Cognitive Skills Models: Towards Increasing Autonomy in Underwater Intervention Missions*

Luís Santos[1], Jorge Sales[2], Pedro J. Sanz[3] and Jorge Dias[4]

***Abstract*— In this paper a set of computational cognitive skills is developed and implemented on a Autonomous Underwater Vehicle (AUV) towards increasing its autonomy in intervention missions. Literature reveals that very few underwater systems have the ability to perform fully autonomous interventions without any kind of user specification/interaction. Our implementation divides the cognitive process two-fold: (1) learning actions from human demonstration and (2) autonomous task execution. Task parametrizations are generalized and encoded using Gaussian Mixture Models (GMM), which in turn are allocated into memory. On run-time execution, the system will perform continuous environment and self assessment using Bayesian Inference. The estimated information is used to probe the memory, locating the adequate (learned) solution. The manipulator configuration sequences are synthesized via Gaussian Mixture Regression (GMR) methods. This work is integrated in a larger project, where evaluation follows a process of increasing experimental complexity, in this specific case, validated using a realistic underwater simulator. Promising results, indicate that the developed cognitive skills can be transferred and tested on the real system.**

I. INTRODUCTION

Underwater manipulation systems are limited, in the sense they usually require a human operator to guide them through their actions. To increase their autonomy, they should exhibit a set of cognitive skills which include memory, learning, reasoning, decision making and reproducing actions. This paper represents a work in progress, focusing on developing the necessary skills for an Autonomous Underwater Vehicle for intervention, which is defined as physical interaction with the environment, using for example a robotic arm.

With this aim, we propose a four stage cognitive process:

1) Sensing the environment;
2) Interpreting the data;
3) Learning and memorizing manipulation skills;
4) Reproducing those skills for autonomous manipulation.

*This work was partly supported by Spanish Ministry of Research and Innovation DPI2011-27977-C03 (TRITON Project), by Foundation Caixa Castelló-Bancaixa PI.1B2011-17, by Universitat Jaume I under grant E-2013-33, by the FCT - Portuguese Foundation for Science and Technology, Grant # 65935/2009, by the Institute of Systems and Robotics from University of Coimbra, Portugal and Khalifa University, Abu Dhabi, UAE.

[1]Luís Santos is with the Institute of Systems and Robotics, Department of Electrical and Computer Engineering, University of Coimbra, Coimbra, Portugal `luis@isr.uc.pt`

[2]Jorge Sales is with the Computer Science and Engineering Department, University of Jaume-I, Castellón, Spain `salesj@uji.es`

[3]Pedro J. Sanz is with the Computer Science and Engineering Department, University of Jaume-I, Castellón, Spain `sanzp@isr.uc.pt`

[4]Jorge Dias is with the Institute of Systems and Robotics, Department of Electrical and Computer Engineering, University of Coimbra, Coimbra, Portugal and the Robotics Institute, Khalifa University, Abu Dhabi, UAE `jorge@isr.uc.pt`

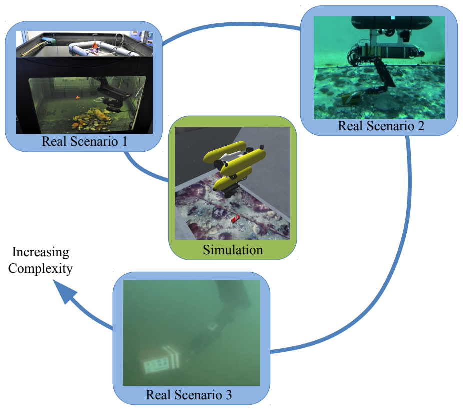


Fig. 1: Development strategy: the core techniques are designed, developed and prototyped inside the UWSim simulator. Then, the research results generated by this activity will be tested on real scenarios of increasing complexity.

We present task oriented approach, in which upon interpreting the scene, the system will probe its memory into retrieving an appropriate action. Memorized actions are built upon task generalization based on "learning by demonstration" principles. Experimental validation will be carried out in the realistic 3D Underwater Simulator UWSim [1] and with specifically developed modules to allow the user interaction for the learning process.

