# Against Amnesic Robots: a Developmental Bayesian Optimization Framework enhanced with Past Experiences and Knowledge from Long-Term Memory

Maxime Petit, Emmanuel Dellandréa, Liming Chen

# Against Amnesic Robots: a Developmental Bayesian Optimization Framework enhanced with Past Experiences and Knowledge from Long-Term Memory

Maxime Petit, Emmanuel Dellandréa, and Liming Chen, *Senior member, IEEE*

***Abstract*—Many robotics applications, softwares, techniques and modules usually require optimizations of hyperparameters in order to be efficient for specific tasks, commonly performed offline by a human expert. In this work, we consider the use case of a grasping robot for industrial bin-picking. We propose a Developmental Cognitive Framework endowing the robot with the capability to self-explore and optimize efficiently by itself such parameters, even from noisy and expensive evaluations, during its own lifetime, after deployment. The robot cognitive architecture is based on reasoning mechanisms (in particular a Bayesian Optimization (BO) module in charge of the exploration) but also a Long-Term Memory (LTM, with *episodic*, *semantic* and *procedural* sub-memories) allowing the robot to take advantage of knowledge from similar past experiences in order to enhance the BO search (using Case-Based Reasoning paradigm applied with some transfer and meta-learning strategies). We evaluated the system with the constrained optimizations of 9 continuous hyperparameters for a professional software in simulated industrial robotic arm bin-picking tasks (a step that is needed each time to handle correctly new object) using only a very small optimization budget of 30 iterations. We show that the BO is significantly benefiting from the combination of meta (ML) and transfer (TL) learning strategies for each of the 10 objects tested to achieve very good performance in any case, despite a very little search budget (overall from 79.52%of success from vanilla BO to 83.96% with TL, 83.63% with ML and 85.62% with ML+TL).**

***Index Terms*—developmental and cognitive robotics, long-term memory, transfer and meta learning, hyperparmeter automatic optimization, case-based reasoning**

## I. Introduction

CURRENT Machine Learning (ML) methods and applications, in particular those in robotics that we target here, *e.g.* Deep Neural Networks (DNN) for grasp prediction [1], affordable object instance segmentation [2], generally require careful selection of a number of hyperparameters to optimize the performance of the overall systems. In some cases, a human expert, based on his knowledge and past experience, can manually selection them, however the task is tedious, time consuming and subject to errors, thereby unsuitable for real robotics applications where rapidity, flexibility and adaptability are required. This issue becomes even more critical when a given robot is frequently assigned novel tasks, *e.g.*, novel object to grasp, which require each time adaptation of novel optimal hyperparameters.

In this paper, we aim to tackle the issue of optimal selection of hyperparameters for robotic systems that we treat as blackboxes for the generality of the proposed approach which can be applied to any hyperparameter tuning problem, *e.g.*, various machine learning-based robotic tasks. Specifically, we address the task of robotized bin-picking in industrial settings where hyperparameters for grasping robots have to be carefully tuned in order to manipulate specific objects, and the introduction of new ones require novel tuning of these. We aim at minimizing the number of trials of these hyperparameters with respect to a performance score that we want to maximize.

The authors are with the Ecole Centrale de Lyon, LIRIS Laboratory, Imagine Team, Lyon, France. E-mail:name.surname@ec-lyon.fr

Fig. 1. Real robotics setup with an industrial Fanuc robot for a grasping task from homogeneous highly cluttered heap of elbowed rubber tubes.

Instead of tedious and error prone manual tuning of hyperparameters, there exist in the literature automatic optimization methods, using learning from trials and errors, by providing a way toward a constrained numerical hyperparameters search. Among these, the Bayesian Optimization (BO, see [3]–[5]) is particularly suited for robotics applications where external modules in need of this automatic optimization, *e.g.* Grasp planner, can be seen as blackboxes whose evaluations are usually: 1) expensive in term of time, and 2) producing noisy score. However, these optimization methods are most of the time only used ”offline” before the deployment of the system *in-situ*, otherwise manually launched by a human expert when he estimates that it is needed when the robot is confronted to new tasks. In short, the process for the optimal selection of hyperparameters is classically separated from the autonomous ”life cycle” of the robot, and applied ’offline’ similarly to

a physical maintenance of the robot. Because of that, the optimization process always starts from scratch (*i.e.* in cold start settings) like an amnesic robot without taking advantage of its past experience and knowledge to warm-start (*e.g.* see [6]) the hyperparameters' optimization of new tasks that harness previously learned ones.

In this paper, we propose a novel Developmental Cognitive Framework (DCF) endowed with a long-term memory and a BO-based inference engine for automatic selection of optimal hyperparameters so that the robot can capitalize on its past experience and knowledge, thereby increase the efficiency of classical "amnesic" automatic optimization methods. Our work is inspired by the human ability to learn across different tasks, and not starting from zero but adapting known skills when confronted to new situations, a strategy that is usually applied from the human expert who begins and guides its search from previous optimizations.

Our contributions are fourfold:

- A novel developmental cognitive framework (DCF) is designed to make it possible for the robot effective optimization of hyperparameters on different tasks during its lifetime. DCF is endowed with a Long-Term Memory (LTM), composed of an episodic, semantic and procedural memory, memorizing the robot past experience, and reasoning modules for automatic optimization of hyperparameters;
- DCF is endowed with a Bayesian Optimization (BO)-based inference engine for optimal tuning of hyperparameters. The BO inference engine is further enhanced with a case-based reasoning inspired Transfer Learning (TL) strategy which forces the robot to initially test the optimized hyperparameters' values from a known task sharing similarities with a new one (*e.g.* trying to grasp an unknown apple with the optimized parameters of a known orange).
- We further implement a Meta Learning (ML) to reduce the search space of the DCF's BO inference engine by discarding the unpromising areas from the search space of similar optimization tasks;
- Using both simulation and a real robot for grasping 10 objects, we show that the proposed DCF enables significant grasping success rate when using the hyperparameters tuned through the TL and ML enhanced BO inference engine in comparison with the raw BO engine despite a very little search budget.

Preliminary version of this work appeared in [7], [8], respectively. This paper significantly extend these preliminary results in several ways. First, we have implemented and tested a third strategy to enhance the BO process of the DCF by combining both method TL and ML methods. Then, we have also compared the performance of the 3 different methods (TL, ML, TL+ML) not only versus the raw amnesic version of the BO (*i.e.* a robot without the DCF) but also between them (TL vs ML, TL vs ML+TL and ML vs ML+TL). In addition, the design of our experiments in order to estimate the performance of the different versions of the DCF is more complete, with 10 different objects in simulation (vs 4 and 7 respectively in [7] and [8]) from 4 databases.

The reminder of the article is organized as follows. Sec. II discusses some related works of interest, in particular those related to Bayesian Optimization methods, Long-Term Memory robotics architecture, as well as similar Transfer and Meta-Learning strategies. Sec. III introduces the proposed DCF in detailing its different components and the links between them. Sec. IV defines the experimental setup for the validation of the the proposed approach while Sec. V discusses the experimental results. Sec. VI draws the conclusion and gives some perspectives for future work.

## II. Related Work

**Bayesian Optimization** [3], [5] is a method of choice for hyperparameter tuning, often used in the machine learning community. It is especially suited when the cost function is expensive and noisy, which is usually the case in robotics applications, under the condition that the number of parameters is limited. Indeed, BO has been implemented with great success in the robotic field, usually with focus on automatic gait optimization (*e.g.* more recently [9]–[11]). One particular work from Cully *et al.* used this technique to empowered a 6-legs walking robot with the capability to adapt its gait very quickly after random damages to its legs [12]. This is of particular interest because this is achieved with a similar strategy to ours: the robot is taking advantage of previous simulated experiences where legs were damaged. The robot was then able to store the best walking strategies for each type of damages in a 6-dimensional behavioral space (one dimension being the duration of contact between the ground and each leg, discretized in 5 parts). It has to be noted that we cannot always easily define metrics for such behavioral space when applying this notion to other task not related to robotics gait. However, we take inspiration from this work and follow the same principle where the behavioral space will be represented by the similarity between objects the robot will have to learn to manipulate. There are a few works exploring the advantage of Bayesian Optimization to robotics grasping applications, more recently explored with a humanoid robot iCub [13], [14]. The robot is self-exploring locally using BO the strategies to perform grasping in a safe manner under uncertainties of both the object model and the robot sensors. Like our application, they have thus to deal with a noisy objective function, but their method only concerns single object grasping (*i.e.* no cluttering) that can be manipulated in a collision-free environment from every direction, using a highly-actuated 5 fingers robotics hand. However, this is a requirement that is not available for our application context, as bin-picking is by essence used in a highly cluttered environment with constrained angles for the approach of the robotic arm.

Endowing robots with **Long-Term Memory** gains a strong interest from the scientific community, with recent works in mobile robotics [15], conversational agents [16], robot perception [17] or HRI in general [18], [19]. Indeed, a Long-Term Memory system (see [20] for review) can be of used by the robot to allow and enhance its learning during lifetime, adapting to user preferences or new situations and

tasks. Long-Term Memory for robotics manipulation and action in particular relies at least on the implementation of a *procedural memory*, responsible of storing motor skills. A lifelong autobiographical memory framework has been provided such feature for the iCub [21], [22], allowing it to store multi-modal data during its interactions with the environment, making them available for *a posteriori* reasoning leading to additional information and concepts discovery. They implemented a declarative memory (with both an *episodic* and *semantic* memory) and a non-declarative memory (*i.e.* the *procedural memory*) linked to skills and actions related to the joints trajectories. However, the reasoning modules developed within these works were somehow limited to direct data analysis and visualization, contrary to our architecture where they are machine learning techniques more tightly coupled with the LTM and thus take advantage of it to be more efficient. Another work of interest has been proposed by Vernon *et al.* [23] with the benefit of integrating an episodic and procedural memory systems for cognitive robotics, providing thus a sense of prospection. The framework is combining the knowledge extracted from past experiences with skillful know-how information endowing the robot with the capacity to change its perspective from immediate sensory experience in order to anticipate the consequences of future actions (its own or from other agents). In their cognitive architecture, the procedural memory is however only a simple repository, pre-defined with primitive actions (*e.g.* reach, push, grasp, wait). Such memory is indeed not capable of growing, lacking flexibility and forbidding the learning of new actions, making the robot less adaptive to new situations.

When trying to reduce the duration and trials needed for a robot to acquire new skills, **Transfer Learning** is a commonly used strategy, in particular when Reinforcement Learning method is applied (for reviews, see [24]–[26]). Our implementation of the Transfer Learning is following a concept coming from the cognitive science research on human memory defined as Case-Based Reasoning (CBR) from Schank [27]. It implies that an agent generates solutions when confronted to a new problem by relying on the use of its experiences from previously solved problems: in short one solve new problem by remembering similar experiences about similar events and environments. When confronted to a new problem, robotics system with CBR should then be able to retrieve relevant similar memories, reuse them for new problem and revise the solutions accordingly and eventually retaining the new optimized one (See [28] for a review on such applications). This idea of such transfer learning specifically applied to Bayesian Optimization methods have been successfully implemented by Feurer *et al.* [29], with a method to start the optimization with the most promising results performing well on similar datasets. We will follow the same strategy, with the main difference on applying it to self-explored robotics manipulation data (and not fixed-sized datasets), using a shape-based similarity metrics. This is because our hypothesis is that objects that are similar between each other will be grasped properly with similar parameters values from a given grasping algorithm.

