{LP_i \in \boldsymbol{R}}(LP_i)\right|}$
    o Let $\boldsymbol{R_k}$ be a subregion in $\boldsymbol{R}$ chosen with probability $\boldsymbol{P_k}$, $k \in \{0, \dots, n\}$ in a roulette wheel manner ;
    o Let $\mathbf{M(t)}$ be a random vector such that $\boldsymbol{SM(t)} \in \boldsymbol{R_k}$ (uniform distribution);
• If mode = mode 3
    o For $l = 0 \dots n$,
      let $\boldsymbol{P}_l = \frac{\left|LP_l - min_{LP_i \in \boldsymbol{R}}(LP_i)\right|}{\sum_{i=1}^{|R|}\left|LP_i - min_{LP_i \in \boldsymbol{R}}(LP_i)\right|}$
    o Let $\boldsymbol{R_k}$ be a subregion in $\boldsymbol{R}$ chosen with probability $\boldsymbol{P_k}$, $k \in \{0, \dots, n\}$in a roulette-wheel manner ;
    o Let $\boldsymbol{Err_{R_k}}$ be a model of the errors made in prediction in $\boldsymbol{R_k}$ in the past, built with a $l$ -nearest neighbor algorithm on the last η learning examplars collected in $\boldsymbol{R_k}$ , belonging to $\boldsymbol{Lex_{R_k}}$ ;
    o Let Mmax(t) = $\boldsymbol{argmax_{M(t)}Err_{R_k}}\left(\boldsymbol{SM(t)}\right)$ obtained by sampling uniformly randomly Γ candidates M(t) ;
    o Let M(t) = Mmax(t) + $\boldsymbol{\varepsilon}$ with $\boldsymbol{\varepsilon}$ a small random number between 0 and σ along a uniform distribution.
• Execute M(t) ;
```

```
Prediction and measurement of the consequences of action

• Estimate the predicted consequence $\tilde{\mathbf{S}}(\boldsymbol{t}+\mathbf{1})$ of executing $\mathbf{M(t)}$ in the environment with state S(t) using the prediction machine PM ;
• Measure the real consequence $\mathbf{S(t+1)}$ after execution of $\mathbf{M(t)}$ in the environment with state S(t);
• Compute the error $\boldsymbol{e}(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{abs}\left(\tilde{\mathbf{S}}(\boldsymbol{t}+\mathbf{1}) - \mathbf{S(t+1)}\right)$;
• Update the prediction machine PM with the new learning plar $\left(\mathbf{SM(t)}, \mathbf{S(t+1)}\right)$;
```

```
Update of region models

• Let Ex = $(\mathbf{SM(t)}, \mathbf{S(t+1)}, \mathbf{e(t+1)})$
• Let $\gamma$ be the total number of regions created by the system so far;
• For all regions $\boldsymbol{R_k}$ such that SM(t) $\in \boldsymbol{R_k}$
    o Let $\boldsymbol{\omega}$ be the maximum $\boldsymbol{\omega_i}$ index in $\boldsymbol{Lex_{R_k}}$;
    o $\boldsymbol{Lex_{R_k}} = \boldsymbol{Lex_{R_k}} + \{((\mathbf{SM(t)}, \mathbf{S(t+1)}), \mathbf{e(t+1)}, \boldsymbol{\omega} + \mathbf{1})\}$ where $\boldsymbol{\omega} + \mathbf{1}$ is an indice used to keep track of the order in which this learning examplar was stored in relation to others (see below);
    o If card($\boldsymbol{Lex_{R_k}}$) $= \boldsymbol{T_{split}}$
    Create two new regions $\boldsymbol{R}_{\gamma+1}$ and $\boldsymbol{R}_{\gamma+2}$ as subregions of $\boldsymbol{R_k}$ with $j$, a cutting dimension and $v_j$, an associated cutting value optimized through random uniform sampling of $\kappa$ possible candidates and such that:
        1. $\boldsymbol{Lex}_{\boldsymbol{R}_{\gamma+1}}$ is initialized with all the elements in $\boldsymbol{Lex_{R_k}}$ that have a $j^{th}$ component of their $\mathbf{SM}(\boldsymbol{t})$ smaller than $v_j$;
        2. $\boldsymbol{Lex}_{\boldsymbol{R}_{\gamma+2}}$ is initialized with all the elements in $\boldsymbol{Lex_{R_k}}$ that have a $j^{th}$ component of their $\mathbf{SM}(\boldsymbol{t})$ greater than $v_j$;
        3. The difference between learning progresses $LP_{\gamma+1}$ and $LP_{\gamma+2}$ measured in both subregions is maximal, i.e.
            $$(\boldsymbol{LP}_{\boldsymbol{\gamma}+\mathbf{1}}\left(\left\{\mathbf{e_i}(\boldsymbol{t}+\mathbf{1})_{\boldsymbol{\upsilon}} | (\mathbf{SM_i}(\boldsymbol{t}), \mathbf{S_i}(\boldsymbol{t}+\mathbf{1}), \mathbf{e_i}(\boldsymbol{t}+\mathbf{1}), \boldsymbol{\omega_i}) \in \boldsymbol{Lex}_{\boldsymbol{R}_{\gamma+1}}\right\}\right) - \boldsymbol{LP}_{\boldsymbol{\gamma}+\mathbf{2}}\left(\left\{\mathbf{e_i}(\boldsymbol{t}+\mathbf{1})_{\boldsymbol{\upsilon}} | (\mathbf{SM_i}(\boldsymbol{t}), \mathbf{S_i}(\boldsymbol{t}+\mathbf{1}), \mathbf{e_i}(\boldsymbol{t}+\mathbf{1}), \boldsymbol{\omega_i}) \in \boldsymbol{Lex}_{\boldsymbol{R}_{\gamma+2}}\right\}\right))^2$$
        is maximal, where errors are indexed by their relative order of measurement $\boldsymbol{\upsilon}$ calculated from $\boldsymbol{\omega}$ values where
            $$LP(E) = \frac{\sum_{i=card\,(E)-\zeta}^{card\,(E)-\frac{\zeta}{2}} e_i - \sum_{i=card\,(E)-\frac{\zeta}{2}+1}^{card\,(E)} e_i}{card(E)}$$
        where $\zeta$ defines the time window used to compute learning progress achieved through the acquisition of most recent learning examplars in each region;
    o Store the learning progresses $\mathbf{LP}_{\boldsymbol{\gamma}+\mathbf{1}}$ and $\mathbf{LP}_{\boldsymbol{\gamma}+\mathbf{2}}$ of the two newly created regions;
    o $\gamma = \gamma + 1$
• For all regions $\boldsymbol{R_k}$ such that SM(t) $\in \boldsymbol{R_k}$ (except $\boldsymbol{R}_{\gamma+1}$ and $\boldsymbol{R}_{\gamma+2}$ if a split was performed), recompute $\boldsymbol{LP_k}$ and store the value;

EndLoop
```

1.2.6 Software

An open-source Matlab-based software library containing the source code of the **IAC** and **R-IAC** algorithms, as well as tools and a tutorial that allow to reproduce all experiments presented in sections IV and V below is made publicly available at: http://flowers.inria.fr/riac-software.zip

1.2.7 Remarks

Regulation of the growth of complexity. As argumented in detail in [28], the heuristics consisting in preferentially exploring subregions of the sensorimotor space where learning progress is maximal has the practical consequence to lead the robot to explore zones of intermediate complexity/difficulty/contingency, which has been advocated by developmental psychologists (e.g. [22,23,38]) as being the key property of spontaneous exploration in humans. Indeed, subregions which are trivial to learn are quickly characterized by a low plateau in prediction errors, and thus become uninteresting. On the other end of the complexity spectrum, subregions which are unlearnable are characterized with a high plateau in prediction errors and thus are also quickly identified as uninteresting. In between, exploration first focuses on subregions where prediction errors decrease fastest, which typically correspond to lower complexity situations, and when these regions are mastered and a plateau is reached, exploration continues in more complicated subregions where large learning progress is detected.

Key advances of R-IAC over IAC and robustness to potential inaccurate and large number of region splits. Among the various differences between IAC and R-IAC, the two most crucial ones are 1) the ***probabilistic choice of regions*** in R-IAC as opposed to the winner take all strategy in IAC, and 2) the ***multiresolution monitoring of learning progress*** in R-IAC as opposed to the only lowest scale monitoring of IAC. The combination of these two innovations allows the system to cope with potentially inaccurate and supernumerary region splits. Indeed, a problem in IAC was that if for example one homogeneous region with high learning progress was split, the winner-take-all strategy typically biased the system to explore later on only one of the two subregions, which was very inefficient. Furthermore, the more regions were split, which happened continuously given the splitting mechanism, the smaller they became, and because only child regions were monitored, exploration was becoming increasingly focused on smaller and smaller subregions of the sensorimotor space, which was also often quite inefficient. While the new splitting mechanism introduced in this paper allows the system to minimize inaccurate splits, the best strategy to go around these problems was to find a global method whose efficiency depends only loosely on the particular region split mechanism. The probabilistic choice of actions makes the system robust to the potentially unnecessary split of homogeneous regions, and the multi-resolution scheme allows the system to be rather insensitive to the creation of an increasing number of small regions.

1.3 The Prediction Machine: incremental regression algorithms for learning forward and inverse models

The **IAC** and **R-IAC** system presented above are mostly agnostic regarding the kind of learning algorithm used to implement the prediction machine, i.e. used to learn forward models. The only property that is assumed is that learning must be incremental, since exploration is driven by measures of the improvement of the learnt forward models as new learning examplars are collected. But among incremental algorithm, methods based on neural networks, memory-based learning algorithms, or incremental statistical learning techniques could be used [43]. This agnosticity is an interesting feature of the system since it constitutes a single method to achieve active learning with multiple learning algorithms, i.e. with multiple kinds of learning biases that can be peculiar to each application domain, as opposed to a number of statistical active learning algorithms designed specifically for particular learning methods such as support vector machines, Gaussian mixture regression, or locally weighted regression [41]. Nevertheless, what the robot will learn eventually will obviously depend both on **IAC** or **R-IAC** and on the capabilities of the prediction machine/regression algorithm for which **IAC/R-IAC** drives the collection of learning exemplars.