A. Related Work

In the field of the underwater intervention, including physical interaction, it is worth mentioning previous projects like SAUVIM [2], intended for deep interventions, which demonstrated the autonomous recovery of seafloor objects by using a very bulky and expensive system; and TRIDENT [3] [4], that demonstrated the first multipurpose object search and recovery strategy in 2012, able to operate in shallow waters, but still requiring user interaction for the specification of the intervention mission. To the best of our knowledge, there exists at least one ongoing project running in the underwater intervention context (including physical interaction) funded by European Commission: PANDORA [5], which aims to increase AUVs autonomy. In this context, a learning solution for autonomous robot valve turning, using Extended Kalman Filtering and Fuzzy Logic to learn manipulation trajectories via kinaesthetic teaching was recently proposed [6].

Reinforcement learning has been applied by Lopes et al. [7], towards learning and imitating actions. In case of unknown conditions, the robot will simply keep repeating to known motion pattern. Pastor et al. [8] investigated "learning by demonstration" methods using Dynamic Movement Primitives. A library is recorded using symbolic descriptors, using kinaesthetic learning. To execute, the system is manually given a specific set of task parameters to perform a single execution. Support Vector Machines are used to learn and actualize object affordances, through behavioral parameters, to establish object-effect relations in [9]. Pushing and pulling household objects by learning the impact and kinematic parameters is addressed in [10]. They centroid and pose information, extracted from a Microsoft Kinect. The goal is giving a robot the ability to select between left and right arm and detect the reach space of its actions.

B. Our Approach

As seen from the state of the art, underwater systems still have limited autonomy. Moreover, most "learning by demonstration" methodologies are implemented in controlled environments and do not show the ability to modify the task during execution. Underwater scenarios are subject to uncertainty and noise, given its own environmental nature, with constantly changing conditions, and might require the AUV to modify its course of action in run-time execution.

This work represents a stage within an experimental process of increasing complexity (see Figure 1). There is an initial phase, where developed models are executed in a realistic simulation environment. The objective is to refine and validate the AUV's implemented models in a simulation environment, after which they will be transferred into a real system, conveying real hardware in real environments with increasing number of uncontrolled variables (disturbances, visibility, noise, etc.). Moreover, when moving to real robotic platforms, the scenarios and skills will also be expanded. The presented work refers to the first stage, model design and implementation for simulation testing.

A human expert is given a target action, which he has to teleoperate in UWSim. The simulator has the ability to give a realistic sense of the underwater conditions during an action performance. Bear in mind, we only use sensed information that is consistent with what a real AUV has access to, in real scenarios. After a number of different trials, performed actions are generalized and stored in a categorically organized memory. Symbolic properties are associated to sensed data, which define environment, object and action models. These are used by the system to infer of a solution for a task, using a Bayesian based decision process. Our framework is task oriented, where tasks are parametric representations of a global objective. From our solution we can identify the following main contributions:

- An analysis model, which can adapt to changing conditions of underwater scenarios;
- A memory which can store generalized action knowledge and is searchable by context/symbolic information;
- An AUV which is cognitive in the sense it can autonomously perceive the environment, decide is course of action and provide the mechanical system with a kinematic solution for a given identified task.

II. UWSim: Data Sensing and Processing

The UWSim is used to simulate a real underwater scenario (i.e. a water tank), including an underwater vehicle equipped with a robotic arm. In our validation arrangement, we consider that the vehicle is located at a fixed position with respect to the target object, so that a user can focus on object manipulation and not on guiding the vehicle.

The UWSim [1] is a software tool for visualization and simulation of underwater robotic missions. The software is able to visualize an underwater virtual scenario that can be configured using standard modeling software, and do the interface with external control programs through the *Robot Operating System (ROS)*. UWSim is currently used in different ongoing projects funded by European Commission (i.e. MORPH [11] and PANDORA) in order to perform HIL (Hardware in the Loop) experiments and to reproduce real missions from the captured logs.

The simulated robotic arm is a virtual representation of the real arm considered for the validation scenario (CSIP Light-weight ARM5E). It has 5 D.O.F. and can be equipped with different kind of grippers, which can also be equipped with sensors to provide contact information, being this information useful when grasping an object. An example of a successful reactive tactile sensor test recently performed in laboratory conditions (water tank) can be seen on-line [12].