In this work, we also implemented a **Meta Learning** (ML) concept, where we want to provide to the system a reduced initial search space, instead of a default larger one, for constrained numerical parameters optimization. We define it as Meta-Learning because the strategy acts on some hyperparameters of the learning mechanism (*i.e.* the boundaries from which the parameters to be optimized are picked to be tested from the BO), in order to increase its performance. With our ML strategy, the BO learns to learn by changing its default settings and, depending on the task to learn, avoids specific unpromising areas. This strategy is following the principle of Viability Evolution defined by Maesani *et al.* [30], [31]. Applied to evolutionary algorithms, the concept is to eliminate before any costly proper evaluations the newly evolved agents that does not seems to be viable (*i.e.* exploring a zone that is already known to be not promising). In such case, that means that the agents do not satisfy a viability criteria, defined as bounds on constraints and regularly shortened over generations. This methods is effectively forcing the generated agents to evolve within a shorter search space than the original one, avoiding unnecessary exploration and thus increasing the performance of the optimization algorithm. In this work, we follow this method applied on Bayesian Optimization by reducing the parameters bounds at the beginning of process, based on similar past experience.

## III. Material and Method

### A. Overview

We designed the Developmental Cognitive Framework (DCF) for robotics by merging the transfer and meta-learning strategies, studied in isolation from our previous work [7], [8], and presented in Fig. 2. In short, it allows the robot to construct and exploit with different reasoning capacities its Long-Term Memory in order to be more efficient in optimizing its behavior when confronted to news tasks similar to a known one, using both strategies of transfer and meta-learning. In this current work, we define a task as bin-picking from homogeneous heap of objects, with a parallel-jaws gripper from an industrial arm.

The Long-Term Memory, detailed in Sec. III-D, is composed of 3 sub-memories present in the human brain and described bu Tulving [32], [33]: 1) an *episodic memory* containing data from experiences and event, that can be linked to a specific place and time, 2) a *semantic memory* where facts and knowledge about the world is stored and 3) a *procedural memory* built with motor skills, using tools and devices *etc.* (*i.e.* "How to do things?"). This memory is interlinked with a reasoning module, a Bayesian Optimization framework allowing the robot to optimize a set of parameters to successfully achieve a task. This module, explained more deeply in Sec.III-B, populates the *episodic memory* with data about each iteration of the optimization (in particular, the parameters values used for each iteration of the optimization and the corresponding score) but also stores directly the optimized set of parameters in the *procedural memory* at the end of each run. The *semantic memory* designed here

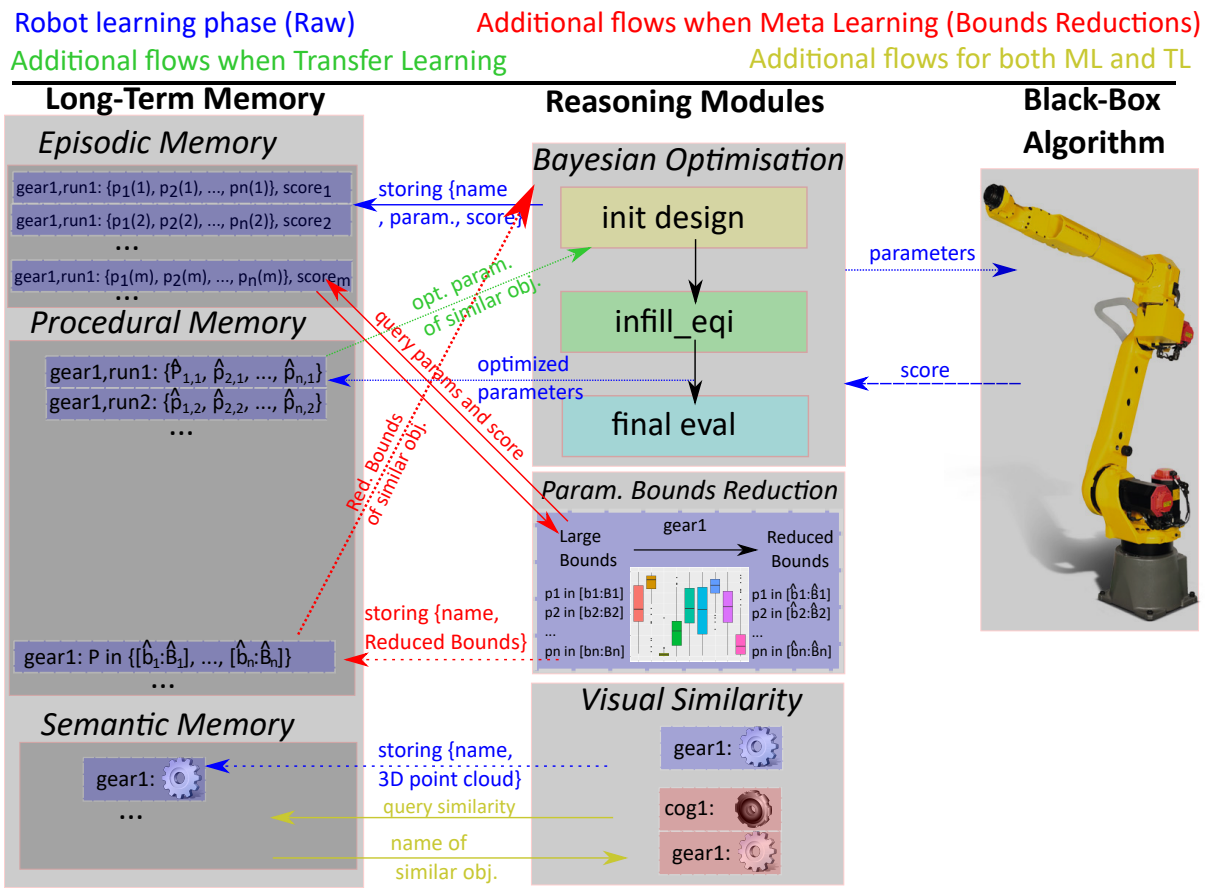


Fig. 2. Architecture of the extended cognitive developmental framework, based on Long-Term Memory (with episodic, procedural and semantic memories) and Reasoning Modules (Bayesian Optimization, Visual Similarity and the Parameters Bounds Reduction) allowing a robot to learn how to grasp objects. This learning consists of guiding an efficient continuous hyperparameters constrained optimization of a black-box algorithm controlling the robot. The blue arrows represent the data flows during a learning phase from a raw BO (*i.e.* without taking advantage of the Long-Term Memory, just storing the experiences). The red and green arrows shows the additional queries and exchanges of information during a learning phase with respectively ML and TL, based on the visual similarity between objects the robot knows how to grasp and the new one (yellow arrows).

is mainly focusing on the visual modality by storing the 3D points clouds of objects along their names and can be accessed by the Visual Similarity module (see Sec.III-C).

Two different mechanisms allow the robot to take advantage of its past experiences to speed up and increase the performance of new optimization and learning for similar tasks, with such similarity links provided by the Similarity Module. Detailed in Sec. III-E, the Transfer Learning strategy consists on forcing the Bayesian Optimization to explore initially optimized set of parameters stored in the *procedural memory*. For instance, when trying to grasp a new unknown object like an apple, the robot will detect this fruit is similar to an orange, that it knows how to grasp. The Transfer Learning method will provide the Bayesian Optimization with such set of optimized parameters to be tested at the beginning of the learning process, guaranteeing some promising point of the search space will be explored. The Meta Learning strategy (See Sec. III-F) relies on an analysis of the parameters distribution from the best iterations of specific task optimization. It detects if some range value of parameters are completely absent from these iterations, indicating that these are unpromising search space areas that can be ignored. By reducing the boundaries from these parameters, the robot will reduce its search space and make the problem easier for the Bayesian Optimization when confronted to similar tasks.

### B. Bayesian Optimization Module

The Bayesian Optimization module is implemented through the R package *mlrMBO*[1] [34] with a Kriging model as surrogate function, also known as Gaussian Processes (GP). It is one of the most widely used in order for the optimization to exploit (by estimating the fitness) and explore (by estimating the uncertainty) the search space [35].

The BO will optimize a number of constrained (*i.e.* within some boundaries) numerical parameters with a budget of specific iterations (*i.e.* trials) exploring the parameters values and being provided in return a noisy score relative to the performance of the blackbox module when using them. In order to select the set of parameters to be tested for each iterations, the BO is composed of three main steps:

- *"initial design"*: This step is not using any prior knowledge but instead is trying to cover the search space as much as possible, in order to provide a first good

[1]https://github.com/berndbischl/mlrMBO

estimation of its landscape. Because we want to have the maximum coverage, points are selecting here using a Maximin Latin Hypercube function [36] effectively maximizing the minimum distance between points (*i.e.* initially forcing the exploration)

- The core Bayesian search mechanism (named here*"infill eqi"*) relies on both a surrogate model (using GP) and an acquisition function (using a specific criteria and a way to parse the function) to define a strategy for efficiently maximizing a black-box function. The chosen GP's kernel is a Matern 3/2 classically used in machine learning (as explained in [37], p85). The next point is thus extracted from the acquisition function (constructed from the posterior distribution over the objective function) using a specific criteria. Several are available, for instance a commonly used one is the Expected Improvement (EI) criteria that favors point with a respectively low or high mean prediction for minimizing or maximizing optimizations but also take into account the variance (*i.e.* modeling the uncertainty about the real performance in this area) which is preferred high. In this work, we have chosen to use an extension of the EI named Expected Quantile Improvement (EQI) from Pichney *et al.* [38]. This criteria has the advantage to be designed specifically to deal with difficult function providing noisy scores. To find the best next point to try according to this specific criteria, we are using a Covariance Matrix Adapting Evolutionary Strategy (CMA-ES) [39], [40] from the package *cmaes*[2] on the acquisition function. It is a stochastic derivative-free numerical optimization algorithm for difficult (non-convex, ill-conditioned, multi-modal, rugged, noisy) optimization problems in continuous search spaces.
- *"final eval"* corresponds to the final evaluation of the best found solution, which will be repeated and tested several times. The best set of parameters is defined by the best score predicted over all points ever visited using the final model, instead of taking the actual performance obtained. This has the advantage of averaging-out noisy function values.

The Bayesian Optimisation module is at a central place in the overall architecture and participates in the workflow through communication and interaction with almost every components.

