Noname manuscript No.
(will be inserted by the editor)

Discovering the hierarchy of tasks with intrinsic motivation enables transfer learning in curriculum learning

Robots learning increasingly complex tasks in continual learning by hierarchical reinforcement learning, emergence of affordances and active imitation learning

Sao Mai Nguyen [1,2,5] · **Nicolas Duminy** [3,5] · **Alexandre Manoury** [2,5] · **Dominique Duhaut** [3,5] · **Cedric Buche** [4,5]

Received: date / Accepted: date

Abstract Multi-task learning by robots poses the challenge of the domain knowledge: complexity of tasks, complexity of the actions required, relationship between tasks for transfer learning. In this article, we demonstrate that this domain knowledge can be learned to address the challenges of high-dimensionality and unboundedness in life-long learning. Specifically, the hierarchy between tasks of various complexities can be learned to bootstrap transfer of knowledge from simple to composite tasks. We propose a framework for robots to learn sequences of actions of unbounded complexity in order to achieve multiple control tasks of various complexity. Our hierarchical reinforcement learning framework, named Socially Guided Intrinsic Motivation for Sequence of Actions through Hierarchical Tasks (SGIM-SAHT), relies on intrinsic motivation to explore the action space and task space, and to discover the relationship between tasks and learnable subtasks. Through experiments on robot arms and mobile robots, we outline our contributions to enable robots to efficiently associate sequences of actions to multiple control tasks: representations of task dependencies, emergence of affordances mechanism, curriculum learning and active imitation learning. SGIM-SAHT mainly advantages tasks at a high level of hierarchy, as our active learning algorithm chooses the most appropriate exploration strategy based on empirical measures of competence and learning progress. It infers its curriculum by deciding which tasks to explore first, how to transfer knowledge, and when, how and whom to imitate. We compare these properties with the state of the art.

Keywords Intrinsic motivation · Continual learning · Curriculum learning · Transfer learning · Multi-task learning · Hierarchical reinforcement learning

[1] U2IS, ENSTA, IP Paris & Inria, FLOWERS team, France, [2]IMT Atlantique, Brest, France, [3]Université Bretagne Sud, Lorient, France, [4] ENIB, Brest, France, [5] Lab-STICC, UMR 6285, team RAMBO
E-mail: nguyensmai@gmail.com

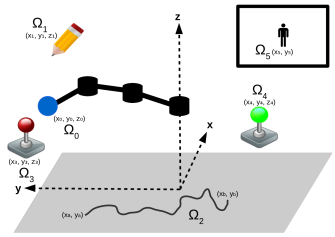

Fig. 1: Setup1: the robot arm can move the pen, draw, move the joysticks, move the videogame character on the screen.

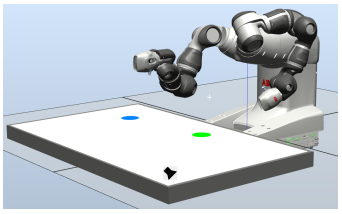

Fig. 2: Setup2: the robot arm can produce sounds by moving the blue and green objects

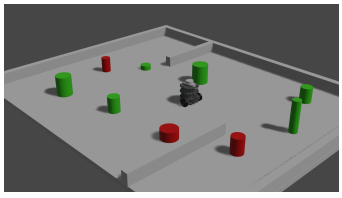

Fig. 3: Setup3: the mobile robot can avoid red obstacles, move green objects, push green objects with other objects

1 Introduction

In the mainstream approaches based on classical artificial intelligence and machine learning, robotic engineering approaches have made several valuable application-specific impacts. Yet, the achievements are often subject to restrictions that involve domain knowledge, a bounded and specific environment, or a limited set of tasks of the same complexity.

To face the challenges of multi-task learning in a continual manner for embodied agents interacting with their

stochastic environment and with humans, methods have taken inspiration in the living, and especially in how adults and infants learn in a life-long learning manner as they develop, adapt and create new skills all along their lives to tackle the various situations they face. These methods fall into the field named *cognitive developmental robotics* [1, 20], within which we will consider representations of actions, the notion of task complexity in curriculum learning and intrinsic motivation as an exploration heuristic.

1.1 Learning Sequences of Motor Policies

In the case of multiple tasks with various complexities and dimensionalities, the complexities of actions considered to complete them should be unbounded, without a priori knowledge. If we relate the complexity of actions to their dimensionality, actions of unbounded complexity should belong to spaces of unbounded dimensionality. For instance in setup Fig.1, if an action of dimensionality n is sufficient to draw a letter, a sequence of 2 actions, i.e. an action of dimensionality $2n$ is sufficient to draw two letters. Nevertheless, in general, texts have variable lengths, therefore there is no bound to the length of this sequence of actions. Likewise, in setup Fig.2, an unbounded sequence of actions is needed to make any tune composed of sounds; and in setup Fig.3, to move all green objects side by side.

