Explore to See, Learn to Perceive, Get the Actions for Free: SKILLABILITY

Varun R. Kompella, Marijn F. Stollenga, Matthew D. Luciw and Juergen Schmidhuber

Abstract— **How can a humanoid robot autonomously learn and refine multiple sensorimotor skills as a byproduct of curiosity driven exploration, upon its high-dimensional unprocessed visual input? We present SKILLABILITY, which makes this possible. It combines the recently introduced Curiosity Driven Modular Incremental Slow Feature Analysis (Curious Dr. MISFA) with the well-known options framework. Curious Dr. MISFA's objective is to acquire abstractions as quickly as possible. These abstractions map high-dimensional pixel-level vision to a low-dimensional manifold. We find that each learnable abstraction augments the robot's state space (a set of poses) with new information about the environment, for example, when the robot is grasping a cup. The abstraction is a function on an image, called a slow feature, which can effectively discretize a high-dimensional visual sequence. For example, it maps the sequence of the robot watching its arm as it moves around, grasping randomly, then grasping a cup, and moving around some more while holding the cup, into a step function having two outputs: when the cup is or is not currently grasped. The new state space includes this grasped/not grasped information. Each abstraction is coupled with an option. The reward function for the option's policy (learned through Least Squares Policy Iteration) is high for transitions that produce a large change in the step-function-like slow features. This corresponds to finding bottleneck states, which are known good subgoals for hierarchical reinforcement learning - in the example, the subgoal corresponds to grasping the cup. The final skill includes both the learned policy and the learned abstraction. SKILLABILITY makes our iCub the first humanoid robot to learn complex skills such as to topple or grasp an object, from raw high-dimensional video input, driven purely by its intrinsic motivations.**

I. Introduction

In reinforcement learning (RL; [1, 2]), an important problem is to learn distinct skills without any external guidance [3]. Skills can act as building blocks for learning more complex skills [4], in a hierarchical learning system. The *options* framework [5] formalizes skills as RL policies, active within a subset of the state space, which can terminate at subgoals, after which another option takes over. When the agent has high-dimensional input, like vision, a skill will require a dimensionality reducing mapping called a *feature abstraction* (or simply an abstraction), so that policy learning becomes tractable [6]. There have been attempts to find skills with abstractions in domains such as those of humanoid robotics [6, 7, 8, 9]. These rely, however, either on demonstrations and explicit task-descriptions from humans, or a readily available state-description and externally defined goals for guidance.

V. R. Kompella*, M. F. Stollenga, M. D. Luciw and J. Schmidhuber are with the IDSIA, Galleria 2, Manno-Lugano 6928, Switzerland. Email: {varun, marijn, matthew, juergen}@idsia.ch. *Corresponding author.

We present here SKILLABILITY, which combines the recently introduced Curiosity Driven Modular Incremental Slow Feature Analysis (Curious Dr. MISFA; [10, 11]), in which an agent explores its environment and learns slow-feature-based abstractions, with the options framework, used to build the abstractions into skills. Curious Dr. MISFA comprises Incremental Slow Feature Analysis (IncSFA; [12, 13]), and Artificial Curiosity (AC; [14, 15]),

IncSFA is used for learning slow features incrementally from high-dimensional sensory input. Slow features [16] can encode underlying causes of the rapidly changing raw sensory inputs. They are based on the slowness principle [17, 18, 19], which posits that these underlying causes of a signal can be extracted via a temporal consistency objective. For example, gray-scale pixel values of a video typically change quickly compared to more abstract latent variables, such as the position of a moving objects. Slow feature outputs over a visual sequence in which an event occurs can take the form of a step function, encoding the two possible states of the event. With such a feature, potentially useful information about the states of objects in the environment can be encoded.

AC is used for for directing the agent's actions. The theory of AC introduces a mathematical formalism for describing curiosity and creativity, wherein agents are interested in the learnable but as yet unknown aspects of their environment, but are disinterested in the predictable or inherently unlearnable aspects. More specifically, a creative agent needs two learning components: an adaptive encoder of the growing history of perceptions and actions and a reinforcement learner. The *learning progress* of the encoder, which in our case is IncSFA, becomes an intrinsic reward for the reinforcement learner [14]. Curious Dr. MISFA learns multiple slow feature abstractions directly on the raw visual input, in order from lowest to highest learning difficulty, which is predicted by the theory of AC.

The novel contribution of this work is to show how skills can be built upon the slow-feature-based abstractions. In SKILLABILITY, each feature learned by Curious Dr. MISFA augments the robot's default state space, which in our case is a set of poses learned using Task Relevant Roadmaps [20]. This augmented state space is further clustered to create new, distinct states. A Markovian transition model is learned by exploring the new state space. The reward function is also learned through exploration, with the agent being intrinsically rewarded for making state-transitions that produce a large change in the step-function-like slow feature outputs. This specialized reward function is used to build the skill policies, to drive the agent to states where such transitions

will occur. These states resemble *bottleneck states*, i.e., "doorways", which are known to be good subgoals in the absence of externally imposed goals [21, 22]. Once the transition and reward functions are learned, the skill's policy is learned via Least-Squares Policy Iteration (LSPI; [23]). Experimental results conducted using the iCub robot illustrate how SKILLABILITY leads to the learning of two skills, of cup toppling and cup grasping.

The rest of this paper is organized as follows. Section II presents an overview of the Curious Dr. MISFA. Section III presents our approach to do skill learning using Curious Dr. MISFA. Section IV contains results of experiments on our iCub; Section V concludes the paper.