The low-level control architecture, including the arm kinematics, was implemented in C++ and makes use of *ROS* for inter-module communications. The kinematic module accepts either Cartesian or joint information (i.e. pose, velocities). The *ARM5Control* module uses joint velocities in order to compute motor RPM [13].

III. Data Interpretation

The system continuously performs inference about the scene and its own state, based on information acquired from built-in sensing capabilities. A Dynamic Bayesian Network is proposed to perform estimation about the various states, in an interpretation process, developed using Bayesian Programming [14], [15], [16]. The relevant variables are defined in the first step of Bayesian Programming, such that:

- $A \in \{pick\text{-}up, push\}$ is a random variable denoting the different actions our manipulator can perform.
- $G \in \{top, lateral, front\}$ is a random variable denoting the end-effector configuration to act on the object.
- $P \in \{approach, reach, manipulation\}$ is a random variable which identifies the current action phase. The initial motion towards an object occurs on the approach phase ; The end-effector takes the appropriate grasp configuration in the reach-to-contact phase; Manipulation correspond to mechanically acting on the object.
- $O \in \{box, stone, vessel\}$ is a random variable defining the known object classes.
- $\Theta \equiv [\theta_1, \cdots \theta_n] \in [-\pi, \pi]$ is a random variable representing the angle in radians of a given joint n.
- $F \equiv (f_1, \cdots, f_b) \in \mathbb{R}^b$ represents the sensed information as a vector of processed laser range measures.

- $d \in \mathbb{R}$ is a random variable measuring the distance from the *end-effector* to the target object.

To estimate the joint state of an action and its corresponding characteristics, we use Bayesian Inference on the decomposition equation of Figure 2. The most likely candidate object class is estimated upon weighing the uncertainty of each evidence $\in F$. Each different action phase P depends on the existing knowledge about the manipulator configuration Θ together with the relative distance to the target object d. The geometric configuration is used as evidence for inference over the Grasp Type state. These dependencies are reflected as conditional probabilities, elements of the decomposition of the joint distribution, which are also reflected in the Direct Acyclic Graph (DAG) in Figure 3. The DAG is divided into different abstraction levels for easier comprehension. There is the action space, the characterization space encompassing the phase and the grasp type, the object space with information about the target object and the feature space, which holds the observable evidence.

Likelihood distributions, which are function of random variables $\in \mathbb{R}$, are formulated upon Gaussian distributions, as they are expected to be normally distributed. The density functions that depend on discrete variables, as is the case of the variables whose space state is symbolic, do not follow any particular known parametric form. They represent statistical information of the different symbols being observed for a given state, and are encoded using stochastic matrices.

Once the model is fully specified we can now query it for information. To obtain an estimate of the most probable action state given observable evidence, we apply the *Maximum A Posteriori* (MAP) method, in which inference over the action variable is given as in equation (1).

$$\hat{A}_{\text{MAP}} = arg_A max\ P(\Theta|G)P(O|G)P(d|P) \\ P(\Theta|P)P(F|O)P(P|A)P(O,G|A)P(A) \quad (1)$$

$$\textit{description}\left\{\begin{array}{l}\textit{specification}\left\{\begin{array}{l}\textit{Variables:}\\ A,G,P,O : \text{scene and state symbolic information;}\\ \Theta \in [-\pi,\pi] : \text{real values of each joint } n;\\ F \in \mathbb{R}^b : \text{laser range finder characteristic vector;}\\ D \in \mathbb{R} : \text{distance from the } \textit{end-effector} \text{ to the object;}\\ \textit{Decomposition:}\\ P(A,G,P,\Theta,D,O,F) =\\ \quad P(\Theta|G)\ P(O|G)\ P(D|P)\ P(\Theta|P)\\ \quad P(F|O)\ P(P|A)P(O,G|A)\ P(A)\\ \textit{Formulation:}\\ P(A) = \textit{Histogram}\\ P(\Theta|G), P(O|G), P(D|P),\\ P(\Theta|P) \text{ and } P(F|O) = \textit{Gaussian}\\ P(P|A), P(O,G|A) = \textit{Stochastic matrix}\end{array}\right.\\ \textit{identification:} \text{ parameters learned from training data } F,\Theta,D\end{array}\right.$$

Fig. 2: Bayesian Program description: (1) enumerates the relevant variables; (2) joint distribution decomposition; (3) formulation of the conditional distributions in parametric forms; (4) Identification stage where parameters of the Gaussian distributions are estimated from experimental data.