Hence, in this article, we consider actions of unbounded complexity and suppose that they can be expressed as a sequence of action primitives. We will consider *primitive actions* and *sequences of actions*, also named in [40] respectively *micro actions* and *compound actions*. The agent thus needs to estimate the complexity of the task and deploy actions of corresponding complexity. The algorithm needs to address the curse of dimensionality.

To tackle high-dimensional sensory inputs, methods such as *DQN* [24], using a deep neural network architecture have successfully handled higher dimensional continuous outcome and context spaces while still considering discrete action spaces. More recently, Mnih et al. [25] proposed an asynchronous variant of the *Actor Critic* algorithm, relying both on deep neural networks and gradient policies, which successfully handles continuous action spaces.

The *options* framework proposes a temporally abstract representation of actions [35]. It enables the agent to learn sequences of actions and reuse them later. Approaches extending the *options* framework propose a temporal abstraction to optimise learning representation [33].

Skill chaining has proposed chains of options in order to reach a given target event. Learning simple skills and planning sequences of actions instead of learning a sequence directly has been shown to greatly simplify the learning problem [17]. This idea of multi-step plans using forward chaining successfully generated totally ordered plans starting from the initial state to a specified goal in in [38].

Following the ideas of a temporally abstract representation of actions and of multi-step planning, we propose a goal-directed representation of compound actions and a multi-step plan using inverse chaining of self-discovered subtasks.

1.2 Task Hierarchy for Curriculum Learning

Multi-task learning in biological agents is progressive and continual. Humans and other species develop and create new skills all along their lives as they adapt to their environment and to their own needs. In particular, infants learn skills of increasing level of difficulty as they grow up: learning simple skills first and then expanding their repository by mastering more complex skills based on the previous simple ones. Indeed, in multi-task learning problems, some tasks can be compositions of simpler tasks, which we call *'complex tasks´* or *'composite tasks´*. The learning agent should be *starting small* before trying to learn more complex tasks, as phrased in [12]. Devising the order in which tasks should be learned has been coined *'curriculum learning'* in [4]: a learning agent needs to decide at each episode both which tasks it wants to learn to control (goal) and which actions to try (means). In our works, we thus take the hypothesis that tasks can be hierarchically related, some may be considered as subtasks of more complex ones (*'hierarchically organised tasks´*). We conjecture that this hierarchy can help bootstrap the learning, by transfer of knowledge from simple to complex tasks.

Indeed, when given a task hierarchy, the robot can exploit this domain knowledge to reuse previously acquired skills to build more complex ones for tool use, as shown in [13, 8]. Approaches for hierarchical multi task learning with neural networks have also been proposed, such as *Hierarchical DQN* [18], that uses intrinsic motivation to train a neural network in a fixed hierarchical manner.

To depend less on domain knowledge, the works presented in this article seek to learn the hierarchy between tasks, by exploring the different combinations between tasks. While learning the dependencies between tasks, we show in [11], that reusing the knowledge of simple tasks as subgoals for more complex tasks indeed greatly reduced the exploration, and in [21], that planning can be used in combination of emerging hierarchical models of tasks.

1.3 Intrinsic Motivation as an Exploration Heuristic

To allow multi-task learning and overcome the limitation of domain knowledge such as the reward shaping of an extrinsic reward, *Universal Value Function Approximators* (UVFA) [32] have been developed, letting the reward function be

parametrised. Furthermore, developmental methods have proposed to transpose into algorithms the notion of intrinsic motivation, that has been outlined as a key mechanism for exploration [22]. These methods use an intrinsic reward function that is not shaped to fit a specific task but is general to all tasks the robot will face and thus tending towards lifelong learning. This approach is called intrinsic motivation – or artificial curiosity – and may be seen as a particular case of reinforcement learning using an intrinsic reward function.

Methods based on *Q-Learning* and intrinsic motivation have been proposed in [39] for discrete environments. The reward obtained by the agent depends on how much new information have been acquired. Other methods such as algorithms *IAC* [31] and *RIAC* [2], used intrinsic motivation to explore the action space of the robot, based on estimations of prediction progress. More recently Baranes and Oudeyer [3] presented the *SAGG-RIAC* algorithm, using goal-babbling to generate goals in order to explore the outcome space – inspired by how infants generate and explore goals by themselves. More recently, IMGEP [14], GEP-GP [6] and CURIOUS [5] have combined intrinsic motivation and goal babbling with deep neural networks and replay mechanisms.

We ground our studies in cognitive developmental robotics and aim for a robot capable **of discovering and learning multiple tasks as well as its curriculum with compound actions** (sequences of actions) leveraging relationship between tasks and exploration based on intrinsic motivation. In this paper, we present our research on the emergence of representations of the task hierarchy. We present the algorithms we developed for robots to learn multiple tasks in curriculum learning based on the heuristic of intrinsic motivation. The main outcome of these works is a representation of compound actions for life-long reinforcement learning algorithms to learn multiple control tasks by transfer of knowledge. We propose in this paper a common framework of three algorithms which proved efficient for manipulation tasks and emergence of affordances. In the following sections, we present the approaches we investigated :

- how knowledge from easier tasks can be transferred to complex task once the robot learns their relationship?
- how a robot can discover new learnable tasks from which to transfer knowledge to more complex tasks?
- how human demonstrations can be beneficial to an active learning robot tackling multiple control tasks?