An important detail is the fact that the $n$ parmeters $p_1, p_2, ..., p_n$ of the blackbox to be optimized are actually normalized $\in [0:1]$ for the Bayesian Optimization process, before being scaled back to their real values when sending them to update the robot parameters. Then it triggers the command for the robot to execute the task (*i.e.* try to grasp X times a specific object). At the end of it, the robotics platform will return a performance score $s_i \in [0:100]$ (*i.e.* the percentage of successful grasps among K attempts), where $i$ is the iteration number. The data from each iteration (*e.g.* iteration number, task evaluated, run number, parameters value, score, execution time) are then stored in the *episodic memory*. In addition, at the end of the optimization run, the optimized set of parameters for the task will be added to the *procedural memory* allowing the robot to quickly load and use the best behavior when confronted to this task in the future.

[2]https://cran.r-project.org/package=cmaes

### C. Similarity Module

The Similarity module is required for the robot to understand what part of its previous experiences is needed to increase its learning performance when confronted to a new one. It relies on the hypothesis that similar tasks should have nearly equal performance when using the same parametric behavior.

The Similarity module (see Fig. 3) designed for this work relies on the visual modality, thus interacting with the visual part of the semantic memory. Indeed, the component is used by the robot to retrieve the most visually similar known object when confronted to the learning of manipulation for a new one. It is based on a deep neural network called PointNet [41] designed to classify and segment point clouds geometrical shape in 3D. It has the advantage of learning both global and local point features, providing an effective approach for 3D recognition tasks. As input, we provide the coordinates of $N$ points from CAD models to create a point cloud. Next, they are transformed with an affine transformation matrix by a network named T-Net. Each point will be learnt by a convolutional kernel (size of 1) and eventually aggregated by symmetric operations (*i.e.* max-pooling) into a global feature vector of dimension 1024. Finally three fully connected layers are stacked on to learn the object classes.
To build our semantic memory, the 3D CAD models of the object to be grasped are represented first with point clouds. Next we modified the last layer in PointNet to correspond to our reference database (*i.e.* changes in the number of classes), and we further fine-tune the network by freezing earlier layers of *conv5*, based on pre-trained model from ModelNet40 [42].

For the retrieval operation of the most similar object from a new given 3D model, we begin by sampling the CAD model into point clouds, then we extract its global features with 1024 dimension before calculating the distance between each reference model. The metrics is inversely correlated to the degree of visual similarity: the most similar references corresponding to the smaller distances.

We want to highlight the initial hypothesis that was made to define this similarity module: similar tasks according to the metrics should have similar performance with the same parameters. But the similarity metrics can take different modalities depending on the tasks at hands, and we consider the term "similar" are not only being visually identical. For instance, grasping solid objects using an industrial robot with a two parallel jaws gripper, as our application task in this work, can rely mostly on visual similarity, with a module that we just explained. However, extension of this framework to other tasks might require to extend the Similarity metrics with adequate additional modalities in order to keep the correlation between similarity and performance true. Indeed, similarity for fragile or soft-body objects grasping tasks might take into account information about the material or its softness to be efficient,

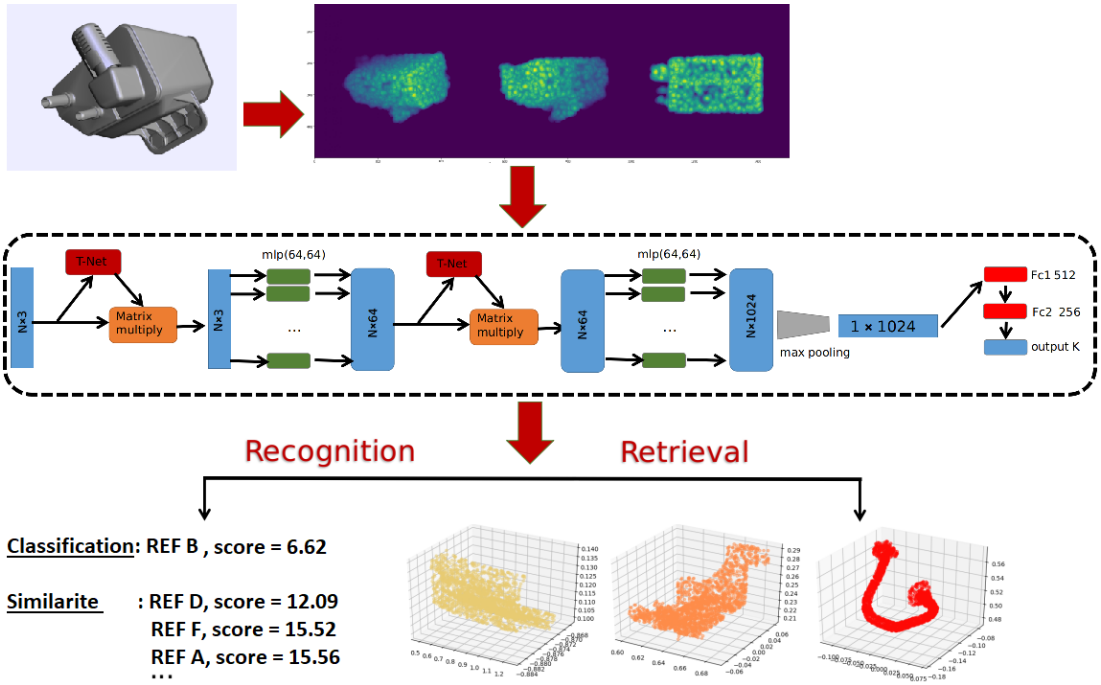


Fig. 3. Main architecture of the Visual Similarity module. A 3D CAD model is first sampled randomly and normalized into set of points $(x, y, z)$, then it is fed into a deep neural network based on PointNet, which learns a global geometry shape by aggregating results of all points into a 1024 feature vector.

and the use of suction gripper will probably need to adjust the metrics with the weight of the target object.

Eventually, such module can also be replaced with a Human-In-The-Loop method, switching slightly the learning paradigm from purely self-supervised to an hybrid form of guided learning with human kind of feedback. It will create a situation that can be linked to keeping the robot in its Zone of Proximal Development, originally conceptualized by Vygotsky before being more extended (as detailed in [43]). It is defined as the difference between what a learner can do without help and what he can achieve with guidance from a skilled partner. In such setup, the human teacher or expert will be involved but in a lighter and easier task than the full optimization of the parameters, only providing the adequate known tasks the robot should inspired from to increase its learning performance for a new but similar task.

### D. Long-Term Memory

The Long-Term Memory is implemented as a relational database in PostgreSQL[3], similarly to other architectures of autobiographical memory for robotics and previous work [7], [8], [21], [44], allowing the robot to store its own experience and knowledge during its interactions with the environment. As introduced earlier, the memory is mainly sub-divised into three parts, each containing specific type of information.

The largest one in the *episodic memory* where the data about each robot's experiences is written and accessible later on for reasoning and analyses. It is split in high-level episodes $r$ , as optimization runs in this work, divided as smaller level with all of its iterations $i$. Data stored about episodes includes a unique identifier, the optimized task $T_O$ (where $O$ is the object id to be grasped), the time of beginning/end, and the hyper-parameters of the Bayesian Optimization run $r$. They consist on the number of iterations for each steps (*"initial design", "infill eqi" and "final eval"*), number of grasp repetition per iteration, the parameters boundaries used (related in particular for the Meta-Learning part) as well as the potential Transfer Learning set of parameters forced at the initial part of the Bayesian Optimisation. Data about iterations $i$ are the unique id of the related run $r$, the set of $m$ hyperparameters tested $\{p_1(i), p_2(i), ..., p_m(i)\}$, the corresponding score $s_i$ obtained with such setup and the duration of the iteration.

The *semantic memory* regroups the data identifying a task and specifically linked to the Similarity module. In our case, a task $T_O$ being to grasp specific objects $O$ from homogeneous heap, the robot stores the visual information about $O$, as point cloud model.

The *procedural memory* possesses two kinds of data: 1) the optimized set discovered at the end of each run $r$, $\{\hat{p}_{1,r}, \hat{p}_{2,r}, ..., \hat{p}_{m,r}\}$, describing how to effectively achieve the task $T_o$ and available to be quickly loaded by the robot if needed, and 2) the reduced parameter bounds for each task, that is the constrained boundaries obtained from analysing the parameter distributions only for the $n^{th}$ best iterations of a specific task. That specific information is produced *a posteriori* by the Parameters Bounds Reduction module that will be detailed later. It has also to be noted that such analysis might be applied to a broader group of tasks instead of a specific one, for instance by analysing all the task involving "object with handle" or "flat object", potentially leading to symbol and concept emergence. Such notion will be expanded briefly in the discussion at Sec.VI.

[3]http://www.postgresql.org/

### E. Transfer Learning

The Transfer Learning module, first developed in previous work [7], takes advantage of a human-like strategy where experts having to manually optimize hyperparameters on a task or dataset start by exploring the configurations that were performing well for similar previous dataset [45], [46]. The module is inspired from an application that has proven the efficiency of this theory to Bayesian Optimisation applied on different datasets by Feurer *et al.* [29]. However, our work is different because we are not building on proper datasets but instead on related similar experiences (with its own new similarity metrics, based on visual information from object). Indeed, when confronted to the optimization of a new task $T_O$ (*i.e.* the grasping from a new object $O$), the Visual Similarity module will provide a list of the closer known objects (*e.g.* $T'_O$). The robot will then be able to query its procedural memory to extract the best X set of optimized parameters obtained from X different runs (or alternatively, but not explored in this work, the best set of optimized parameters from the X closer objects). These sets are considered promising *a priori* solutions and will be tested at the first step of the algorithm, to be more precise at the end of the "init design", where the other previous iterations parameters are still obtained from the maximinLHS function. In order to be effectively estimating the effect of the transfer learning, the number of total points during this phase will remained fixed (*e.g.* 7 points from maximinLHS and 3 from TL for a budget of 10 iterations in total for the "initial design" with 3 as budget for the TL step.)

### F. Meta-Learning

The Meta-Learning strategy has been initially designed from another previous work [8] with the idea of increasing the performance of the learning by constraining the search space to a smaller, thus more easily explored and exploited one. Indeed, the robot will determine in a first step *a posteriori* reduced parameters bounds, discarding a whole range of values that are none promising for each task (*e.g.* among all the runs for bin-picking a specific object). In a second step, the robot will provide such smaller search space to the Bayesian Optimization, instead of a larger default one, when confronted to a similar unknown task. In order to obtain such reduced parameters bounds, the robot analyses the distribution of the parameter values explored during the iterations leading to the best $n^{th}$ scores. An efficient parameters bounds should result in a roughly uniform distribution, meaning the whole range provides interesting parameter values with good results in conjunction with others. On the other hand, a very narrow distribution indicates that the BO is focusing on a specific area, and the rest corresponds to a non-promising part of the search space that is sub-optimized but will anyhow cost some precious optimization budget to be explored, mostly at the beginning.