II. Curious Dr. MISFA

In this section, we briefly review Curiosity Driven Modular Incremental Slow Feature Analysis.

v

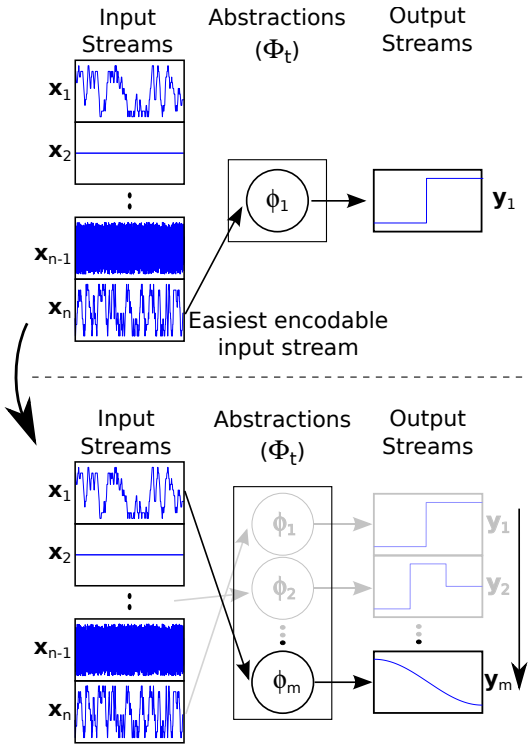

Fig. 1. Learning Process of Curious Dr. MISFA. Given a set of input streams, abstractions are learned in order of learning difficulty, from the easiest to the most difficult. The result is a set of abstractions $\{\phi_1, ..., \phi_m\}$ learned sequentially. However, since the learning difficulty of input streams are not known *a priori*, the learning process involves estimating not just the abstractions, but also the order in which the input streams are encoded.

From a set of high-dimensional input streams $X = \{\mathbf{x_1}, ..., \mathbf{x_n} : \mathbf{x_i}(t) \in \mathbb{R}^I\}$, Curious Dr. MISFA learns abstractions $\Phi_t = \{\phi_1, ..., \phi_m; m \leq n\}$ in order of learning difficulty, where t denotes time. Each abstraction ϕ_i takes the form of a set of orthogonal slow features, is unique, and maps one or more input streams $\mathbf{x} \in X$ to low-dimensional output $\mathbf{y}(t) \in \mathbb{R}^J, J \ll I$, where $\mathbf{y}(t) = \phi_i(\mathbf{x}(t))$. Since the learning difficulty of the input streams are not known

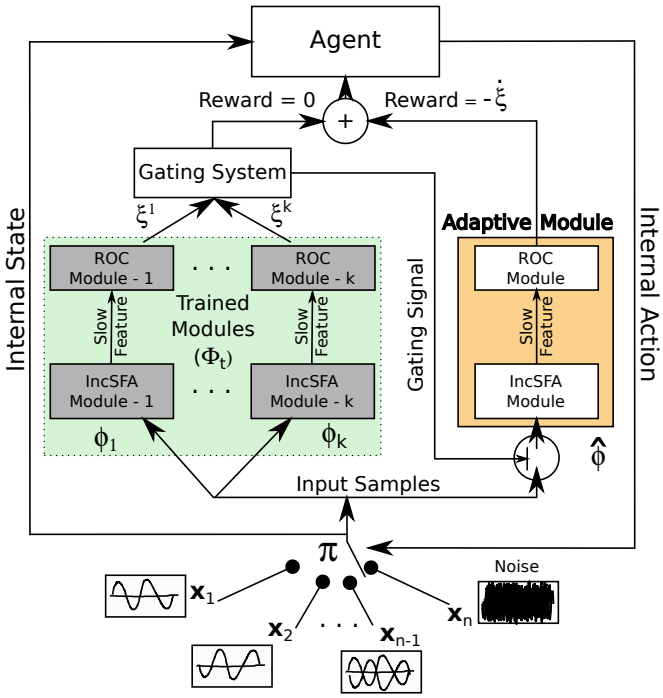

Fig. 2. The architecture of Curious Dr. MISFA includes (a) an RL agent that learns an input stream selection policy from intrinsic rewards, (b) an adaptive IncSFA-ROC module that updates an abstraction based on the incoming input observations, and (c) a gating system that prevents learning duplicate abstractions.

a priori, the learning process involves estimating not just the abstractions, but also identifying the current easiest-to-learn input stream, which is not redundant with already learned abstractions. To this end, Curious Dr. MISFA uses Reinforcement Learning to learn an optimal *input stream selection policy* $\boldsymbol{\pi}$.

Figure 2 shows the architecture of Curious Dr. MISFA, which includes (a) an RL agent that generates π based on intrinsic rewards, (b) an adaptive IncSFA-Robust Online Clustering (ROC; [24, 25]) module that updates an abstraction based on the incoming input observations, and (c) a gating system that prevents encoding inputs that have been previously encoded.

The Curious Dr. MISFA RL *agent* (not the same as a skill-learning agent, which we will discuss in Sec. III) is within an internal environment that has a set of discrete states $\mathcal{S}^I = \{s_1^I, ..., s_n^I\}$, equal to the number of input streams. In each state s_i^I, the agent is allowed to take only one of the two actions ($\mathcal{A}^I$): *stay* or *switch*. The action *stay* causes the agent to stay in the same state, while *switch* randomly shifts the agent's state to one of the other states. In between actions, at state s_i^I, the estimator receives a fixed τ time step sequence of input observations ($\mathbf{x}$) of the corresponding input stream $\mathbf{x}_i$. The adaptive abstraction $\hat{\phi} \notin \Phi_t$ updates based on $\mathbf{x}$ via the IncSFA-ROC abstraction-estimator.

The agent receives intrinsic rewards for transitions proportional to the corresponding learning progress made by IncSFA-ROC algorithm. These intrinsic rewards contribute to estimating a reward function, which is used to compute the

input stream selection policy π. This policy is used to select the input stream for the next iteration, yielding new samples $\mathbf{x}$. These new samples, if not encodable by previously learned abstractions, are used to update the adaptive abstraction. The updated abstraction $\hat{\phi}$ is added to the abstraction set Φ_t, when the IncSFA-ROC's estimation error falls below a low threshold δ. When an abstraction is added, a new adaptive abstraction $\hat{\phi}$ is instantiated and the process continues.

The following sections discuss more details on different parts of Curious Dr. MISFA algorithm.

A. Abstraction-Estimator: IncSFA-ROC

Curious Dr. MISFA's abstraction-estimator is the Incremental Slow Feature Analysis coupled with a Robust Online Clustering algorithm. IncSFA is used to learn real-valued abstractions of the input, while ROC is used to learn a discrete mapping from the current abstraction activations to the agent's *subjective state-space* $\mathcal{S}^{\dagger}$, which contains discrete (clustered) values of all previously encoded abstractions.

