

R-IAC: Robust intrinsically motivated exploration and active learning

Adrien Baranes, Pierre-Yves Oudeyer

▶ **To cite this version:**

Adrien Baranes, Pierre-Yves Oudeyer. R-IAC: Robust intrinsically motivated exploration and active learning. IEEE Transactions on Autonomous Mental Development, 2009, 1 (3), pp.155-169. 10.1109/TAMD.2009.2037513 . hal-00818174

HAL Id: hal-00818174

https://inria.hal.science/hal-00818174

Submitted on 26 Apr 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

R-IAC: Robust Intrinsically Motivated Exploration and Active Learning

Adrien Baranès and Pierre-Yves Oudeyer
INRIA, France

***Abstract*— IAC was initially introduced as a developmental mechanism allowing a robot to self-organize developmental trajectories of increasing complexity without pre-programming the particular developmental stages. In this paper, we argue that IAC and other intrinsically motivated learning heuristics could 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 (Robust Intelligent Adaptive Curiosity), 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 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 presented in this paper is made publicly available.**

***Index Terms*— active learning, intrinsic motivation, exploration, developmental robotics, artificial curiosity, sensorimotor learning.**

I. 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 conceived 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 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.

A. 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 becomes 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.

B. 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 mechanism 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 of 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 reaching, biting and simple vocal imitation with an AIBO robot. In these experiments, the focus was on the study of how developmental stages could self-organize into a developmental trajectory without a direct pre-specification of these stages and their number.

This paper aims to propose a new version of the Intelligent Adaptive Curiosity algorithm (IAC) presented in [27], called **R-IAC** for Robust Intelligent Adaptive Curiosity, and to show that it can be used as an efficient active learning algorithm to learn forward and inverse models in a complex unprepared sensorimotor space with unlearnable subspaces. Furthermore, together with the complete pseudo-code, we provide access to accompanying publicly available open-source software that implements the algorithm and contains tools to reproduce all the experiments presented in this paper.

II. Robust Intelligent Adaptive Curiosity (**R-IAC**) as active learning

A. 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 showed how IAC allowed 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 were learnt in spite of not having been specified initially. In [36], this system 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:

1) ***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;

2) ***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;

3) ***A new region splitting mechanism*** that is based on the direct optimization of learning progress dissimilarity among regions;

4) ***The introduction of a third exploration mode*** hybridizing learning progress heuristics with more classic heuristics based on the exploration of zones of maximal unpredictability;

B. Prediction Machine and Analysis of Error Rate

We consider a robot as a system with motor/actions channels **M** and sensory/state channels **S. M** and **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, **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 $\mathbf{M}(\mathbf{t})$, and sensors, as $\mathbf{S}(\mathbf{t})$, at a time t. $\mathbf{SM}(\mathbf{t})$ 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 $(\mathbf{SM(t)}, \mathbf{S(t+1)})$, 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 $\tilde{\mathbf{S}}(\boldsymbol{t+1})$ (estimated using the created model), and real consequences $\mathbf{S}(\boldsymbol{t+1})$. It is then able to produce a measure of error $\boldsymbol{e(t+1)}$, which represents the quality of the model for sensorimotor context $\mathbf{SM}(\boldsymbol{t})$. This is summarized in Fig. 1.

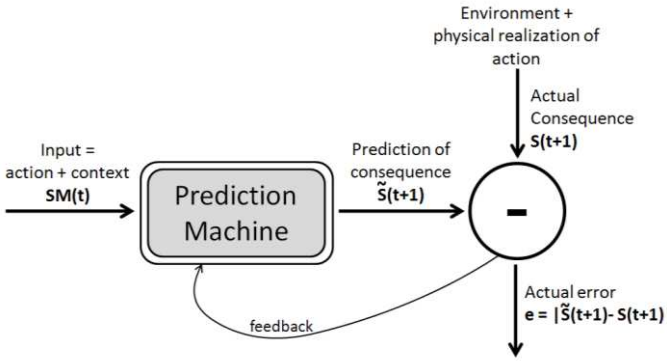


Fig. 1. The prediction learning machine (e.g. a neural network, an SVM, or Gaussian process regression based algorithm)

Then, we consider a module able to analyze learning progress over time, for a given subregion $\boldsymbol{R_n}$ of the sensorimotor space **SM**. This system, called Prediction Analysis Machine **PAM** (Fig. 2) considers the set $\boldsymbol{E_n}$ of all the $\boldsymbol{m}$ experimented exemplars $\mathrm{Ex}_i = \left(\mathbf{SM}_i(\mathbf{t}), \mathbf{S}_i(\mathbf{t+1}), \boldsymbol{e}_i(\boldsymbol{t+1})\right), i \in [1{:}\,m]$, collected inside $\boldsymbol{R_n}$, sorted by their execution order (from the older to the last performed):

$$\boldsymbol{E_n} = \left\{ \begin{matrix} \mathrm{Ex}_1, \\ \mathrm{Ex}_2, \\ \ldots, \mathrm{Ex}_m \end{matrix} \right\}_n = \left\{ \begin{matrix} \left(\mathbf{SM}_1(\mathbf{t}), \mathbf{S}_1(\mathbf{t+1}), \boldsymbol{e}_1(\boldsymbol{t+1})\right), \\ \left(\mathbf{SM}_2(\mathbf{t}), \mathbf{S}_2(\mathbf{t+1}), \boldsymbol{e}_2(\boldsymbol{t+1})\right), \\ \ldots, \quad \left(\mathbf{SM}_i(\mathbf{t}), \mathbf{S}_i(\mathbf{t+1}), \boldsymbol{e}_i(\boldsymbol{t+1})\right) \end{matrix} \right\}_n$$

where $\boldsymbol{e_i(t+1)}$ is the prediction errors of **PM** associated to the prediction of $\mathbf{S}_i(\mathbf{t+1})$ given $\mathbf{SM}_i(\mathbf{t})$.