2 Framework for Learning Tasks of Various Complexities

In this section, we propose a formalisation of the problem of learning compound actions to achieve hierarchically organised tasks, and propose a framework for algorithms based on intrinsic motivation to learn the curriculum.

2.1 Formalisation

Let us consider a robot interacting with a non-rewarding environment by performing sequences of motions of unbounded length in order to induce changes in its surroundings.

Each of these motions is named a *primitive action*, described by a parametrised function with p parameters: $a \in \mathcal{A} \subset \mathbb{R}^p$. We call $\mathcal{A}$ the *primitive action space*. Our robot can perform sequences of primitive actions. Let a *compound action* be a sequence of any length $n \in \mathbb{N}$ primitive actions, and be described by $n * p$ parameters : $a = [a_1, \ldots, a_n] \in \mathcal{A}^n$. Thus the action space exploitable by the robot is a continuous space of infinite dimensionality $\mathcal{A}^{\mathbb{N}} \subset \mathbb{R}^{\mathbb{N}}$.

The actions performed by the robot have consequences on its environment, which we call outcomes $\omega \in \Omega$, where Ω is a subspace of the state space S defining the control tasks to learn. Once the robot knows how to cause an outcome ω, we say the outcome is then *controllable*. The set of controllable outcomes is $\Omega_{controllable} \subset \Omega$ and this set changes as the robot learns new tasks. For convenience, we define the *controllable space* $\mathcal{C} = \mathcal{A} \cup \Omega_{controllable}$, regrouping both primitive actions $\mathcal{A}$ and observables that may be controlled, $\Omega_{controllable}$.

The robot learns tasks T each mapping controllable $c \in \mathcal{C}_T \subset \mathcal{C}$ and outcomes $\omega \in \Omega_T \subset \Omega$ within a given context $s \in S_T \subset S$. More formally, a task is a set of : a **forward model** $M_T : (S_T, \mathcal{C}_T) \rightarrow \Omega_T$ and an **inverse model** $L_T : (S_T, \Omega_T) \rightarrow \mathcal{C}_T$. The forward model is used to predict the observable consequence $\tilde{\omega}$ of a controllable c in a given context s. Conversely, the inverse model is used to estimate a controllable $\tilde{c}$ to be performed in a given context s to induce a goal observable state ω as a result of $\tilde{c}$.

These models are trained on the data acquired by the robot all along its exploration and recorded in its dataset $\mathcal{D}$. We define a strategy σ any process enabling the choice of a type of exploration or selection of a source of data. For instance, we wlll consider autonomous exploration strategy or imitation learning strategy.

Let us also note $\mathcal{H}$ the hierarchy of the models used by our robot. As our robot also learns this hierarchy, $\mathcal{H}$ varies along time. Its representation will be detailed in section 2.3.

2.2 Algorithmic Architecture

We propose a generic algorithm SGIM-SAHT to learn and then take advantage of task hierarchy to solve increasingly complex tasks.

The SGIM-SAHT algorithm learns by episodes in which a task T to work on, a goal outcome $\omega_g \in \Omega_T$ and a strategy σ have been selected. The selected strategy σ applied to the chosen goal outcome ω_g choooses a sequence of controllables l_c to try to reach the goal (Alg.1, l.3).

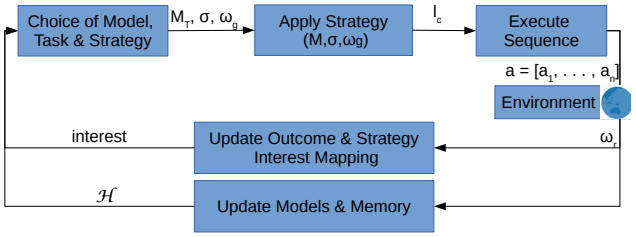

Fig. 4: The SGIM-SAHT algorithmic architecture

This sequence of controllables $l_c = [c_1, \ldots, c_m]$ is broken down in a compound action $a = [a_1, \ldots, a_n] \in \mathcal{A}^{\mathbb{N}}$, to be executed by the robot. The outcomes reached in the environment ω_r are recorded in the memory, along with the primitive actions and the built controllables sequence (Alg.1, l.4). This breakdown process is potentially recursive, based on the learned hierarchy between tasks.

Then, it computes its competence on the outcomes reached with hindsight experience replay. The measure depends on the Euclidean distance between the goal outcome ω_g and the reached outcome ω_r (Alg.1, l.5). Its exact definition depends on the implementation, and is detailed in [11, 21].

The memory and competence are used to update the models M_T and L_T, and the hierarchy of models $\mathcal{H}$ (Alg.1, l.6). More importantly, the learner updates its interest map, by computing the interest of the goal outcome for the used strategy $interest(\omega_g, \sigma)$. This interest depends on the progress measure $p(\omega_g)$ which is the derivative of the competence.