IncSFA: Slow feature analysis [16] is an unsupervised learning technique that extracts features from an input stream with the objective of maintaining a slowly-changing feature response over time. SFA is concerned with the following optimization problem:

Given an I-dimensional input signal $\mathbf{x}(t) = [x_1(t), ..., x_I(t)]^T$, find a set of J instantaneous real-valued functions $\mathbf{g}(\mathbf{x}) = [\mathbf{g_1}(\mathbf{x}), ..., \mathbf{g_J}(\mathbf{x})]^{\mathbf{T}}$, which together generate a J-dimensional output signal $\mathbf{y}(t) = [y_1(t), ..., y_J(t)]^T$ with $y_j(t) := g_j(\mathbf{x}(t))$, such that for each $j \in \{1, ..., J\}$

$$\Delta_j := \Delta(y_j) := \langle \dot{y}_j^2 \rangle \quad \text{is minimal}- \tag{1}$$

under the constraints

$$\langle y_j \rangle = 0 \quad \text{(zero mean)}, \tag{2}$$

$$\langle y_j^2 \rangle = 1 \quad \text{(unit variance)}, \tag{3}$$

$$\forall i < j : \langle y_i y_j \rangle = 0 \quad \text{(decorrelation and order)}, \tag{4}$$

with $\langle \cdot \rangle$ and $\dot{y}$ indicating temporal averaging and the derivative of y, respectively.

The goal is to find instantaneous functions g_j generating different output signals that are as slowly varying as possible. The decorrelation constraint (4) ensures that different functions g_j do not code for the same features. The other constraints (2) and (3) avoid trivial constant output solutions.

SFA operates on the covariance of observation derivatives, so it scales with the size of the observation vector instead of the number of states. SFA is originally realized as a batch method, requiring all data to be collected before processing. The algorithmic complexity is cubic in the input dimension I. In contrast, IncSFA has a linear update complexity [13], and can adapt the features to new observations, achieving the slow-feature objective robustly in open-ended learning environments.

ROC maintains estimates (via nearest neighbor) of slow-feature outputs. These clusters act as discrete subjective states, in space $\mathcal{S}^{\dagger}$. In learning, ROC is similar to an incremental K-means algorithm — a set of cluster centers is maintained, and with each new input, the most similar cluster center (the winner) is adapted to become more like the input. Unlike k-means, with each input it follows the adaptation step by *merging* the two most similar cluster centers, and *creating a new cluster center* at the latest input. In this way, ROC can quickly adjust to non-stationary input distributions by directly adding a new cluster for the newest input sample, which may mark the beginning of a new input process.

B. Estimation Error and Curiosity Reward

Each ROC-Estimator node j has an associated error ξ^j. These errors are initialized to 0 and then updated whenever the node is activated by: $\xi^j(t) = \min_w \|\mathbf{y}(t) - \mathbf{v}_w\|$, where $\mathbf{y}(t)$ is the slow-feature output vector, $\mathbf{v}_w$ is the estimate of the w^{th} cluster of the activated node and $\|.\|$ represents L^2 norm. The total estimation error is calculated as the sum of stored errors of the nodes: $\xi(t) = \sum_{j=1}^{p} \xi^j(t)$. The agent receives rewards proportional to the *derivative* of the total estimation error, which motivates it to continue towards yielding a learnable abstraction. The agent's reward function is computed at every iteration from the curiosity rewards ($\dot{\xi}$) as follows:

$$R^I(s^I, s^I_{\text{-}}, a) := (1 - \eta)\ R^I(s^I, s^I_{\text{-}}, a) + \eta \sum_{t}^{t+\tau} -\dot{\xi}(t)$$

where $0 < \eta < 1$ is a discount factor, τ is the duration of the current option until its termination, $(s^I, s^I_{\text{-}}) \in \mathcal{S}^I$ and $a \in \{\text{stay, switch}\}$.

C. Input Stream Selection Policy

The transition-probability model P^I of the internal environment is similar to a complete graph and is given by:

$$\left(P^I_{i,j,\text{stay}},\ P^I_{i,j,\text{switch}}\right) = \begin{cases} (1,\ 0), & \text{if } i = j \\ (0,\ \frac{1}{N-1}), & \text{if } i \neq j \end{cases} \tag{5}$$

$\forall i, j \in [1, ..., N]$. Using the current updated model of the reward function R^I and the internal state transition-probability model P^I, we use model-based Least Squares Policy Iteration [23] to generate the agent's internal-policy ($\pi : \mathcal{S}^I \times \{\text{stay}, \text{switch}\} \rightarrow [0, 1]$) for the next iteration. The agent uses a decaying ϵ-greedy strategy over the internal policy to carry out an internal-action (stay or switch) for the next iteration.

D. Module Freezing and New Module Creation

Once the adaptive (training) module's $\hat{\phi}$ estimation error gets lower than a threshold δ, the agent freezes and saves the IncSFA-ROC module, resets the ϵ-greedy value and starts training a new module.

E. Gating System and Abstraction Assignment

The already trained (frozen) modules represent our learned library of abstractions Φ_t. If a *trained* module's estimation error within an option is below the threshold δ, that option is assigned with that module's abstraction and the adaptive

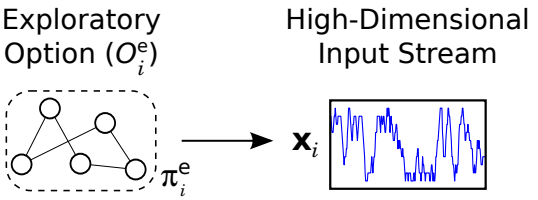

Fig. 3. Each exploratory option O_i^e when executed, generates a high-dimensional observation stream $\mathbf{x}_i$, which is an input to Curious Dr. MISFA algorithm.