The Prediction Analysis Machine **PAM** monitors the learning process inside $\boldsymbol{R_n}$ by analyzing the evolution of errors. More precisely, the system computes the value opposite to the derivative of errors, which is called Learning Progress $\boldsymbol{LP_n}$. This value is computed using a sliding window which contains the ζ ($\zeta = 2.k, k > 1$) most recent exemplars of the considered region $\boldsymbol{R_n}$: $\left\{\mathrm{Ex}_{m-\zeta}, \mathrm{Ex}_{m-\zeta+1}, \ldots, \mathrm{Ex}_m\right\}_n$ (if the region contains less than ζ examplars, then the learning progress is computed over a shorter window with all the current collected exemplars).

Therefore, considering $|E_n|$ as the cardinality of E_n we define the Learning Progress $\boldsymbol{LP(E_n)}$, also noted $\boldsymbol{LP_n}$, in region $\boldsymbol{R_n}$ and at a given moment in time as:

$$\boldsymbol{LP(E_n)} = \frac{\sum_{i=|E_n|-\zeta}^{|E_n|-\frac{\zeta}{2}} e_i - \sum_{i=|E_n|-\frac{\zeta}{2}}^{|E_n|} e_i}{\zeta}$$

Because $\boldsymbol{LP_n}$ is computed over the recent past through a sliding window, it is not necessary to memorize all past learning exemplars, and makes the whole system both computationally efficient in terms of speed and memory usage. Actually, in a given region, only the last $\boldsymbol{T_{split}}$ exemplars need to be memorized, with $\boldsymbol{T_{split}} > \zeta$ being the splitting parameter described below. $\boldsymbol{LP_n}$ is then used as a measure of interestingness 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.

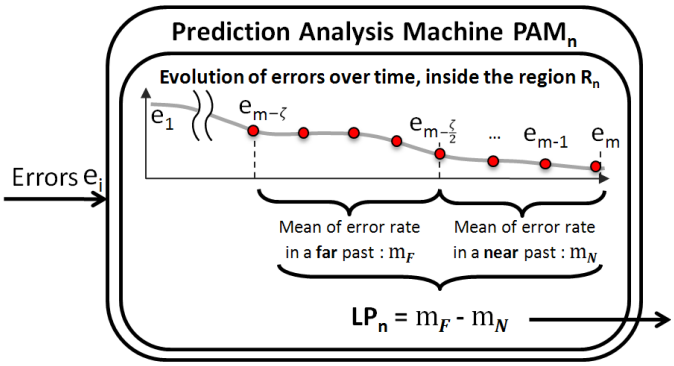


Fig. 2. Internal mechanism of the Prediction Analysis Machine $\mathbf{PAM}_n$ associated to a given subregion $\boldsymbol{R_n}$ of the sensorimotor space. This module considers errors detected in prediction by the Prediction Machine **PM**, and returns a value representative of the learning progress in the region. Learning progress is the derivative of errors computed over the last ζ examplars collected in the subregion $\boldsymbol{R_n}$.

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 $\boldsymbol{R_n}$ of the sensorimotor space, need to be built. Therefore, the learning progress $\boldsymbol{LP_n}$ provided as the output values of each **PAM** becomes representative of the interestingness of the associated region $\boldsymbol{R_n}$. 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.

C. The Split Machine

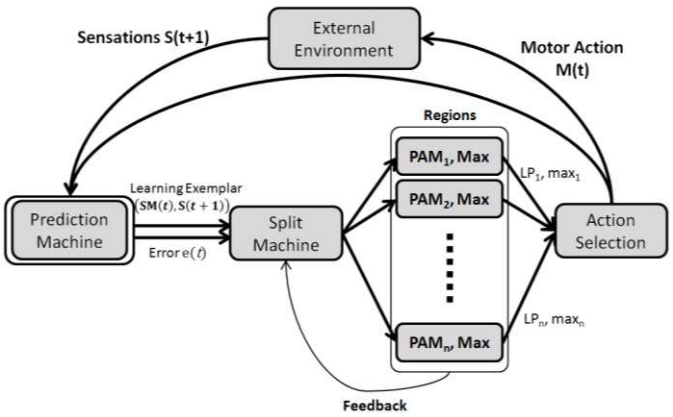


Fig. 3. General architecture of **IAC** and **R-IAC**. The Prediction Machine **PM** is used to create a forward model of the world, and measures the quality of its predictions (errors values). Then, a split machine cuts the sensorimotor space into different regions, whose quality of learning over time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

The Split Machine **SpM** (Fig. 3) is both responsible of 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.

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 a FIFO (first-in first-out) list of recently collected exemplars that it covers, with a maximum length of $\boldsymbol{T_{split}}$. When this threshold is reached, different mechanisms are triggered in **IAC** and **R-IAC**:

1. in **IAC**, the region is split into two daughter regions according to the mechanism described below, and the parent region is deleted ;
2. in **R-IAC**: if the region is a leaf region, then it is split into two daughter regions with the mechanism described below, but the initial region is kept and becomes a parent region; If the region is already a parent region, then it is not split anymore and any subsequent exemplars added to the FIFO list provokes the deletion of the oldest exemplar already in the list ;
3. In both **IAC** and **R-IAC**, splitting is done with hyperplanes perpendicular to one dimension. An example of split execution is shown in Fig. 4, using a two dimensional input space. **IAC** and **R-IAC** differ in the way the hyperplane is chosen.

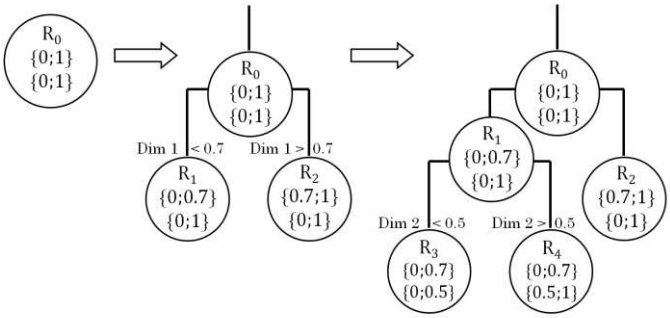


Fig. 4. The sensorimotor space is iteratively and recursively split into sub-spaces, called "regions". Each region R_n is responsible for monitoring the evolution of the error rate in the anticipation of consequences of the robot's actions, if the associated contexts are covered by this region.

