2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

Autonomous Trajectory Design System for Mapping of Unknown Sea-floors using a team of AUVs

Georgios Salavasidis, Athanasios Ch. Kapoutsis, Savvas A. Chatzichristofis,
Panagiotis Michailidis and Elias B. Kosmatopoulos

Abstract— **This research develops a new on-line trajectory planning algorithm for a team of Autonomous Underwater Vehicles (AUVs). The goal of the AUVs is to cooperatively explore and map the ocean seafloor. As the morphology of the seabed is unknown and complex, standard non-convex algorithms perform insufficiently. To tackle this, a new simulation-based approach is proposed and numerically evaluated. This approach adapts the Parametrized Cognitive-based Adaptive Optimization (PCAO) algorithm. The algorithm transforms the exploration problem to a parametrized decision-making mechanism whose real-time implementation is feasible. Upon that transformation, this scheme calculates off-line a set of decision making mechanism's parameters that approximate the - non-practically feasible - optimal solution. The advantages of the algorithm are significant computational simplicity, scalability, and the fact that it can straightforwardly embed any type of physical constraints and system limitations. In order to train the PCAO controller, two morphologically different seafloors are used. During this training, the algorithm outperforms an unrealistic optimal-one-step-ahead search algorithm. To demonstrate the universality of the controller, the most effective controller is used to map three new morphologically different seafloors. During the latter mapping experiment, the PCAO algorithm outperforms several gradient-descent-like approaches.**

I. Introduction

Autonomous Underwater Vehicles (AUVs) have been successfully applied in large range of applications such as: deep sea mining, locating and mapping new hydrothermal vents [1], harbor security [2], mine countermeasures [3], underwater archaeology [4], etc. A key AUV requirement in all these applications is the ability to explore and map the ocean bottom using ranging sensors.

In general terms, two are the fundamental challenges that the underwater robotic research community faces while aims to autonomously construct seafloor maps. First challenge is the widely-known problem termed *Simultaneous Localization and Mapping (SLAM)*. SLAM is a technique aiming to efficiently process on-board sensor readings in order to simultaneously localise robots and create maps of the external environment. Although approximate algorithms there exist for concurrent localization and mapping, SLAM still remains quite challenging problem when it comes to the underwater environment. Aside from the challenge related to the data fusion performed by SLAM algorithms, it is also of high importance to navigate the team of AUVs over highly informative paths. Information rich trajectories affect the SLAM performance in terms of both localization and mapping accuracy. The second challenge to be addressed, therefore, is related to the AUV trajectory generation that would increase the *SLAM efficiency*[1].

For several years, great effort has been devoted to the study of single AUV SLAM and Cooperative SLAM (C-SLAM), where a team of AUVs is considered. These methodologies effectively deal with a diverge set of AUVs systems (different sensor suit, etc.), while the their performance has been demonstrated in field experiments (e.g. [5], [6], [7]). Despite these advances, however, almost all underwater map-building methods are characterized by low autonomy, since they typically rely on a set of pre-defined trajectories and often on human intervention. AUVs usually follow trajectories designed off-line, before the actual deployment, which is a limiting factor when a totally unknown area is to be mapped. An off-line trajectory design system may have high impact on the map quality, since pre-defined trajectories are likely to "miss" areas rich in information or, reversely, AUVs may waste significant time focusing on low informative regions. In practice, a usual approach to tackle these drawbacks is to perform the following repetitive procedure. Initially, AUVs map the seafloor by following blindly pre-defined trajectories (usually in a lawnmover pattern). Once this first step is accomplished, new trajectories are generated, always off-line, but now using the existing bottom knowledge from the already constructed maps. Next, the AUVs are again deployed in order to acquire further seafloor knowledge while following the re-designed trajectories. This repetitive process continues until a required map accuracy is obtained. Some mapping missions, however, are time-critical, which means that an immediate map is required. Time constraints may furthermore be imposed because of dynamically changing environments. For instance, detection of sunken drums, leaking chemicals or search-and-rescue missions require optimized with respect to time procedures, which would allow a quick response. Another important aspect is the fact that the off-line trajectory generation

Georgios Salavasidis is with the National Oceanography Centre, Southampton SO14 3ZH, UK, geosal@noc.ac.uk

Savvas A. Chatzichristofis is with the Neapolis University Pafos, School of Informatics, 8042, Cyprus, schatzic@nup.ac.cy

Panagiotis Michailidis is with the Informatics & Telematics Institute, Center for Research and Technology–Hellas (ITI-CERTH), 57001, Thessaloniki, Greece, panosmih@iti.gr

Athanasios Ch. Kapoutsis and Elias B. Kosmatopoulos are with the Dept. of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi 67100, Greece and ITI-CERTH {akapouts,kosmatop}@ee.duth.gr

[1]The problem of multi-robot *trajectory generation for maximizing the SLAM efficiency* is also referred in the literature as *exploration* or *optimal motion strategy*. In the rest of this paper, these terms will be used interchangeably.

system cannot exploit the advantage of being member of a team. Although cooperative mapping (e.g. by sharing sensor measurements and mapping progress status) can be faster and more effective, pre-defined trajectories force the AUVs to act almost as separate identities.

This paper proposes a new approach that overcomes the previously described shortcomings. The aim of this research is to generate trajectories on-line for a team of AUVs in order to construct fast and accurate seafloor maps (i.e. maximizing *SLAM efficiency*). The proposed approach is based on a recently introduced *approximate optimal control* methodology – abbreviated as Parametrized Cognitive Adaptive Optimization (PCAO) [8], [9] – specifically tailored to the problem of multi-AUV exploration and mapping. Instead of relaxing the original NP-hard problem, *PCAO does the best possible to approximate the optimal solutions by a computationally tractable decision-making mechanism*. In simple words, the PCAO approach solves the following problem: given a parametrized decision-making mechanism whose real-time implementation is practically feasible (for a fixed set of parameters), find the set of the decision making mechanism's parameters that optimally approximate the – non-practically feasible – optimal solution.