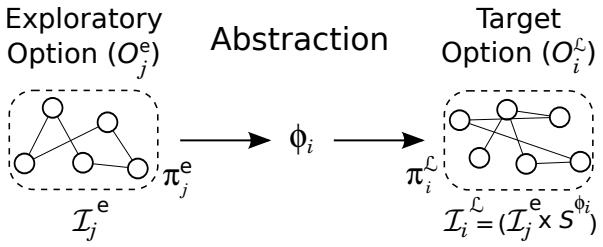

Fig. 4. Once an abstraction ϕ_i is learned corresponding to an exploratory-option stream $\mathbf{x_j} = \mathcal{U}(\mathcal{P}(s, \pi_j^e(s^\dagger)))$, its feature-activation values are quantized to a set of discrete feature-states $\mathcal{S}^{\phi_i}$. A target option is then constructed with the product state-space $\mathcal{I}_i^{\mathcal{L}} = (\mathcal{I}_j^e \times \mathcal{S}^{\phi_i})$ and a learned target-policy $\pi_i^{\mathcal{L}}$.

training module $\hat{\phi}$ will be prevented from learning via a "gating signal" (see Figure 2). There is no intrinsic reward in this case. Hence the training module $\hat{\phi}$ will encode only data from input streams that were not encoded earlier. Input that no other trained modules can encode, serves to train the adaptive module.

III. Skill Learning using SKILLABILITY

SKILLABILITY combines the options framework with Curious Dr. MISFA to autonomously learn skills from high-dimensional video data, in the absence of any external guidance. Curious Dr. MISFA provides slow-feature-based abstractions, and the corresponding abstraction outputs are discretized and used to construct the agent's *subjective* state space, for each abstraction set. Options are learned in the subjective spaces, with a policy that maximizes the variation in the slow feature output.

Recall that $\mathcal{S}^\dagger$ denotes the agent's discrete subjective state space that contains previously encoded abstractions of the high-dimensional observations along with the predefined states. The agent additionally has a set of n options that are pre-defined exploratory behaviors over different parts of the agent's subjective state space, called the *exploratory option set* $\mathcal{O}^e = \{O_1^e, ..., O_n^e\}$. More formally, an exploratory option is a tuple $\langle \mathcal{I}_i^e, \beta_i^e, \pi_i^e \rangle$, where $\mathcal{I}_i^e \subseteq \mathcal{S}^\dagger$ is the initiation set comprising states where the option is available, $\beta_i^e : \mathcal{S}^\dagger \mapsto [0,1]$ is the option termination condition, which will determine where the option terminates (e.g., some probability in each state), and $\pi_i^e : \mathcal{I}_i^e \times \mathcal{A} \to [0,1]$ is a pre-defined option's *exploration* policy, such as a random walk within the applicable state space. *Executing an option* implies that the agent follows the option's policy until its termination condition is met.

As shown in Figure 3, each exploratory option O_i^e when executed, generates a high-dimensional observation stream: $\mathbf{x_i}(t) = \mathcal{U}(\mathcal{P}(s, \pi_i^e(s^\dagger)))$, where $s^\dagger \in \mathcal{I}_i^e$ is the agent's current subjective state while executing O_i^e at time t, and s is the corresponding environment state $s \in \mathcal{S}$.

We have not yet discussed where the input streams might come from. In this case, they come from executing each of these exploration policies. Let $X = \{\mathbf{x_1}, ..., \mathbf{x_n}\}$ denote the set of n different I-dimensional input observation streams generated by the n option's exploration policies. Curious Dr. MISFA algorithm will learn abstractions sequentially from the input streams generated upon executing the selected options. Therefore, the internal states as discussed in Sec. II now correspond to the n exploratory options. When the agent shifts to one of the internal states, it executes the corresponding exploratory option. The termination condition is set to the fixed time-out τ.

Given the input exploratory-option set, the agent is trying to learn a *target-option set* $\mathcal{O}^{\mathcal{L}} = \{O_i^{\mathcal{L}}\}$, which corresponds to a set of *skills*. Each *target option* $O_i^{\mathcal{L}}$, unlike an exploratory option, has additionally a learned abstraction $\phi_i \in \mathbb{R}^{I \times J}$ that maps the input observation ($\mathbf{x}(t) \in \mathbb{R}^I$) to J-dimensional ($J \ll I$) feature output ($\mathbf{y}(t) \in \mathbb{R}^J$). A *target option* is therefore a tuple $\langle \mathcal{I}_i^{\mathcal{L}}, \beta_i^{\mathcal{L}}, \phi_i, \pi_i^{\mathcal{L}} \rangle$. $\mathcal{I}_i^{\mathcal{L}} \subseteq (\mathcal{S}^\dagger \times \mathcal{S}^{\phi_i})$ is the target-option's initiation set defined over *augmented* state-space $(\mathcal{S}^\dagger \times \mathcal{S}^{\phi_i})$, where $\mathcal{S}^{\phi_i}$ denotes a set of discretized states in the feature-output ($\mathbf{y}$) space. β_i is the option's termination condition, and $\pi_i^{\mathcal{L}} : (\mathcal{S}^\dagger \times \mathcal{S}^{\phi_i}) \times \mathcal{A} \mapsto [0,1]$ is a learned target-policy that generates a *predictable* observation-stream $\mathcal{U}(\mathcal{P}(s, \pi_i^{\mathcal{L}}(s^\dagger)))$, $s \in \mathcal{S}$, $s^\dagger \in \mathcal{I}_i^{\mathcal{L}}$.

Learning a Target Option Once an abstraction ϕ_i is learned, corresponding to the easiest but not-yet learned exploratory-option stream $\mathbf{x_j} = \mathcal{U}(\mathcal{P}(s, \pi_j^e(s^\dagger)))$, its feature-activation values ($\mathbf{y_i} = \phi_i \cdot \mathbf{x_j}$) are discretized to a set of discrete feature-states $\mathcal{S}^{\phi_i}$. A target option is then constructed (Figure 4). The initiation set is simply the product state-space: $\mathcal{I}_i^{\mathcal{L}} = (\mathcal{I}_j^e \times \mathcal{S}^{\phi_i})$. The termination condition is problem specific. The target option policy $\pi_i^{\mathcal{L}} : \mathcal{I}_i^{\mathcal{L}} \times \mathcal{A} \mapsto [0,1]$ must be done in such a way to facilitate any subsequent reuse of the target option.