The learner then uses these interest measures to partition the outcome space Ω in regions R_i of high and low progress. This process is described in detail in [28]. In the beginning of the next episode, the learner chooses the strategy, model and goal outcome that could bring the most progress, according to the updated interest map.

Algorithm 1 SGIM-SAHT

Input: the different strategies $\sigma_1, ..., \sigma_n$
Input: the initial model hierarchy H
Initialization: partition of outcome spaces $R \leftarrow \bigsqcup_i \{\Omega_i\}$
Initialization: episodic memory *Memory* $\leftarrow \emptyset$
1: **loop**
2: $\omega_g, \sigma, M \leftarrow$ Select Goal Outcome, Strategy & Model(R, H)
3: $l_c \leftarrow$ Execute Strategy(σ, ω_g)
4: $\mathcal{D} \leftarrow (\omega_r, a, l_c) \leftarrow$ Execute Sequence(l_c)
5: $(competence(\omega_g), competence(\omega_r)) \leftarrow$ Compute Competence (ω_g, ω_r)
6: Update $M, \mathcal{H}$ with $(\mathcal{D}, competence(\omega_g), competence(\omega_r)$
7: $R_i \leftarrow$ Update Outcome and Strategy Interest Map($R, \mathcal{D}, \omega_g$)
8: **end loop**

2.3 Task Hierarchy Representation

The idea of SGIM-SAHT is to use a hierarchical representation of the outcome and controllable spaces, to outline the dependencies between tasks to help the reuse of previous knowledge. In our works, we introduce two rrepresentations called Procedure and CHIME.

The first is a goal-directed representation of action sequences in the form of sequences of subgoals, that enable transfer of knowledge between inter-related tasks, beyond simple hindsight replay. It is a temporally abstract representation of succession of actions that enable the learning agent to discover the task hierarchy and exploit the policy learned for previous tasks to learn the policy of new complex tasks. We define procedures as a way to encourage the robot to reuse previously learned tasks, and chain them to build more complex ones. More formally, **a procedure is defined as a succession of previously known outcomes** $(\omega_1, \omega_2, ..., \omega_n) \in \Omega^n$. The definition of procedures is recursive and the succession is unbounded.The procedure space is thus simply $\Omega^{\mathbb{N}}$. The definition of the procedure space only depends on the outcome space. But the valid procedures, representing the real dependencies between tasks, depend on each application case. Thus the learning agent can explore the procedure space to test which procedures are valid.

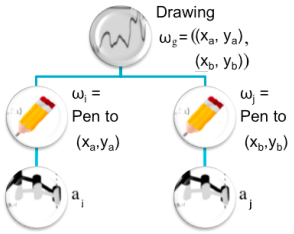

Fig. 5: Illustration of a procedure for setup fig.1. To make a drawing ω_g between points (x_a, y_a) and (x_b, y_b), a robot can recruit subtasks consisting in (ω_i) moving the pen to (x_a, y_a), then (ω_j) moving the pen to (x_b, y_b). These subtasks will be completed respectively with actions a_i and a_j.

Executing a procedure $(\omega_1, \omega_1, ..., \omega_n)$ means building the action sequence a corresponding to the succession of actions $a_i, i \in [\![1, n]\!]$ (potentially action sequences as well) and execute it (where the a_i reach best the ω_i $\forall i \in [\![1, n]\!]$ respectively). Fig. 5 illustrates this idea of task hierarchy.

Fig. 6: Illustration of nested models for setup fig.3. To push an object, the robot needs to control its position, and therefore command its wheel.

The procedures representation is based on a static set of controllable and outcome spaces.

In comparison, the CHIME hierarchical representation is based on **a dynamic set of controllable and outcome spaces and the emergence of nested models**. It is constructed using simple models $M((S_T, \mathcal{C}_i) \rightarrow \Omega_j)$ which can rely on others: lower models map outcomes to actions while higher models map them to other outcomes that should be reached. For instance in the setup3 (fig.3), the robot can move itself with the model $(M_0 : wheelCommand \in \mathcal{A} \rightarrow$

$(x_0, y_0) \in \Omega_0$) and it can learn that the position (x_1, y_1) of the object pushed depends on its own position (M_1 : $(x_0, y_0) \in \Omega_0 \rightarrow (x_1, y_1) \in \Omega_1$. The combination of the nested models M_0 and M_1 enable the robot to control (x_1, y_1) with its wheel commands (fig.6). Contrarily to SGIM-PB, only primitive actions are learned directly. Compound actions are the result of planning that constructs sequences of actions from these learned primitives. At the beginning $H = \emptyset$, no model is present, and the robot chooses itself what model to create or modify: if $\mathcal{C}_i$ seems to be highly correlated to Ω_j it may create the model $M : \mathcal{C}_i \rightarrow \Omega_j$.