The proposed algorithm is first trained over a set of seabed surfaces with variant morphologies. During the learning part, the PCAO algorithm outperforms an unrealistic optimal-one-step-ahead search algorithm (used as a benchmarking tool) which takes decisions after evaluating a number of position candidates. Next, we assess the universality of the obtained controller. We apply the best learnt controller to a set of new morphologically different seafloors. The performance of the algorithm is compared with several gradient-descent-like approaches. The results clearly demonstrate both the effectiveness and simplicity (in terms of implementation and energy requirements) of the PCAO algorithm.

II. Autonomous Multi-AUV Exploration for Mapping of Unknown Sea-floors

This study considers a team of N_R AUVs having as objective to collaboratively construct precise seafloor maps. We turn the mapping task into a problem of estimating the position of N_L landmarks (static features) in the 3D environment. In fact, the actual landmark position corresponds to a specific point of the sea bottom, since landmarks are placed on the top of the seabed. Estimating landmarks implies direct estimates of seafloor points. Two are the benefits of formulating the mapping problem in such way: a) placing a large number of landmarks, while considering potential limitations on available memory and computational power, forces the AUVs to construct a high resolution bathymetric map, b) placing landmarks only at areas of high interest, AUVs are required to focus on these particular regions rather that constructing a full, possible meaningless, regional bathymetric map.

The AUV sensor suite consists of proprioceptive sensors (e.g. IMU, GPS, etc.), primarily for self-localization via dead reckoning, and exteroceptive sensors, such as a multi-beam sonar that enables the mapping task. To simplify our analysis, we assume that the AUVs are perfectly localized[2].

Let x_i^L denote the 3D position of the $i-$th landmark, x_i^R the position of the $i-$th AUV, and

$$X^L = [x_1^L, \ldots, x_{N_L}^L], \quad X^R = [x_1^R, \ldots, x_{N_R}^R]$$

the matrices of landmarks to be estimated and positions of AUVs.

A. AUVs control

At each time-step, the vector of AUV positions X^R is updated according to the following equation:

$$X^R(t_{k+1}) = X^R(t_k) + u(t_{k+1})\, dt \tag{1}$$

where $u(t_{k+1})$ corresponds to the control action at t_{k+1} and dt is the sampling time, which is assumed to be sufficiently small. Although AUVs are typically torpedo-shaped under-actuated vehicles, the vehicle dynamics are simplified. This simplification allows to assume that the displacement vector $u \cdot dt$ is constrained in each direction (independently to the overall 3D displacement) by a maximum allowed displacement in the corresponding direction within the dt time interval. Therefore, the control/velocity vector must equivalently obey to: $|u(i)| \leq u_{max}(i)$, where $u_{max}(i)$ is the maximum AUV velocity in $i-th$ direction.

B. Measurement model

Thanks to their small aperture, pencil-beam echo sounders typically feature small footprints. There is therefore an interest in using them for achieving high-resolution seabed mapping with low spatial uncertainty. As a consequence, this study considers all AUVs to be equipped with narrow multi-beam sonars. The area ensonified by the beam is assumed isotropic, with magnitude as calculated from the first returned echo. We assume that the landmark size is comparable to the size of the ensonified area and the range to this landmark (if located within this patch) is given by the first return. Although this assumption is not strictly valid when observing large areas (larger distance to seabed), the sensor noise model, detailed in a subsequent paragraph, provides an estimate of the confidence on this measurement.

Every time-step, R sonar range-to-bottom measurements (perpendicular direction to the AUV longitudinal axis) are obtained. These ranges form an area called swath. Only landmarks located within the multi-beam swath width can be detected, measured and re-estimated[3]. Let Y denote the measurement vector containing ranges from beams that ensonify at least one landmark. In the general case, sonar measurements are related to X^L and X^R through a nonlinear function that admits the form:

$$Y = H(X^L, X^R, \Xi)$$

[2]This research addresses only with the map construction problem. It has to be emphasized that the proposed approach can be easily extended to further include the localization problem.

[3]Please note that the detection and data association problem is out of the scope of this research.

where H is the nonlinear sensor function and Ξ is the measurement noise vector. Following, we provide a discussion on challenges that an underwater trajectory design system needs to address.

(NL-Noise) The typical assumption made in most underwater applications is that the sensor noise is Additive White Gaussian. This simplified assumption is very restrictive and usually not realistic. Range-based measurements, such the multi-beam echo sounder acquires, are typically affected by an error proportional to the range-to-bottom. As a result, it is more realistic to assume a multiplicative sensor noise of the following general form:

$$y = h(x^R, q) + h_\xi(x^R, q)\xi \tag{2}$$

where y is a single range measurement, x^R is the AUV position, and q is the position of a landmark within the beam's patch. Given a known beam geometry, $h(x^R, q)$ models the beam and provides a noise-free range-to-bottom. To introduce a beam noise, $h_\xi(x,q)$ is a nonlinear function of x^R and q [e.g. $h_\xi(x,q)$ is the slant range between x^R and q], and ξ is a standard Gaussian noise.