In robot learning, a particular important problem is to learn the forward and inverse kinematics as well as the forward and inverse dynamics of the body [44,45,46,47]. A number of regression algorithms have been designed and experimented in this context in the robot learning literature, and because a particularly interesting use of **IAC/R-IAC** is for driving exploration for the discovery of the robot's body, as it will be illustrated in the experiments in the next section (and was already illustrated for **IAC** in [27,36]), it is useful to look at state-of-the-art statistical regression methods for this kind of space. An important family of such algorithms is locally weighted regression [45], among which Locally Weighted Projection Regression (LWPR) has recently showed a strong ability to learn incrementally and efficiently forward and inverse models in high-dimensional sensorimotor spaces [46,45]. Gaussian process regression has also proven to allow for very high generalization performances [48]. Another approach, based on Gaussian mixture regression [49,3], is based on the learning of the joint probability distribution of the sensorimotor variables, instead of learning a forward or an inverse model, and can be used online for inferring specific forward or inverse models by well-chosen projections of the joint density. Gaussian mixture regression (GMR) has recently shown a number of good properties for robot motor learning in a series of real-world robotic experiments [3]. It is interesting to note that these techniques come from advances in statistical learning theory, and seem to allow significantly higher performances than for example approaches based on neural-networks [50].

Because it is incremental and powerful, LWPR might be a good basic prediction algorithm to be used in the R-IAC framework for conducting robot experiments. Yet, LWPR is also characterized by a high number of parameters which tuning is not straightforward and thus makes its use not optimal for repeated experiments about **IAC/R-IAC** in various sensorimotor spaces. On the other hand, Gaussian processes and Gaussian mixture regression have much less parameters (only one parameter for GMR, i.e. the number of Gaussians) and are much easier to tune. Unfortunately, they are batch methods which can be computationally very demanding as the dataset grows. Thus, they cannot be used directly as prediction machines in the **IAC/R-IAC** framework.

This is why we have developed a regression algorithm, called ILO-GMR (Incremental Local Online Gaussian Mixture Regression) which mixes the ease of use of GMR with the incremental memory-based approach of local learning approaches. The general idea is to compute online local few-components GMM/GMR models based on the datapoints in memory whose values in the input point dimensions are in the vicinity of this input point. This local approach allows directly to take into account any novel single datapoint/learning exemplar added to the database since regression is done locally and online. It can be done computationally efficiently thanks to the use of few GMM components, and crucially thanks to the use of an incremental approximate nearest neighbor algorithm derived from recent batch-mode approximate nearest neighbor algorithms [51,52,53]. ILO-GMR has only two parameters: the number of components for local models, and a parameter that defines the notion of local vicinity. Another feature of ILO-GMR is that given its incremental and online nature, with a single set of parameters it can in principle approximate and adapt efficiently to a high variety of mapping to be learnt that may differ significantly in their length scale and might require differ. The technical details and comparison of performances of the ILO-GMR algorithm will be presented in a future paper. Initial experiments to learn the forward kinematics of 6 to 10 DOF's robotic arms have shown that ILO-GMR (tuned with the optimal number of components and vicinity) allows to reach prediction performances in generalization slightly worse than GMR (tuned with the optimal number of components) but similar to LWPR (tuned with the experimentally optimal parameters), the difference between LWPR and ILO-GMR being that ILO-GMR is much easier to tune but slower in prediction due to its only computational of local joint density models. Yet, for the 10 DOF systems of our experiments, these prediction times appear to be compatible with real-time control.

Learning forward motor models is mainly useful if it can be re-used for robot control, hence for inferring inverse motor models [46,48]. This brings up difficult challenges since most robotic systems are highly redundant, which means that the mapping from motor targets in the task space to motor commands in the joint/articulatory space is not a function: one target may correspond to many motor articulatory commands. This is why learning directly inverse models with standards regression algorithm is bound to fail in redundant robots, since when asked to find an articulatory configuration that yields a given target configuration, it will typically output the mean of accurate solutions which is itself not an accurate solutions. Fortunately, there are various approaches to go around this problem [46,48], and one of them is specific to the GMM/GMR approach [50], called the single component least square estimate (SLSE): because this approach encodes joint distributions rather than functions, redundancies are encoded in the GMM and inverse models can be computed by projecting the joint distribution on the corresponding output dimensions and then doing regression based only a the single Gaussian component that gives the highest posterior probability at the given input point. This approach is readily applicable in ILO-GMR, which we have done for the hand-eye-clouds experiment described below.

1.4 Self-organizing developmental trajectories with IAC and motor primitives in the Playground Experiment

In this section, we will present the Playground Experiment in which it is shown how the IAC system can drive the exploration and learning of motor primitives by an AIBO robot, and focus on the self-organization of behavioural developmental trajectories of increasing complexity. An extended presentation of these results is available in [27]. Further sections will then present experiments focused on the compared efficiency of **IAC** and **R-IAC** for active learning.

The Playground Experiment setup involves a physical robot as well as a complex sensorimotor system and environment. We use a Sony AIBO robot which is put on a baby play mat with various toys that can be bitten, bashed or simply visually detected (see figure 6). The environment is very similar to the ones in which two or three month old children learn their first sensorimotor skills, although the sensorimotor apparatus of the robot is here much more limited. We have developed a web site which presents pictures and videos of this set-up: http://playground.csl.sony.fr/.

1.4.1 Motor primitives

The robot is equipped initially with several parameterizable motor primitives that control its fore arms and its head. Its back legs are frozen such that it cannot walk around. There are three motor primitives: turning the head, bashing and crouch biting. Each of them is controlled by a number of real number parameters, which are the action parameters that the robot controls. The ``turning head" primitive is controlled with the pan and tilt parameters of the robot's head. The ``bashing" primitive is controlled with the strength and the angle of a whole leg movement (a lower-level automatic mechanism takes care of setting the individual motors controlling the leg and takes care of choosing which leg –left or right- is used depending on the angle parameter). The ``crouch biting" primitive is a complex movement consisting in sequencing a crouching with the robot chest while opening the mouth, and then closing the mouth. It is controlled by the depth of crouching (and the robot crouches in the direction in which it is looking at, which is determined by the pan and tilt parameters). To summarize, choosing an action consists in setting the parameters of the 5-dimensional continuous vector **M(t)**:

$$\mathbf{M(t)\}} = (\boldsymbol{p, t, bs, ba, d})$$

where ***p*** is the pan of the head, ***t*** the tilt of the head, ***bs*** the strength of the bashing primitive, ***ba*** the angle of the bashing primitive, and ***d*** the depth of the crouching of the robot for the biting motor primitive. All values are real numbers between 0 and 1, plus the value -1 which is a convention used for not using a motor primitive: for example, **M(t)** =(0.3, 0.95, -1, -1, 0.29) corresponds to the combination of turning the head with parameters ***p***=0.3 and ***t***=0.95 with the biting primitive with the parameter ***d***=0.29 but with no bashing movement.

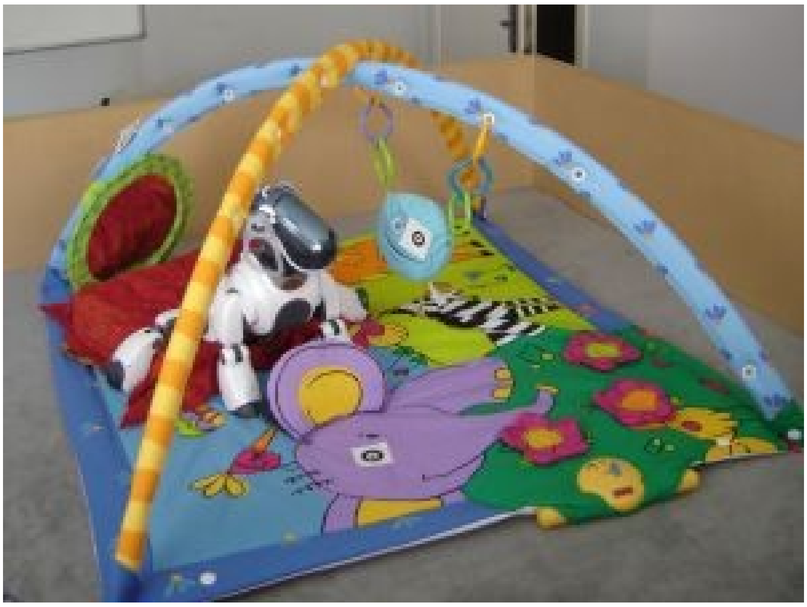


Fig. 6. The Playground Experiment setup

1.4.2 Perceptual primitives

The robot is equipped with three high-level sensors/perceptual primitives based on lower-level sensors. The sensory vector **S(t)** is thus 3-dimensional:

$$\mathbf{S(t)} = (\boldsymbol{Ov, Bi, Os})$$

where:

- ***Ov*** is the binary value of an object visual detection sensor: It takes the value 1 when the robot sees an object, and 0 in the other case. In the playground, we use simple visual tags that we stick on the toys and are easy to detect from the image processing point of view ;
- ***Bi*** is the binary value of a biting sensor: It takes the value 1 when the robot has something in its mouth and 0 otherwise. We use the cheek sensor of the AIBO;
- ***Os*** is the binary value of an oscillation sensor: It takes the value 1 when the robot detects that there is something oscillating in front of it, and 0 otherwise. We use the infra-red distance sensor of the AIBO to implement this high-level sensor. This sensor can detect for example when there is an object that has been bashed in the direction of the robot's gaze, but can also detect events due to human walking around the playground (we do not control the environment).

It is crucial to note that initially the robot knows nothing about sensorimotor affordances. For example, it does not know that the values of the object visual detection sensor are correlated with the values of its pan and tilt. It does not know that the values of the biting or object oscillation sensors can become 1 only when biting or bashing actions are performed towards an object. It does not know that some objects are more prone to provoke changes in the values of the ***Bi*** and ***Os*** sensors when only certain kinds of actions are performed in their direction. It does not know for example that to get a change in the value of the oscillation sensor, bashing in the correct direction is not enough, because it also needs to look in the right direction (since its oscillation sensors are on the front of its head). These remarks allow us to understand easily that a random strategy will not be efficient in this environment. If the robot would do random action selection, in a vast majority of cases nothing would happen (especially for the ***Bi*** and ***Os*** sensors).