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 $\mathbf{S(t+1)}$ 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 $\mathbf{SM(t)} \rightarrow \mathbf{S(t+1)}$. Mathematically, we consider $\varphi_n = \left\{ \left(\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t+1})\right)_{\boldsymbol{i}} \right\}$ as the set of exemplars possessed by the region $\boldsymbol{R_n}$. 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 $(\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t+1}))_{\boldsymbol{i}}$ of φ_{n+1} have a j^{th} component of their $\mathbf{SM}(\boldsymbol{t})$ smaller than v_j
(2) All the exemplars $(\mathbf{SM}(\boldsymbol{t}), \mathbf{S}(\boldsymbol{t+1}))_{\boldsymbol{i}}$ of φ_{n+2} have a j^{th} component of their $\mathbf{SM}(\boldsymbol{t})$ greater than v_j
(3) The quantity :

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

is **minimal**, where

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

where S is a set of vectors, and |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 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 progresses 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 were 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 then done by choosing j and v_j such that:

$$Qual(j, v_j) = (LP_{n+1} - LP_{n+2})^2$$

is **maximal**, where LP_{n+1} and LP_{n+2} are the learning progress computed inside φ_{n+1} and φ_{n+2}.

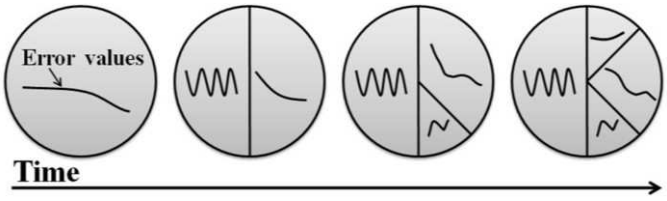


Fig. 5. Evolution of the sensorimotor regions over time. The whole space is progressively subdivided in such a way that the dissimilarity of each sub-region in terms of learning progress is maximal.

D. Action Selection Machine

We present here an implementation of Action Selection Machine **ASM**. The **ASM** decides on actions $\mathbf{M}(t)$ to perform, given a sensory context $\mathbf{S}(t)$. (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:

Outline of the global loop of IAC and R-IAC algorithms

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

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 p_1 typically equal to 30%**.**

*2) **Mode 2**: Learning Progress Maximization Exploration*

This mode, chosen with a probability p_2 typically equal to 70% in **IAC** and 60% in **R-IAC** (which means this is the dominant mode)**,** 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:

- ***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 n) with a probability P_n proportional to its learning progress LP_n, stored in the associated $\mathbf{PAM_n}$:

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

- ***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.

*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 $\boldsymbol{SM}(t) \rightarrow e(t+1)$ 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**).

E. Pseudo-code of R-IAC

R-IAC($\boldsymbol{PM}, \boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p_3}, \boldsymbol{T_{split}}, \boldsymbol{l}, \eta, \boldsymbol{\Gamma}, \kappa, \zeta$)

Init

- **Let** $\boldsymbol{R_0}$ be the whole space of mathematically possible values of the sensorimotor context **SM(t)** (typically a hypercube in $\mathbb{R}^{\boldsymbol{d}}$);
- **Let** $\boldsymbol{LP_0 = 0}$ be the learning progress associated to $\boldsymbol{R_0}$ **;**
- **Let** $\boldsymbol{Lex_{R_0}} = [\emptyset]$ (later on in the algorithm, $\boldsymbol{Lex_{R_k}}$ will be the FIFO list $\left[\left(\left(\mathbf{SM_i(t)}, \mathbf{S_i(t+1)}\right), \mathbf{e_i(t+1)}, \boldsymbol{\omega_i}\right)\right]$ where the set of $\left(\mathbf{SM_i(t)}, \mathbf{S_i(t+1)}\right)$ components is the set of learning exemplars collected in R_k, the set of $\mathbf{e_i(t+1)}$ components is the set of associated prediction errors, and $\boldsymbol{\omega_i}$ is a time stamp whose value is used to find the relative order in which each particular learning exemplar was collected within $\boldsymbol{R_k}$); The size of $\boldsymbol{Lex_{R_k}}$ is bounded by $\boldsymbol{T_{split}}$.
- Init the prediction/learning machine **PM** with an empty set of learning exemplars;

Loop

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

Action Selection

- Select action selection mode *mode* among **mode 1**, **mode 2** and **mode 3** with probabilities $\boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p_3}$;
- **If** *mode* = **mode 1**
  - Let $\mathbf{M(t)}$ be a random vector (uniform distribution)
- **If** *mode* = **mode 2**
  - For $l = 0 \ldots n$, let $\boldsymbol{P}_l = \frac{\left|LP_l - min_{LP_i \in \boldsymbol{R}}(LP_i)\right|}{\sum_{i=1}^{|R_l|}\left|LP_i - min_{LP_i \in \boldsymbol{R}}(LP_i)\right|}$
  - Let $\boldsymbol{R_k}$ be a subregion in $\boldsymbol{R}$ chosen with probability $\boldsymbol{P_k}$, $k \in \{0, \ldots, n\}$ in a roulette wheel manner ;
  - Let $\mathbf{M(t)}$ be a random vector such that $\mathbf{SM(t)} \in \boldsymbol{R_k}$ (uniform distribution);
- **If** *mode* = **mode 3**
  - For $l = 0 \ldots n$, let $\boldsymbol{P}_l = \frac{\left|LP_l - min_{LP_i \in \boldsymbol{R}}(LP_i)\right|}{\sum_{i=1}^{|R|}\left|LP_i - min_{LP_i \in \boldsymbol{R}}(LP_i)\right|}$
  - Let $\boldsymbol{R_k}$ be a subregion in $\boldsymbol{R}$ chosen with probability $\boldsymbol{P_k}$, $k \in \{0, \ldots, n\}$in a roulette-wheel manner ;
  - 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 $\boldsymbol{Lex_{R_k}}$ **;**
  - Let **Mmax(t)** = $\boldsymbol{argmax_{M(t)} Err_{R_k}}\left(\boldsymbol{SM(t)}\right)$ obtained by sampling uniformly randomly $\boldsymbol{\Gamma}$ candidates **M(t) ;**
  - **Let M(t) = Mmax(t) +** $\boldsymbol{\varepsilon}$ with $\boldsymbol{\varepsilon}$ a small random number between **0** and $\boldsymbol{\sigma}$ along a uniform distribution.
- **Execute M(t) ;**