(MaxRange) In addition to the (NL-Noise) limitation, sonar beam models are characterised by their maximum range. Augmenting sonar noise model with the maximum range limitation, sonar model takes the following form:

$$y_{x^R-q} = \begin{cases} \text{undefined} & \text{if } \|x^R - q\| \geq thres \\ h(x^R, q) + h_\xi(x^R, q)\xi & \text{otherwise} \end{cases} \tag{3}$$

where y_{x^R-q} denotes a measurement from an AUV at x^R and a landmark at q. *thres* corresponds to the maximum beam range-to-bottom beyond which the measurements are not considered.

(ObsAvoid) As in real robot application, the navigation system must make sure that the AUVs avoid obstacles. This study assumes that the AUV navigation system is enhanced with collision avoidance techniques.

C. Landmark estimation

This subsection aims to provide a mathematical formulation to the landmark-based mapping problem using a team of AUVs. Let $\hat{X}^L$ denotes the estimate of X^L as generated by an Extended Kalman Filter (EKF) based SLAM algorithm. Since different AUV trajectories produce maps of a different accuracy, the *active exploration* problem is addressed by designing trajectories that the landmark estimation accuracy is maximised.

To optimize the estimation accuracy, we define the Ω matrix that contains the mapping progress as follows[4]:

$$\Omega = \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_{N_L} \end{bmatrix} = \begin{bmatrix} \|x_1^L - \hat{x}_1^L\| \\ \vdots \\ \|x_{N_L}^L - \hat{x}_{N_L}^L\| \end{bmatrix} \tag{4}$$

[4]Please note that $\|x_i^L - \hat{x}_i^L\|$ cannot be calculated in real-life as the calculation requires knowledge of the true landmark positions. In practice, however, information about the term $\|x_i^L - \hat{x}_i^L\|$ can be obtained from, for example, the associated terms of the EKF error covariance matrix.

This matrix is updated using both AUV position $X^R(t)$ and corresponding sonar measurements (3). We consider that the landmark $\hat{x}_i^L$ is accurately-estimated, if the ω_i is below a certain threshold (or, equivalently, the corresponding error covariance matrix is sufficiently small). Following this, we denote $\mathbf{A}$ to be a set of all accurately estimated landmarks. Please note that once a certain landmark is sufficiently estimated, it will remain estimated forever (put differently, it will hereafter belong to the $\mathbf{A}$ set). Therefore, the number of non accurately-estimated landmarks, at each time-step t_k, is given by:

$$\varepsilon(t_k) = N_L - |\mathbf{A}_k| \tag{5}$$

where $|\mathbf{A}_k|$ denotes the *cardinality* of the set $\mathbf{A}_k$, i.e. the number of accurately estimated landmarks until the t_k time-step.

Additionally, we define the objective function for the map construction problem:

$$J = \int_0^N \Pi(x(s), u(s))\, ds \approx \sum_{k=1}^{N-1} \Omega^T(t_k)\Omega(t_k)dt + \kappa\varepsilon(t_N) \tag{6}$$

where Π is an appropriate nonlinear function that corresponds to the instantaneous cost, N is the mapping duration (in time-steps), and κ factor serves as a weight to balance the terms in the objective function. This formulation – apart from the general objective to minimize the number of non-accurately estimated landmarks – aims to reward controllers that rapidly minimize the estimation error.

By using all the preliminaries and definitions described previously, the optimal AUV navigation/exploration problem to map the seafloor surface, can be cast as a dynamic optimization problem as follows:

$$\begin{aligned} &\min_{u(t_1),u(t_2),\ldots,u(t_N)} J \\ s.t. \quad &C(X^R(t_k)) \leq 0, k = 1,\ldots,N \end{aligned} \tag{7}$$

The nonlinear function $C(\cdot)$ is used to constrain the generated waypoints $X^R(t_k)$ within the operation area, and to incorporate obstacle avoidance and maximum speed constraints. Standard algebraic manipulations can be employed to cast all these constraints in the $C(X^R(t_k)) \leq 0$ form. In general the above formulation can embed any type of physical constraints and system limitations that might be imposed from the available AUV-infrastructure or/and the type of the mission. However, given a set of transformations and employing a pre-integrator as in [10], the constrained optimal control problem can be transformed to an unconstrained one.

III. Optimal P-based controller

This section introduces the P-parametrized approximately optimal controller for the dynamic optimization problem of (7). Instead of attempting to explicitly solve, the practically infeasible optimization problem presented in section (II-C), this research aims to approximate the optimal controller.

Without loss of generality, it can be assumed that the system dynamics have the following form:

$$\begin{aligned} \dot{X}^R &= u(t) \\ \dot{\Omega} &= F\left(\Omega, X^R(t), Y(t)\right) \end{aligned} \tag{8}$$

where F is a nonlinear morphology-dependent function, that describes the evolution of the mapping progress. Let us define the augmented system vector x as following

$$\dot{x} = \begin{bmatrix} \dot{X}^R \\ \dot{\Omega} \end{bmatrix} = G(x) + Bu \tag{9}$$

$G(x) = \begin{bmatrix} O \\ F\left(\Omega, X^R(t), Y(t)\right) \end{bmatrix}$ and $B = \begin{bmatrix} I \\ O \end{bmatrix}$, where O denotes an appropriately sized all-zero vector and I denotes the identity matrix.