3 Transfer of knowledge to increasingly complex tasks by discovering the hierarchy of tasks

To learn the hierarchy $\mathcal{H}$ between tasks, we proposed in [10] an implementation, IM-PB (Intrinsically Motivated Procedure Babbling) based on procedures and the identification among all possible dependencies, of those that are valid.

3.1 Learn Task Hierarchy to Adapt the Length of Actions

The experiments on setups of fig.1,2 in [9, 11] show that an intrinsically motivated learner is capable of learning sequences of motor actions. Intrinsic motivation indeed guides the exploration of the action space, the task space and the procedure space to find a curriculum from simple to complex tasks. IM-PB takes advantage of the dependencies between tasks. It explores the procedure space to learn these dependencies. During the learning phase, results show that the robot explores mainly the action space for simple tasks, and it explores considerably more the procedure space for complex hierarchical tasks. Thus it implicitly understands that simple tasks do not need to be decomposed into subtasks and can be reached directly by action primitives. On the contrary for complex tasks, it is more advantageous to seek which subtasks to reuse. On the test phase, the robot uses mainly the decomposition into subgoals correctly, as we designed our setup. Furthermore, It can also adapt the length of its action sequence to the task to achieve: the results show that the length of the sequence of actions increases as the complexity of the task increases in terms of its hierarchy. **Combining these procedures with the learning of simple actions to complete simple tasks, it can build sequences of actions to achieve complex tasks**.

We showed that the robot can take advantage of the procedures representation to improve its performance, especially on high-level tasks. It can also adapt the complexity of its action sequence to the complexity of the task at hand.

Nevertheless, this adaptation is limited to the first two levels of task hierarchy, and the learner can not well adapt this complexity to a deeper hierarchy of tasks. To help the robot improve its understanding of task dependencies, we present in section 5 the benefits of active imitation learning.

4 Emergence of Learnable Sub-tasks for Planning a Complex task

In the previous section, the set of possible tasks (association of inputs and outputs) are given, and the robot needs to choose which are easier to learn first, which are more difficult to learn, and which tasks can be reused as sub-goals for more difficult models. But if we do not have a set of subgoals given, but have only given a high-dimensional set of inputs and observable outputs, the transfer of knowledge becomes much more difficult, as the agent needs to imagine by itself new subgoals to learn, and how to reuse these simple models for more complex tasks. We proposed an algorithm in [21] to discover learnable models (object features that can be controlled and control features) in a hierarchical way. It was applied to wheeled robot with obstacles, movable objects and spots (fig.3). It proposed an emergence of affordances : the algorithm is able to discover learnable models, and once the model is learned, its output can be used as input features of more complex models, leading to hierarchical learning. Learning is based on object features, so capable of generalising to new objects. Objects are only described by the physical properties such as colour, height, diameter, and are generated randomly. This section summarises this work on emergence of affordances.

4.1 Affordance

The concept of *affordance*, has been first introduced by Gibson in 1977 [15] as a way to characterise physical states in an action-oriented fashion, in terms of the possible interactions an agent may have with objects. Even without knowing an object specifically, seeing visual cues of a handle may suggest possible embodied interactions with unknown objects.

In affordances learning, many approaches have been developed [16]: for instance, the traversability affordance has been studied in different works [37, 7, 23]. Likewise, the grasp affordance is a recurrent topic and various approaches exist to learn it such as learning based on visual descriptors or raw image input [26, 19]. However such methods focus on one, or a fixed number of specific affordances, with no mechanism adapting it to new or more complex affordances.

In our case, we aim to continual learning of multiple affordances through the interaction with its environment. Thus, the robot builds itself sensory motor skills using a wide variety of actions. The robot can use actions of unbounded length and duration, in a continuous action space. Ugur and Piater [36] proposed an emergence of a hierarchical structure of affordances. However, affordances were defined as a

list of discrete effects on objects, and the actions considered are manually coded actions. We would like to tackle continuous features of affordances and be able to learn compound actions in continuous action spaces. We extended their work with intrinsic motivation and planning.

4.2 Affordance Formalisation

To precise the formalisation of section2.1, let us note an affordance model $A(\mathcal{C}_i, \Omega_j, S_k)$, defined formally as a set of:

- an observation predictor $pred_A : \Omega \mapsto 0, 1$ that indicates whether A may be applied to an object o in the scene, accordingly to its visual or physical properties.
- a forward model M_A and an inverse model L_A. Both models learn the relationship between $\mathcal{C}_i$ and Ω_j knowing a context S_k.

4.3 CHIME Implementation

At each episode, the algorithm CHIME explores its environment by performing actions, observes the context and the outcomes obtained and processes the acquired data. One episode is composed of multiple iterations, and at each iteration one primitive action is performed.

Starting an episode, the robot decides stochastically either to explore the action space (Alg.2,l. 10), or the outcome space by generating a goal to attain during the episode (l. 5). When choosing the action space (l.10), the robot generates a random controllable $c \in \mathcal{C}$. When choosing to generate a goal (l. 5), an affordance A and a goal ω_g are selected, based on an interest metric detailed in [21]. The robot then uses its inverse models and its planning system to infer a sequence of controllables $c \in \mathcal{C}$ to be performed in order to reach ω_g.