Prediction and measurement of the consequences of action

- Estimate the predicted consequence $\tilde{\mathbf{S}}\mathbf{(t+1)}$ of executing $\mathbf{M(t)}$ in the environment with state **S(t)** using the prediction machine **PM** (e.g. ILO-GMR, LWPR, or a neural net);
- Measure the real consequence $\mathbf{S(t+1)}$ after execution of $\mathbf{M(t)}$ in the environment with state **S(t);**
- Compute the error $\boldsymbol{e(t+1)} = \boldsymbol{abs}\left(\tilde{\mathbf{S}}\mathbf{(t+1)} - \mathbf{S(t+1)}\right)$**;**
- Update the prediction machine **PM** with the new learning exemplar $\left(\mathbf{SM(t)}, \mathbf{S(t+1)}\right)$**;**

Update of region models

- **Let** Ex = $\left(\mathbf{SM(t)}, \mathbf{S(t+1)}, \mathbf{e(t+1)}\right)$
- **Let** γ be the total number of regions created by the system so far;
- **For** all regions $\boldsymbol{R_k}$ such that **SM(t)** $\in \boldsymbol{R_k}$
  - **Let** $\boldsymbol{\omega}$ be the maximum $\boldsymbol{\omega_i}$ time stamp in $\boldsymbol{Lex_{R_k}}$ **;**
  - **Update** $\boldsymbol{Lex_{R_k}}$ by adding $\left(\left(\mathbf{SM(t)}, \mathbf{S(t+1)}\right), \mathbf{e(t+1)}, \boldsymbol{\omega_t}\right)$ (and possibly deleting the oldest exemplar if $|\boldsymbol{Lex_{R_k}}| = \boldsymbol{T_{split}}$) where $\boldsymbol{\omega_t}$ is a time stamp used to keep track of the order in which this learning exemplar was stored in relation to others (see below);
  - **If** $|\boldsymbol{Lex_{R_k}}| = \boldsymbol{T_{split}}$ **AND** $\boldsymbol{R_k}$ is a leaf region

    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 κ 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 $\mathbf{SM(t)}$ smaller than v_j ;

2. $\boldsymbol{Lex_{R_{\gamma+2}}}$ 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. $\left(LP_{\gamma+1} - LP_{\gamma+1}\right)^2$ is **maximal**, where errors are indexed by their relative order of measurement calculated from $\boldsymbol{\omega_i}$ values where $\boldsymbol{LP(E_n)} = \frac{\sum_{i=|E_n|-\zeta}^{|E_n|-\frac{\zeta}{2}} e_i - \sum_{i=|E_n|-\frac{\zeta}{2}}^{|E_n|} e_i}{\zeta}$ and where ζ defines the time window used to compute learning progress achieved through the acquisition of most recent learning exemplars in each region;

- Store the learning progresses $\mathbf{LP_{\gamma+1}}$ and $\mathbf{LP_{\gamma+2}}$ of the two newly created regions;
- $\gamma = \gamma + 1$

- **For** all regions $\boldsymbol{R_k}$ such that $\mathbf{SM(t)} \in \boldsymbol{R_k}$ (except $\boldsymbol{R_{\gamma+1}}$ and $\boldsymbol{R_{\gamma+2}}$ if a split was performed), recompute $\boldsymbol{LP_k}$ and store the value; ***End Loop***

F. 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 reproducing all experiments presented in sections IV and V below is made publicly available at: http://flowers.inria.fr/riac-software.zip

G. Remark

*1) Computational complexity of **R-IAC***

Because regions are stored and accessed in a binary tree, because only leaves regions can be split and only one region per new exemplar can be split, and because the number of exemplars stored in each region is bounded by $\boldsymbol{T_{split}}$ and managed by a FIFO list/stack, it follows that the total number of regions grows logarithmically with the number of collected exemplars, hence the number of stored exemplars grows also logarithmically (with a higher but constant multiplicative factor), and thus global memory usage grows logarithmically. Furthermore, the computational cost of updating the regions-tree structure is dominated by the cost of the splitting algorithm, which is currently done through Monte-Carlo simulation: this allows to control the number of samples in the optimization process and thus can be set to be low to ensure fast computation (at the cost of accuracy, but since multi-resolution makes the system robust to suboptimal splits, this has a limited impact on the performance of the system), which is also permitted by the fact that learning progress is computed only over the last ζ exemplars in each region. Finally, the computational cost of action selection grows linearly with the number of regions, thus logarithmically with the number of collected exemplars, with a very small constant multiplicative factor since it only consists of basic probability computations based on the learning progress values of regions. Thus, the system's memory and computation time requirements grow logarithmically with time, which makes it in practice scalable to many existing robotic experimental setups.

2) Regulation of the growth of complexity

As argued 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.

3) 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 multiresolution scheme allows the system to be rather insensitive to the creation of an increasing number of small regions.

4) Planning learning progress

The central contribution of both the **IAC** and **R-IAC** systems lie in the way rewards are defined and computed, i.e.

through region-based hierarchical multiresolution evaluation of learning progress. This can be readily and efficiently re-used in a traditional active learning regression context where the learning problem can be transformed into an immediate reward maximization problem, such as in the standard self-supervised regression framework in [41], and as we will do in the experiments presented in the next sections. Yet, many real world robotic sensorimotor spaces are such that a given zone of high learning progress might not be immediately reachable and thus might require planning through a potentially uncertain intermediate path which does not necessarily provide learning progress. While the reward system of **R-IAC** is currently integrated into an action selection loop which is compatible with such environments, it does not include such a planning capacity and thus the overall architecture is suboptimal in that case. Hence, the **R-IAC** reward system would need to be integrated with an action selection scheme that allows the system to plan and maximize the cumulated sum of future expected **R-IAC** rewards (i.e. future expected learning progress as defined in **R-IAC**) rather than immediate **R-IAC** rewards. This could be done like in related intrinsically motivated reinforcement learning architecture presented in [28,33], and will be achieved and evaluated in future work.