The optimal trajectory generation problem (described in eq. 7) can be cast as an unconstrained optimal control problem [10] of the form:

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & \int_0^N \Pi\left(x(s), u(s)\right) ds \\ \text{subject to} \quad & \dot{x} = G(x(t)) + Bu(t) \end{aligned} \tag{10}$$

A. Optimal Control Approximation

According to Hamilton-Jacobi-Bellman equation [11], the controller that minimizes the above criterion can be obtained solving the partial differential equation:

$$u^* = \underset{u}{\operatorname{argmin}} \left\{ \left(\frac{\partial V^*}{\partial x} \right)^\tau (G(x) + Bu) + \Pi(x) \right\} \tag{11}$$

where V^* denotes the optimal *cost-to-go* function and u^* the corresponding optimal control actions, which can be transformed to:

$$u^* = -B^\tau \left(\frac{\partial V^*}{\partial x} \right) \tag{12}$$

As shown in [8], the optimal *cost-to-go* function can be approximated, with arbitrary accuracy, using the following positive definite PieceWise Quadratic (PWQ) approximation:

$$V^*(x) \approx V(x) = \sum_{i=1}^{L} \beta_i(x) \left(x^\tau P x\right) = z^\tau(x) P z(x) \tag{13}$$

where $\beta_i, i = 1, 2, \ldots, L$ are a set of smooth mixing signals [10]. The vector z and the matrix P have the following analytical form:

$$z(x) = \begin{bmatrix} \sqrt{\beta_1(x)}x \\ \sqrt{\beta_2(x)}x \\ \vdots \\ \sqrt{\beta_L(x)}x \end{bmatrix}, \; P = \begin{bmatrix} P_1 & 0 & \ldots & 0 \\ 0 & P_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & P_L \end{bmatrix}$$

The deviation from the optimal cost-to-go function is in the order of the number of linear controllers $\mathcal{O}(1/L)$. Applying the above approximation to (12), the optimal controller can now be represented as

$$u^* = -B^\tau \frac{\partial V}{\partial x} + \mathcal{O}(1/L) \tag{14}$$

which, utilizing optimal cost-to-go function approximation, can be written equivalently as

$$u^* = -B^\tau M_z(x) P z(x) + \mathcal{O}(1/L) \tag{15}$$

where $M_z(x)$ is the Jacobian matrix of $z(x)$ with respect to x.

Up to this point, we have approximated the optimal controller u^*, with a P-parametrized controller. In other words, the AUV trajectories can be designed by appropriately translating (through the matrix P) the information vector $z(x)$, as follows

$$u = -B^\tau M_z(x) P z(x) \tag{16}$$

B. Evaluation metric

The last component of the approximately optimal P-based controller is the evaluation function. The assessment of a single experiment is achieved by utilizing the objective function in (6). However, the noise presence in measurements (2) may result in different mapping outcome, although an identical controller is used. However, it is desirable to construct a control matrix P that is "immune" to this factor. Thus, the same control matrix P is evaluated through N_e Monte Carlo simulation runs over the same surface and the summation of the achieved scores is calculated as $\sum_{e=1}^{N_e} J_e$.

Finally, in order to avoid the over-fitting on a single surface, the above described criterion is applied to a number N_s of morphologically different seafloors, constructing the overall mapping score as:

$$\mathscr{E}(P) = \sum_{s=1}^{N_s} \frac{\sum_{e=1}^{N_e} J_e}{N} \tag{17}$$

where N denotes the mapping duration in time-steps.

IV. Specifically-tailored PCAO Approach

The approximation of the afore-formulated optimal P-based controller could be achieved by employing off-the-shelf *gradient descent methodologies* [12],[13], i.e.

$$P = P - \eta \nabla_P \mathscr{E}(P), \; \eta > 0 \tag{18}$$

However, there are two major disadvantages that do not allow the utilization of such techniques, without compromising any of the real-word aspects defined in section II.

- An analytic expression for the gradient $\nabla_P \mathscr{E}(P)$ in (18) is needed. Such an analytic expression is infeasible to obtain as it involves calculations that require an analytic expression for the unknown surface-dependent data acquisition model and the robots motion/sensor model. Even in cases where a standard measurements acquisition model is considered, the calculation of the gradient $\nabla_P \mathscr{E}(P)$ remains a practically infeasible problem, due to the high complexity.
- The second issue arises from the fact that the term $\mathcal{O}(1/L)$ in (13) acts as a disturbance. Due to the presence of this term, the performance of a gradient descent algorithm is bounded under the $\mathscr{E}(P^*) + \mathcal{O}(1/L)$ (where P^* denotes the optimal P-based controller). Therefore, a controller formed given a gradient descent algorithm may significantly deviate form the optimal one, especially in cases where the effect of $\mathcal{O}(1/L)$ is

not negligible. Apart from the acquisition of the optimal solution, such a disturbance term may destroy the convergence properties of the gradient descent algorithm and lead it to divergence, even in cases where this term is relative small.

In order to tackle the above shortcomings, the *Parametrized Cognitive Adaptive Optimization (PCAO)* [8] - specifically tailored to the problem - is proposed. The main steps of the modified PCAO algorithm are outlined in Algorithm 1. Following, the main features of PCAO algorithm are discussed.

• The analytic formulas of the evaluation criterion and its gradient is not required. PCAO algorithm, simultaneously with the searching for the optimal P-controller, learns the different characteristic that govern the relation between the P-controller's elements and the corresponding evaluation cost $\mathscr{E}(P)$. It has been proven that employing an appropriately defined estimator, similar to the one of the Step 2, results in a cognitive adaptive optimization scheme, that approximates the gradient descent performance without carrying out any differentiation on the unknown $\mathscr{E}$ function [18] [17].

• The utilization of random perturbations provides the proposed algorithm with the potential to escape from local minima. In essence, the random perturbations inside the PCAO primary selection mechanism, could have a behavior similar to *simulated annealing*, which has been proved that under specific conditions can overcome local minima [19].