1.4.3 The sensorimotor loop

The mapping that the robot has to learn is:

$$\mathbf{SM(t)} = (\boldsymbol{p, t, bs, ba, d}) \rightarrow \mathbf{S(t+1)} = (\boldsymbol{Ov', Bi', Os'})$$

The robot is equipped with the **IAC** system. In this experiment, the sensorimotor loop is rather long: when the robot chooses and executes an action, it waits that all its motor primitives have finished their execution, which lasts approximately one second, before choosing the next action. This is how the internal clock for the **IAC** system is implemented in this experiment. On the one hand, this allows the robot to make all the measures necessary for determining adequate values of **(*Ov, Bi, Os*)**. On the other hand and most importantly, this allows the environment to come back to its ``resting state''. This means that environment has no memory: after an action has been executed by the robot, all the objects are back in the same state. For example, if the object that can be bashed has actually been bashed, then it has stopped oscillating before the robots tries a new action. This is a deliberate choice to have an environment with no memory: while keeping all the advantages, the constraints and the complexity of a physical embodiment, this makes that mapping from actions to perception learnable in a reasonable time. This is crucial if one wants to do many experiments (already in this case, each experiment lasts for nearly one day). Furthermore, introducing an environment with memory frames the problem of the maximization of internal reward within delayed reward reinforcement problems, for which there exists powerful and sophisticated techniques whose biases would certainly make the results more advanced but would make it more difficult to understand the specific impact and properties of the intrinsic motivation system.

1.4.4 Results

During an experiment we continuously measure a number of features which help us characterize the dynamics of the robot's development. First, we measure the frequency of the different kinds of actions that the robot performs in a given time window. More precisely:

- The percentage of actions which do not involve the biting and the bashing motor primitive in the last 100 actions (i.e. the robot's action boils down to ``just looking" in a given direction).
- The percentage of actions which involve the biting motor primitive in the last 100 actions.
- The percentage of actions which involve the bashing motor primitive;

Then, we track the gaze of the robot and at each action measure if it is looking towards 1) the bitable object, or 2) the bashable object, or 3) no object. This is possible since from an external point of view we know where the objects are and so it is easy to derive the information from the head position.

Third, we measure the evolution of the frequency of successful biting actions and the evolution of successful bashing actions. A successful biting action is defined as an action which provokes a ``1" value on the ***Bi*** sensor (an object has actually be bitten). A successful bashing action is defined as an action which provokes an oscillation in the ***Os*** sensor.

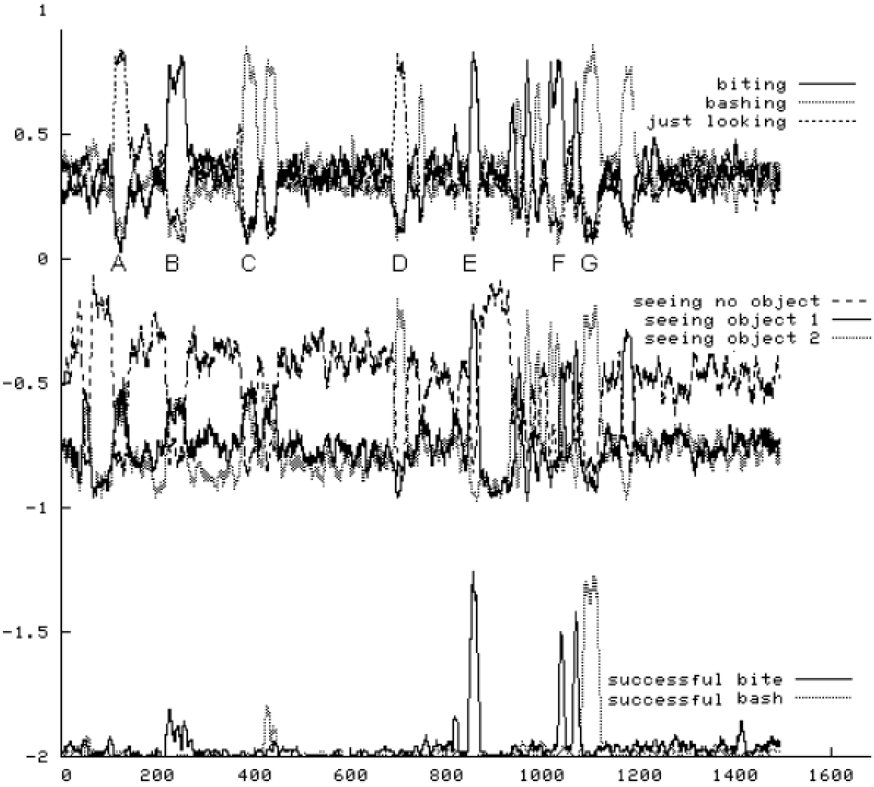


Fig. 7. Curves describing a run of the Playground Experiment.

Top 3: Frequencies for certain action types on windows 100 time steps wide.

Mid 3: Frequencies of gaze direction towards certain objects in windows 200 time steps wide: ``object 1" refers to the bitable object, and ``object 2" refers to the bashable object.

Bottom 3: Frequencies of successful bite ans successful bash in windows 200 time steps wide.

Figure 7 shows an example of result, showing the evolution of the three kinds of measures on three different levels. A striking feature of these curves is the formation of sequences of peaks. Each of these peaks means basically that at the moment it occurs the robot is focusing its activity and its attention on a small subset of the sensorimotor space. So it is qualitatively different from random action performance in which the curves would be stationary and rather flat. By looking in details at these peaks and at their co-occurence (or not) within the different kinds of measures, we can make a description of the evolution of the robot's behaviour. On figure 7, we have marked a number of such peaks with letters from A to G. We can see that before the first peak, there is an initial phase during which all actions are produced equally often, that most often no object is seen, and that a successful bite or bash only happens extremely rarely. This corresponds to a phase of random action selection. Indeed, initially the robot categorizes the sensorimotor space using only one big region (and so there is only one category), and so all actions in any contexts are equally interesting. Then we observe a peak (A) in the ``just looking" curve: this means that for a while, the robot stops biting and bashing, and focuses on just moving its head around. This means that at this point the robot has split the space into several regions, and that a region corresponding to the sensorimotor loop of ``just looking around" is associated to the highest learning progress from the robot's point of view. Then, the next peak (B) corresponds to a focus on the biting action primitive (with various continuous parameters), but it does not co-occur with the looking towards the bitable object. This means that the robot is trying to bite basically in all directions around him : it did not discover yet the affordances of the biting actions with particular objects. The next peak (C) corresponds to a focus on the bashing action primitive (with various continuous parameters) but again the robot does not look towards a particular direction. As the only way to discover that a bashing action can make an object move is by looking in the direction of this object (because the IR sensor is on the cheek), this means that the robot does not use at this point the bashing primitive with the right affordances. The next peak (D) corresponds to a period within which the robot stops again biting and bashing and concentrates on moving the head, but this times we observe that the robot focuses these ``looking" movement in a narrow part of the visual field : it is basically looking around one of the objects, learning how it disappears/reappears in its field of view. Then, there is a peak (E) corresponding to a focus on the biting action, which is this time coupled with a peak in the curve monitoring the looking direction towards the bitable object, and a peak in the curve monitoring the success in biting. It means that during this period the robot uses the action primitive with the right affordances, and manages to bite the bitable object quite often. This peak is then repeated a little bit later (F). Then finally a co-occurrence of peaks (G) appears that corresponds to a period during which the robot concentrates on using the bashing primitve with the right affordances, managing to actually bash the bashable object quite often.

This example shows that several interesting phenomena have appeared in this run of the experiment. First of all, the presence and co-occurrence of peaks of various kinds shows a self-organization of the behavior of the robot, which focuses on particular sensorimotor loops at different periods in time. Second, when we observe these peaks, we see that they are not random peaks, but show a progressive increase in the complexity of the behaviour to which they correspond. Indeed, one has to remind that the intrinsic dimensionality of the ``just looking" behaviour (pan and tilt) is lower than the ``biting" behaviour (which adds the depth of the crouching movement), which is itself lower than the ``bashing" behaviour (which adds the angle and the strength dimensions). The order of appearance of the periods within which the robot focuses on one of these activities is precisely the same. If we look in more details, we also see that the biting behaviour appears first in a non-affordant version (the robot tries to bite things which cannot be bitten), and then only later in the affordant version (where it tries to bite the biteable object). The same observation holds for the bashing behaviour: first it appears without the right affordances, and then it appears with the right affordances. The formation of focused activities whose properties evolve and are refined with time can be used to describe the developmental trajectories that are generated in terms of stages: indeed, one can define that a new stage begins when a co-occurence of peaks that never occured before happens (and so which denotes a novel kind of focused activity).