Summarized in Alg.1, we will detail now how the analysis is performed to compute the new reduced parameters bounds, that can be used later for similar tasks to increase the performance of the BO. Initially, the module is querying the robot's *episodic memory* to retrieve every iterations $I(O)$ (extracting in particular the parameter values and the corresponding score) of past optimization run for a task $O$ (*e.g.* grasping object $O$). It will then only keep a subset $I_n(0)$ of the $n^{th}$ best iterations. Next, the module will check the uniformity of the distribution in [0:1] from the normalized values for each parameters in $I_n(O)$. This is achieve by applying the Dudewicz-van der Meulen test [47], which is an entropy-based test specifically designed to check the uniformity over this specific distribution. By checking the p-value $p_{dm}$ against an alpha risk $\alpha_{dm}$, we can reject the uniformity hypothesis (when $p_{dm} < \alpha_{dm}$),

**Algorithm 1** Algorithm for Bounds Reduction (ML condition)

**Input:** All iterations of all runs for object $O$ with scaled parameters values ($\in [0:1]$)
**Output:** New reduced bounds for object $O$

```
Select I_n(O) the n% best iterations for O
for each parameters p_j(O) do
    Compute p_dm, p-value of Dudewicz-van der Meulen test for uniformity for p_j(O) values from I_n(O)
    if (p_dm < α_dm) then
        Compute p_w, p-value of Wilcoxon test (H0: μ = 0.5)
        if (p_w < α_w and median(p_j(O))> 0.5) then
            Increase lower bound for p_j(O) to the 5% percentile of p_i(O) values from I_n(O)
        else if (p_w < α_w median(p_j(O))< 0.5) then
            Reduce upper bound for p_j(O) to the 95% percentile of p_j(O) values from I_n(O)
        else
            Reduce upper & increase lower bounds for p_i(O)
        end if
    end if
end for
return Modified Parameters bounds
```

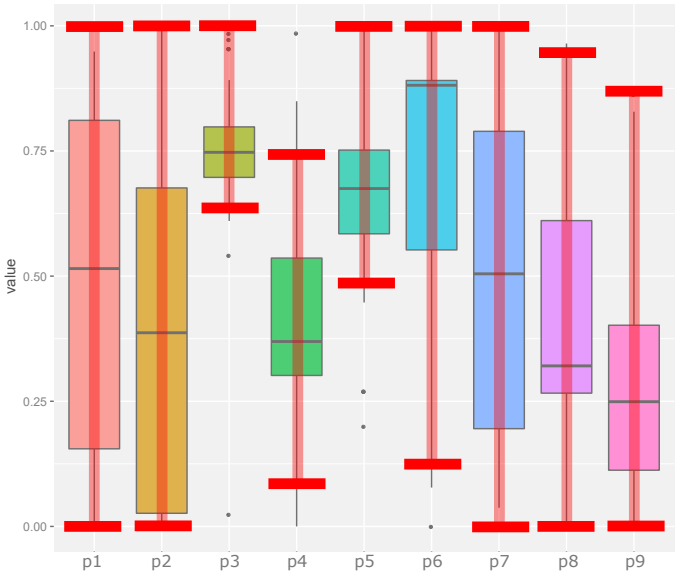


Fig. 4. Example of the reduced boundaries (in red) with object *m782*. Parameters p1, p2, p7 are following a uniform distribution on [0:1]. All the other are not under such law, and p3, p5, p6 have a median $> 0.5$; p8, p9 have a median $< 0.5$ and p4 has a median = 0.5. done!

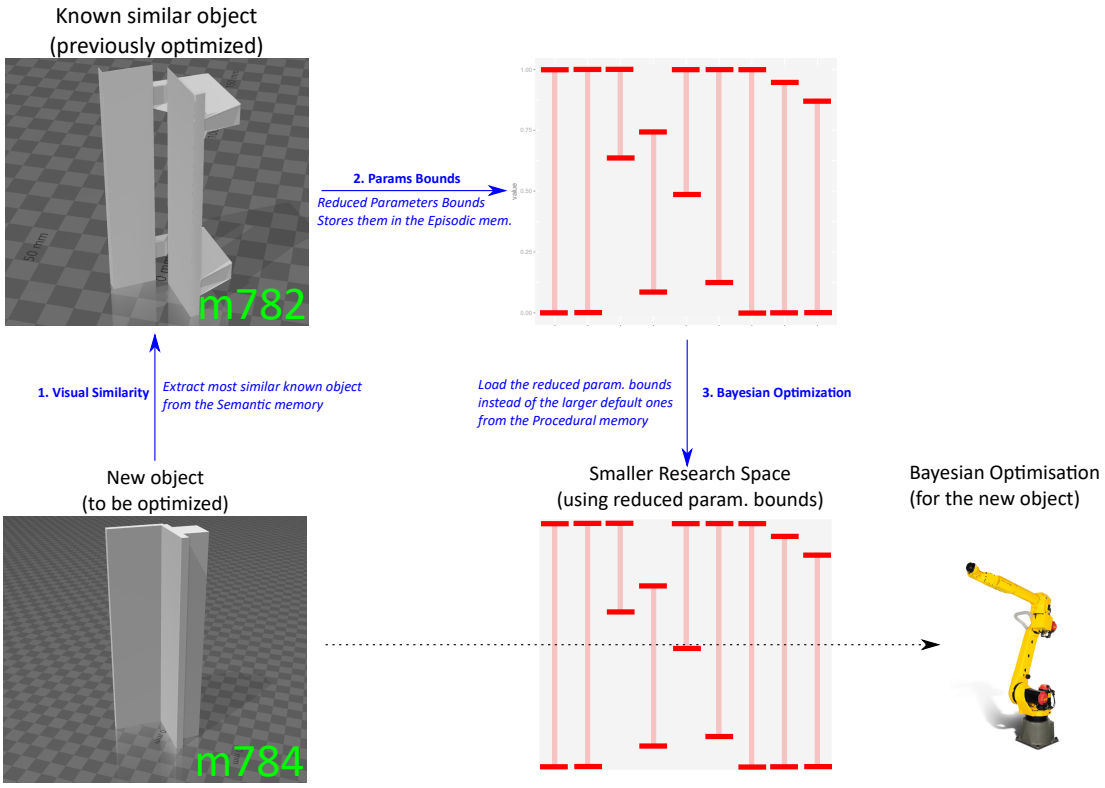


Fig. 5. Focus on the robotics workflow during the Meta-Learning condition, with an example of learning how to grasp a new heap of object m784 based on a known m782 object. The robot first use the Visual Similarity to detect that the m782 is a very similar and known object, then it loads the m782 reduced parameters bounds. Eventually, the robot force the BO to use them, reducing the search space instead of the default larger boundaries, forbidding the algorithm to test parameters values that are inside unpromising areas.

meaning some range of the parameter value are not explored and can be eliminated. The module can then either increase the lower bound (if smaller values of the parameters are ignored), or lower the upper bound (if larger values of the parameters are not explored), or do both. Such decision is dependent on the median of the distribution that will be statistically tested in order to determine if $\mu = 0.5$, using a one-sample Wilcoxon signed rank test as we cannot assume the normality of the distribution. This test will produce a p-value $p_w$ and confronted to another alpha risk $\alpha_w$ we will determine statistically if the distribution is centered around 0.5. If it is the case, we reduce the upper bound and increase the smaller bound as well. If $p_w < \alpha_w$, we can reject such hypothesis and only change one of the bound (*i.e.* the lower bounds will be increased if $\mu > 0.5$ or the upper bounds will be smaller if $\mu < 0.5$). The new values for the bounds are dependent on the $x^{th}$ and $X^{th}$ percentile value of the parameters, respectively the new value for the lower bounds and the upper bounds, with $0 \leq x < X \leq 1$. An example of the new boundaries obtained with this method is presented in Fig. 4

At the end of this *a posteriori* analysis, the corresponding reduced boundaries for the task are stored into the *procedural memory*, available quickly in the future to be given to the Bayesian Optimization process instead of the default larger parameters bounds. This can occurs when the robot is trying to optimize the parameters for a similar tasks using the meta-learning method, as described in Fig. 5.

## IV. Experiments

The experiment setup will be similar to the ones described in [21] and [8] allowing first to confirm their results but with a larger number of tasks (*i.e.* grasping optimizations for a greater number of different objects). That is, the benefit of cognitive architecture, with the use of a long-term memory to enhance the performance of a BO algorithm with a Transfer Learning strategy [21] (condition 'TL') or a Meta Learning method [8] (condition 'ML') when compared to an amnesic robot with only a vanilla BO (condition 'Raw'). In addition, we will be able to compare the performances between the ML only and TL only enhanced version of the BO in this study. Eventually, we will also have optimizations under a 4th condition, when both enhancements are available to the BO (condition 'ML+TL'), the robot using then all the available reasoning methods to take advantage of its autobiographical memory, with the three episodic, semantic and procedural memories been used.

The blackbox function to be optimized is a professional robotics grasping software called Kamido from Sileane[4]. It is designed to analyse an RGB-D image from a top-view camera in order to provide grasping opportunities for an industrial robotic arm with parallel-jaws gripper[5] in a bin-picking task from homogeneous heaps (*i.e.* composed by several instances

[4]http://www.sileane.com/en/solution/gamme-kamido

[5]Kamido has another mode for suction gripper but this is not explored in this work

TABLE I
BOUNDS FOR EACH PARAMETER TO BE OPTIMIZED, WITH THE LARGER "DEFAULT" AND THE REDUCED BOUNDS OBTAINED FROM SEVERAL OBJECTS USING THE PARAMETERS BOUNDS REDUCTIONS MODULE.

| Obj. | p1 | p2 | p3 | p4 | p5 | p6 | p7 | p8 | p9 |
|---|---|---|---|---|---|---|---|---|---|
| Def. | -20:20 | 5:15 | 16:100 | 5:30 | 5:30 | 5:40 | 30:300 | 5:20 | 1:10 |
| C2 | -20:20 | **8**:15 | **46**:**92** | 5:30 | **13**:30 | **24**:**37** | **100**:**220** | 5:**15** | **3**:**9** |
| D' | **-18**:**10** | 5:15 | **49**:**99** | **8**:**23** | 5:30 | 5:40 | 30:300 | **8**:20 | **2**:**8** |
| P_2 | -20:20 | **6**:**14** | **20**:**69** | 5:30 | 5:30 | **18**:**37** | **114**:**267** | 5:**19** | 1:10 |
| ham_t | -20:20 | 5:15 | **46**:100 | **8**:30 | 5:30 | **17**:40 | 30:300 | 5:20 | 1:10 |
| m782 | -20:20 | 5:15 | **68**:100 | **7**:**23** | **12**:30 | **9**:40 | 30:300 | 5:**19** | 1:**8** |
| bathDet | **-15**:**9** | **9**:15 | **69**:100 | **10**:30 | **18**:30 | **27**:40 | 30:**276** | 5:20 | 1:10 |

of the same object) scenarii. These targets are computed using several parameters that a robotics expert need to fine-tuned for each new object to grasp (*e.g.* minimum and maximum opening of the gripper, different thresholds, offsets or margin from computer vision analysis). We are proposing here to apply our self-learning cognitive architecture in order for the robot to optimize nine of them (the ones that were often and largely 'tweaked' from previous tedious and long optimization processes guided by a human expert). Indeed, we want to keep the optimization problem for low to medium-dimension applications (maximum between 10 and 20 dimensions) as it is the standard usage for BO techniques where it shines particularly well. Some recent techniques and modifications of the core BO concepts are being investigated in order to allow efficient optimizations for high-dimension domains [48], [49] but this is outside the scope of this paper.