V. Simulation experiments

This section presents simulation results using the PCAO algorithm. A MATLAB-based simulator has been developed where the controller is, first, trained and, later, evaluated over various terrains. This simulation environment incorporates all environmental constraints and sensor limitations discussed in the section II-B. For the learning part of the algorithm, the simulator runs the PCAO algorithm over $N_s = 2$ morphologically different terrains, see figure 1. The first area is an actual Digital Elevation Model (DEM), hereafter referred to as Real Map 1, whereas the second is an artificially generated map aiming to enclose various terrain structures (hereafter referred to as Artificial Map 1). The choice of these maps is primarily driven by the necessity to train the controller both on realistic data but also over a seabed with variant morphology.

Table I lists the considered parameters for setting up the simulator. These parameters remain constant throughout the training and evaluation part of the algorithm.

One step for evaluating our approach is to compare its performance with the Random Search (RS) algorithm during the training period. In contrast to the PCAO algorithm which forms the controller in (16), according to the procedure described in the Algorithm 1, the RS algorithm selects randomly a symmetric and positive definite P matrix (obeying

[5]According to [17] it suffices to choose M to be any positive integer larger or equal to 2 x [the number of variables being optimized by PCAO]. In our case the variables optimized are the P elements and thus it suffices for M to satisfy $M \geq 2 \times (3n_r(2L+1)^2)^2$

Algorithm 1 PCAO algorithm

Initialization

1) Choose positive integer M
2) Initialize $P(0)$ to be a matrix satisfying the constraints $e_1 I \preceq P(0) \preceq e_2 I$
3) Choose a positive scalar function $a(k)$ satisfying

$$a(k) > 0,\ \lim_{k\to\infty} a(k) = 0,\ \sum_{k=0}^{\infty} a(k) = \infty,\ \sum_{k=0}^{\infty} a(k)^2 < \infty$$

4) Set $k = 0$

[The reader is referred to [14] and [15] for guidelines on the choice of $a(k)$]

Step 1. Calculate the robot trajectories for all the time-steps T of the simulation period, based on the $P = P(k)$ controller (16) and derive the corresponding evaluation $\mathscr{E}(P(k))$ (17)

Step 2. Construct the Linear-In-the-Parameters estimator of $\mathscr{E}(P)$, based on stored tuples $\langle P(j), \mathscr{E}(P(j))\rangle$ of historical evaluations as

$$\hat{\mathscr{E}}(P) = \theta_{best}^{\tau}\phi(P)$$

$$\theta_{best} = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{k} \left(\mathscr{E}(P(i)) - \theta^{\tau}\phi(P(i))\right)^2$$

where θ denotes the vector of parameter estimates and standard least-squares optimization algorithms can be used to calculate θ_{best}. ϕ denotes the non-linear vector of L_ϕ regressor terms and L_ϕ is a positive user-defined integer. The vector ϕ of regressor terms must be chosen so that it is a universal approximator [20], such as polynomial approximators, radial basis functions, kernel-based approximators, etc.

Step 3. Calculate $P_{best}(t)$ to be the best P *obtained so far*, i.e.,

$$P_{best}(k) = \underset{P(s),s=0,\ldots,k}{\operatorname{argmin}} \{\mathscr{E}(P(s))\}$$

Step 4. Generate M perpetuated candidates (*random perturbations*)[5] of $P_{best}(k)$:

$$P_{cand}^{(i)} = (1 - a(k))\, P_{best}(k) + a(k)\Delta P^{(i)},\ \ i = 1, 2, \ldots, M$$

where $\Delta P^{(i)}$ are random symmetric positive definite matrices with the same structure as in [16], and satisfying $e_1 I \preceq \Delta P^{(i)} \preceq e_2 I$

Step 5. The P-controller for the next evaluation is selected from the M available candidates, as

$$P(k+1) = \underset{P_{cand}^{(i)}}{\operatorname{argmin}} \left(\theta_{best}^{\tau}\phi\left(P_{cand}^{(i)}\right)\right)$$

where $\theta_{best}^{\tau}\phi(\cdot)$ denotes the best available estimation of the evaluation function (as calculated at **Step 2.**) until the kth timestamp

Step 6. Set $k = k+1$ and *GO TO* **Step 1.**

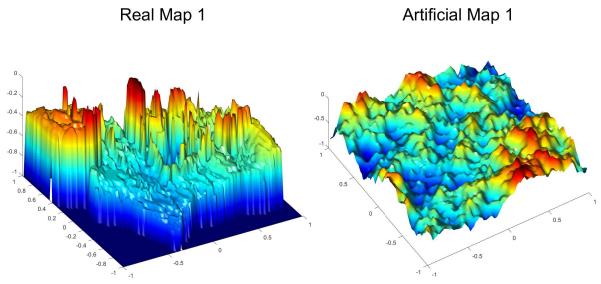

Fig. 1. Two morphologically different seafloors used for the PCAO controller training.