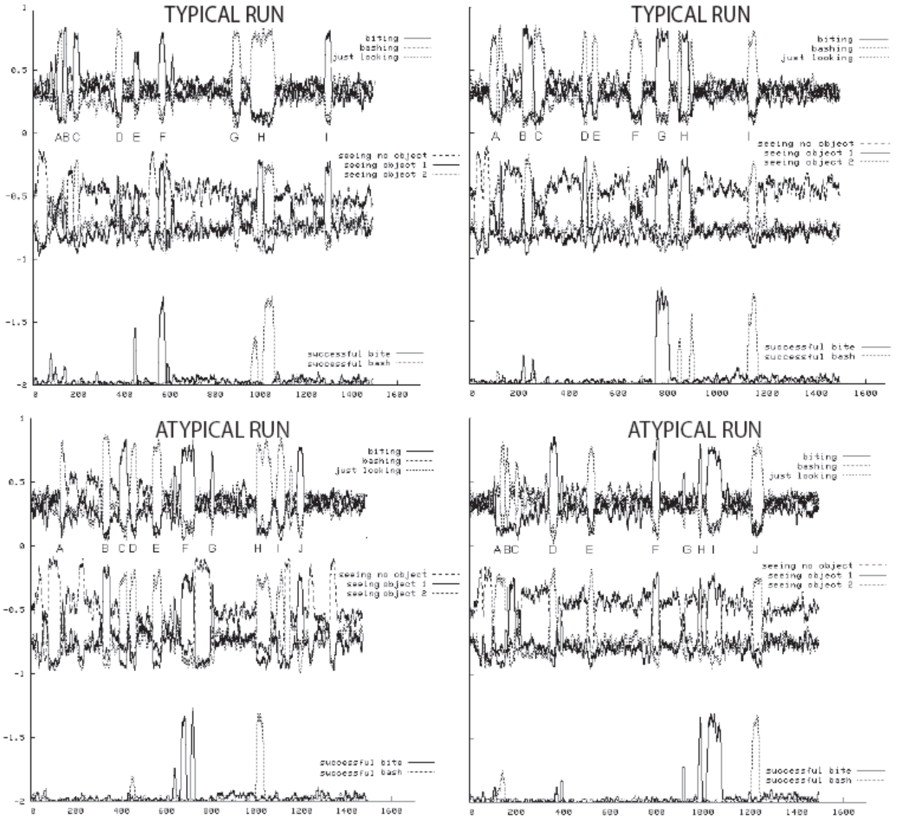


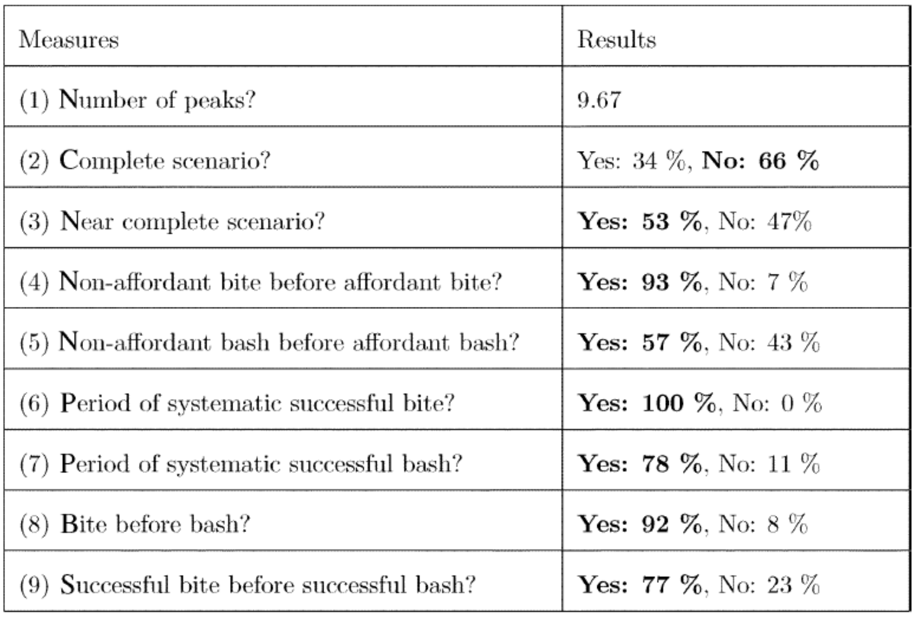
Fig. 8. Various runs of the simulated experiments. In the top squares, we observe two typical developmental trajectories corresponding to the ``complete scenario'' described by measure 1. In the bottom curve, we observe rare but existing developmental trajectories.

We ran several times the experiment with the real robots, and whereas each particular experiment produced curves which were different in the details, it seemed that some regularities in the patterns of peak formation, and so in terms of stage sequences, were present. We then proceeded to more experiments in order to assess precisely the statistical properties of these self-organized developmental trajectories. Because each experiment with the real robot lasts several hour, an in order to be able to run many experiments (200), we developed a model of the experimental set-up. Thanks to the fact that the physical environment was memoryless after each action of the robot, it was possible to make an accurate model of it using the following procedure: we let the robot perform several thousands actions and we recorded each time **SM(t)** and **S(t+1)**. Then, from this database of examples we trained a prediction machine based on locally weighted regression. This machine was then used as a model of the physical environment and the IAC algorithm of the robot was directly plugged into it.

Using this simulated world set-up, we ran 200 experiments, each time monitoring the evolution using the same measures as above. We then constructed higher-level measures about each of the runs, and based on the structure of the peak sequence. Peaks where here defined using a threshold on the height and width of the bumps in the curves. These measures correspond to the answer to these following questions:

- (Measure 1) **Number of peaks?**: How many peaks are there in the action curves (top curves) ?
- (Measure 2) **Complete scenario?**: Is the following developmental scenario matched: first there is a ``just looking" peak, then there is a peak corresponding to ``biting" with the wrong affordances which appears before a peak corresponding to ``biting" with the right affordances, and there is a peak corresponding to ``bashing" with the wrong affordances which appears before a peak corresponding to ``bashing" with the right affordance (and the relative order between ``biting"-related peaks and ``bashing"-related peaks is ignored). Biting with the right affordance is here defined as the co-occurence between a peak in the ``biting" curve and a peak in the ``seeing the biteable object" curve, and biting with the wrong affordances is defined as all other situations. The corresponding definition applies to ``bashing".
- (Measure 3) **Nearly complete scenario?**: Is the following less constrained developmental scenario matched: there is a peak corresponding to ``biting" with the wrong affordances which appears before a peak corresponding to ``biting" with the right affordances, and there is a peak corresponding to ``bashing" with the wrong affordances which appears before a peak corresponding to ``bashing" with the right affordances (and the relative order between ``biting"-related peaks and ``bashing"-related peaks is ignored).
- (Measure 4) **Non-affordant bite before affordant bite?**: Is there is a peak corresponding to ``biting" with the wrong affordances which appears before a peak corresponding to ``biting" with the right affordances?
- (Measure 5) **Non-affordant bash before affordant bash?**: there is a peak corresponding to ``bashing" with the wrong affordances which appears before a peak corresponding to ``bashing" with the right affordances?

- (Measure 6) **Period of systematic successful bite?** Does the robot succeeds systematically in biting often at some point (= is there a peak in the ``successful bite'' curve)?
- (Measure 7) **Period of systematic successful bash?** Does the robot succeeds systematically in bashing often at some point (= is there a peak in the ``successful bash'' curve?
- (Measure 8) **Bite before bash?** Is there a focus on biting which appears before a focus on bashing (independantly of affordance) ?
- (Measure 9) **Successful bite before successful bash?** Is there a focus on successfully biting which appear before a focus on successfully bashing ?



Measures	Results
(1) Number of peaks?	9.67
(2) Complete scenario?	Yes: 34 %, **No: 66 %**
(3) Near complete scenario?	**Yes: 53 %**, No: 47%
(4) Non-affordant bite before affordant bite?	**Yes: 93 %**, No: 7 %
(5) Non-affordant bash before affordant bash?	**Yes: 57 %**, No: 43 %
(6) Period of systematic successful bite?	**Yes: 100 %**, No: 0 %
(7) Period of systematic successful bash?	**Yes: 78 %**, No: 11 %
(8) Bite before bash?	**Yes: 92 %**, No: 8 %
(9) Successful bite before successful bash?	**Yes: 77 %**, No: 23 %

Table 1 Statistical measures on the 200 simulation-based experiments.

The numerical results of these measures are summarized in table 1. This table shows that indeed some structural and statistical regularities arise in the self-organized developmental trajectories. First of all, one has to note that the complex and structured trajectory described by Measure 2 appears in 34 percent of the cases, which is high given the number of possible co-occurences of peaks which define a combinatorics of various trajectories. Furthermore, if we remove the test on ``just looking'', we see that in the majority of experiments, there is a systematic sequencing from non-affordant to affordant actions for both biting and bashing. This shows an organized and progressive increase in the complexity of the behaviour. Another measure confirms this increase of complexity from another point of view: if we compare the relative order of appearance of periods of focused bite or bash, then we find that ``focused bite'' appears in the large majority of the cases before the ``focused bash", which corresponds to their relative intrinsic dimension (3 for biting and 4 for bashing). Finally, one can note that the robot reaches in 100 percent of the experiments a period during which it repeatedly manages to bite the biteable object, and in 78 percent of the experiments it reaches a period during which it repeatedly manages to bash the bashable object. This last point is interesting since the robot was not pre-programmed to achieve this particular task.

These experiments show how the intrinsic motivation system which is implemented (**IAC**) drives the robot into a self-organized developmental trajectory in which periods of focused sensorimotor activities of progressively increasing complexity arise. We have seen that a number of structural regularities arose in the system, such as the tendency of non-affordant behaviour to be explored before affordant behaviour, or the tendency to explore a certain kind of behaviour (bite) before another kind (bash). Yet, one has also to stress that these regularities are only statistical: two developmental trajectories are never exactly the same, and more importantly it happens that some particular trajectories observed in some experiments differ qualitatively from the mean. Figure 8 illustrate this point. The figures on the top-left and top-right corners presents runs which are very typical and corresponds to the ``complete scenario" described by Measure 1. On the contrary, the runs presented on the bottom-left and bottom-right corners corresponds to atypical results. The experiment of which curves are presented in the bottom-left corner shows a case where the focused exploration of bashing was performed before the focused exploration of biting. Nevertheless, in this case the regularity ``non-affordant before affordant" is preserved. On the bottom-right corner, we observe a run in which the affordant bashing activity appears very early and before any other focused activity. This balance between statistical regularities and diversity has parallels in infant sensorimotor development [60]: there are some strong structural regularities but from individual to individual there can be some substantial differences (for e.g. some infants learn how to crawl before they can sit and other do the reverse).

1.5 Experimenting and Comparing R-IAC and IAC With a Simple Simulated Robot

In this section, we describe the behavior of the **IAC** and **R-IAC** algorithms in a simple sensorimotor environment that allows us to show visually significant qualitative and quantitative differences, as well as compare them with random exploration.

1.5.1 Robotics configuration

We designed a simulated mechanical system, using the *Matlab robotics toolbox* [54]. It consists of a robotic arm using two degrees of freedom, represented by the two rotational axes q_1, q_2 as shown on figure 6. The upper part of the arm has been conceived as a bow, which creates a redundancy in the system: for each position and orientation of the tip of the arm, there are two corresponding possible articulatory/joint angle configurations.

This system's sensory system consists in a one-pixel camera, returning an intensity value $\boldsymbol{p}$, set on its extremity as shown on figure 8. The arm is put in a cubic painted environment $\boldsymbol{V}$, whose wallpapers are visible to the one-pixel camera, according to articulatory configurations.

Intensity values measured by the cameras are consequences of both environment $\boldsymbol{V}$ and rotational axes $\boldsymbol{q_1}, \boldsymbol{q_2}$. **So,** we can describe the system input/output mapping with two input dimensions, and one output as:

$$\boldsymbol{p} = \boldsymbol{V}(\boldsymbol{q_1}, \boldsymbol{q_2})$$

Thus, in this system the mapping to be learnt is state independent since here trajectories are not considered (only end positions are measured) and the perceptual result of applying motor joint angle commands does not depend on the starting configuration.

1.5.2 Environment configuration

The front wall consists of an increasing precision checker (Fig. 10), conceived with a black and white pattern. The designed ceiling contains animated wallpaper with white noise, returning a random value to the camera when this one is watching upward bound. Finally, other walls and ground are just painted in white (Fig. 9).