III. The Prediction Machine: Incremental Regression Algorithms for Learning Forward and Inverse Models

The **R-IAC** system presented above is 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 exemplars are collected. But among incremental algorithms, methods based on neural networks, memory-based learning algorithms, or incremental statistical learning techniques could be used [43]. This agnosticism 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 **R-IAC** and on the capabilities of the prediction machine/regression algorithm for which **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 **R-IAC** is for driving exploration for the discovery of the robot's body, as 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 whose tuning is not straightforward and thus makes its use not optimal for repeated experiments about **R-IAC** in various sensorimotor spaces. On the other hand, Gaussian processes and Gaussian mixture regression have fewer 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 **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 taking 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 (typically 2 or 3), 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]. A 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.

ILO-GMR has only two parameters: the number $\boldsymbol{k}$ of components for local models, and a parameter $\boldsymbol{N}$ that defines the notion of local vicinity (see the pseudo-code outline below). A related approach, based on Gaussian process regression rather than Gaussian mixture regression, has been

described in [42] and in depth comparisons of these approaches will be of high interest in further work.

Outline of the pseudo-code of **ILO-GMR**

ILO − GMR(Data, X, N, k)
Data *is the set of* $(Xi, Yi = f(Xi))$ *points already observed and stored in a hierarchichal k-means tree structure.*
$\boldsymbol{X}$ *is the input point.*

- Find the approximate $\boldsymbol{N}$ closest points $\boldsymbol{Xj}$ to $\boldsymbol{X}$ in ***Data*** (e.g. with incremental hierarchical k-means);
- Build a local $\boldsymbol{k}$-components GMM model based on the $(\boldsymbol{Xj}, \boldsymbol{Yj})$ corresponding examplars;
- Predict $\boldsymbol{y' = f(X)}$ with GMR using a least square estimate;
- Measure the true $\boldsymbol{y = f(X)}$;
- Update incrementally ***Data*** to include the novel exemplar $(\boldsymbol{y}, \boldsymbol{f(X)})$;

We have compared the performance of **ILO-GMR** with other state-of-the-art regression methods on the hard regression task defined in the SARCOS dataset which has been used several times in the literature as a benchmark for regression techniques in robotics, e.g. [45,60,61]. This dataset encodes the inverse dynamics of the arm of the SARCOS robot, with 21 input dimensions (position, velocity and acceleration of 7 DOFs) and 7 output dimensions (corresponding torques). It contains 44484 exemplars in the training database and 4449 test exemplars. It is available at: http://www.gaussianprocess.org/gpml/data/. The regression methods to which we compared performances on this dataset are: Gaussian Mixture Regression (GMR, [3]), Gaussian Process Regression (GPR, [62]), Local Gaussian Process Regression (LGP, [61]), support vector regression (ν-SVR, [63]) and Locally Weighted Projection Regression (LWPR, [44]). All those algorithms were tuned with reasonable effort to obtain the best generalization results. For ILO-GMR, optimal tuning was done with $\boldsymbol{N = 200}$ and $\boldsymbol{k = 2}$, but results degrade very slowly when moving away from these parameters. Table 1 shows the comparison of the performances of those algorithms for predicting the torques of the first joint in the SARCOS database. We observe that the performance of ILO-GMR matches nearly the best performance (GPR), is slightly better than ν-SVR, LGP and GMR, and clearly better than LWPR while being also incremental but much easier to tune.

GMR	V-SVR	ILO-GMR	LWPR	GPR	LGP	LR
0.014	0.011	0.0075	0.024	0.0065	0.011	0.081

Table 1. Normalized mean square errors of regression methods using the SARCOS dataset

Furthermore, in spite of the fact that our current implementation of ILO-GMR was done in Matlab and is not optimized, it is already able to make a single prediction and incorporate a new learning exemplar in around 10 milliseconds on a standard laptop computer and when 44484 SARCOS data examples are already in memory. Furthermore, we have measured experimentally the evolution of training and prediction time per new exemplar: it increases approximately linearly with a small slope in the range $0 - 44484$ learning exemplars. Further work will study systematically the computational complexity and scalability of ILO-GMR.

Furthermore, learning forward motor models is mainly useful if they 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 second experiment described below.

IV. Experiment 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. In these experiments, the parameters of **IAC** and **R-IAC** are $\boldsymbol{T_{split}} = 250$, and the learning progress window is 50. Also, probabilities are $\boldsymbol{p_1} = 0.3$, $\boldsymbol{p_2} = 0.7$ in **IAC** and $\boldsymbol{p_1} = 0.3$, $\boldsymbol{p_2} = 0.6$, $\boldsymbol{p_3} = 0.1$. in **R-IAC**, The incremental learning algorithm that is used to learn the forward model is the ILO-GMR system described in part III, with the same parameters in both **IAC** and **R-IAC** experiments ($\boldsymbol{k} = 2$ and $\boldsymbol{N} = 100$).

A. 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 $\boldsymbol{q_1}, \boldsymbol{q_2}$ as shown on Fig. 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 equipment consists of a one-pixel camera, returning an intensity value $\boldsymbol{p}$, set on its extremity as shown on Fig. 6. 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 $\boldsymbol{q_1, q_2}$. So, we can describe the system input/output mapping with two input dimensions, and one output as: $\boldsymbol{p = V(q_1, q_2)}$.

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.

B. Environment configuration

The front wall consists of an increasing precision checker (Fig. 7), 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. 6).