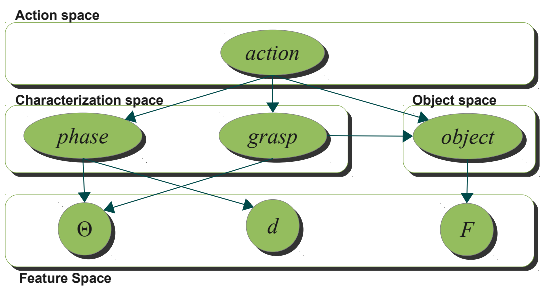


Fig. 3: Directed Acyclic Graph of the proposed Dynamic Bayesian Network. Nodes represent variables and directed arcs represent variable dependencies.

The *MAP* method is a point estimate, which will return the most probable state from each of the symbolic variables.

A. Experimental Results

In this section we present the experiments which demonstrate the interpretation capabilities of our system.

TABLE I: Confusion Table of Symbolic Classifications. Acronyms: (A) Approach; (R) Reach-to-Contact; (M) Manipulation; (T) Top; (L) Lateral; (F) Front; (V) Vessel; (B) Box; (S) Stone; (Pu) Pick-Up; (P) Push.

(a) Action Confusion Table.

	(Pu)	**(P)**
(Pu)	1.00	0.00
(P)	0.10	0.90

(b) Object Confusion Table.

	(V)	**(B)**	**(S)**
(V)	0.95	0.00	0.05
(B)	0.00	1.00	0.00
(S)	0.01	0.00	0.99

(c) Grasp Type Confusion Table.

	(T)	**(L)**	**(F)**
(T)	1.00	0.00	0.00
(L)	0.00	0.95	0.05
(F)	0.10	0.00	0.90

(d) Phase Confusion Table.

	(A)	**(R)**	**(M)**
(A)	0.93	0.04	0.03
(R)	0.32	0.55	0.13
(M)	0.00	0.03	0.96

Action classification shows high precision. It is a fact that we have a reduced number of Action classes, which is in part justified by the reduced number of DoF of our manipulator. Increasing the number of DoF will allow the system to perform a wider range of actions, while simultaneously providing the model with extra variables. Discrimination will naturally increase, hence precision is expected to hold.

Object detection is robust, given they have different shapes. Considering scenarios where a gripper is to pick an object, the shape is probably the most important attribute. Therefore, in case we augment our object database, confusion might exist in similarly shaped objects, which is not critical from a grasping point of view. However, inferring additional object properties from sensed data, might increase discrimination in these cases.

We can see that the results for action phases in Table I (d) suggest a delay when detecting the *Reach-to-Contact*. After a thorough step-by-step analysis, we found this confusion to come from the distance d thresholds in the model training. In fact, the Reach-to-Contact phase is defined by short ranges and time periods, generating Gaussian Distributions of low

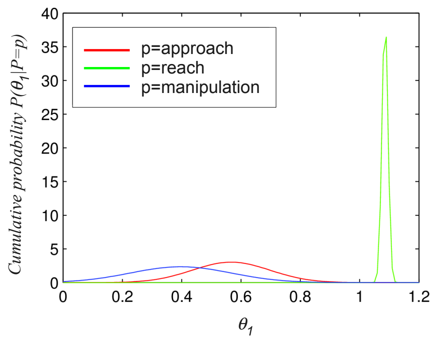


Fig. 4: Learned Gaussian Distributions for $P(\theta_1|P = p)$, with $p = \{approach, reach, manipulation\}$.

variance (see Figure 4). Hence, when in the presence of noise, it easily diverges to the nearest phase class, more specifically, *Approach*. In light of models' precision and discussed issues, the achieved promising results (see Table II), indicate that we can move into a more complex experimental phase, the underwater tank.

TABLE II: Global precision per symbolic variable.