The set up of the system is such that we can sort three kinds of subregions in the sensorimotor space:

- The arm is positioned such that the camera is watching the front wall: for most learning algorithm, this subregion is rather difficult to learn with an increasing level of complexity from left to right (on fig. 7). This feature makes it particularly interesting to study whether **IAC** or **R-IAC** are able to spot these properties and control the complexity of explored sub-subregions accordingly.
- The arm is positioned such that the camera is watching the ceiling: the measured intensity values are random, and thus there are no correlations between motor configurations and sensory measures. Hence, once a few statistical properties of the sensory measures have potentially been learnt (such as the mean), nothing more can be learnt and thus no learning progress can happen.
- The arm is positioned such that a white wall is in front of the camera: the measured intensity value is always 0, so the input/output correlation is trivial. Thus, after it has been learnt that intensity values are constant in this area, nothing can be further learnt.

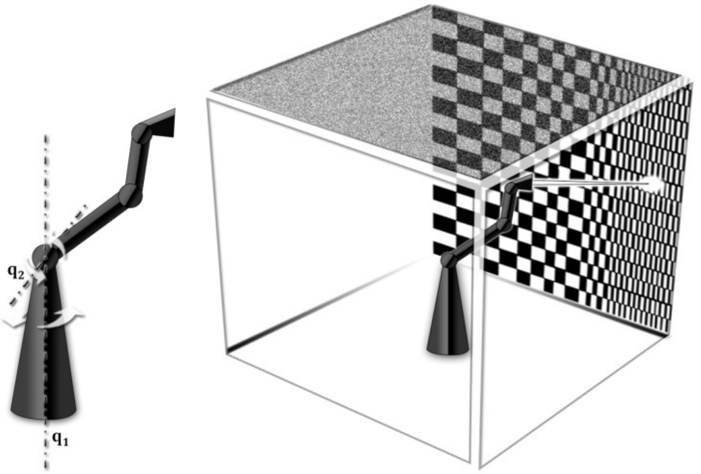


Fig. 9. Representation of a 2 axes arm, with a one pixel camera mounted on its extremity. This arm is put in the center of a cubic room, with different painted walls of different complexities.

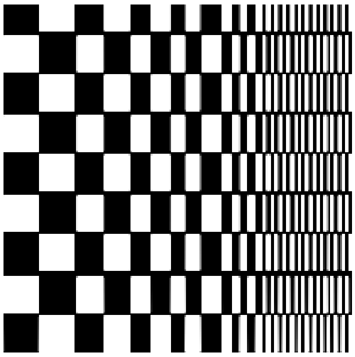


Fig. 10. Wallpaper disposed in the front wall. For many learning algorithms, the complexity increases from left to

Because the system has just two motor dimensions and one sensory dimension, it can be visualized using a 2D projection on a plane such as in figure 11. This projection shows a central vertical zone corresponding to the dynamic noise projected on the ceiling. Then, we can easily distinguish the front wall, represented on both sides of the noisy area, because of the redundancy of the arm. The remaining white parts correspond to other walls and the floor.

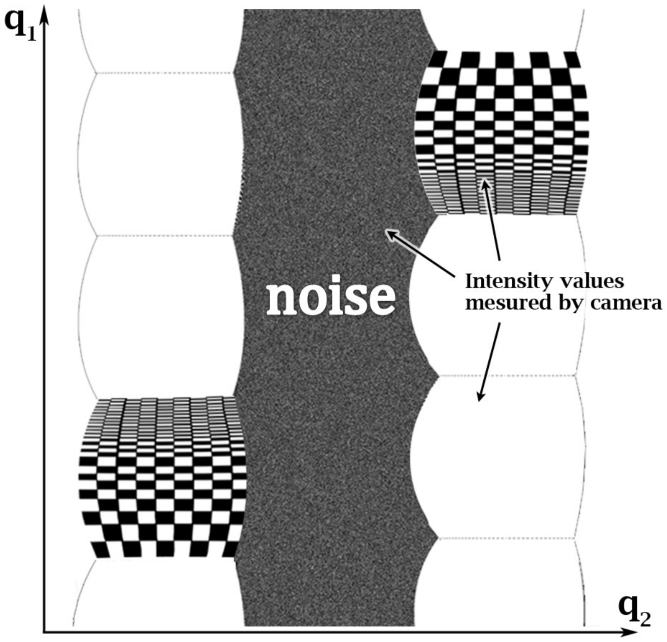


Fig. 11. 2D visualization of the sensorimotor space of the robot, with two motor dimensions one sensory dimension.

1.5.3 Results: Exploration Areas

First, it is interesting to perform qualitative comparisons of the exploration behavior generated by random exploration, **IAC** exploration and **R-IAC** exploration methods.

For each exploration method, the system is allowed to explore its sensorimotor space through 20000 sensorimotor experiments, i.e. it is allowed to collect 20000 learning exemplars. During each run of a given method, every 2000 sensorimotor experiments made by the system one computes a 2D smoothed histogram representing the distribution of explored sensorimotor configurations in the last 500 sensorimotor experiments. This allows us to visualize the evolution of the exploration focus, over time, for each system.

Random exploration obviously leads to a flat histogram.

Fig. 12 presents typical results obtained with **R-IAC** (on the left) and **IAC** (on the right), on a grey scale histogram where darker intensities denote low exploration focus and lighter intensities denote higher exploration focus. First, we observe that **R-IAC** is focalizing on the front wall, containing the image of the checker, using its two possible redundant exploration positions. It avoids the region which contains the white noise, and also the regions just containing a white color. In contrast, we cannot observe the same accuracy to concentrate sensorimotor experiments over interesting areas with the **IAC** exploration method. Here, the algorithm is indeed avoiding the noise, but we cannot observe precisely some interest toward the front wall, and the system seems to find some things to learn in the back wall, as we can see, watching the bottom-right part of the two last images.

The histograms in figure 12 were smoothed with a large spatial frequency filter to allow us to visualize well the global exploratory behavior. Nevertheless, it is also interesting to use a smaller spatial frequency smoother in order to zoom in and visualize the details of the exploration behavior in the front wall region. Fig. 13 shows a typical result obtained with **R-IAC**, just considering exemplars performed watching the front wall in the bottom-left side of the 2D projection. This sequence shows very explicitly that the system first focuses exploration on zones of lower complexity and progressively shifts its exploration focus towards zones of higher complexity. The system used is thus here able to evaluate accurately the different complexities of small parts of the world, and to drive the exploration based on this evaluation.

1.5.4 Results: Active Learning

We can now compare the performances of random exploration, **IAC** exploration and **R-IAC** exploration in terms of their efficiency for learning as fast as possible the forward model of the system. For the **R-IAC** method, we included here a version of **R-IAC** without the multi-resolution scheme to assess the specific contribution of multi-resolution learning progress monitoring in the results.

For each exploration method, 30 experiments were run in order to be able to measure means and standard deviations of the evolution of performances in generalization. In each given experiment, every 5000 sensorimotor experiment achieved by the robot, we freezed the system and tested its performances in generalization for predicting $\boldsymbol{p}$ from $(\boldsymbol{q_1}, \boldsymbol{q_2})$ on a test database generated beforehand and independently consisting of random uniform queries in the sensorimotor subspace where there are learnable input/output correlations (i.e. excluding the zone with white noise). Results are provided on figure 14. As we can easily observe, and as already shown in [27], using **IAC** leads to learning performances that are statistically significantly higher than with **RANDOM** exploration. Yet, as figure 14 shows, results of **R-IAC** are statistically significantly much higher than **IAC**, and the difference between **IAC** and **R-IAC** is larger than between **IAC** and random exploration. Finally, we observe that including the multi-resolution scheme into **R-IAC** provides a clear improvement over **R-IAC** without multi-resolution, especially in the first half of the exploration trajectory where inappropriate or too early region splits can slow down the efficiency of exploration if only leaf regions are taken into account for region selection.

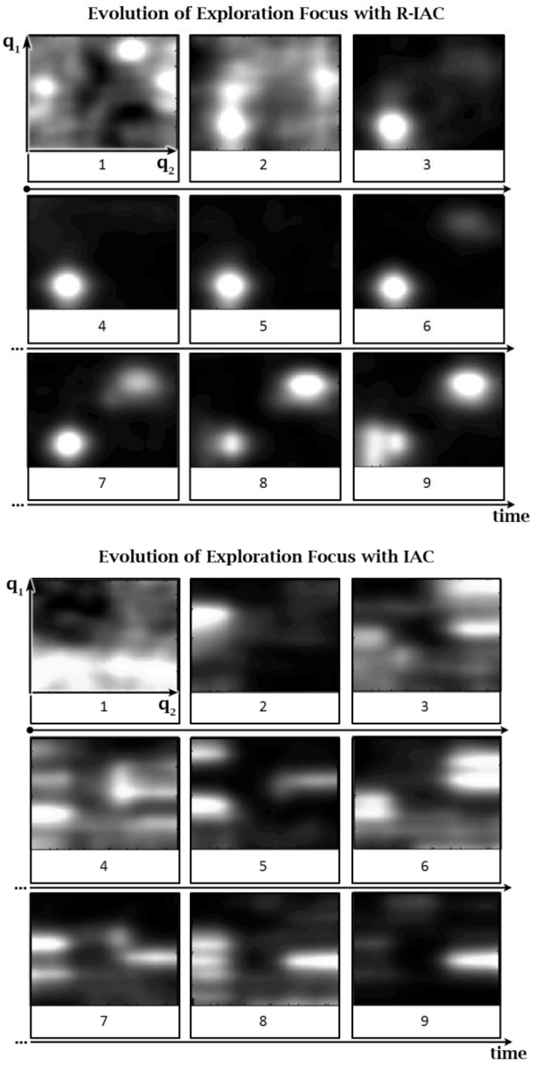


Fig. 12. Evolution of the exploration focus when using **R-IAC** as an exploration heuristics (left) or **IAC** (right). Each square represents the smoothed distribution of explored motor configurations at different times in a given run and over a sliding time window. Darker intensities denote low exploration focus and lighter intensities denote higher exploration focus. We observe that **R-IAC** leads the system to explore preferentially motor configurations such that the camera is looking at the checkerboard, while avoiding zones that are trivial to learn or unlearnable zones. On the contrary, **IAC** is unable to organize exploration properly and “interesting” zones are much less explored.