A target-policy $\pi_i^{\mathcal{L}}$ is developed using Model-based Least-Squares Policy Iteration Technique (LSPI; [23]) using estimated transition and reward models. The target-option's transition model ($P^{O_i^{\mathcal{L}}}$) has been continually estimated from the (s, a, s') samples generated via the exploratory-option's policy π_j^e. The reward function is learned from intrinsic rewards, which are proportional to the *change of subsequent feature activations*:

$$r^{O_i^{\mathcal{L}}}(t) = \|\mathbf{y_i}(t) - \mathbf{y_i}(t-1)\| \tag{6}$$

Since the slow feature outputs often resemble step functions, intrinsically rewarding states include *bottleneck states*, which are known to be good *subgoals* for hierarchical reinforcement learning in the absence of external goals [22].

Figure 5 illustrates the complete learning process. Given a set of input exploratory options generating n input streams,

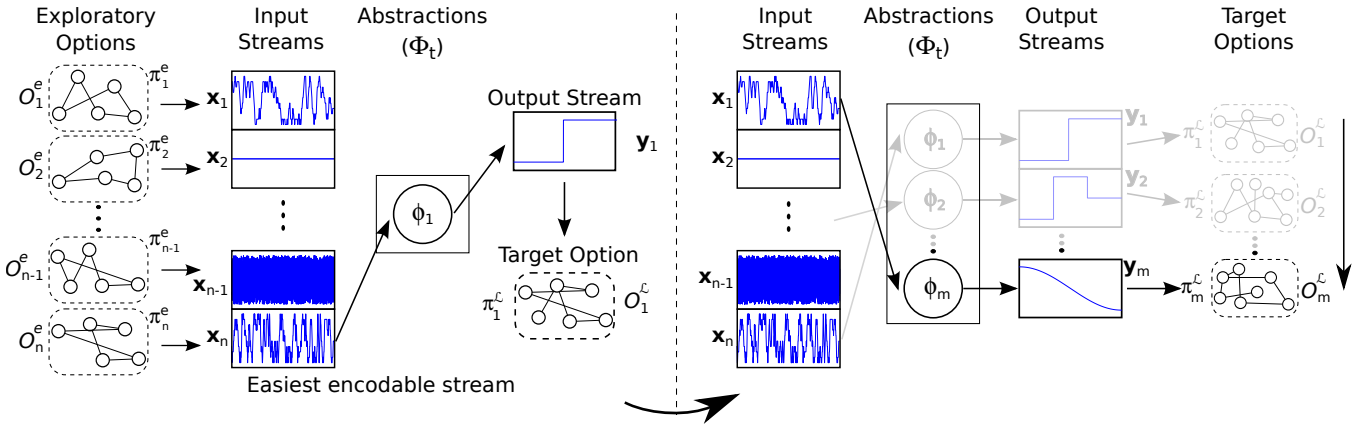

Fig. 5. Skill Learning. Given a set of input exploratory options generating n input streams, abstractions are learned in order of learning difficulty, from easiest to most difficult ones. For each abstraction, a corresponding target option is learned, resulting in a target-option skill set.

abstractions are learned in order of learning difficulty, from the easiest to the most difficult one. When an abstraction is learned, a corresponding target option is learned. This learning process results in learning skills in order of learning difficulty from high-dimensional input streams and avoiding uncompressible noisy streams, therefore increasing the number of skills acquired by the agent over time.

IV. Experimental Results

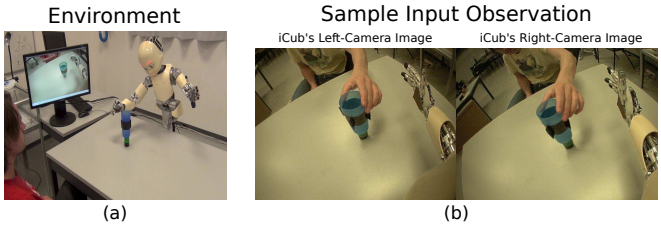

Fig. 6. (a) Our iCub robot is placed next to a table, with an object (a plastic cup) in reach of its right arm and within its field-of-view. (b) Sample input images captured from both left and right iCub camera-eyes are an input to the algorithm.

We present here experimental results using an iCub humanoid. More studies on the types of representations learned by the IncSFA algorithm and curiosity-based abstraction learning with Curious Dr. MISFA can be found elsewhere [10, 11, 13, 26]. The results here show how the iCub translates the abstractions learned using SKILLABILITY to learn skills of toppling a cup and grasping a cup.

Environment: We pre-selected a safe environment for the iCub to explore, yet the iCub is mostly unaware of the environment properties. The iCub is placed next to a table, with the cup in reach of its right arm and within its field-of-view (Figure 6(a)). The cup topples over upon contact, and the cup often ends up in one of a few locations afterwards, so the resulting images after toppling are fairly predictable. There is a human interactor present, who monitors the robot's safety and replaces the cup in its original position after it is toppled. The iCub is not given the structure of the environment, therefore it does not "know" that the plastic-cup and the interactor exist. It continually observes the gray-scale pixel values from the high-dimensional images (75×100) captured by the left and right camera eyes (Figure 6(b)). In addition to the interactor and the cup, it also cannot recognize its own moving hand in the incoming image stream, as shown in the Figure 6(b).

Task-Relevant Roadmap We do not induce exploration at the level of joint angles, due to the complexity of the robot's joint space. Instead we give the robot a map of poses to move between. This compressed actuator joint-space representation is called a Task-Relevant Roadmap (TRM; [20]). This map contains a family of iCub postures that adhere to relevant constraints. The TRM is grown offline by repeatedly optimizing cost-functions that represent the constraints, using a Natural Evolution Strategies (NES; [27]) algorithm, such that the task-space is covered. This allows us to deal with complex cost-functions and the full 41 degrees-of-freedom of the iCub's upper body. The constraints used: (a) the iCub's hand is positioned on a plane parallel to the table while keeping its palm oriented horizontally, (b) the left hand is kept within a certain region to keep it out of the way, and (c) the head is pointed towards the table. The task-space of the TRM comprises the x- and y-position of the hand, which forms the initial discretized 10×5 subjective space $\mathcal{S}^{\dagger}$. The action space contains 6 actions: move *North*, *East*, *South*, *West*, *Hand-close* and *Hand-open*.

Because the full body is used, the movements look more dynamic, but as a consequence, the head moves around and looks at the table from different directions, making the task more difficult. Even so, IncSFA still finds the resulting regularities in the raw camera input stream, and the skill learner continues to learn upon these regularities, without any external rewards.