	Symbolic Variables			
	action	*object*	*grasp*	*phase*
Precision (%)	95.00	99.12	96.66	86.50

IV. Action Generalization and System Memory

Learning which *attitude* the end-effector should exhibit at the different phases, requires the definition of the manipulation skills based on sensed information and learning them, from human demonstrations. Consider a set Ω, containing ω trajectory samples for a given action $a \in A$, characterized by a specific Grasp Type $g \in G$ for a determined object $o \in O$. Let each ω be manually segmented in time with respect to the different phases $p \in P$. We consider that a trajectory can be parametrized as $\omega = f(\theta_n, d, t)$, where t represents the variable time. This function represents the behaviour of a single manipulator joint θ_n in time, but also encodes a constraint value range θ_n function of the distance d to the target object. We propose the following parametrization functions to represent ω.

$$\omega = \begin{cases} d & = f(t) \\ \theta_i & = f(t) \quad ,i = 1, \cdots, n \\ \theta_i & = f(d) \end{cases} \tag{2}$$

The graphics in Figure 5 present collection of m human demonstrations, which plot an example graph for each of the aforementioned functions.

For an efficient representation, we need to obtain a generalized model for each of the functions. We propose an encoding process which uses Gaussian Mixture Models (GMM). A mixture model can efficiently be applied to reduce the dimensionality of a large number of sampled action trials. In cases where the number of available learning trials is high, there is an obvious advantage of saving a fixed

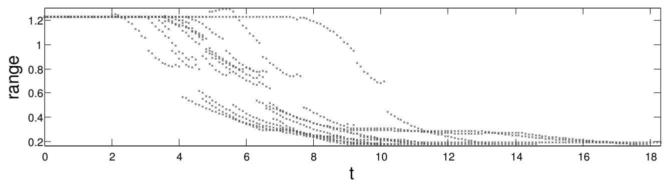


Fig. 5: Raw data functions for $m = 10$ trials of a **Pick** a **Box** from the **Top**, at the **Approach** *Phase*.

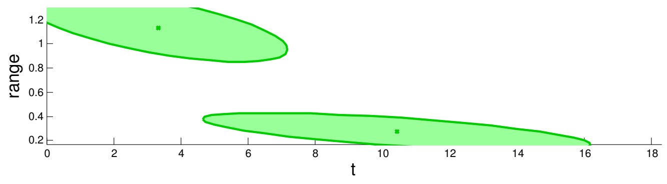


Fig. 6: GMM representation of raw data function.

number of representative parameters, rather than the raw data. Moreover, we can apply regression function, which using those parameters can retrieve an approximation of the trial average.

In equation (3), the formulation for the GMM encoding is presented for variables where $\Gamma \equiv (\gamma_1, \cdots, \gamma_z)$, where $\gamma = \{\theta_1, \cdots, \theta_n, d, t\}$.

$$p(\Gamma|\chi) = \sum_{i=1}^{k} \phi_i f(\Gamma|\mu_i, \Sigma_i) \tag{3}$$

Each function is represented by a set of weights and Gaussian Parameters $\chi \equiv (\phi_1, \cdots, \phi_k, m_1, \cdots, m_k, \sigma_1, \cdots, \sigma_k)$ for a number of k different Gaussian distributions. Mean and variance vectors are given by $\mu = (m_1, \cdots, m_k)$ and $\Sigma = (\sigma_1, \cdots, \sigma_k)$. Figure 6 presents the Gaussian Mixture Model of the functions of Figure 5, considering $k = 2$. The optimal Gaussian parameters are estimated from a Maximum Likelihood (ML) approach, for which the Expectation Maximization algorithm was used.

$$L(\Gamma|\chi) = \sum_{j=1}^{z} log \sum_{i=1}^{k} \phi_i f(\gamma_j|m_i, \sigma_i) \tag{4}$$

Given the estimation process follows ML, we selected a small number of components, $k = 2$, so as to avoid the effect of over-fitting.