In both cases, the robot converts c into a sequence of primitive actions $a = [a_1, \ldots, a_n] \in \mathcal{A}^n$.These actions are then executed by the robot (l. 18) to collect data into $\mathcal{D}$ (l. 27). $\mathcal{D}$ is used to improve existing affordances (l. 26), decide whether creating a new affordance is necessary, and update the intrinsic motivation mapping. These different processes are described in [21].

4.4 Developmental Emergence of Affordances

The results on setup3 (fig.3) show in [21] that CHIME can discover non-predefined affordances, and can use unbounded sequences of learned actions to complete all tasks. Even without predefinition of possible dependencies between observables and controllable features, the agent is able to discover the inputs and outputs of learnable models, and how they are related to each other. We show that these models emerge and are learned in a developmental order from the lowest to the highest level of hierarchy. We show that planning based on these emergent models enables the robot to infer a sequence of actions to complete complex tasks. We have compared our algorithm to the state of the art to outline two main properties : the developmental learning process and the hierarchy of the nested models of affordance.

Algorithm 2 CHIME layout

```
1: i = 0
2: loop
3:    D_episodic = ∅
4:    if H ≠ ∅ & Random() ≤ α then
5:       A = AffordanceSelection(H)
6:       ω = GoalSelection(A)
7:       ω_g = ObjectSelection(A, ω)
8:       c = Plan(ω_g)
9:    else
10:      C_i = RandomControllableSpace(C)
11:      c_r = RandomValue(C_i)
12:      c = [c_r]
13:   end if
14:   a = TransformToPrimitive(c)
15:   for a_k ∈ a do
16:      ω_before = GetObservations(Ω)
17:      a_i = [c_k] if c_k ∈ A else TransformToPrimitive(c_k)
18:      Execute(a_i)
19:      ω_after = GetObservations(Ω)
20:      ω_i = ω_after − ω_before
21:      s_i = s_before
22:      D_episode ← (a_i, ω_i, s_i)
23:      i += 1
24:   end for
25:   UpdateInterestMaps(D, D_episodic)
26:   UpdateAffordances(D, D_episodic)
27:   D ← D_episodic
28: end loop
```

The learning is based on active learning to collect data through new interactions with the environment, guided by the heuristics of intrinsic motivation. Once learned, these affordance control models are used to plan complex tasks with known or unknown objects, by using their physical properties to decide whether a learned affordance may be applied.

5 Who, What, How to imitate

Beyond mere autonomous intrinsically motivated exploration, we show that in high dimensional and unbounded task spaces, the performance is improved even more when the robot can imitate actions and procedures from teacher demonstrations, when it actively chooses between self-supervised intrinsic motivation and imitation learning strategies.

5.1 For primitive actions

Methods taking advantage of human demonstrations have shown that they can tackle more varied and large goal spaces, and that the learning could be bootstrapped [29]. The bootstrapping effect is all the more efficient when the learning

robot uses active learning based on intrinsic motivation to choose what to learn, and who, when and how to imitate. This choice on the source of information, has been called a *strategy*. The SGIM-ACTS algorithm proposed in [28] for multi-task learning has been applied to 3D object recognition by active learning using curiosity-driven manipulation by the iCub robot [30] or sound production and mother tongue imitation by a vocal tract [27].

5.2 For compound actions

For hierarchically organised tasks, we have studied how the limitations of IM-PB can be overcome with imitation learning. We proposed in [11, 9] the implementation SGIM-PB (Socially Guided Intrinsic Motivation by Procedure Babbling) that merged IM-PB with SGIM-ACTS. It is a strategic learner that can choose at every episode whether to autonomously explore the action or procedure space, but also whether to request a demonstration of actions or procedures to one of the available teachers. It can devise its curriculum to learn in a developmental manner different kinds of tasks. It can choose when imitation learning can be more beneficial than autonomous exploration, who among the different teachers are most expert in the field of knowledge it needs at the moment, and what kind of demonstrations is most beneficial. In terms of imitation learning, SGIM-PB self-determines who, what and when to imitate. Through setups fig.1,2, we show that demonstrations of procedures are beneficial to bootstrap the learning for tasks of the highest level of hierarchy. Another conclusion is that demonstrations seem the most beneficial when they are demonstrations of policies for simple tasks, and when they are indications of procedures (i.e. subtasks) for the most complex tasks.