A real-time physics Pybullet simulation with a parallel-jaws gripper and a top-view RGB-D camera is implemented. Objects can be loaded from Wavefront OBJ format, commonly used in most object database, on which we apply a volumetric hierarchical approximate convex decomposition [50]. The score of an iteration is defined as the percentage of success at bin-picking, a total success being to grasp one of the cluttered object and release it into another box. A partial reward (half success) is applied when the robot is grasping the object but fails to drop it properly (*e.g.* due to slippery).

We will use the same objects as in [8], composed mostly with models from several commonly used 3D objects databases (Turbosquid, Princeton Shape Benchmark [51] and KIT [52]) and completed with CAD model from real references used in the industry. The objects to be optimized are presented in Fig. 6 with their labels, along the most similar known objects by the robot that will be used for both TL and ML methods, defined in previous works. Because the Princeton Shape Database also offers a classification from its instances, we introduce three additional objects (namely m1109, m1110 and m1111, from the class "hammer") from it to have several different versions of the same class (along with previously used object, even if they are not coming from the same database, with *hammer_j* and *hammer_t* extracted from Turbosquid).

In total, we will optimize 10 objects (See Fig. 6) from 4 different databases using 4 different conditions for the BO: Raw (amnesic robot using a plain BO without the use of memory and reasoning), BO with TL only, BO with ML only and BO with ML+TL methods. For each object and condition, 6 independents runs will be performed with the same budget for the optimization of 30 iterations (10 iterations for *"initial design"* and 20 for *"infill eqi"*). However, differently from previous experiments [7], [8], we increase the number of final evaluations, to go from 5 to 10 allowing to have a better estimation of the final performance, and so stronger results. One iteration is still defined as 15 grasping trials. The other details of the setup are identical, with a Bayesian Optimization relying on GP with a classic Matern 3/2 kernel, and the EQI criteria defined with a $\beta = 0.65$ (indicator in [0.5:1], with high value favoring repetitions or clustering, and low value leaning toward a more exploratory behavior). A stochastic derivative-free numerical optimization algorithm, the Covariance Maxtrix Adapting Evolutionary Strategy (CMA-ES [39], [40] from the package *cmaes*), is responsible of the infill criterion optimization. This method is known to be efficient for difficult (non-convex, ill-conditioned, multi-modal, rugged, noisy) optimization problems in continuous search spaces. For the Transfer Learning mode, we will extract the best 3 optimization results to be passed on to the BO. We still want a fixed 10 iterations budget for the *init design* phase, so in such setup only 7 points will be chosen by maximinLHS and the last 3 will be from the TL. For the Meta Learning mode, the best subsets for each object of iterations are the top 35% because the objective function is very noisy so we did not want to be too conservative. The alpha risks for both the Dudewicz-van der Meulen and Wilcoxon tests (*i.e.* $\alpha_{dm} and \alpha_w$) is set to 0.15 for the same reason. The percentile responsible for the bounds reductions are x=0.05 and X=0.95, allowing to discard potential outliers that would otherwise block the method to perform stronger reductions to the boundaries. The obtained reduced boundaries for these objects of reference are presented in Table I, which have been generated from our previous work on the ML strategy [8].

We will also confront our cognitive architecture on a real robotic arm Fanuc (the version M10iA12) with a parallel-jaws gripper. The task will be bin-picking from an homogeneous heap of highly cluttered elbowed rubber tubes (See Fig. 1). We use here a real-soft-body object which represents an interesting property related to grasping tasks but one that was not possible to represent realistically in simulation. We will use an optimization budget of 30 iterations, with additional 10 iterations to estimate the final performance of each run . We will compare the results under the 4 conditions for BO: raw BO, BO with TL, BO with ML and BO with ML+TL.

## V. RESULTS

In this section, we present the results of the experiments from simulations, and compare the effect of amnesic robot (*i.e.* vanilla BO, condition 'Raw') against a robot with long-term memory, using solely a Transfer Learning (TL) Meta-Learning (ML) strategy or the combination of both (ML+TL). Compared with our previous works on TL [7] and ML [8], in addition to estimate the optimization scores for the condition ML+TL and the gain of performance compared to other strategies, we have also performed experiments with additional objects. The new

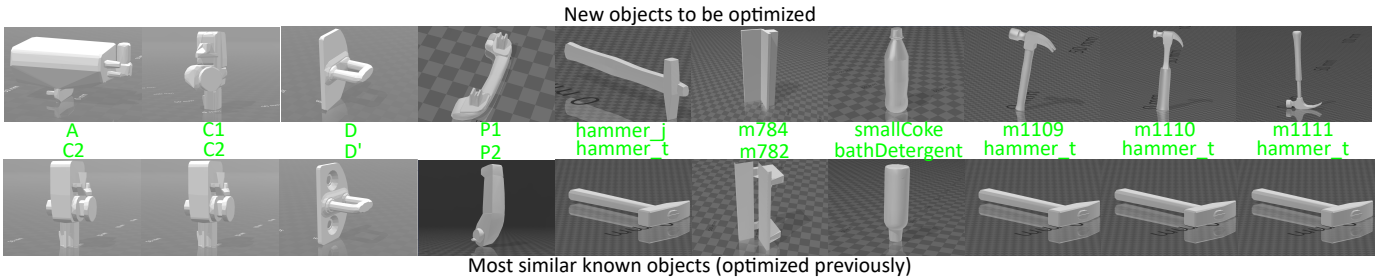


Fig. 6. CAD models of the simulated objects from Turbosquid, Princeton Shape Benchmark, KIT and our private database of industrial objects.

TABLE II
BAYESIAN OPTIMIZATION RESULTS (% OF GRASP SUCCESS) UNDER THE 4 CONDITIONS (RAW, TL, ML, ML+TL)

| Reference | % success all run mean±sd, median | % success (worst run) | % success (best run) |
|---|---|---|---|
| A | 67.6±7.56, 65.2 | 60.6 | 81.2 |
| A_TL_C2 | 77.2±2.24, 78.0 | 73.9 | 80 |
| A_ML_C2 | 75.5±2.09, 75.2 | 73.3 | 78.8 |
| A_ML+TL_C2 | **81.2**±4.22, **80.0** | **75.8** | **87.6** |
| C1 | 76.8±7.08, 76.1 | 66.7 | **86.1** |
| C1_TL_C2 | 78.5±3.33, 78.2 | 74.5 | 83.6 |
| C1_ML_C2 | 78.8±4.47, 78.2 | 73.3 | 84.2 |
| C1_ML+TL_C2 | **80.3**±3.58, **80.3** | **75.5** | 84.8 |
| D | 64.6±25.2, 75.2 | 27.9 | 87.9 |
| D_TL_D' | 89.1±3.23, **89.7** | 83.0 | 92.7 |
| D_ML_D' | 89.0±6.90, **89.7** | 78.2 | **95.8** |
| D_ML+TL_D' | **89.1**±2.55, 89.1 | **85.5** | 92.1 |
| P1 | 91.3±5.40, 92.6 | 83.0 | **99.4** |
| P1_TL_P2 | 90.5±4.40, 90.3 | 84.2 | 94.8 |
| P1_ML_P2 | **92.2**±3.35, **91.5** | **89.4** | 98.8 |
| P1_ML+TL_P2 | 91.8±4.72, 92.74 | 85.5 | 97.0 |
| ham_j | 84.8±5.17, 83.6 | 78.2 | 91.5 |
| ham_j_TL_ham_t | 85.1±4.36, 84.5 | 80.0 | 91.5 |
| ham_j_ML_ham_t | 85.8±1.94, 85.5 | 83.0 | 88.9 |
| ham_j_ML+TL_ham_t | **90.0**±2.69, **89.1** | **85.8** | **93.3** |
| m784 | 74.6±7.64, 75.2 | 63.6 | **86.7** |
| m784_TL_m782 | 77.1±6.37, 78.2 | 66.7 | 84.8 |
| m784_ML_m782 | 76.2±3.62, 76.4 | 71.5 | 83.0 |
| m784_ML+TL_m782 | **77.5**±1.38, **77.6** | **75.2** | 79.4 |
| coke | 87.5 ±2.70, 87.6 | 83.6 | 90.3 |
| coke_TL_det. | 88.1±1.90, 87.6 | **85.5** | 90.3 |
| coke_ML_det. | 87.9±3.34, 87.9 | 84.2 | 92.7 |
| coke_ML+TL_det. | **89.1**±3.7, **88.2** | 84.2 | **94.5** |
| m1109 | 84.4±6.28, **87.1** | 72.7 | 89.1 |
| m1109_TL_ham_t | 84.7±3.17, 84.2 | **81.2** | 88.5 |
| m1109_ML_ham_t | 84.9±2.17, 85.2 | **81.2** | 87.6 |
| m1109_ML+TL_ham_t | **85.2**±3.39, 85.5 | 80.0 | **90.0** |
| m1110 | 84.8±3.99, 84.5 | 79.7 | 89.7 |
| m1110_TL_ham_t | 86.2±4.10, 85.6 | 80.6 | **91.5** |
| m1110_ML_ham_t | 85.1±3.69, 86.5 | 78.2 | 87.9 |
| m1110_ML+TL_ham_t | **88.9**±2.12, **89.7** | **84.8** | 90.3 |
| m1111 | 78.6±3.19, 80.0 | 74.8 | 82.4 |
| m1111_TL_ham_t | 83.1±1.41, 82.7 | **81.8** | 85.5 |
| m1111_ML_ham_t | 81.1±4.48, 80.6 | 76.4 | 86.7 |
| m1111_ML+TL_ham_t | **83.3**±3.20, **83.6** | 78.5 | **87.0** |

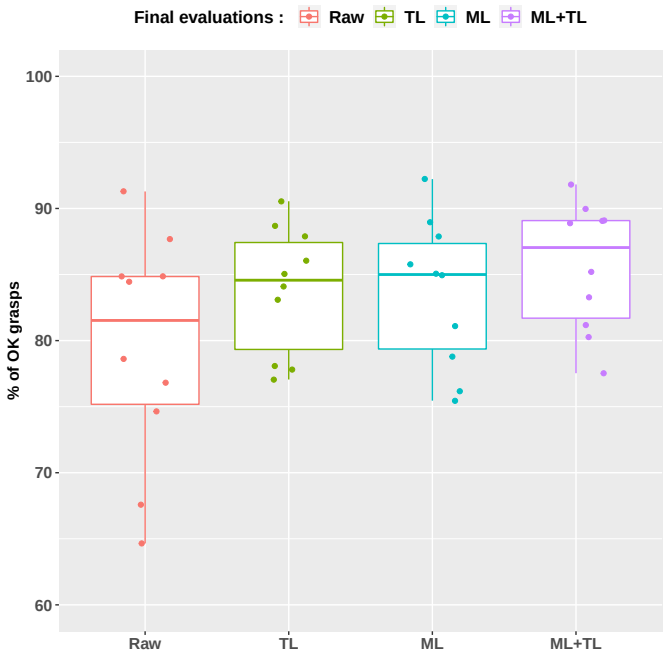


Fig. 7. Boxplot of the mean final performance after Bayesian Optimization on every object for all of their runs (zoom in [60:100]), under the 4 conditions (Raw, TL, ML, ML+TL). Each dot is the average final performance after the optimization runs for a specific object and condition. We can see the higher median, Q1 and Q3 quantiles for ML+TL overall.