A. Memorizing and Accessing Manipulation Skills

Given different sets of GMM parameters χ for different actions and characteristics, there is the need to develop a system action memory. We propose an indexing scheme, where the memory location is encoded by enumerating variable state, creating a discrete 4-D manifold space.

$$index = f(a, o, g, p), \text{ where } \begin{cases} a \in A \\ o \in O \\ g \in G \\ p \in P \end{cases} \tag{5}$$

Indexes a, o, p, g correspond to the states for symbolic variables. We propose an intuitive way of representing this cognitive memory, using a B-Tree structured memory, where the index at each level will lead to the selection of a leaf. Each leaf contains the generalized manipulator configuration parameters χ for solving a task with specific characteristics (see Skills Memory block in Figure 8a). It keeps data sorted and allows for searching or sequential access, insertions, and update in logarithmic time.

Given that we are using a supervised learning process, at the beginning of each demonstration, the user is instructed to perform a specific action, using a determined grasp type to interact with a target object. The different phases are also automatically assigned upon defining specific distance thresholds from the *end-effector* to the object. Therefore, the system automatically records that experiment to the specific memory location given by those index selections. Memory size benefits from the mixture approach, as the number of leafs is exponentially proportional to the number of different actions. Having a parametric representation of a generalized action, helps maintaining a sizeable, efficient memory.

B. Manipulation Skills Synthesis

In this paper, we are proposing a Task Oriented approach where, given a task to be executed, the system will retrieve the required learned Gaussian parameters to perform it. To decode these parameters into generalized configuration function sequences $\hat{\omega}$, we apply a Gaussian Mixture Regression (GMR) method. Let the joint data samples (Γ, ω), where Γ and ω are observations and target motion functions respectively, follow a Gaussian Mixture distribution as in equation (3). The parameters for the model are given by χ, and the joint distribution can be expressed as a sum of the products of the marginal density over Γ and the probability density function of ω conditioned on Γ:

$$P(\Gamma, \omega) = \sum_{i=1}^{k} \phi_i P(\omega|\Gamma, m_i, \sigma_i) P(\Gamma, \mu_k, \Sigma_k). \tag{6}$$

The marginal distribution is given by:

$$P(\Gamma) = \sum_{\omega} P(\Gamma, \omega) = \sum_{i=1}^{k} \phi_i P(\Gamma, \mu_i, \Sigma_i). \tag{7}$$

The regression function can be obtained from (6) and (7):

$$P(\omega|\Gamma) = \frac{\sum_{i=1}^{k} \phi_i P(\omega|\Gamma, m_i, \sigma_i) P(\Gamma, \mu_i, \Sigma_i)}{\sum_{i=1}^{k} \phi_i P(\Gamma, \mu_i, \Sigma_i)} \tag{8}$$

Where the mean and covariance of the conditional distribution $P(\omega|\Gamma)$ can be computed as:

$$\begin{aligned} m_k &= \mu_k + \Sigma_k \Sigma_k^{-1} (\Gamma - \mu_k) \\ \sigma_k^2 &= \Sigma_k - \Sigma_k \Sigma_k^{-1} \Sigma_k \end{aligned} \tag{9}$$

The graphs in Figure 7[1] illustrate the covariance space after the regression (color *blue*) and the synthesized function $\hat{\omega}$ (in *red*), obtained upon a Maximum-Likelihood Estimation.

$$\hat{\omega} = MLE(m_k, \sigma_k^2) \tag{10}$$

[1]A MatLab code developed by Calinon et al. [17] was used.

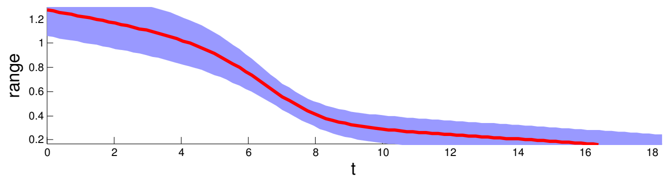


Fig. 7: Synthesized function using GMR technique.

Each of these functions is computed for all θ_n in the manipulator joint structure, which will give a synthesized configuration along time and range. The data is given to the simulator, in order to be executed by the kinematic control in the underwater manipulator UWSim.

The system is continuously assessing task characteristics and goals using its interpretation capabilities. Whenever a modification in task parameters occurs, a memory probe function is triggered to provide a new $\hat{\omega}$, satisfying the required changes. We consider our approach as task oriented, in the sense that we provide a $\hat{\omega}$ for a complete action phase. This functions are updated based on two conditions:

1) If the task characteristics A, O, G change.
2) When a new phase $p \in P$ begins.