6 Conclusion

Through this article, we have presented three implementations of an algorithmic framework for learning multiple control tasks through curriculum learning by discovering the dependencies between tasks and exploiting this hierarchy to transfer knowledge from the easy tasks to the compositional tasks. Table 1 summarises the properties of IM-PB, SGIM-PB and CHIME in learning to perform complex tasks with compound actions, in contrast to the state of the art. While IM-PB and SGIM-PB rely on a static set of controllable and outcome features and explore the dependencies between tasks to learn sequences of actions, CHIME builds dynamically its set of features from emergent control models that are then used to plan sequences of actions. Whereas IM-PB and CHIME rely only on autonomous exploration, SGIM-PB requests different kinds of demonstrations depending on the complexity of the task to the appropriate teacher. All three rely on a temporally abstract representation of compound actions using task hierarchy. They efficiently manage to learn them through the discovery of relationships between tasks to enable transfer of knowledge. We have also proposed a framework unifying both approaches and regrouping similar aspects, such as the intrinsically guided strategical learning and the hierarchical representation.

Algorithm	Reward	Cont. goal	Multi-task	Hierarchy	Actions	Imitation
Qlearning [34]	Ext.				Discrete	
Qlearning & curiosity [39]	Int.				Primitive	
Options [33]	Ext.				Option	
Skill-chaining [17]	Ext.			Yes	Option	
DQN [24]	Ext.				Discrete	
h-DQN [18]	Int.		Yes	Yes	Primitive	
Asynch. Actor Critic [25]	Ext. self eval				Primitive	
UVFA [32]	Ext.	Yes				
GEP-PG [6]	Int.	Yes	Yes			
CURIOUS [5]	Int.	Yes	Yes		Primitive	
IAC [31] RIAC [2]	Int.	Yes	Yes		Primitive	
SAGG-RIAC [3]	Int.	Yes	Yes		Primitive	
IMGEP [14]	Int.	Yes	Yes		Primitive	
SGIM-ACTS [28]	Int.	Yes	Yes		Primitive	Yes
IM-PB [10, 11]	Int.	Yes	Yes	Yes	Proced.	
SGIM-PB [9, 11]	Int.	Yes	Yes	Yes	Proced.	Yes
CHIME [21]	Int.	Yes	Yes	Yes	Planning	

Table 1: Comparison between the algorithms on : intrinsic vs extrinsic reward, the goal space is continuous (parametrised), single task vs multi-task problem, hierarchical learning, the action representation and whether imitation learning is used.

In future works, we consider developing an implementation of the SGIM-SAHT algorithm using all the described features: learning primitive actions and then planning sequences of them, then once learned, optimising directly these sequences owing to the procedure framework. The emergent subtasks will reduce the dependency on domain knowledge, whereas learning a representation of a compound action will result in better optimised policies and reduce the planning complexity. We also wish to compare the different performances and features from each algorithm on a common experiment.

Acknowledgements The research work presented is partially supported by the European Regional Developemnt Fund (ERDF) via the VITAAL Contrat Plan Etat Region, by Institut Mines Telecom (IMT) and by the French Ministry of Research.

References

1. Asada M, MacDorman KF, Ishiguro H, Kuniyoshi Y (2001) Cognitive developmental robotics as a new paradigm for the design of humanoid robots. Robotics and Autonomous Systems 37(2-3):185–193