Experiment parameters: We used a fixed parameter setting for the entire experiment.

IncSFA Algorithm: CCIPCA used learning rate $1/t$ with amnesic parameter 0.4, while the MCA learning rate was 0.01. CCIPCA did variable size dimension reduction by calculating how many eigenvalues would be needed to keep 99% of the input variance — typically this was between 5-10 — so the 7500 pixels could be effectively reduced to

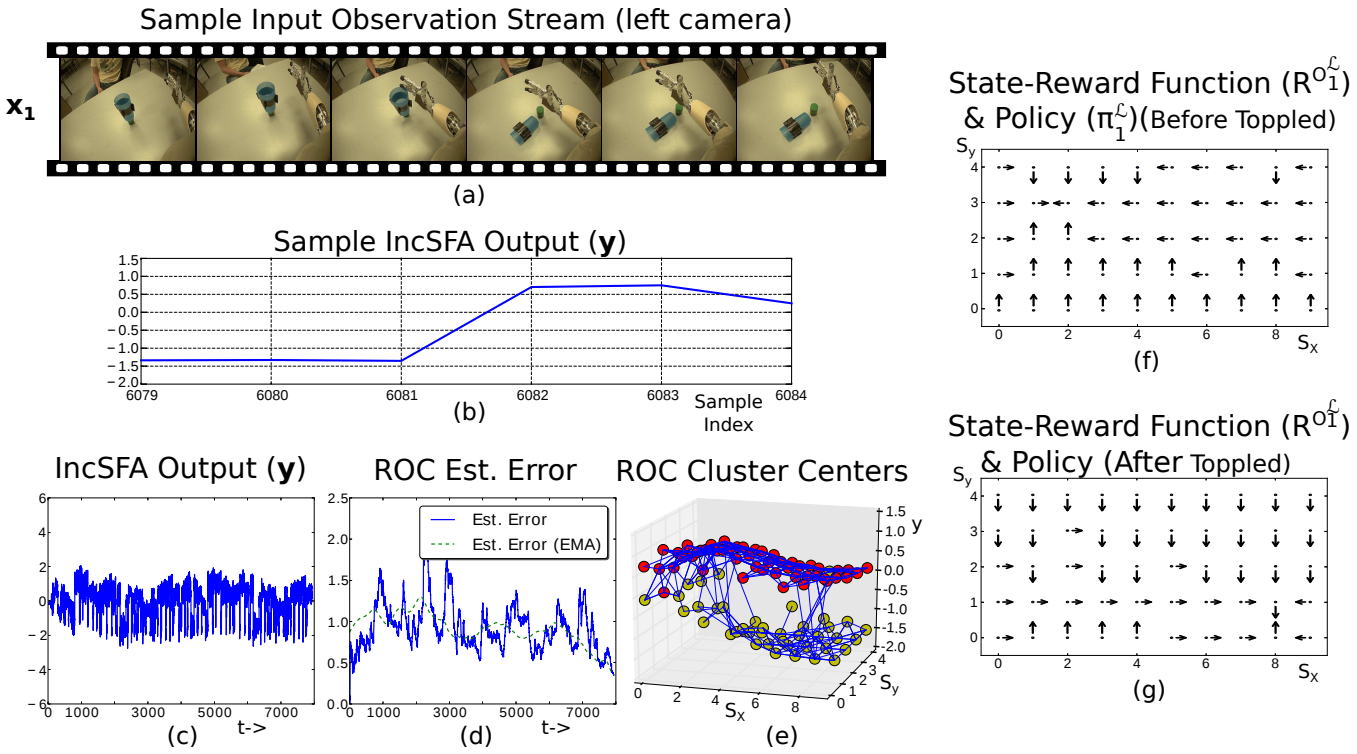

Fig. 7. **Experiment 1: Topple Skill.** (a) A sample image stream of the iCub's left-eye camera showing the topple-event, and (b) the corresponding sample IncSFA output after a few iterations. The result is a step-function encoding the topple-event. (c) IncSFA output over algorithm execution time. (d) ROC estimation error over algorithm execution time. The estimation error eventually drops below the threshold ($\delta = 6$). (e) The resultant ROC cluster centers. (f)-(g) State-reward function estimate and the learned target-option's policy.

only about 10 dimensions. The output dimension was set to 1, therefore, we use only the first IncSFA feature as an abstraction.

ROC Clustering Algorithm: Each clustering implementation had its maximum number of clusters set to $N^{max} = 3$. The estimation error threshold, below which the current module is saved and a new module is created, was set to $\delta = 0.3$. The amnesic parameter was set to $\beta^{amn} = 0.01$.

Internal Reinforcement Learner (LSPI): The initial ϵ-greedy value was 1.0, with a 0.995 decay multiplier. The window-averaging time constant was set to $\tau = 20$, that is, 20 sample images were used to compute the window-averaged progress error ξ and the corresponding curiosity-reward.

Skill-Learning Reinforcement Learner (LSPI): Each feature-abstraction output values were quantized to either $(-1, 1)$, therefore into two $|\mathcal{S}^{\phi_i}| = 2$ feature-states.

A. Experiment 1: iCub Learns to Topple a Cup

First, the iCub learns a skill to topple the cup. The iCub's subjective space ($\mathcal{S}^\dagger$) at $t = 0$ is a 10×5 grid found using TRM. The plastic-cup placed is roughly placed around $(2, 2)$ grid-point on the table. The input exploratory-option set ($\mathcal{O}^e$) has only one exploratory option: $\mathcal{O}^e = \{O_1^e\}$. The exploratory option terminates after $\tau = 20$ time-steps after it begins execution. The internal state-space at $t = 0$ is $\mathcal{S}^I = \{s_1^I\}$, where s_1^I corresponds to the exploratory option O_1^e. The exploratory-option's policy π_1^e is a random-walk over the subjective-state space $\mathcal{S}^\dagger$. It explores by taking one of the six actions (*North*, *East*, *South*, *West*, *Hand-close* and *Hand-open*) and gets high-dimensional images from its camera-eyes. The exploration causes the outstretched hand to eventually displace and topple the plastic-cup placed on the table. It continues to explore and after an arbitrary amount of time-steps the interactor replaces the cup to its original position. After every τ time-steps, currently executing option terminates. Since, there is only one exploratory option the iCub re-executes the same option. Figure 7(a) shows a sample input image-stream of only the left camera[1].