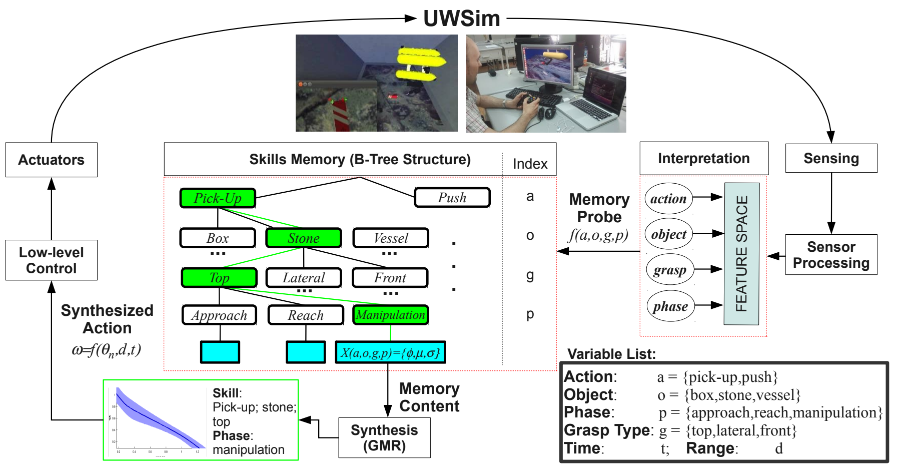
Hence, we can define a $\hat{\omega}_p$ as the manipulator motion sequence at phase p. To perform a task T encompassing a set of characteristics, the manipulator should be able to get a sequence, such that:

$$\begin{aligned} &T_{(A=a,O=o,G=g)} = \\ &\underbrace{\hat{\omega}_{p=approach} \rightarrow \hat{\omega}_{p=reach} \rightarrow \hat{\omega}_{p=manipulation}}_{\text{Action phases } p \in P \text{ are sequential and assessed from classification.}} \end{aligned} \tag{11}$$

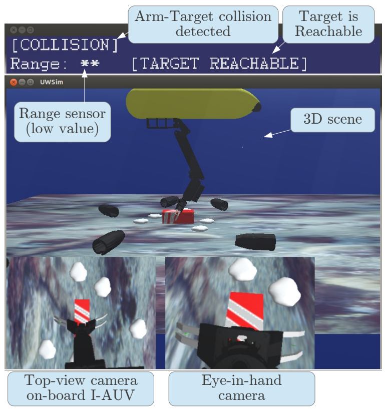
This parametric approach, allows to construct a chain of motion sequences to perform more complex tasks. High discrepancies for θ_n between two phases are solved using kinematic control. In case the system selects and empty memory cell, it will search its closest neighbours for a solution. Neighbourhood is defined as a leaf which shares at least 3 common states with the original solution.

V. Autonomous Execution Experiments

To perform the autonomous execution experiments, a scenario is loaded into the simulator. The goal is for the AUV to pick an object autonomously after a learning process, from multiple expert user teleoperations. The human-robot interaction involves: (1) the use of a gamepad, (2) a complete 3D visual information display, and also (3) sensor information, such as the distance to the target object and collision detection. A *User Monitor* module was specifically developed to display this information, connecting to the simulator via *ROS* interface (see Figure 8b). The module also sends vibro-tactile information to the gamepad. This is done to give the user a sense of contact between the arm and/or the end-effector and the target object. The first step in the learning process conveys data acquisition from several trials, from different users. The second step involves learning the actions and scene parameters, which are generalized and stored into memory.



(a) Global system block diagram.



(b) User Monitor Module.

Fig. 8: Global system block diagram, encompassing acquisition, interpretation, memory and execution stages; and User Monitor module connected to UWSim simulator to provide feedback to the user during the learning stage.

On execution, the memory is constantly probed according to sensed information. The adequate motion sequence functions (one for each joint) are synthesized and sent to the kinematics module (low-level controller). At this stage, the proper input to the simulator is generated considering a specified rate, resolution and velocity ranges. Discontinuities are automatically solved using smooth interpolation.