objects m1109, m1110 and m1111 we introduced have been labelled as "hammer" from the Princeton Shape Database. Instead of relying on a specific similarity metric, we are investigating here the possibility to replace or complement the similarity module with simpler, more generic and commonly use Object Classification methods (either using pure machine learning and computer vision methods, or allowing the user to provide such classification when new objects are presented). Interestingly, it has to be noted that the known object of reference used for ML and TL will be the hammmer_t object, which comes from a different database (Turbosquid). The final evaluations of the performance are numerically presented on the Table II where the results are shown for each object and condition (Raw, TL, ML, ML+TL), including the average grasping performance, the median as well as the score for the worst and best run. The average performance distribution among all the object according to the 4 versions of the BO can also be seen as boxplots in the Fig. 7. Eventually, in order to estimates the gain of performance when considering each object among the conditions, the Fig. 8 and 9 represent the paired average scores split among objects, respectively for Raw vs TL vs ML+TL, and Raw vs ML vs ML+TL conditions.

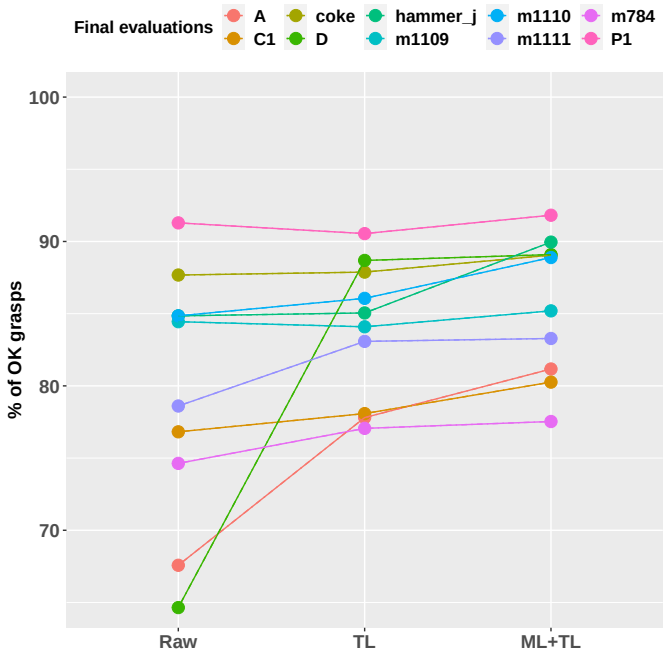


Fig. 8. Final mean performance of all runs, grouped by objects and paired on BO with 3 conditions: Raw (amnesic robot), enhanced with TL or using both ML and TL. This shows the gain of performance when using ML+TL strategy, surpassing previous gains from using TL or compared to Raw.

### A. Confirmation of the benefit of TL vs amnesic raw BO

We focus first on the comparison of performance between the Raw BO and BO enhanced with the TL method, in order to confirm the results obtained in our previous work [7]. Among all the objects, the mean performance jumps from 79.52% with a Raw BO to 83.96% with the TL strategy, a statistically significant increase (p-value of 0.0068 for a paired sampled Wilcoxon signed-rank test with the alternative hypothesis H1 'TL induces an increase of performance compared to Raw', when comparing the mean performance per objects). This can also be seen in the Fig. 8, when focusing on the Raw and TL condition. It seems to be in part due to the fact that the TL methods guarantees a higher 'worst performance optimization': the minimum performance among the optimization runs for each object is consistently higher when the BO is using the TL method compared to without (p-value of 0.0295 for a paired sampled Wilcoxon signed-rank test with H1) with an average worst score of 69.09% (Raw) versus 79.15% (TL). This is highlighted in Fig. 7 when the lower tail and Q1 of the green TL boxplot are much higher that the corresponding part for the Raw boxplot. This confirms the fact that the TL strategy guides initially the BO toward adequate solutions, avoiding it to be lost within unpromising areas.

### B. Confirmation of the benefit of ML vs amnesic raw BO

We want here to confirm the results obtained in [8], namely the benefit of the ML strategy for BO versus an amnesic raw BO. Again, we observe an increase of mean performance of the BO among all the objects when using the ML strategy, taking advantage of the long-term memory compared to an amnesic robot, with 83.63% (ML) versus 79.52% (Raw). This is a significant increase of performance when following the ML strategy compared to without (p-value=0.00098 for H1 with a paired sample Wilcoxon signed-rank test). The gain of performance for each object is shown with the Fig. 9, when focusing on the Raw and ML condition. Similarly to the TL method, the ML strategy makes the BO avoid unpromising regions in the search space, leading to a statistically significant greater minimum expected score, with an average score for the worst run per object of 78.88% (ML) versus 69.09% (Raw) (p-value=0.0062 for H1 with paired sample Wilcoxon signed-rank test). This can be visualised in Fig. 7 with a shorter tail from the cyan boxplot representing the BO enhanced by ML against the red one for the raw BO.

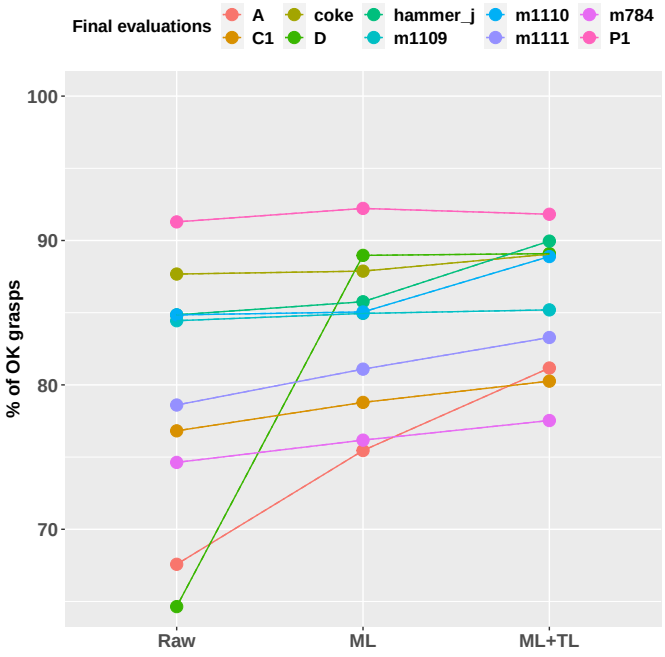


Fig. 9. Final mean performance of all runs, grouped by objects and paired on BO with 3 conditions: Raw (amnesic robot), enhanced with ML or using both ML and TL. This shows the gain of performance when using ML+TL strategies, surpassing previous gains from using ML or compared to Raw.

### C. Comparing ML with TL

This comparison corresponds to a new result, when we want to check if one of the strategy (*i.e.* TL and ML) using a long-term memory to enhanced a BO is better than the other. The performance seems very similar between both methods, TL being better with 6 objects (A, m784, coke, m1109, m1110 and m1111 for TL) and ML achieving a greater score with the other 4 (C1, D, P1 and hammer_j). The mean absolute difference between their respective scores per object is a meager 0.9% with the biggest difference of 1.98% for the object m1111. With an average performance among every object of 83.96% for TL vs 83.63% for ML, the methods indeed appear to provide equivalent results. This is statistically confirmed as we cannot reject the H0 hypothesis of equality with a paired sample Wilcoxon signed-rank test (p-value=0.4443).

Because we have shown that both TL and ML strategies provide better learning for the robot, but TL and ML seems equivalent, we can now check if the combination of those approaches can provide an additional edge versus the ML or TL only mode.

### *D. Increase of performance with the combination of ML and TL compared to TL only*

We obtained an mean overall performance, all objects considered, of 85.62% for ML+TL versus 83.96% for TL only (with the mean of the worst run equal 81.06% and the mean of the best run is 89.61% for ML+TL compared to respectively 79.15% and 88.33% for TL). The average performance for each object is then significantly higher with ML+TL vs TL only (p-value of 0.0049 with a paired sample Wilcoxon signed-rank test). In fact, the average score is increased for almost every object (except equal for object D), with an average gain of 1.85% and among others, with the object hammer_j benefiting from it the most (+4.91%).

If we look at the performance of the worst run per object, there seems to be a tendency for better results for ML+TL compared to TL only, though not statistically significant yet (p-value=0.09 with a paired-sample Wilcoxon signed-rank test with H1). ML+TL provide better worst runs for 7 objects (with an average improvement of 3.55%, and a maximum of 8.48% for object m784), TL is more efficient at worst run for the 3 others (average improvement of 1.92%, and a maximum of 3.33% for object m1111).

The combination of ML+TL to enhance the BO indeed provides significantly better results than using TL only in average. Because we already have shown that the use of TL enhanced the BO compared to the raw amnesic version, we can also conclude that the ML+TL strategy is more efficient than the Raw BO.

### *E. Increase of performance with the combination of ML and TL compared to ML only*

We can first look at the data overall, without taking the object into account and only focusing on each final scores obtains from every run, in ML and ML+TL mode. We observe higher performance when the BO is enhanced with the combination of both ML+TL instead of ML only (mean=85.62%, mean of the min=81.06%, mean of the max=89.61% for ML+TL vs respectively 83.63%, 78.88% and 88.45% for ML). The improvement for ML+TL compared to ML only is then statistically significant when considering the mean performance for each object (p-value of 0.0049 for a paired sample Wilcoxon signed-rank test with H1). Indeed, when looking at the impact for each object separately, there is a gain of performance for 9 objects out of 10 (for an average increase of 2.26%) and only a slight decrease for the objects P1 (-0.40%). It has to be noted that the P1 object is actually the one with the best performance anyhow. Also, despite the fact that the global shapes are similar between P1 and P2, they are not a very good match compared to the other pairs. Thus in such case, the TL version seems to not be as efficient as usual, and the 3 points proposed are not providing promising set of parameters, which is roughly equivalent to take away 3 iteration budgets in the initial design phase by forcing the BO to explore point that might actually not be so interested to try, especially when compare to the very high expectation for the results.