The outcome of IncSFA abstraction-learning over such an input sequence is a step function (Figure 7(b)), which, when quantized, indicates the pose of the cup (toppled vs non-toppled). Figure 7(c) shows the IncSFA output over the input samples. Figure 7(d) shows the ROC estimation error (blue solid line) and an Expected Moving Average of the error (green dashed line) over samples. As the process continues, the error eventually drops below the threshold $\delta = 0.3$ and the abstraction-module ϕ_1 is saved. Figure 7(e) shows the ROC cluster-centers C that map the feature output-activations to each of the 10×5 states. It is clear that there are two well-separated clusters each representing the state of the plastic-cup.

Immediately after the abstraction is saved, the transition model (represented by the blue lines in Figure 7(e)) and

[1]We, however, used both the left and right camera images as an input observation by concatenating them together.

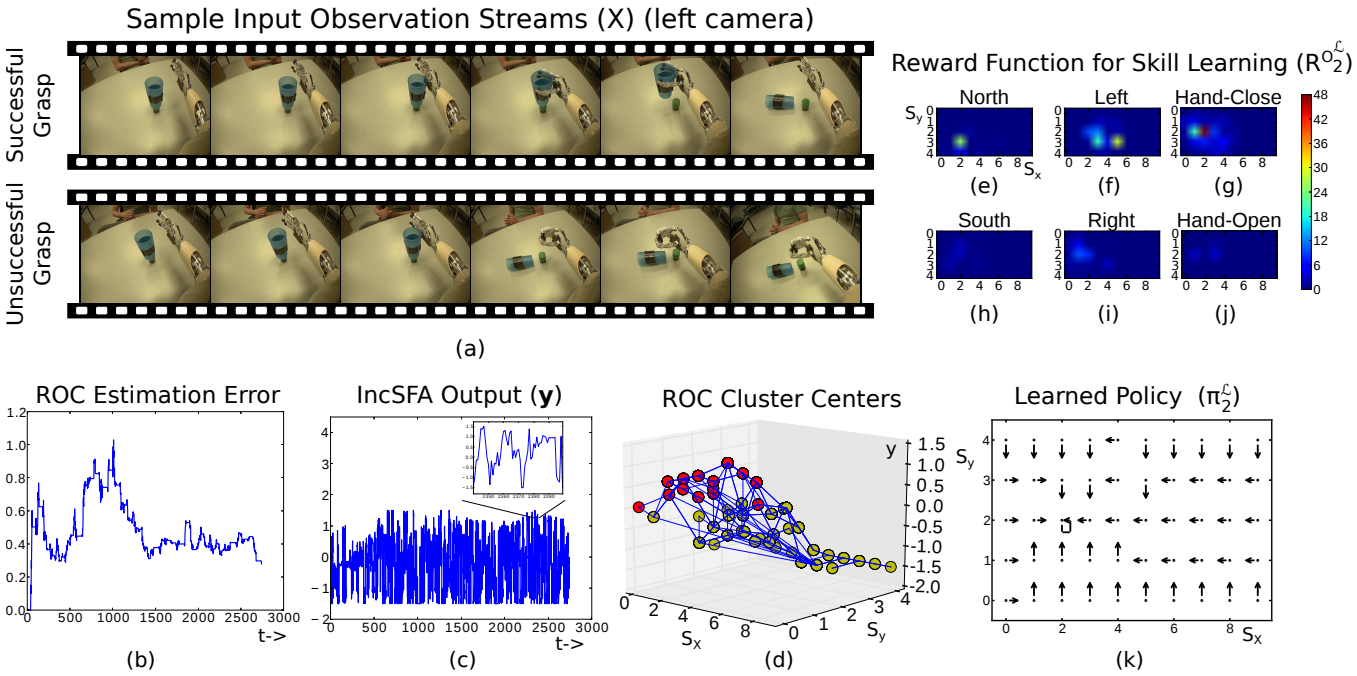

Fig. 8. **Experiment 2: Grasp Skill.** (a) Sample iCub's left-eye camera images corresponding to the input exploratory option $\mathbf{x}_2$ that corresponds to the *biased-init & explore* exploratory option. (b) ROC estimation error of the adaptive-module that is encoding the new regularities. (c) IncSFA output over execution-time. (d) Resultant ROC cluster centers. (e)-(j) Estimated reward function for learing the target option. (k) Learned target-option's policy representing the grasp-skill. The circular arrow represents the hand-close action.

reward model of $O_1^{\mathcal{L}}$ are learned, followed by a corresponding target-option's policy $\pi_1^{\mathcal{L}}$. Figure 7(f) shows the learned policy $\pi_1^{\mathcal{L}}$ (which makes the iCub to move its hand *west* to topple the cup) and an averaged state reward function (averaged over its 6 actions). Figure 7(g) shows the policy and the averaged reward function after the cup has been toppled. The policy makes the iCub to move its hand towards *east*. This is because, during the experiment the interactor happened to replace the cup only when the iCub's hand is around far east. Therefore, the learned target option $O_1^{\mathcal{L}}$, for the given environment, represents a "Topple" skill.

B. Experiment 2: iCub Learns to Grasp a Cup

Here, we present results of another experiment where the iCub learns a skill to grasp the cup. We add another exploratory-option policy, which is biased to explore close to the cup. This improves the probability for a successful grasp while exploring (Figure 8(a) Top). The experiment would still work without this biased exploration, but would take much longer.

To avoid re-coding the topple event in the case of an unsuccessful grasp (Figure 8(a) Bottom), the previously learned topple-abstraction is stored in the learned abstraction-set $\Phi_t = \{\phi_1\}$ at $t = 0$. Therefore, whenever the cup gets toppled, the gating system prevents the adaptive abstraction to train on those samples (see II-E). The adaptive-abstraction $\hat{\phi}$ now encodes only the successful grasp events.