This way, the training and learning loop is closed and the simulator is able to display the automatically generated grasp execution after a training stage. Examples for the training and execution are available on-line at the following website: `https://sites.google.com/a/uji.es/learning`.

VI. CONCLUSIONS

With the aim of increasing the autonomy levels for Underwater Intervention Missions, we have developed a set of computational cognitive skills that allow the system to automatically grasp an object after a learning process. The developed scheme allows the robot to learn by demonstration, memorize, decide and access autonomously the suitable solution to solve a task. The implemented solution (to be publicly available) works as independent *ROS* nodes that can connect both to the simulator and the real environments, as they share the same interface. The proposed models have been validated in a realistic underwater simulation environment, paving the way for testing them in more complex environments (as previously described in Figure 1).

REFERENCES

[1] M. Prats, J. Pérez, J. Fernández, and P. Sanz, "An open source tool for simulation and supervision of underwater intervention missions," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 2577–2582.

[2] G. Marani, S. K. Choi, and J. Yuh, "Underwater autonomous manipulation for intervention missions AUVs," *Ocean Engineering*, vol. 36, pp. 15–23, 2009.

[3] FP7-TRIDENT, "Marine Robots and Dexterous Manipulation for Enabling Autonomous Underwater Multipurpose Intervention Missions (TRIDENT)," http://www.irs.uji.es/trident/.

[4] P. J. Sanz, P. Ridao, G. Oliver, G. Casalino, C. Insaurralde, C. Silvestre, C. Melchiorri, and A. Turetta, "TRIDENT: Recent improvements about autonomous underwater intervention missions," in *3rd IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles (NGCUV 2012)*, Porto, Portugal, 04 2012.

[5] FP7-PANDORA, "Persistent Autonomy through learNing, aDaptation, Observation and Re-plAnning (PANDORA)," http://persistentautonomy.com/.

[6] A. Carrera, S. R. Ahmadzadeh, A. Ajoudani, P. Kormushev, M. Carreras, and D. G. Caldwell, "Towards Autonomous Robotic Valve Turning," *Journal of Cybernetics and Information Technologies (CIT)*, vol. 12, no. 3, pp. 17–26, 2012.

[7] M. Lopes, F. Melo, and L. Montesano, "Affordance-based imitation learning in robots," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2007, pp. 1015–1021.

[8] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 2009, pp. 763–768.

[9] E. Ugur, E. Oztop, and E. Sahin, "Going beyond the perception of affordances: Learning how to actualize them through behavioral parameters," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 4768–4773.

[10] T. Hermans, J. M. Rehg, and A. F. Bobick, "Decoupling Behavior, Control, and Perception in Affordance-Based Manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) Workshop on Cognitive Assistive Systems*, October 2012.

[11] FP7-MORPH, "Marine Robotic System of Self-Organizing, Logically Linked Physical Nodes (MORPH)," http://morph-project.eu/.

[12] "Reactive tactile sensor test." [Online]. Available: http://youtu.be/42ZklVwNaqc

[13] J. J. Fernández, M. Prats, J. C. García, R. Marín, and A. Peñalver, "Manipulation in the Seabed: A New Underwater Manipulation System for Shallow Water Intervention," in *1st Conference on Embedded Systems, Computational Intelligence and Telematics in Control, CESCIT 2012*, University of Würzburg, Germany, April 2012, pp. 314–319.

[14] P. Bessiere, J.-M. Ahuactzin, K. Mekhnacha, and E. Mazer, *Bayesian Programming*. Taylor & Francis, 2012.

[15] F. Colas, J. Diard, and P. Bessière, "Common bayesian models for common cognitive issues," *Acta Biotheoretica*, vol. 58, pp. 191–216, 2010.

[16] J. F. Ferreira, J. Lobo, P. Bessière, M. Castelo-Branco, and J. Dias, "A bayesian framework for active artificial perception," *IEEE Transactions on Cybernetics (Systems Man and Cybernetics, part B)*, vol. 43, pp. 699–711, 2013.

[17] S. Calinon, F. Guenter, and A. Billard, "On learning, representing and generalizing a task in a humanoid robot," *IEEE Trans. on Systems, Man and Cybernetics, Part B, Special issue on robot learning by observation, demonstration and imitation*, vol. 37, pp. 286–298, 2007.