2. Baranes A, Oudeyer Py (2009) R-IAC: Robust intrinsically motivated exploration and active learning. IEEE Transactions on Autonomous Mental Development 1(3):155–169
3. Baranes A, Oudeyer PY (2013) Active learning of inverse models with intrinsically motivated goal exploration in robots. Robotics and Autonomous Systems 61(1):49–73
4. Bengio Y, Louradour J, Collobert R, Weston J (2009) Curriculum learning. In: Proceedings of the 26th Annual International Conference on Machine Learning, ACM, New York, NY, USA, ICML '09, pp 41–48
5. Colas C, Fournier P, Sigaud O, Chetouani M, Oudeyer PY (2018) CURIOUS: Intrinsically Motivated Modular Multi-Goal Reinforcement Learning
6. Colas C, Sigaud O, Oudeyer PY (2018) GEP-PG: Decoupling Exploration and Exploitation in Deep Reinforcement Learning Algorithms. In: International Conference on Machine Learning (ICML), Stockholm, Sweden
7. Dongshin Kim, Jie Sun, Sang Min Oh, Rehg JM, Bobick AF (2006) Traversability classification using unsupervised on-line visual learning for outdoor robot navigation. In: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., pp 518–525
8. Duminy N, Nguyen SM, Duhaut D (2016) Strategic and interactive learning of a hierarchical set of tasks by the Poppy humanoid robot. In: ICDL-EPIROB 2016 : 6th Joint IEEE International Conference Developmental Learning and Epigenetic Robotics
9. Duminy N, Nguyen SM, Duhaut D (2018) Effects of social guidance on a robot learning sequences of policies in hierarchical learning. In: IEEE (ed) International Conference on Systems Man and Cybernetics
10. Duminy N, Nguyen SM, Duhaut D (2018) Learning a set of interrelated tasks by using sequences of motor policies for a strategic intrinsically motivated learner. In: IEEE International on Robotic Computing, pp 288–291
11. Duminy N, Nguyen SM, Duhaut D (2019) Learning a set of interrelated tasks by using a succession of motor policies for a socially guided intrinsically motivated learner. Frontiers in Neurorobotics 12:87
12. Elman J (1993) Learning and development in neural networks: The importance of starting small. Cognition 48:71–99
13. Forestier S, Oudeyer PY (2016) Curiosity-driven development of tool use precursors: a computational model. In: 38th Annual Conference of the Cognitive Science Society (CogSci 2016), pp 1859–1864
14. Forestier S, Mollard Y, Oudeyer P (2017) Intrinsically motivated goal exploration processes with automatic curriculum learning. CoRR abs/1708.02190
15. Gibson J (1977) The theory of affordances. In: Perceiving, Acting, and Knowing, Robert Shaw and John Bransford
16. Jamone L, Ugur E, Cangelosi A, Fadiga L, Bernardino A, Piater J, Santos-Victor J (2016) Affordances in psychology, neuroscience, and robotics: A survey. IEEE Transactions on Cognitive and Developmental Systems 10(1):4–25
17. Konidaris G, Barto AG (2009) Skill Discovery in Continuous Reinforcement Learning Domains using Skill Chaining. Advances in Neural Information Processing Systems pp 1015–1023
18. Kulkarni TD, Narasimhan K, Saeedi A, Tenenbaum J (2016) Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In: Advances in neural information processing systems, pp 3675–3683
19. Levine S, Pastor P, Krizhevsky A, Ibarz J, Quillen D (2018) Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. The International Journal of Robotics Research 37(4-5):421–436
20. Lungarella M, Metta G, Pfeifer R, Sandin iG (2003) Developmental robotics: a survey. Connection Science 15(4):151–190
21. Manoury A, Nguyen SM, Buche C (2019) Hierarchical affordance discovery using intrinsic motivation. In: Human Agent Interaction
22. Miller KA, Deci EL, Ryan RM (1988) Intrinsic Motivation and Self-Determination in Human Behavior. Contemporary Sociology 17(2):253
23. Mitriakov A, Papadakis P, Nguyen SM, Garlatti S (2020) Staircase traversal via reinforcement learning for active reconfiguration of assistive robots. In: IEEE (ed) World Congress on Computational Intelligence
24. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533
25. Mnih V, Badia AP, Mirza M, Graves A, Lillicrap TP, Harley T, Silver D, Kavukcuoglu K (2016) Asynchronous methods for deep reinforcement learning. CoRR abs/1602.01783
26. Montesano L, Lopes M (2009) Learning grasping affordances from local visual descriptors. In: 2009 IEEE 8th International Conference on Development and Learning, pp 1–6
27. Moulin-Frier C, Nguyen SM, Oudeyer PY (2014) Self-organization of early vocal development in infants and machines: The role of intrinsic motivation. Frontiers in Psychology 4(1006)
28. Nguyen SM, Oudeyer PY (2012) Active choice of teachers, learning strategies and goals for a socially guided intrinsic motivation learner. Paladyn Journal of Behavioural Robotics 3(3):136–146
29. Nguyen SM, Oudeyer PY (2014) Socially guided intrinsic motivation for robot learning of motor skills. Autonomous Robots 36(3):273–294
30. Nguyen SM, Ivaldi S, Lyubova N, Droniou A, Gerardeaux-Viret D, Filliat D, Padois V, Sigaud O, Oudeyer PY (2013) Learning to recognize objects through curiosity-driven manipulation with the icub humanoid robot. In: IEEE International Conference on Development and Learning - Epirob
31. Oudeyer PY, Kaplan F, Hafner V (2007) Intrinsic motivation systems for autonomous mental development. IEEE Transactions on Evolutionary Computation 11(2):265–286
32. Schaul T, Horgan D, Gregor K, Silver D (2015) Universal value function approximators. In: Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37, JMLR.org, ICML'15, pp 1312–1320
33. Sutton REJ R S, A K (2006) Temporal abstraction in temporal-difference networks. In: Advances in Neural Information Processing Systems 18
34. Sutton RS, Barto AG (1998) Reinforcement Learning: an introduction. MIT Press
35. Sutton RS, Precup D, Singh S (1999) Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. Artificial Intelligence 112:181 – 211
36. Ugur E, Piater J (2016) Emergent structuring of interdependent affordance learning tasks using intrinsic motivation and empirical feature selection. IEEE Transactions on Cognitive and Developmental Systems
37. Uğur E, Şahin E (2010) Traversability: A case study for learning and perceiving affordances in robots. Adaptive Behavior 18(3):258–284
38. Ugur E, Piater J, Sahin E, Oztop E (2009) Affordance learning from range data for multi-step planning. In: International Conference on Epigenetic Robotics
39. Vigorito C, Barto A (2010) Intrinsically Motivated Hierarchical Skill Learning in Structured Environments. IEEE Transactions on Autonomous Mental Development 2(2):132–143
40. Zech P, Renaudo E, Haller S, Zhang X, Piater J (2019) Action representations in robotics: A taxonomy and systematic classification. International Journal of Robotics Research 38(5):518–562