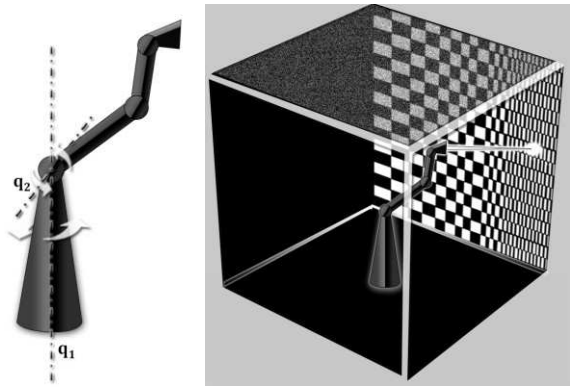


Fig. 6. Representation of a 2 axes arm, with a one pixel camera mounted on its extremity. This arm is put in the center of a cubic room, with different painted walls of different complexities.

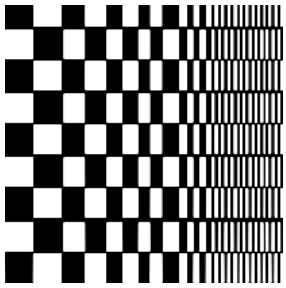


Fig. 7. Wallpaper disposed in the front wall. For many learning algorithms, the complexity increases from left to right.

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 (see 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.

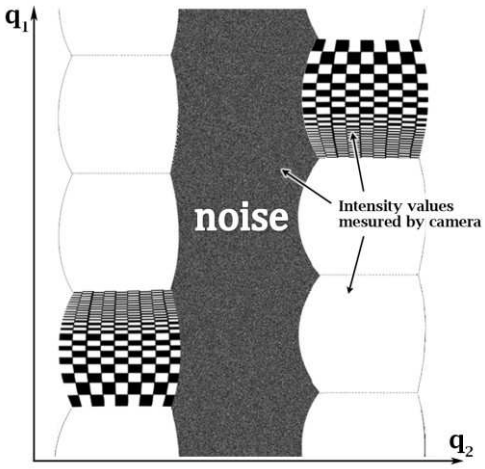


Fig. 8. Two-Dimensions visualization of the sensorimotor space of the robot, with two motor dimensions $(\boldsymbol{q_1, q_2})$ and one sensory dimension.

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 Fig. 8. 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.

A. 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. 9 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 focusing 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 Fig. 9 were smoothed with a gaussian 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. 10 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.

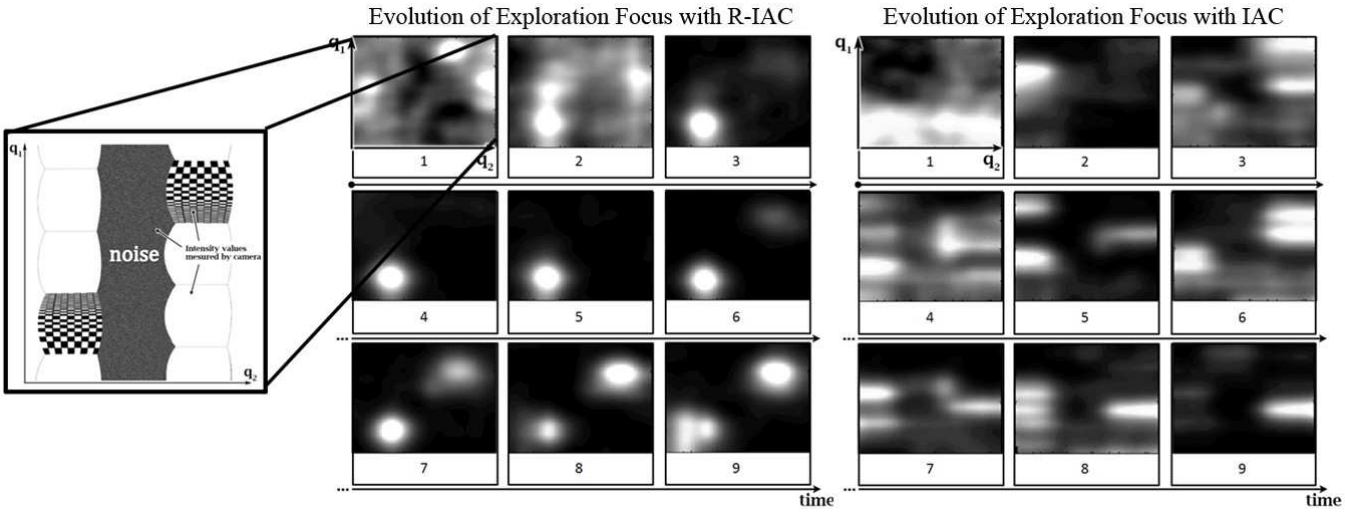


Fig. 9. Evolution of the exploration focus when using **R-IAC** as an exploration heuristics (left) or **IAC** (right). Each square represents the smoothed distribution of explored motor configurations at different times in a given run and over a sliding time window. Darker intensities denote low exploration 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.