TABLE I
SIMULATION PARAMETERS

$t_p^{max} = 500$	Number of iterations, P matrix updates
$N_e = 100$	Number of Monte Carlo runs per each map
$N_s = 2$	Number of seafloors
$N = 500$	Duration of seafloor mapping [time-steps]
$N_R = 3$	Number of AUVs
$N_L = 1100$	Number of landmarks
$[-1,1]^3$	Area to be mapped - [m^3]
$u_{max}(i) = 0.5$	Maximum speed in i-th direction - [m/s]
$dt = 0.1$	Sampling time - [sec]
$\kappa = 100$	Weight factor
$R = 20$	Number of sonar beams
$thres = 0.2$	Maximum beam range - [m]
$L = 1$	Set of smooth mixing signals (13)
$L_\phi = 100$	Size of the vector ϕ of regressor terms

the PCAO restrictions). As a matter of fairness, algorithms start with an identical landmark spread and control initialization. Figure 2 demonstrates, in comparative way, the learning progress of both algorithms. The horizontal axis of the figure shows the learning time (P matrix updates), while the vertical axis provides numerical measure of the seafloor mapping performance, according to (17), for each control update iteration. As this figure shows, the RS is incapable of learning the map underlying information in order to improve the controller. While the RS performance remains, in average, constant as the training proceeds, the PCAO algorithm constantly increases the effectiveness of the controller. After t_p^{map} iterations, which is the allowed learning period, the most successful controllers are obtained at points B and C, for RS and PCAO respectively. Table II translates

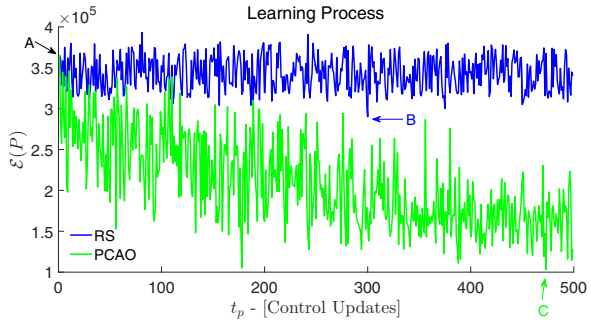

Fig. 2. Learning Process: PCAO algorithm (green line) compared to Random Search algorithm (blue line).

TABLE II
ε%, NON-ACCURATELY ESTIMATED LANDMARKS

Instances	Real Map 1	Artificial Map 1
A - Initialization	99%	78%
B - Best RS	71%	68%
C - Best PCAO	13%	29%

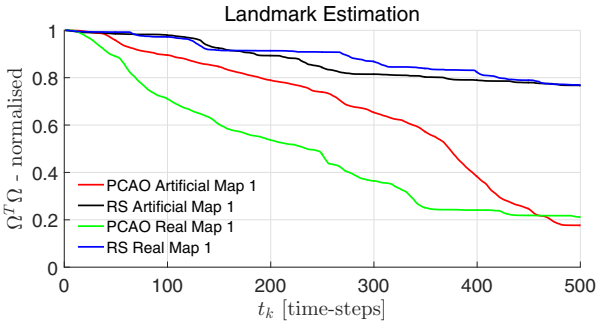

Fig. 3. Cost function evolution over time for the most effective controllers constructed using the PCAO and the RS algorithm. The evaluation is performed over the two seafloors considered for the training process.

the numerical values of the cost function of the three marked instances into the ε% performance criterion (5).

Following, figure 3 demonstrates the cost function evolution over N time-steps of mapping procedure using the most effective controllers. In this figure, vertical axis refers to the instantaneous cost whereas the horizontal axis is the mapping time. The PCAO performance is shown with green and red, while blue and black are for the RS (for Real Map 1 and Artificial Map 1 respectively). Again, the figure shows that the PCAO outperforms the RS as it is able to rapidly and significantly reduce the estimation error.

Table III compares the performance of the best PCAO controller against the Semi-Exhaustive (SE) algorithm [20, Algorithm 1] with respect to ε% criterion (average results of 100 MC simulation runs). SE search is an approach which approximates the optimal-one-step-ahead exhaustive search algorithm. This algorithm attempts to provide greedy optimal solution at each time-step by evaluating a number of valid candidate positions. In essence, SE performs a series of actual motions per decision iteration. All these motions are then passed through the evaluation function (5) and the one which achieves the best score is selected as final action to be performed. By definition, therefore, SE is an impractical algorithm and cannot be considered as an online-trajectory-generation alternative. Application time constraints and AUV power would prevent the algorithm from use. Dealing with simulations, though, this study uses SE as a powerful benchmarking tool for analysis/validation purposes. The mapping time required by the SE is parametrized by the number of candidates, Cand., under evaluation. Considering N discrete iterations, the SE operational time increases following:

$$T_{SE} = N(1 + \text{Cand.})$$

The PCAO algorithm outperforms the SE algorithm with

TABLE III

PCAO VS SEMI-EXHAUSTIVE

Control Algorithm	Real Map 1	Artificial Map 1	Mapping Time
PCAO	13.63%	29.54%	500
SE (Cand.=0)	86.85%	91.58%	500
SE (Cand.=5)	23.2%	40.67%	3000
SE (Cand.=50)	3.89%	13.4%	25500
SE (Cand.=500)	1.81%	11.16%	250500

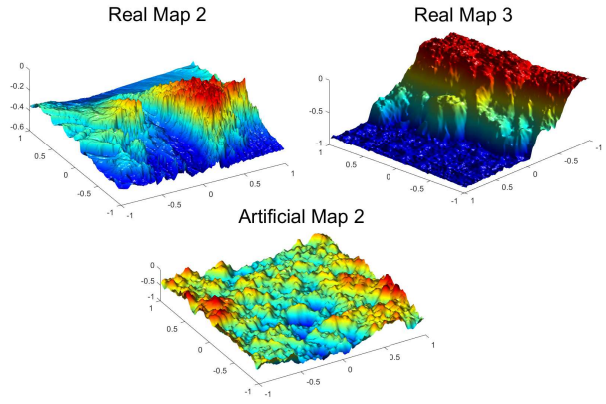

Fig. 4. Three additional seafloors for evaluating the universality of the controller.

up to five candidates. Alternatively, the SE algorithm requires more than 3000 time-steps to construct a more accurate map.