Figure 8(b) shows the ROC estimation error plot. When the estimation error drops below the threshold $\delta = 0.3$, it saves the module $\phi_2 = \hat{\phi}$ and adds it to the abstraction-set $\Phi_t = \{\phi_1, \phi_2\}$. Figure 8(c) shows the IncSFA output over the samples received. The result is a step-like function indicating a grasp event. Figure 8(d) shows the cluster centers and the transitions. The iCub then begins to learn the target-policy $\pi_2^{\mathcal{L}}$ by learning the target-option's transition and reward model. Figure 8(e)-(j) show the target-option's state-action reward model developed after 8000 observation samples (t=8000). And finally, Figure 8(k) shows the corresponding skill learned, *i.e.*, to perform a Hand-Close at $(2, 2)$ (the counterclockwise circular arrow represents the Hand-Close action). This experiment demonstrated how the iCub learns an abstraction to represent whether the cup has been successfully grasped-or-not, and to learn a subsequent "grasp" skill.

V. Conclusions and Future Work

We have presented SKILLABILITY, which combines the Curious Dr. MISFA algorithm, for learning slow-feature-based abstractions, with the options framework, and a special reward function to maximize the expected change in slow feature outputs, for learning policies. These three pieces enabled an iCub humanoid robot to learn skills with high-dimensional video input. The iCub learns both a a "Topple" and "Grasp" skill (with respect to a cup) without any prior knowledge of its environment and itself, except for a set of poses used as a roadmap for motion planning. In future work, we plan to build a continual learning system that autonomously re-uses previously learned skills to learn more complex skills, making it an open-ended skill-learning system.

ACKNOWLEDGEMENTS

This work was funded through SNF grant #138219 (Theory and Practice of Reinforcement Learning II) and through the 7th framework program of the EU under grant #270247 (NeuralDynamics project).

REFERENCES

[1] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: a survey. *Journal of AI research*, 4:237–285, 1996.

[2] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. Cambridge, MA, MIT Press, 1998.

[3] J. Schmidhuber. Powerplay: Training an increasingly general problem solver by continually searching for the simplest still unsolvable problem. *Frontiers in psychology*, 4, 2013.

[4] M. B. Ring. *Continual Learning in Reinforcement Environments*. PhD thesis, University of Texas at Austin, Austin, Texas 78712, August 1994.

[5] R. S. Sutton, D. Precup, and S. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1):181–211, 1999.

[6] G. Konidaris and A.G. Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. *Advances in Neural Information Processing Systems*, 22:1015–1023, 2009.

[7] J. Peters, K. Mülling, J. Kober, D. Nguyen-Tuong, and O. Krömer. Towards motor skill learning for robotics. *Robotics Research*, pages 469–482, 2011.

[8] G. Konidaris, S. Kuindersma, A. G. Barto, and R. Grupen. Constructing skill trees for reinforcement learning agents from demonstration trajectories. *Advances in Neural Information Processing Systems*, 23:1162–1170, 2010.

[9] J. Mugan and B. Kuipers. Autonomous learning of high-level states and actions in continuous environments. *IEEE Transactions on Autonomous Mental Development*, 4(1):70–86, 2012.

[10] M. Luciw, V. R. Kompella, S. Kazerounian, and J. Schmidhuber. An intrinsic value system for developing multiple invariant representations with incremental slowness learning. *Frontiers in Neurorobotics*, 7, 2013.

[11] V. R. Kompella, M. Luciw, M. Stollenga, L. Pape, and J. Schmidhuber. Autonomous learning of abstractions using curiosity-driven modular incremental slow feature analysis. In *Proc. of the Second Joint Conference on Development and Learning and Epigenetic Robotics (ICDL-EPIROB)*, San Diego, 2012. IEEE.

[12] V. R. Kompella, M. Luciw, and J. Schmidhuber. Incremental slow feature analysis. In *Proc. 20th International Joint Conference of Artificial Intelligence (IJCAI)*, pages 1354–1359, 2011.

[13] V. R. Kompella, M. Luciw, and J. Schmidhuber. Incremental slow feature analysis: Adaptive low-complexity slow feature updating from high-dimensional input streams. *Neural Computation*, 24(11):2994–3024, 2012.

[14] J. Schmidhuber. Curious model-building control systems. In *Proceedings of the International Joint Conference on Neural Networks, Singapore*, volume 2, pages 1458–1463. IEEE press, 1991.

[15] J. Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990-2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230 –247, 2010.

[16] L. Wiskott and T. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.

[17] P. Földiák and M. P. Young. Sparse coding in the primate cortex. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 895–898. The MIT Press, 1995.

[18] G. Mitchison. Removing time variation with the anti-hebbian differential synapse. *Neural Computation*, 3(3):312–320, 1991.

[19] G. Wallis and E.T. Rolls. Invariant face and object recognition in the visual system. *Progress in Neurobiology*, 51(2):167–194, 1997.

[20] M. Stollenga, L. Pape, M. Frank, J. Leitner, A. Förster, and J. Schmidhuber. Task-relevant roadmaps: A framework for humanoid motion planning. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Tokyo, Japan, 2013. Accepted for publication.

[21] Ishai Menache, Shie Mannor, and Nahum Shimkin. Q-cutdynamic discovery of sub-goals in reinforcement learning. pages 295–306, 2002.

[22] Özgür Simsek and Andrew G. Barto. Skill characterization based on betweenness. In *NIPS'08*, pages 1497–1504, 2008.

[23] M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

[24] I. D. Guedalia, M. London, and M. Werman. An online agglomerative clustering method for nonstationary data. *Neural Computation*, 11(2):521–540, 1999.

[25] D. Zhang, D. Zhang, S. Chen, K. Tan, and K. Tan. Improving the robustness of online agglomerative clustering method based on kernel-induce distance measures. *Neural processing letters*, 21(1):45–51, 2005.

[26] M. Luciw and J. Schmidhuber. Low complexity proto-value function learning from sensory observations with incremental slow feature analysis. In *Proc. 22nd International Conference on Artificial Neural Networks (ICANN)*, Lausanne, 2012.

[27] Daan Wierstra, Tom Schaul, Jan Peters, and Juergen Schmidhuber. Natural evolution strategies. In *IEEE World Congress on Computational Intelligence*, pages 3381–3387. IEEE, 2008.