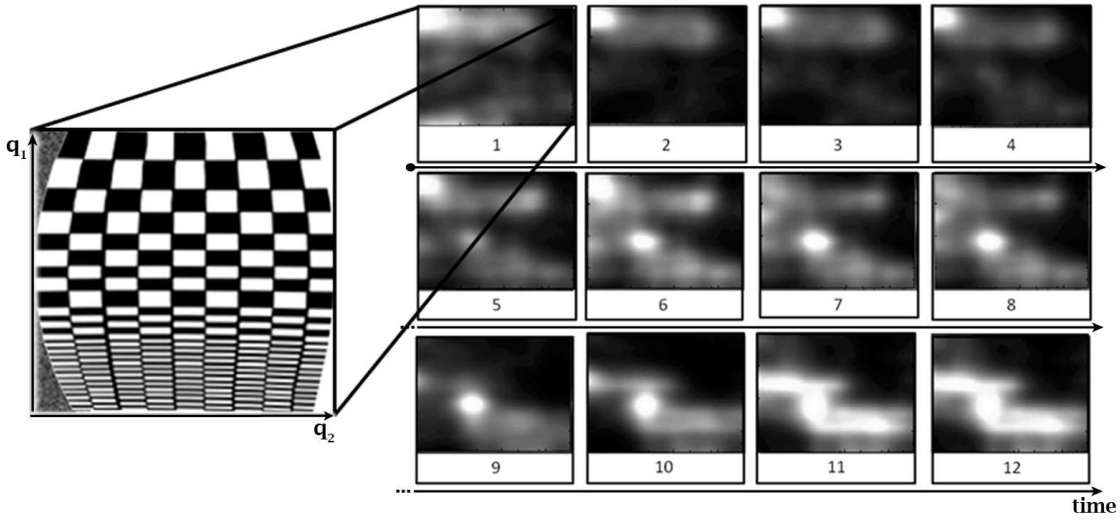


Fig. 10. A zoom into the evolution of the distribution of explored sensorimotor experiments in one of the two subregions where the camera is looking at the checkerboard when **R-IAC** is used. We observe that exploration is first focused on zones of the checkerboard that have a low complexity (for the given learning algorithm), and progressively shifts towards zones of increasing complexity.

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 is thus able to evaluate accurately the different complexities of small parts of the world, and to drive the exploration based on this evaluation.

A. 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 froze 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 Fig. 11. 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 Fig. 11 shows, results of **R-IAC** are statistically significantly 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.

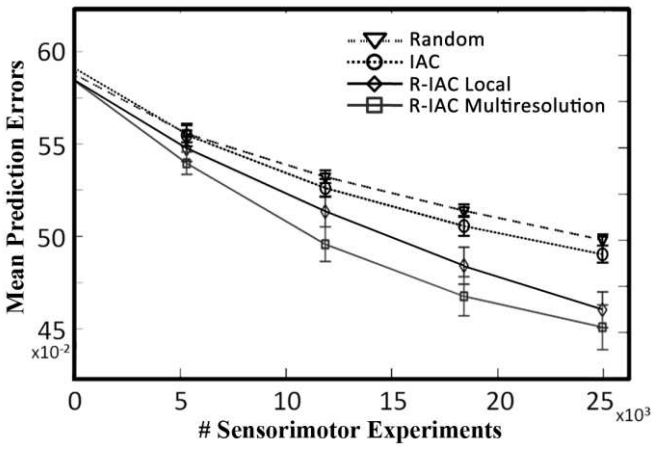


Fig. 11. Mean and standard deviation of prediction errors with **IAC**, **R-IAC** with only local resolution, and **R-IAC** with multi-resolution, compared with the random exploration approach.

V. 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.

A. 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 Fig. 12). On the tip of one of the two arms is attached a square camera capable of detecting 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 Fig. 12). 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 reference frame $\boldsymbol{C}$ 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 nor to the tip of the other arm).

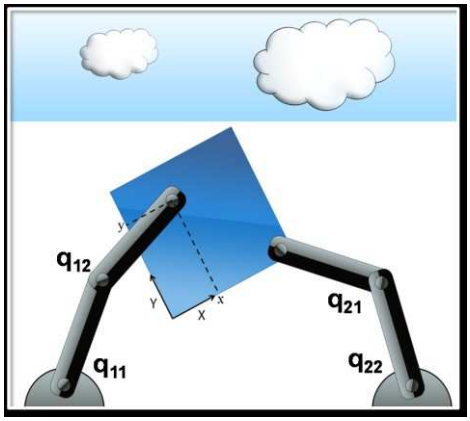


Fig. 12. Experimental setup. The 2D robot has two arms, each with two links and two revolute joints. At the tip of the right arm is rigidly attached a square camera/eye which can sense either the position of the tip of the other arm in its own reference frame $(\boldsymbol{X},\boldsymbol{Y})$ if it is 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.

B. Results

In these experiments, the parameters of **IAC** and **R-IAC** are $\boldsymbol{Tsplit} = 250$, the learning progress window is 50. Also, probabilities are $\boldsymbol{p_1} = 0.3$, $\boldsymbol{p_2} = 0.7$ in **IAC** and $\boldsymbol{p_1} = 0.3$, $\boldsymbol{p_2} = 0.6$, $\boldsymbol{p_3} = 0.1$. in **R-IAC.** 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 part III, with the same parameters in both **IAC** and **R-IAC** experiments ($\boldsymbol{k} = 2$ and $\boldsymbol{N} = 100$).

A first study of what happens consists of 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. 13 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 **R-IAC** shows a large difference with both **IAC** and random exploration: the system spends three times more time in a distance less than 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.

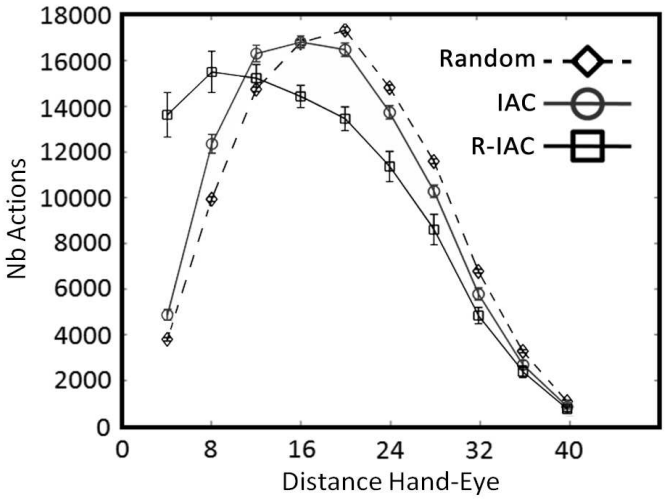


Fig. 13. Mean distributions (and standard deviations) over 30 simulations of distances between the hand and the center of the eye when exploration is random, guided by **IAC**, or guided by **R-IAC**. We observe that while **IAC** pushes the system to explore slightly more than random exploration the 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

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 independently 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 Fig. 14. 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 $(\boldsymbol{x},\boldsymbol{y})_C$ objectives corresponding to particular $\boldsymbol{q_{21}}, \boldsymbol{q_{22}}$ values**.** Control was realized through inferring an inverse model using ILO-GMR as presented in part III. 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 Fig. 14.
Both curves on Fig. 14 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 Fig. 15 configurations obtained, considering fixed goals $\{(\boldsymbol{x},\boldsymbol{y})_C, \boldsymbol{q_{21}}, \boldsymbol{q_{22}}\}$, and estimated positioning of the left hand.