As the training process is completed and analyzed, it is time to assess the controller universality. By universality it is meant the ability of the controller to perform the mapping task over areas where it has never been trained. To do so, a new set of three seafloors is introduced, see figure 4, where the controller is evaluated against the SE and two widely known stochastic approximation algorithms [21]: a) Simultaneous Perturbation Stochastic Approximation (SPSA), b) Finite Differences Stochastic Approximation (FDSA). It must be emphasized that these algorithms are quite standard in serving as base cases for analyzing the performance of stochastic approximation algorithms. Similar to the PCAO algorithm, SPSA and FDSA are gradient free algorithms and, moreover, they do not require an analytical form of the objective function. Mathematical analysis has established that their performance is approximately the same as that of the standard gradient descent algorithm. However, it is worth noticing that these algorithms, similarly to SE, are not really practical as they require AUVs to perform a set of auxiliary motions before the actual decision takes place.

Tables IV to VI show results (both best and average after 100 MC runs) over the new set of seafloors. Figures 5 to 7 demonstrate the time evolution of the instantaneous cost during the mapping procedure for all, under comparison, algorithms.

In all cases, PCAO average results outperform the respective results from FDSA, SPSA, and SE with $Cand. = 0$ showing, further, the algorithm's consistency and robustness to sensor noise and initialization values. SE with $Cand. = 5$ shows to outperform the PCAO effectiveness, but this comes at six times increased mapping time.

TABLE IV

REAL MAP 2: PCAO VS SEMI-EXHAUSTIVE, FDSA AND SPSA

Control Algorithm	Average Result $\varepsilon\%$	Best Performance $\varepsilon\%$	Mapping Time
PCAO	19.84%	13.27%	500
SE (Cand.=0)	77.3%	67.9%	500
SE (Cand.=5)	13.23%	5%	3000
SPSA	33.96%	22.72%	1500
FDSA	39.54%	2.36%	9500

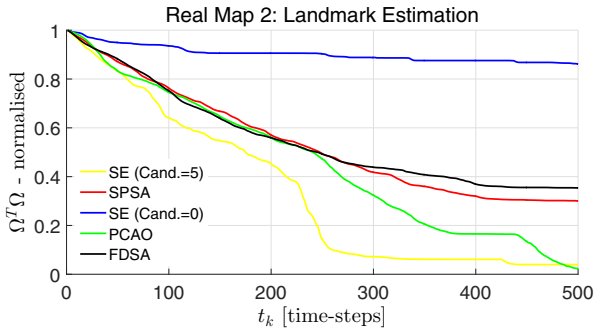

Fig. 5. Performance evaluation of the PCAO algorithm against the SPSA, FDSA and SE algorithms over the Real Map 2. Each line is associated to a control algorithm and represents the corresponding instantaneous cost function evolution over time.

TABLE V

REAL MAP 3: PCAO VS SEMI-EXHAUSTIVE, FDSA AND SPSA

Control Algorithm	Average Result $\varepsilon\%$	Best Performance $\varepsilon\%$	Mapping Time
PCAO	34.6%	20.18%	500
SE (Cand.=0)	77.4%	70.18%	500
SE (Cand.=5)	15.07%	10.54%	3000
SPSA	48.59%	28.36%	1500
FDSA	62.75%	41.54%	9500

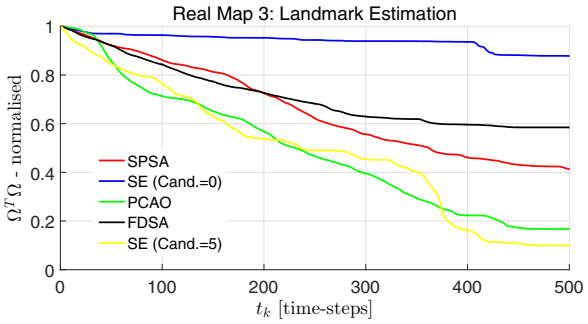

Fig. 6. Performance evaluation of the PCAO algorithm against the SPSA, FDSA and SE algorithms over the Real Map 3. Each line is associated to a control algorithm and represents the corresponding instantaneous cost function evolution over time.

VI. CONCLUSIONS

Current multi-AUV systems are far from being capable of fully autonomously taking over real-life complex situation-awareness operations. As such operations require advanced reasoning and decision-making abilities, current

TABLE VI

ARTIFICIAL MAP 2: PCAO VS SEMI-EXHAUSTIVE, FDSA AND SPSA

Control Algorithm	Average Result $\varepsilon\%$	Best Performance $\varepsilon\%$	Mapping Time
PCAO	27.25%	23.18%	500
SE (Cand.=0)	80.16%	72%	500
SE (Cand.=5)	20.4%	10.54%	3000
SPSA	52.59%	40.9%	1500
FDSA	54.14%	11.72%	9500

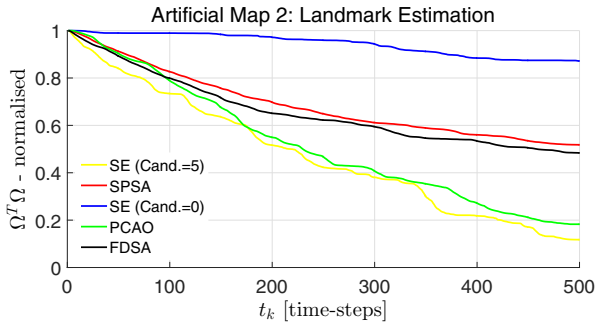

Fig. 7. Performance evaluation of the PCAO algorithm against the SPSA, FDSA and SE algorithms over the Artificial Map 2. Each line is associated to a control algorithm and represents the corresponding instantaneous cost function evolution over time.

methods have to heavily rely on human intervention. This research presented a new approach, called PCAO, that is able to efficiently and fully-autonomously navigate a team of AUVs. The AUVs are deployed in order to explore and map completely unknown underwater environments. Simulation experiments demonstrated the efficiency of the PCAO again other stochastic based approaches. Although these algorithms might be sometimes more accurate, they are naturally impractical. They require an considerable increase in computational power and mapping time duration, factors that forbid they applicability in real-life missions. On the other hand, PCAO algorithm showed its superiority in terms of computational simplicity (simple matrix multiplication) and implementation straightforwardness.