In addition, when looking at the average performance for the worst run per object, we can observe that the ML+TL strategy provides a significant improvement when compared to ML, with 81.06% for ML+TL versus 78.88% for ML (p-value of 0.048 with a Wilcoxon signed-ranked test with alternative hypothesis). Indeed, the score of the worst run increased for 7 objects (a jump up to 7.27% for D and 6.67% for m1110) and decreased only slightly for 2 (3.9% for P1, but still achieving 85.5%, and 1.2% for m1109).

The use of the long-term memory to enhance the BO with both ML+TL mechanisms is overall significantly better at finding hyperparameters for efficient grasps than a BO guided through ML strategy only. Again, because we have previously seen that the ML enhanced BO provides better result than the amnesic BO, we know that the combination of ML+TL for the BO is also more efficient than the Raw BO.

### *F. Experiment with a real Fanuc robot*

The results of the experiments with a real robot are presented in Fig. III, showing the average final performance (in term of % of grasping success) and the final performance of the best run, with the 4 conditions for the BO. They confirm the benefit of the cognitive architecture framework for the learning mechanism shown in the simulated experiments, with a better average performance when using either one of the three methods that takes advantage of the long-term memory (TL BO=79.8%, ML BO=83.3%, ML+TL BO=88.1% vs only 74.0% for Raw BO). Moreover, we observe again that the robot achieves a more efficient learning when using the new combined techniques of ML+TL BO compared to other condition, either in average or when looking at the best run (best run for Raw=81.8%, for TL=80.3%, for ML=84.2% and for ML+TL=91.5%).

TABLE III
RESULTS FOR REAL GRASPING (IN % OF SUCCESS) WITH A FANUC ARM AND AN ELBOWED RUBBER SOFT OBJECTS

| Learning Method | Average final perf. among runs | final perf. of the best run |
|---|---|---|
| Raw BO | 74.0 | 81.8 |
| TL BO | 79.8 | 80.3 |
| ML BO | 83.3 | 84.2 |
| ML+TL BO | **88.1** | **91.5** |

## VI. CONCLUSION AND FUTURE WORK

In this study, we have shown how the robot can rely on its experience and long-term memory to improve the performance of an automatic Bayesian Optimization (BO) module in order to optimize continuous numerical parameters in a constrained setting for bin-picking application tasks for 10 different objects in simulation. The cognitive architecture allows both a Meta Learning (ML) and Transfer Learning (TL) mechanisms to enhance the optimizations runs of the BO, with a larger improvement in performance for the combination of both

(ML+TL). Even with a very small budget of 30 trials to optimize 9 parameters from expensive and noisy evaluations, the robot manages to achieve very good performances, with statistically significant gains overall on the expected scores when using TL vs Raw (83.96% vs 79.5%), ML vs Raw (83.6% vs 79.5%), ML+TL vs TL (85.6% vs 83.96%) and ML+TL vs ML (85.6% vs 83.6%). We indeed confirm previous results, using additional objects for benchmarks, that both strategies ML and TL using past experiences to enhance a BO provide better results for the robot than an amnesic one (*i.e.* Raw BO). In addition, the combined usage of both ML+TL methods allows better results for the optimization and the grasping performance that using only either ML or TL.

The gain of performance from ML+TL is consistent among all the objects when compared to Raw and TL methods only, and almost consistent with ML (only 1 objects out of 10 have a slightly reduced average performance). Interestingly, the object sharing the same class "hammer" (hammer_j, m1109, m1110 and m1111), but extracted from different databases, are benefiting from past experiences from another "hammer" class member (hammer_t). This leads us to assume that the similarity metric can in some extend be combined with a standard computer vision based classification method.

We can argue that the benefit of ML+TL consists in part of increasing the worst expected score for an optimization (*i.e.* the performance of the worst run among 6 for each object) with the average score of the worst run among all the object increasing from 69.09% (Raw) to 79.15% (TL) and 78.88% (ML), and finally reaching 81.06% (ML+TL). Indeed, the methods are guiding the BO in order to focus initially on more promising areas of the search space (TL) or to provide an easier search space to the BO by reducing the boundaries of the parameters to optimize (ML). This leads the BO to avoid some unpromising areas of the search space, not loosing precious iteration budget to explore them and reducing the possibility to be attracted toward local maxima within these regions and thus guaranteeing a better optimization overall especially with a low budget.

Those results have also been validated with an in-situ experiment, by controlling a real robotic arm Fanuc to realize a bin-picking task from an homogeneous heap of a rubber (*i.e* soft-body) elbowed tube. We observe a better performance when using either one of the methods exploiting the cognitive architecture and the long-term memory (*i.e.* either TL, ML or ML+TL BO). We find also the same results than the simulated experiments, with the best learning being achieved with the new techniques combining both ML+TL for the BO (average grasping performance of 88.1% with the best run obtaining 91.5% of success).

Overall, we have shown that our cognitive developmental architecture, composed by a long-term memory, reasoning modules and a Bayesian Optimization framework is efficient to endow the robot with the capacity to autonomously fine-tune hyperparameters from its software and adapt to the world quickly. By providing the robot with the capacity to retrieve data from similar past experiences when confronting to a new task, the system is able to start each new learning with a-priori knowledge (either from Meta Learning or Transfer Learning strategy) to guide its optimization, eventually achieving better results especially under a low budget constraint (*i.e* when the robot has to quickly learn without too much time). In this study, we have chosen to confront our cognitive architecture with the optimization of a professional grasping software used in the industry. This shows that developmental systems can be of used not only for service robots but also for industrial ones. In addition, because the BO is treating the software to optimize as a blackbox, without any assumptions, it can be easily adapted to fine-tune any other methods. We are thinking for instance to the Deep Learning methods used in robotics for diverse tasks (*e.g.* Computer Vision, Natural Language Processing, Object Manipulation) that usually requires a computationally costly fine-tuning step each time the domain is changing.

As for the next steps, we want to use this developmental learning framework in a real-to-sim transfer settings, as explored recently by Marco *et al.* for the cart-pole balancing problem [53] and more linked to our applications, by Breyer *et al.* for grasping problems [54] or by Kaspar *et al.* for manipulation tasks [54] with a robotic arm. In a sense, this will allow the robot to make use of its imagination, simulating future tasks (*e.g.* learning how to manipulate a simulated object $O_s$) in order to gather information about them, before using the knowledge to warmstart the optimization for the real interactions with the environment (*e.g.* learning how to manipulate the same real object $O_r$). This mental simulation endows the robot with a greater capacity to learn, possibly using several of its computer to run parallel scenarii or allowing him to 'dream' about it and exploring safely the simulated world when its physical body is shutdown at night in the factory.

Interestingly, in this setup, the robot is by essence not bound to the physical world and is not constrained by the objects put in front of him by external human agents for learning. Thus, the robot will have to decide itself which objects to explore during its mental simulations of training. Among others considerations, it will have to decide if it should 1) continue to explore grasping for already optimized objects, if it thinks it can find an even better solution, or 2) focus on completely unexplored objects in order to be prepared for future possible manipulations with them or similar instances. The robot will indeed need a self-driven continual learning approach to guide its mental learning, prioritizing tasks (*i.e.* exploiting known objects or exploring new ones) depending on future goals, for instance with the implementation of a curiosity mechanism as an intrinsic motivational framework as proposed recently [55], [56].

On the optimization process *per se*, the BO is currently evaluating a single objective function, where the score is based on a unique criteria (*i.e.* here the percentage of good grasping). Future work might extend this evaluation toward a multi-objective function, allowing to add other constrains on the score, for instance the speed of execution or the safety for the robot. This might be achieved by the BO with the use of a Pareto front of multiple criteria when evaluating the performance, as provided by the *mlrMBO* R-package [57],

[58].

The results about the use of classification labels in replacement of the visual similarity metrics, with the "hammer" object (hammer_$j$, m1109, m1110, m1111 optimized with the knowledge from hammer_$t$) is encouraging and can lead to interesting prospects. Indeed, such label can also be provided simply and on-the-fly in an intuitive manner (*e.g.* natural language processing) from the human, in order to designate the object name and class but also potentially some descriptions, with corresponding affordances or properties (*e.g.* a "flat" and "heavy" object, a "thin" object with a "handle") and use them to guide the robot in its manipulation. Some correlations and co-occurrences (as used to discover pronouns [59] or body-parts and basic motor skills [60]) between the reduced boundaries and the symbolic descriptions might potentially lead to embodied concept emergence [61] for the robot, related to the physical properties of the objects (*e.g.* the "thin" property being defined at some very thing range of one or several parameters allowing to grasp objects sharing this property).

## ACKNOWLEDGMENT

The authors would like to thank the members of the Imagine team at ECL for their constructive feedbacks during the paper internal reviews. We also thank Siléane for providing access and support to their Kamido solution, the software to control the Fanuc robot and the samples of real objects. This work is supported by the EU FEDER funding (FUI PIKAFLEX project), by the French National Research Agency, l'Agence Nationale de Recherche (ARES LabCom under grant ANR 16-LCV2-0012-01) and by the CHIST-ERA EU project "Learn-Real".

## REFERENCES

[1] A. Depierre, E. Dellandréa, and L. Chen, "Jacquard: A large scale dataset for robotic grasp detection," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3511–3516.

[2] M. Grard, E. Dellandrea, and L. Chen, "Deep multicameral decoding for localizing unoccluded object instances from a single rgb image," *International Journal of Computer Vision*, vol. 128, no. 5, pp. 1331–1359, 2020.

[3] J. Mockus, "The bayesian approach to local optimization," in *Bayesian Approach to Global Optimization*. Springer, 1989, pp. 125–156.

[4] ——, "Application of bayesian approach to numerical methods of global and stochastic optimization," *Journal of Global Optimization*, vol. 4, no. 4, pp. 347–365, 1994.

[5] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[6] D. Yogatama and G. Mann, "Efficient transfer learning method for automatic hyperparameter tuning," in *Artificial Intelligence and Statistics*, 2014, pp. 1077–1085.

[7] M. Petit, A. Depierre, X. Wang, E. Dellandréa, and L. Chen, "Developmental bayesian optimization of black-box with visual similarity-based transfer learning," in *The 9th Joint IEEE Int. Conf. on Development and Learning and on Epigenetic Robotics (ICDL-Epirob)*, 2018.

[8] M. Petit, E. Dellandréa, and L. Chen, "Bayesian optimization for developmental robotics with meta-learning by parameters bounds reduction," in *The 11th Joint IEEE Int. Conf. on Development and Learning and on Epigenetic Robotics (ICDL-Epirob)*, 2020.

[9] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, "Bayesian optimization for learning gaits under uncertainty," *Annals of Mathematics and Artificial Intelligence*, vol. 76, no. 1-2, pp. 5–23, 2016.

[10] B. Yang, G. Wang, R. Calandra, D. Contreras, S. Levine, and K. Pister, "Learning flexible and reusable locomotion primitives for a microrobot," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1904–1911, 2018.

[11] A. Rai, R. Antonova, F. Meier, and C. G. Atkeson, "Using simulation to improve sample-efficiency of bayesian optimization for bipedal robots." *J. Mach. Learn. Res.*, vol. 20, pp. 49–1, 2019.