VI. CONCLUSION

IAC was initially introduced as a developmental mechanism allowing a robot to self-organize developmental trajectories of increasing complexity without pre-programming the particular developmental stages [27]. In this paper, we have argued that **IAC** and other intrinsically motivated learning heuristics could be viewed as active learning algorithms, and were based on heuristics that are more suited than traditional active learning algorithms for operation in unprepared sensorimotor spaces with large unlearnable subspaces. Then, 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 presented in this paper 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 increasing 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].

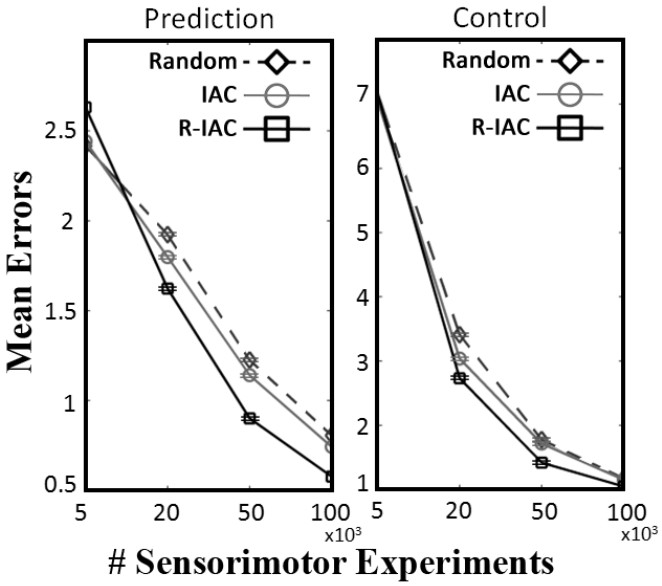


Fig. 14 Left: evolution of the generalization capabilities of the learnt forward model with Random exploration, **IAC**, and **R-IAC**, averaged over 30 simulations. Right : evolution of performances in control based on the forward model learnt through Random exploration, **IAC** exploration, and **R-IAC** exploration, averaged over 30 simulations.

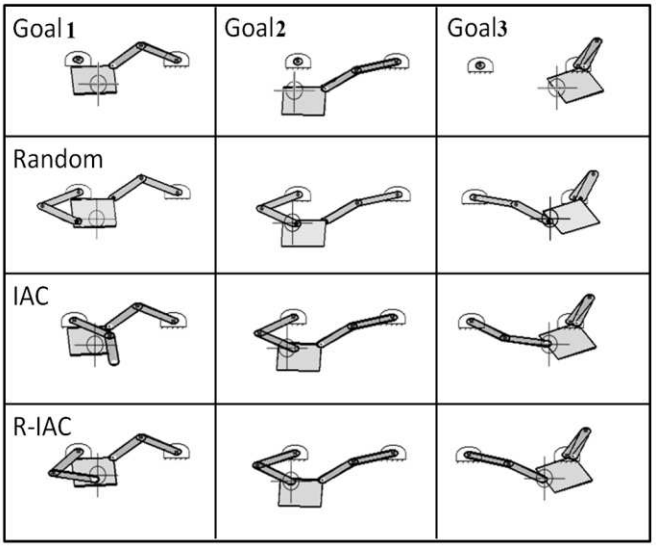


Fig. 15. Examples of performances obtained in control. the first row shows three fixed goals (defined as joint position of the right hand, and a position of the left hand extremity in the reference frame of the eye). The robot has to reach the targets (position fixed in the eye) with the left arm.We observe that the robot which has explored its sensorimotor space with R-IAC reaches all three goals with high accuracy (fourth row), while the robot that explored its sensorimotor space through random (second row) or IAC (third row) exploration are either slightly imprecise for goals 2 and 3 and very imprecise for goal 1.

VII. ACKNOWLEDGEMENTS

We thank Ming Li for his crucial help in the elaboration and evaluation of the ILO-GMR regression algorithm.

VIII. 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] D. Koch and A. Billard (2009) Gaussian Mixture Regression and its Application in Robotics, RSS'09 Workshop on Regression in Robotics.

[61] Nguyen-Tuong, D. and J. Peters: Local Gaussian Processes Regression for Real-time Model-based Robot Control. Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008), 380-385, IEEE Service Center, Piscataway, NJ, USA

[62] 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.

[63] C.-C. Chang and C.-J. Lin, LIBSVM: a library for support vector machines, 2001, http://www.csie.ntu.edu.tw/ cjlin/libsvm.



Adrien Baranès received the M.S. degree in Artificial Intelligence and Robotics from the University Paris VI, France, in 2008.

He is now working towards the Ph.D. degree at the French National Institute of Computer Sciences and Automatic (INRIA) where his research topic is the intrinsic motivations paradigm, studied in both knowledge and competence based frameworks. His research interests are machine learning, psychological and developmental approaches of robotics, and computational neurosciences.



Pierre-Yves Oudeyer studied theoretical computer science at Ecole Normale Supérieure de Lyon, France, and received the Ph.D. degree in artificial intelligence from the University Paris VI, France. After eight years as a permanent researcher at Sony CSL Paris, he is now research scientist at INRIA, France, where he heads the FLOWERS team. He is interested in the mechanisms that allow humans and robots to develop perceptual, motivational, behavioral, and social capabilities to become capable of sharing cultural representations.

He has published a book, more than 60 papers in international journals and conferences, and received several prizes for his work in developmental robotics and on the origins of language. He is editor of the IEEE CIS Newsletter on Autonomous Mental Development, and associate editor of IEEE Transactions on Autonomous Mental Development, Frontiers in Neurorobotics, and of the International Journal of Social Robotics.