ACKNOWLEDGMENT

This paper has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 programme under grant agreement no 740593 (ROBORDER) and the project EQUILY from Swedish Energy Agency.

REFERENCES

[1] S. E. Webster, R. M. Eustice, H. Singh, and L. L. Whitcomb, "Advances in single-beacon one-way-travel-time acoustic navigation for underwater vehicles," *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 935–950, 2012.

[2] A. Rodningsby and Y. Bar-Shalom, "Tracking of divers using probabilistic data association filter with a bubble model," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 3, pp. 1181–1193, 2009.

[3] V. Yordanova and H. Griffiths, "Synchronous rendezvous technique for multi-vehicle mine countermeasure operations," in *OCEANS'15 MTS/IEEE Washington*. IEEE, 2015, pp. 1–6.

[4] B. Bingham, B. Foley, H. Singh, R. Camilli, K. Delaporta, R. Eustice, A. Mallios, D. Mindell, C. Roman, and D. Sakellariou, "Robotic tools for deep water archaeology: Surveying an ancient shipwreck with an autonomous underwater vehicle," *Journal of Field Robotics*, vol. 27, no. 6, pp. 702–717, 2010.

[5] L. Paull, G. Huang, M. Seto, and J. J. Leonard, "Communication-constrained multi-AUV cooperative SLAM," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 509–516.

[6] P. L. N. Carrasco, F. Bonin-Font, M. M. Campos, and G. O. Codina, "Stereo-Vision Graph-SLAM] for robust navigation of the AUV SPARUS II," *IFAC-PapersOnLine*, vol. 48, no. 2, pp. 200–205, 2015.

[7] M. Solbach, F. Bonin-Font, A. Burguera, G. Oliver, and D. Paulus, "Robust world-centric stereo EKF localization with active loop closing for AUV s," *Pattern Recognition and Image Analysis*, vol. 26, no. 1, pp. 205–215, 2016.

[8] S. Baldi, I. Michailidis, E. B. Kosmatopoulos, P. Ioannou *et al.*, "A" plug and play" computationally efficient approach for control design of large-scale nonlinear systems using cosimulation: a combination of two" ingredients"," *Control Systems, IEEE*, vol. 34, no. 5, pp. 56–71, 2014.

[9] A. C. Kapoutsis, G. Salavasidis, S. Chatzichristofis, J. Braga, J. Pinto, J. Sousa, and E. B. Kosmatopoulos, "The noptilus project overview: A fully-autonomous navigation system of teams of auvs for static/dynamic underwater map construction," *IFAC-PapersOnLine*, vol. 48, no. 2, p. 231237, 2015.

[10] E. Kosmatopoulos, S. Baldi, K. Aboudolas, D. Rovas, A. Papachristodoulou, and P. Ioannou, "Nonlinear control of large scale complex systems using convex control design tools," in *Control & Automation (MED), 2011 19th Mediterranean Conference on*. IEEE, 2011, pp. 1022–1027.

[11] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena Scientific Belmont, MA, 1995, vol. 1, no. 2.

[12] C. Audet and J. E. Dennis Jr, "Analysis of generalized pattern searches," *SIAM Journal on Optimization*, vol. 13, no. 3, pp. 889–903, 2002.

[13] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 2013.

[14] E. B. Kosmatopoulos and J. L. Piovesan, "CLF-based control design for unknown multiinput nonlinear systems with good transient performance," *Automatic Control, IEEE Transactions on*, vol. 55, no. 11, pp. 2635–2640, 2010.

[15] A. Renzaglia, L. Doitsidis, A. Martinelli, and E. B. Kosmatopoulos, "Multi-robot three dimensional coverage of unknown areas," *The International Journal of Robotics Research*, p. 0278364912439332, 2012.

[16] L. Rodrigues, "Stability analysis using controlled invariant sets for piecewise-affine systems," in *Control and Automation, 2003. ICCA'03. Proceedings. 4th International Conference on*. IEEE, 2003, pp. 198–202.

[17] E. B. Kosmatopoulos, "An adaptive optimization scheme with satisfactory transient performance," *Automatica*, vol. 45, no. 3, pp. 716–723, 2009.

[18] E. B. Kosmatopoulos, M. Papageorgiou, A. Vakouli, and A. Kouvelas, "Adaptive fine-tuning of nonlinear control systems with application to the urban traffic control strategy tuc," *Control Systems Technology, IEEE Transactions on*, vol. 15, no. 6, pp. 991–1002, 2007.

[19] V. Granville, M. Křivánek, and J.-P. Rasson, "Simulated annealing: A proof of convergence," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 6, pp. 652–656, 1994.

[20] A. C. Kapoutsis, S. A. Chatzichristofis, L. Doitsidis, J. B. de Sousa, J. Pinto, J. Braga, and E. B. Kosmatopoulos, "Real-time adaptive multi-robot exploration with application to underwater map construction," *Autonomous Robots*, pp. 1–29, 2015.

[21] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE transactions on automatic control*, vol. 37, no. 3, pp. 332–341, 1992.