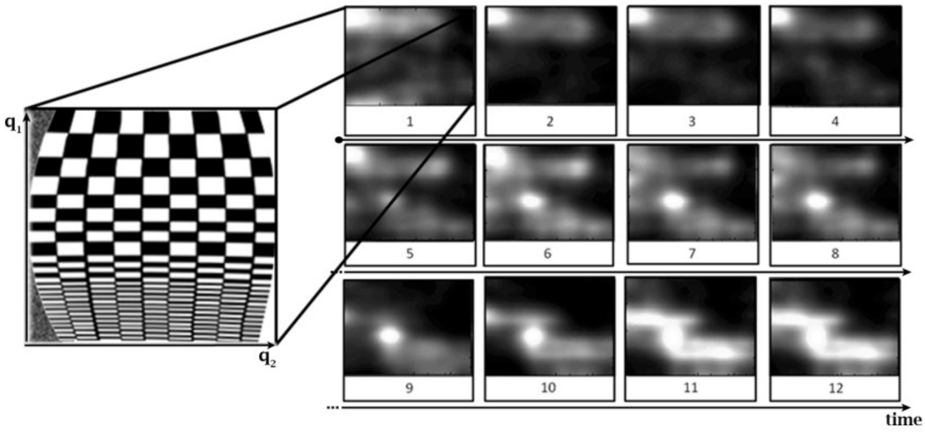


Fig. 13. A zoom into the evolution of the distribution of explored sensorimotor experiments in one of the two subregions where the camera is looking at the checkerboard when **R-IAC** is used. We observe that exploration is first focused on zones of the checkerboard that have a low complexity (for the given learning algorithm), and progressively shifts towards zones of increasing complexity.

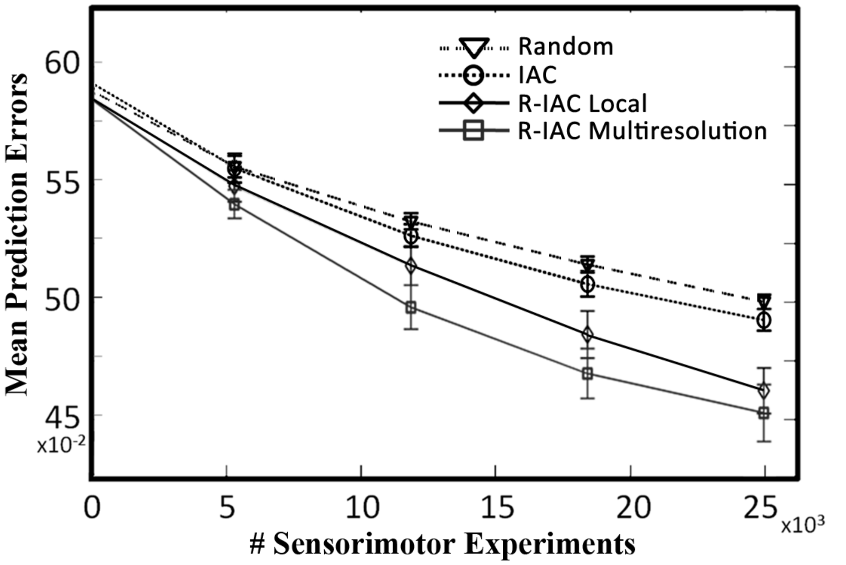


Fig. 14. Comparison of performances of the first and two new implementation of IAC, compared with the random exploration approach.

1.6 The Hand-Eye-clouds Experiment

We will now compare the performances of **IAC** and **R-IAC** as active learning algorithms to learn a forward model in a more complex 6-dimensional robotic sensorimotor space that includes large unlearnable zones. Both algorithms will also be compared with baseline random exploration.

1.6.1 Robotics Configuration

In this experiment, a simulated robot has two 2-D arms, each with two links and two revolute joints whose angles are controlled by motor **inputs** $\boldsymbol{q_{11}}$, $\boldsymbol{q_{12}}$, $\boldsymbol{q_{21}}$, $\boldsymbol{q_{22}}$ (see figure 15). On the tip of one of the two arms is attached a square camera capable to detect the sensory position $(\boldsymbol{x},\boldsymbol{y})$ of point-blobs relative to the square. These point-blobs can be either the tip of the other arm or clouds in the sky (see figure 15). This means that when the right arm is positioned such that the camera is over the clouds, which move randomly, the relation between motor configurations and perception is quasi-random. If on the contrary the arms are such that the camera is on top of the tip of the other arm, then there is an interesting sensorimotor relationship to learn. Formally, the system has the relation:

$$(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{E}(\boldsymbol{q_{11}},\boldsymbol{q_{12}}, \boldsymbol{q_{21}},\boldsymbol{q_{22}})$$

where $(\boldsymbol{x},\boldsymbol{y})$ is computed as follows:

(1) The camera is placed over the white wall: nothing has been detected: $(\boldsymbol{x},\boldsymbol{y}) =$ (-10, -10);

(2) The camera is on top of the left hand: the value $(\boldsymbol{x},\boldsymbol{y})$ of the relative position of the hand in the camera referential $\boldsymbol{C}$ is taken. According to the camera size, the **x** and **y** values are in the interval [0; 6];

(3) The camera is looking at the window: Two random values $(\boldsymbol{x},\boldsymbol{y})$ playing the role of random clouds displacement are chosen for output. The interval of outputs corresponds to camera size.

(4) The camera is looking at the window and sees both hand and cloud: the output value $(\boldsymbol{x},\boldsymbol{y})$ is random, like if just a cloud had been detected.

This setup can be thought to be similar to the problems encountered by infants discovering their body: they do not know initially that among the blobs moving in their field of view, some of them are part of their "self" and can be controlled, such as the hand, and some other are independent of the self and cannot be controlled (e.g. cars passing in the street or clouds in the sky).

Thus, in this sensorimotor space, the "interesting" potentially learnable subspace is next to a large unlearnable subspace, and also next to a large very simple subspace (when the camera is looking neither to the clouds not to the tip of the other arm).

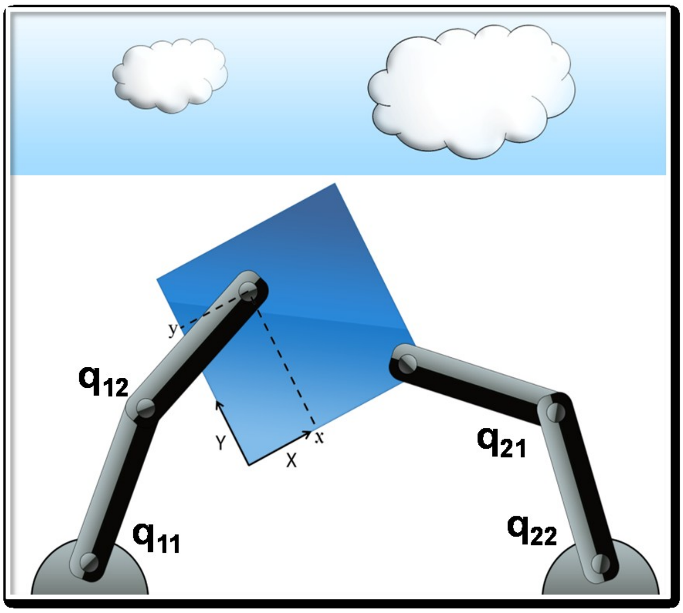


Fig. 15. Experimental setup. The 2D robot has two arms, each with two links and two revolute joints. At the tip of the right arm is rigidly attached a square camera/eye which can sense either the position of the tip of the other arm in its own referential (X, Y) if it is above it, but which can also sense the position of randomly moving clouds when the right arm motor configuration is such that the camera is looking over the top grey area (the « window »). When the camera senses something, the robot does not know initially whether this corresponds to the tip of its left arm or to a cloud. In subregions corresponding to the first alternative, the motor/sensor mapping is correlated and a lot can be learnt. In subregions corresponding to the second alternative, there are no correlations between motors and sensors and nothing can be learnt except some basic statistical properties of the random movement of clouds. There is a third alternative, which actually happens most of the time if the joint space is sampled randomly: the camera looks below the window but does not see its left arm tip. In this very large subregion, the motor to sensor mapping is trivial.

1.6.2 Results

In these experiments, the parameters of **IAC** and **R-IAC** are $\boldsymbol{Tsplit} = 250$, the learning progress window is 50, $\boldsymbol{p_1}$ = 0.3, $\boldsymbol{p_2} = 0.6$, $\boldsymbol{p_3} = 0.1$. Experiments span a duration of 100000 sensorimotor experiments. The incremental learning algorithm that is used to learn the forward model is the ILO-GMR system described in section 3.

A first study of what happens consists in monitoring the distance between the center of the eye (camera), and the hand (tip of the other arm). A small distance means that the eye is looking the hand, and a high, that it is focusing on clouds (noisy part) or on the white wall. Fig. 16 shows histograms of these distances. We first observe the behavior of the Random exploration algorithm. The curve shows that the system is, in majority, describing actions with a distance of 22, corresponding to the camera looking at clouds or at the white wall. Interestingly, the curve of the **IAC** algorithm is similar but slightly displaced towards shorter distance: this shows that **IAC** pushed the system to explore the "interesting" zone a little more. We finally observe that **RIAC** shows a large difference with both **IAC** and random exploration: the system spends three times more time in a distance inferior to 8, i.e. exploring sensorimotor configurations in which the camera is looking at the other arm's tip. Thus, the difference between **R-IAC** and **IAC** is more important than the difference between **IAC** and random exploration.

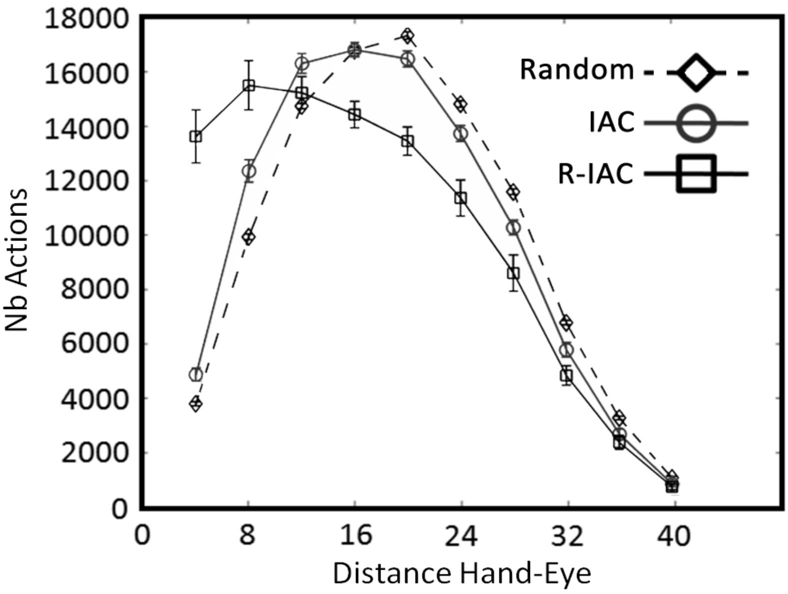


Fig. 16. Mean distributions of hand-center of eye distances when exploration is random, guided by **IAC**, or guided by **R-IAC**. We observe that while **IAC** pushes the system to explore slightly more than random exploration the zones of the sensorimotor space where the tip of the left arm is perceived by the camera or near the camera, **R-IAC** is significantly more efficient than **IAC** for driving exploration in the "interesting" area.