[12] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, "Robots that can adapt like animals," *Nature*, vol. 521, no. 7553, p. 503, 2015.

[13] J. Nogueira, R. Martinez-Cantin, A. Bernardino, and L. Jamone, "Unscented bayesian optimization for safe robot grasping," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 1967–1972.

[14] J. Castanheira, P. Vicente, R. Martinez-Cantin, L. Jamone, and A. Bernardino, "Finding safe 3d robot grasps through efficient haptic exploration with unscented bayesian optimization and collision penalty," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1643–1648.

[15] T. Krajník, J. P. Fentanes, J. M. Santos, and T. Duckett, "Fremen: Frequency map enhancement for long-term mobile robot autonomy in changing environments," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 964–977, 2017.

[16] G. Sarthou, A. Clodic, and R. Alami, "Ontologenius: A long-term semantic memory for robotic agents," in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2019, pp. 1–8.

[17] F. Balint-Benczedi and M. Beetz, "Amortized object and scene perception for long-term robot manipulation," *arXiv preprint arXiv:1903.12302*, 2019.

[18] M. Edirisinghe, M. Muthugala, and A. Jayasekara, "Application of robot autobiographical memory in long-term human-robot social interactions," in *2018 2nd International Conference On Electrical Engineering (EECon)*. IEEE, 2018, pp. 138–143.

[19] C. Moulin-Frier, T. Fischer, M. Petit, G. Pointeau, J.-Y. Puigbo, U. Pattacini, S. C. Low, D. Camilleri, P. Nguyen, M. Hoffmann, *et al.*, "Dac-h3: a proactive robot cognitive architecture to acquire and express knowledge about the world and the self," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 4, pp. 1005–1022, 2017.

[20] R. Wood, P. Baxter, and T. Belpaeme, "A review of long-term memory in natural and synthetic systems," *Adaptive Behavior*, vol. 20, no. 2, pp. 81–103, 2012.

[21] M. Petit, T. Fischer, and Y. Demiris, "Lifelong augmentation of multimodal streaming autobiographical memories," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 8, no. 3, pp. 201–213, 2016.

[22] ——, "Towards the emergence of procedural memories from lifelong multi-modal streaming memories for cognitive robots," 2016.

[23] D. Vernon, M. Beetz, and G. Sandini, "Prospection in cognition: the case for joint episodic-procedural memory in cognitive robotics," *Frontiers in Robotics and AI*, vol. 2, p. 19, 2015.

[24] Y. Lu, L. Luo, D. Huang, Y. Wang, and L. Chen, "Knowledge transfer in vision recognition: A survey," *ACM Comput. Surv.*, vol. 53, no. 2, Apr. 2020. [Online]. Available: https://doi.org/10.1145/3379344

[25] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1633–1685, 2009.

[26] F. L. Da Silva and A. H. R. Costa, "A survey on transfer learning for multiagent reinforcement learning systems," *Journal of Artificial Intelligence Research*, vol. 64, pp. 645–703, 2019.

[27] R. C. Schank, *Dynamic memory: A theory of reminding and learning in computers and people*. cambridge university press, 1983.

[28] R. L. De Mantaras, D. McSherry, D. Bridge, D. Leake, B. Smyth, S. Craw, B. Faltings, M. L. Maher, M. T COX, K. Forbus, *et al.*, "Retrieval, reuse, revision and retention in case-based reasoning," *The Knowledge Engineering Review*, vol. 20, no. 3, pp. 215–240, 2005.

[29] M. Feurer, J. T. Springenberg, and F. Hutter, "Initializing bayesian hyperparameter optimization via meta-learning." in *AAAI*, 2015, pp. 1128–1135.

[30] A. Maesani and D. Floreano, "Viability principles for constrained optimization using a (1+ 1)-cma-es," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2014, pp. 272–281.

[31] A. Maesani, G. Iacca, and D. Floreano, "Memetic viability evolution for constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 125–144, 2015.

[32] E. Tulving *et al.*, "Episodic and semantic memory," *Organization of memory*, vol. 1, pp. 381–403, 1972.

[33] E. Tulving, "Memory and consciousness." *Canadian Psychology/Psychologie canadienne*, vol. 26, no. 1, p. 1, 1985.
[34] B. Bischl, J. Richter, J. Bossek, D. Horn, J. Thomas, and M. Lang, "mlrMBO: A Modular Framework for Model-Based Optimization of Expensive Black-Box Functions." [Online]. Available: http://arxiv.org/abs/1703.03373
[35] A. I. J. Forrester, A. J. Keane, and N. W. Bressloff, "Design and analysis of" noisy" computer experiments," *AIAA journal*, vol. 44, no. 10, pp. 2331–2339, 2006.
[36] M. Stein, "Large sample properties of simulations using latin hypercube sampling," *Technometrics*, vol. 29, no. 2, pp. 143–151, 1987.
[37] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced lectures on machine learning*. Springer, 2004, pp. 63–71.
[38] V. Picheny, D. Ginsbourger, Y. Richet, V. Picheny, D. Ginsbourger, and Y. Richet, "Optimization of Noisy Computer Experiments with Tunable Precision," *Technometrics*, vol. 55, no. 1, pp. 2–13, 2013.
[39] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*. IEEE, 1996, pp. 312–317.
[40] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary computation*, vol. 11, no. 1, pp. 1–18, 2003.
[41] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, vol. 1, no. 2, p. 4, 2017.
[42] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.
[43] S. Chaiklin *et al.*, "The zone of proximal development in vygotsky's analysis of learning and instruction," *Vygotsky's educational theory in cultural context*, vol. 1, no. 2, pp. 39–64, 2003.
[44] G. Pointeau, M. Petit, and P. F. Dominey, "Successive developmental levels of autobiographical memory for learning through social interaction," *IEEE Transactions on Autonomous Mental Development*, vol. 6, no. 3, pp. 200–212, 2014.
[45] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for lvcsr using rectified linear units and dropout," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 8609–8613.
[46] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in *International conference on machine learning*, 2013, pp. 1319–1327.
[47] E. J. Dudewicz and E. C. Van Der Meulen, "Entropy-based tests of uniformity," *Journal of the American Statistical Association*, vol. 76, no. 376, pp. 967–974, 1981.
[48] A. Nayebi, A. Munteanu, and M. Poloczek, "A framework for bayesian optimization in embedded subspaces," in *International Conference on Machine Learning*, 2019, pp. 4752–4761.
[49] R. Moriconi, M. P. Deisenroth, and K. S. Kumar, "High-dimensional bayesian optimization using low-dimensional feature spaces," *Machine Learning*, vol. 109, no. 9, pp. 1925–1943, 2020.
[50] K. Mamou, "Volumetric hierarchical approximate convex decomposition," in *Game Engine Gems 3*, E. Lengyel, Ed. A K Peters, 2016, pp. 141–158.
[51] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The princeton shape benchmark," in *Proceedings Shape Modeling Applications, 2004.* IEEE, 2004, pp. 167–178.
[52] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 927–934, 2012.
[53] A. Marco, F. Berkenkamp, P. Hennig, A. P. Schoellig, A. Krause, S. Schaal, and S. Trimpe, "Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with bayesian optimization," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1557–1563.
[54] M. Breyer, F. Furrer, T. Novkovic, R. Siegwart, and J. Nieto, "Flexible robotic grasping with sim-to-real transfer based reinforcement learning," *arXiv preprint arXiv:1803.04996*, 2018.
[55] A. Laversanne-Finot, A. Péré, and P.-Y. Oudeyer, "Curiosity driven exploration of learned disentangled goal spaces," in *Conference on Robot Learning (CoRL)*, 2018.
[56] C. Colas, P. Fournier, M. Chetouani, O. Sigaud, and P.-Y. Oudeyer, "Curious: intrinsically motivated modular multi-goal reinforcement learning," in *International conference on machine learning*, 2019, pp. 1331–1340.
[57] "Multi-objective parameter configuration of machine learning algorithms using model-based optimization," *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*, 2017.
[58] D. Horn, T. Wagner, D. Biermann, C. Weihs, and B. Bischl, "Model-based multi-objective optimization: taxonomy, multi-point proposal, toolbox and benchmark," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2015, pp. 64–78.
[59] G. Pointeau, M. Petit, G. Gibert, and P. F. Dominey, "Emergence of the use of pronouns and names in triadic human-robot spoken interaction," in *Development and Learning and Epigenetic Robotics (ICDL-Epirob), 2014 Joint IEEE International Conferences on*. IEEE, 2014, pp. 146–152.
[60] M. Petit and Y. Demiris, "Hierarchical action learning by instruction through interactive grounding of body parts and proto-actions," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3375–3382.
[61] T. Taniguchi, T. Nagai, T. Nakamura, N. Iwahashi, T. Ogata, and H. Asoh, "Symbol emergence in robotics: a survey," *Advanced Robotics*, vol. 30, no. 11-12, pp. 706–728, 2016.

**Maxime Petit** received the M.Sc. degree in computer science from the University of Paris-Sud, France, and an engineering degree in biosciences (bio-informatics and modelling) from the National Institute of Applied Sciences (INSA) Lyon, France, both in 2010. In 2014, he received a Ph.D. in Neurosciences from the National Institute of Science and Medical Research (INSERM), Lyon, within the Robot Cognition Laboratory. He was a Research Associate at the Personal Robotics Lab, Imperial College London, and is Research Associate at Ecole Centrale de Lyon (France) since 2017. His research interests include developmental robotics, memory and reasoning in robotics, especially linked to social interaction through spoken language with a human.

**Emmanuel Dellandréa** is an Associate Professor at Ecole Centrale de Lyon, France, since 2004. He was awarded a Master and Engineering degrees in Computer Science from the Université de Tours, France, in 2000 followed by a Ph.D. in Computer Science in 2003. He obtained the Habilitation to Drive Research (HDR) in 2020 from Ecole Centrale de Lyon. His research activities led in the LIRIS laboratory (CNRS UMR 5205) are related to image and video understanding, computer vision, machine learning, robotics, as well as affective computing, including recognition of emotion induced by images and videos.

**Liming Chen** is a distinguished Professor at the Department of Mathematics and Computer Science, Ecole Centrale de Lyon, University of Lyon, France. He received his BSc in Mathematics and Computer Science from the University of Nantes in 1984, his MSc and PhD in computer science from the University Pierre and Marie Curie Paris 6 in 1986 and 1989 respectively. He was an associate professor at the Université de Technologie de Compiègne before he joined Ecole Centrale de Lyon as Professor in 1998. He served as the Chief Scientific Officer in the Paris-based company Avivias from 2001 to 2003, and the scientific multimedia expert in France Telecom R&D China in 2005. He was the head of the Department of Mathematics and Computer science from 2007 through 2016. His current research interests include computer vision, machine learning, and multimedia with a particular focus on robot vision and learning since 2016. He is an associate editor for Eurasip Journal on Image and Video Processing and a senior IEEE member.