Then, we evaluated the quality of the learnt forward model using the three exploration algorithms. We considered this quality in two respects: 1) the capability of the model to predict the position of the hand in the camera given motor configurations for which the hand is within the field of view of the robot; 2) the capacity to use the forward model to control the arm: given a right arm configuration and a visual objective, we tested how far the forward model could be used to drive the left arm to reach this visual objective with the left hand.The first kind of evaluation was realized by first building independantly a test database of 1000 random motor configurations for which the hand is within the field of view, and then using it for testing the learnt models built by each algorithm at various stages of their lifetime (the test consisted in predicting the position of the hand in the camera given joint configurations). Thirty simulations were run, and the evolution of mean prediction errors is shown on the right of figure 17. The second evaluation consisted in generating a set of $\{(\boldsymbol{x},\boldsymbol{y})_C, \boldsymbol{q_{21}},\boldsymbol{q_{22}}|\boldsymbol{x} > 0\ and\ \boldsymbol{y} > 0\}$ values that are possible given the morphology of the robot, and then use the learnt forward models to try to move the left arm, i.e. find $(\boldsymbol{q_{11}},\boldsymbol{q_{12}})$ to reach the $(\boldsymbol{x},\boldsymbol{y})_{\boldsymbol{C}}$ objectives corresponding to particular $\boldsymbol{q_{21}},\boldsymbol{q_{22}}$ values**.** Control was realized through inferring an inverse models using ILO-GMR as presented in section 3. The distance between the reached point and the objective point was each time measured, and results, averaged over 30 simulations, are reported in the left graph of figure 17.

Both curves on figure 17 confirm clearly the qualitative results of the previous figure: **R-IAC** outperforms significantly **IAC**, which is only slighlty better than random exploration. We have thus shown that **R-IAC** is much more efficient in such an example of complex inhomogeneous sensorimotor space. We also illustrate on figure 18 configurations obtained, considering fixed goals $\{(\boldsymbol{x},\boldsymbol{y})_C, \boldsymbol{q_{21}},\boldsymbol{q_{22}}\}$, and estimated positioning of the left hand.

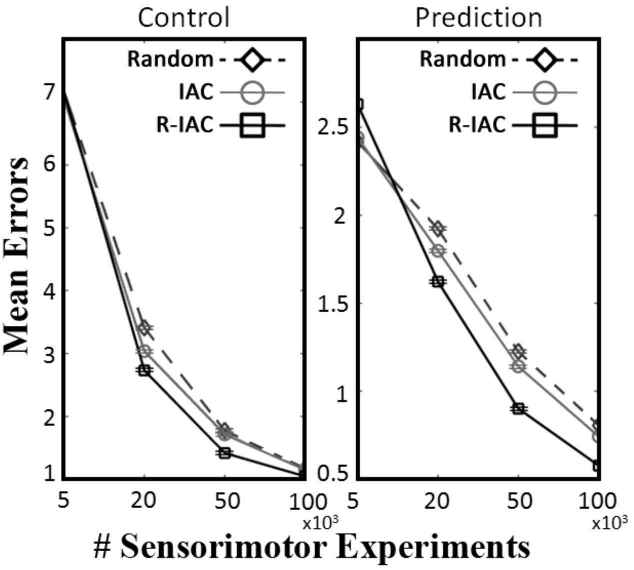


Fig. 17 Left: evolution of performances in control based on the forward model learnt through Random, IAC, and R-IAC exploration heuristics, averaged over 30 simulations. Right : evolution of the generalization capabilities of the learnt forward model with Random, IAC, and R-IAC explo., av. over 30 simulations.

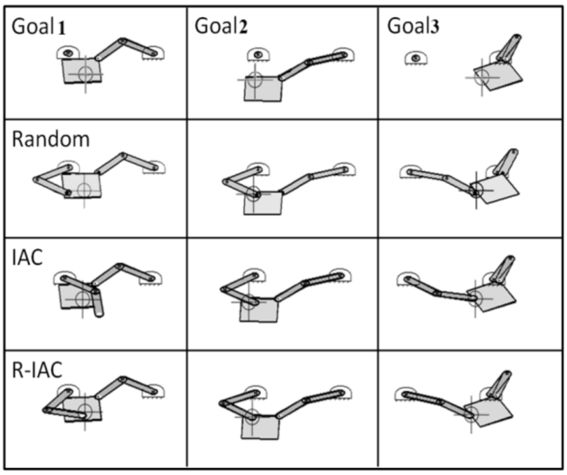


Fig. 18. Examples of performances obtained in control. The first row corresponds to goals fixed. Here, values fixed are the joints of the right hand, and the position in the referential of its eye. The challenge consists of reaching the target (position fixed in the eye) with the left arm.

1.7 Conclusion

In this chapter, we have presented two computational intrinsic motivation systems, **IAC** and **R-IAC**, that share substantial properties with human intrinsic motivation systems. As for humans, we have shown that they can be successfully used to self-organize complex sensorimotor developmental trajectories in learning robots, with the formation of stages of increasing complexity. This opens in return new modelling insigths to understand better developmental dynamics in humans [27]. Furthermore, thanks to their capacity to actively regulate the growth of complexity in the exploration process, we have shown that these systems can also be very efficient to drive the motor learning of forward and inverse models in spaces which contain large subregions that are either trivial or unlearnable. For this kind of sensorimotor spaces, typically encountered by developmental robots, we have explained why these intrinsic motivation systems, which we may call developmental active learning systems, will be much more efficient than more traditional active learning heuristics based on the maximization of uncertainty or unpredictability.

Furthermore, we have introduced a novel formulation of **IAC**, called **R-IAC**, and shown that its performances as an intrinsically motivated active learning algorithm were far superior to **IAC** in a complex sensorimotor space where only a small subspace was interesting. We have also shown results in which the learnt forward model was reused in a control scheme.

Further work will study extensions of the current results in several directions. First, experiments with **R-IAC** presented in this chapter were achieved in simulated robots. In spite of the fact that **IAC** was already evaluated in high-dimensional real robotic systems [27,36,34], these experiments were focusing on the self-organization of patterns in developmental trajectories. Evaluating **IAC** and **R-IAC** as active learning methods in high-dimensional real sensorimotor robotic spaces remains to be achieved. Second, both **IAC** and **R-IAC** heuristics could also be conceptualized as mechanisms for generating internal immediate rewards that could serve as a reward system in a reinforcement learning framework, such as for example in intrinsically motivated reinforcement learning [28,33,35]. Leveraging the capabilities of advanced reinforcement learning techniques for sequential action selection to optimize cumulated rewards might allow **IAC** and **R-IAC** to be successfully applied in robotic sensorimotor spaces where dynamical information is crucial, such as for example for learning the forward and inverse models of a force controlled high-dimensional robot, for which guided exploration has been identified as a key research target for the future [47,48].

Also, as argued in [55], it is possible to devise “competence-based” intrinsic motivation systems in which the measure of interestingness characterizes goals in the task space rather than motor configurations in the motor/joint space such as in knowledge-based intrinsic motivation systems like **IAC** or **R-IAC.** We believe that a competence based version of **R-IAC** would allow us to increase significantly exploration efficiency in massively redundant sensorimotor spaces. Finally, an issue of central importance to be studied in the future is how intrinsically motivated exploration and learning mechanisms can be fruitfully coupled with social learning mechanisms, which would be relevant not only for motor learning [56,57,58], but also for developmental language learning grounded in sensorimotor interactions [59].

Acknowledgements

The work presented here was partly realized and supported by the Sony CSL Laboratory, Paris, France (the Playground Experiment in particular), and partly realized and supported by INRIA.

References

[1] J. Weng, J. McClelland, A. Pentland, O. Sporns et al, “Autonomous mental development by robots and animals”, *Science*, vol. 291, pp. 599–600, 2001.

[2] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini, “Developmental robotics: A survey”, *Connection Sci.*, vol. 15, no. 4, pp. 151–190, 2003.

[3] S. Calinon, F. Guenter, and A. Billard, “On Learning, Representing and Generalizing a Task in a Humanoid Robot”. *IEEE Transactions on Systems*, Man and Cybernetics, Part B, Special issue on robot learning by observation, demonstration and imitation, 37:2, 286-298, 2007.

[4] M. Lopes, F.S. Melo, L. Montesano, (2007) “Affordance-based imitation learning in robots”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1015–1021. USA, 2007.

[5] P. Abbeel, A.Y. Ng, “Apprenticeship learning via inverse reinforcement learning”. In: *Proceedings of the 21st International Conference on Machine Learning* (ICML’04), pp. 1–8, 2004.

[6] C.G. Atkeson and S. Schaal, “Robot learning from demonstration”. In *Proc. 14th International Conference on Machine Learning*, pp. 12–20. Morgan Kaufmann, 1997.

[7] A. Alissandrakis, C.L. Nehaniv, K. Dautenhahn, “Action, state and effect metrics for robot imitation”. In: *15th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN 06)*, pp. 232–237. Hatfield, United Kingdom, 2006.

[8] B. Argall, S. Chernova, M. Veloso, “A survey of robot learning from demonstration”. *Robotics and Autonomous Systems* 57(5), 469–483, 2009.

[9] M. Asada, M. Ogino, S. Matsuyama, J. Oga, “Imitation learning based on visuo-somatic mapping”. In: *O.K. Marcelo*, H. Ang (eds.) 9th Int. Symp. Exp. Robot.,, vol. 21, pp. 269–278. Springer-Verlag, Berlin, Germany, 2006.

[10] P. Andry, P. Gaussier, S. Moga, J.P. Banquet and J. Nadel, “Learning and communication via imitation: an autonomous robot perspective”. *IEEE Transactions on Systems*, *Man, and Cybernetics*, Part A 31(5): 431-442, 2001.

[11] Y. Demiris and A. Meltzoff, “The Robot in the Crib: A developmental analysis of imitation skills in infants and robots”, *Infant and Child Development*, 17:43-53, 2008.

[12] M. Pardowitz S. Knoop, R.D. Zollner, R.Dillmann, “Incremental learning of tasks from user demonstrations, past experiences, and vocal comments*”. IEEE Transactions on Systems, Man and Cybernetics* - Part B 37(2), 322–332, 2007.

[13] E. Oztop, M. Kawato, M. Arbib, “Mirror neurons and imitation: A computationally guided review”. *Neural Networks* 19(3), 254–271, 2006.

[14] R. Rao, A. Shon, A. Meltzoff, “Imitation and social learning in robots, humans, and animals”, chap. *A Bayesian model of imitation in infants and robots*. Cambridge University Press, 2007.

[15] R.C. Arkin, (2005) "Moving Up the Food Chain: Motivation and Emotion in Behavior-based Robots", in *Who Needs Emotions: The Brain Meets the Robot*, Eds. J. Fellous and M.~Arbib, Oxford University Press, 2005.

[16] J.M. Fellous and M. Arbib, (eds), "Who Needs Emotions: The Brain Meets the Robot", *Oxford University Press*, 2005.

[17] D. McFarland, T. Bosser, "Intelligent Behavior in Animals and Robots", *MIT Press*, Cambridge, MA, 1993.

[18] R. Manzotti, V. Tagliasco, "From behaviour-based robots to motivation-based robots". *Robot. Auton. Syst*. 51, No. 2-3, 175-190, 2005.

[19] A. Stoytchev, R. Arkin, "Incorporating Motivation in a Hybrid Robot Architecture". *JACIII 8*(3): 269-274, 2004.

[20] R.C. Arkin, M. Fujita, T. Takagi and R. Hasegawa "An ethological and emotional basis for human-robot interaction", *Robotics and Autonomous Systems*, Volume 42, Number 3, pp. 191-201(11), 2003.

[21] R. White, "Motivation reconsidered: The concept of competence. Psychological", 66:297–333, 1959.

[22] D. Berlyne, "Curiosity and Exploration", *Science,* Vol. 153. no. 3731, pp. 25 – 33, 1966.

[23] E. Deci and R. Ryan, "Intrinsic Motivation and Self-Determination in Human Behavior". *Plenum Press*, 1985.

[24] W. Schultz, "Getting Formal with Dopamine and Reward", *Neuron*, Vol. 36, pp. 241-263, 2002.

[25] P. Dayan and B. Balleine, "Reward, Motivation and Reinforcement Learning", *Neuron*, Vol. 36, pp. 285-298, 2002.

[26] P. Redgrave and K. Gurney, "The Short-Latency Dopamine Signal: a Role in Discovering Novel Actions?", *Nature Reviews Neuroscience*, Vol. 7, no. 12, pp. 967-975, 2006.

[27] P-Y. Oudeyer, F. Kaplan and V. Hafner, "Intrinsic Motivation Systems for Autonomous Mental Development", *IEEE Transactions on Evolutionary Computation*, 11(2), pp. 265—286, 2007.

[28] A. Barto, S. Singh and N. Chentanez, "Intrinsically motivated learning of hierarchical collections of skills", *in Proc. 3rd Int. Conf. Development Learn*., San Diego, CA, 2004, pp. 112–119, 2004.

[29] A. Blanchard and L. Cañamero, "Modulation of Exploratory Behavior for Adaptation to the Context". *Biologically Inspired Robotics (Biro-net) in AISB'06 : Adaptation in Artificial and Biological Systems,* Bristol UK, 2006.

[30] R. Der, M. Herrmann, R. Liebscher, "Homeokinetic approach to autonomous learning in mobile robots". *In Robotik* 2002; Dillman, R., Schraft, R. D., W¨orn, H., Eds.; VDI: Dusseldorf, Germany; pp. 301-306, 2002.

[31] D.S. Blank, D. Kumar, L. Meeden, and J. Marshall, "Bringing up robot: Fundamental mechanisms for creating a self-motivated, self-organizing architecture". *Cybernetics and Systems*, 36(2), 2005.

[32] X. Huang and J. Weng, "Novelty and Reinforcement Learning in the Value System of Developmental Robots", *in Proc. Second International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, Edinburgh, Scotland, August 10 - 11, 2002.

[33] J. Schmidhuber, "Curious model-building control systems", in *Proc.Int. Joint Conf. Neural Netw.*, Singapore, 1991, vol. 2, pp. 1458–1463, 1991.

[34] P-Y. Oudeyer, F. Kaplan, "Discovering Communication", *Connection Science*, 18(2), pp. 189—206, 2006.

[35] M. Schembri, M. Mirolli, G. Baldassarre, "Evolution and Learning in an Intrinsically Motivated Reinforcement Learning Robot". *ECAL* 2007: 294-303, 2007.

[36] F. Kaplan and P-Y Oudeyer, "Intrinsically Motivated Machines", in M. Lungarella and Iida, F, Bongard, J,. and Pfeifer, R. (Eds.): 50 Years of AI, Festschrift, LNAI 4850, pp. 304–315, 2007.

[37] V. Fedorov, "Theory of Optimal Experiment". New York, NY: Academic, 1972.

[38] E. J. Gibson, *Principles of perceptual learning and development*. New-York: Appleton-Century-Crofts, 1969.

[39] D. Berlyne, *Conflict, Arousal, and Curiosity*. New York: McGraw-Hill, 1960.

[40] M. Csikszentmihalyi, *Creativity-Flow and the Psychology of Discovery and Invention.* New York: Harper Perennial, 1996.

[41] D. Cohn, Z. Ghahramani and M. Jordan, "Active learning with statistical models", *J. Artif. Intell. Res*., vol. 4, pp. 129–145, 1996.

[42] M. Hasenjager and H. Ritter, “Active Learning in Neural Networks”. In: *New learning paradigms in soft computing*. Berlin, Germany: Physica-Verlag GmbH, pp. 137–169, 2002.

[43] R.O. Duda , P.E. Hart and D.G. Stork, *Pattern Classification,* Wiley, 2006.

[44] S. Vijayakumar and S. Schaal, “LWPR : An O(n) Algorithm for Incremental Real Time Learning in High Dimensional Space”, *Proc. of Seventeenth International Conference on Machine Learning (ICML2000)* Stanford, California, pp.1079-1086, 2000.

[45] A. D'Souza, S. Vijayakumar, S. Schaal, “Learning inverse kinematics”, *IEEE International Conference on Intelligent Robots and Systems (IROS 2001)*, Piscataway, NJ: IEEE, 2001.

[46] J. Peters, S. Schaal, “Learning to control in operational space”, *International Journal of Robotics Research*, 27, pp.197-212, 2008.

[47] C. Salaün, V. Padois and O. Sigaud, “Control of redundant robots using learned models : an operational space control approach”, *IEEE International Conference on Intelligent Robots and Systems (IROS 2009),* 2009.

[48] D.Y. Yeung and Y. Zhang, “Learning inverse dynamics by Gaussian process regression under the multi-task learning framework”. *In The Path to Autonomous Robots*, G.S. Sukhatme (ed.), pp.131-142, Springer, 2009.

[49] Z. Ghahramani, “Solving inverse problems using an EM approach to density estimation”, in M. C. Mozer, P. Smolensky, D.S. Toureztky, J.L. Elman, A.S. Weigend, *Proceedings of the 1993 Connectionist Models Summer School*, pp. 316—323, Hillsdale, NJ : Erlbaum Associates, 1993.

[50] C. E. Rasmussen, “Evaluation of Gaussian Process and other Methods for Non-linear Regression”. *PhD thesis*, Department of Computer Science, University of Toronto, 1996.

[51] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, “An Optimal Algorithm for Approximate Nearest Neighbor Searching”, *Journal of the ACM*, 45, 891-923, 1998.

[52] S. Maneewongvatana and D. M. Mount, “Analysis of Approximate Nearest Neighbor Searching with Clustered Point Sets, Data Structures, Near Neighbor Searches, and Methodology”: *Fifth and Sixth DIMACS Implementation Challenges,* eds. M. H. Goldwasser, D. S. Johnson, C. C. McGeoch, in the DIMACS Series in Discr. Math. and Theoret. Comp. Sci., Vol. 59, AMS, 2002, 105-123, 2002.

[53] D. Filliat, “A visual bag of words method for interactive qualitative localization and mapping”. *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2007.

[54] P.I. Corke, “A robotics toolbox for Matlab”, *IEEE Robotics and Automation Magazine*, 1(3), pp. 24—32, 2006.

[55] P-Y. Oudeyer, and F. Kaplan, “How can we define intrinsic motivation?” *Proceedings of the 8th International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, Lund University Cognitive Studies, Lund:LUCS, Brighton, 2008.

[56] Y. Kuniyoshi, Y. Yorozu, M. Inaba, H. Inoue : “From visuo-motor self learning to early imitation-a neural architecture for humanoid learning”. In: *IEEE Int. Conf. Robotics and Automation*, vol. 3, pp. 3132–3139, 2003.

[57] M. Lopes, F. Mello, L. Montesano, J. Santos-Victor (to appear) “Cognitive processes in imitation : overview and computational approaches”, in *From motor to interaction learning in robots*, ed. O. Sigaud and J. Peters, Springer LNCS.

[58] A. L. Thomaz and C. Breazeal, "Experiments in Socially Guided Exploration: Lessons learned in building robots that learn with and without human teachers." *Connection Science, Special Issue on Social Learning in Embodied Agents*, 20(2&3), pg91-110, 2008.

[59] F. Kaplan, P-Y. Oudeyer, B. Bergen, “Computational Models” in *the Debate over Language Learnability, Infant and Child Development*, 17(1), pp. 55—80, 2008.

[60] E. Thelen and L. B. Smith*, A Dynamic Systems Approach to the Development of Cognition and Action*. Cambridge, MA: MIT Press, 1994.

[61] A. Baranes and P-Y. Oudeyer, « R-IAC : Robust Intrinsically Motvated Active Learning », Proceedings of the IEEE International Conference on Development and Learning, 2009.