Auton Robot (2016) 40:1279–1306
DOI 10.1007/s10514-015-9511-7

Toward persistent autonomous intervention in a subsea panel

Narcís Palomeras[1] · Arnau Carrera[1] · Natàlia Hurtós[1] · George C. Karras[2] · Charalampos P. Bechlioulis[2] · Michael Cashmore[3] · Daniele Magazzeni[3] · Derek Long[3] · Maria Fox[3] · Kostas J. Kyriakopoulos[2] · Petar Kormushev[4] · Joaquim Salvi[1] · Marc Carreras[1]

Received: 15 December 2014 / Accepted: 14 October 2015 / Published online: 30 October 2015
© Springer Science+Business Media New York 2015

Abstract Intervention autonomous underwater vehicles (I-AUVs) have the potential to open new avenues for the maintenance and monitoring of offshore subsea facilities in a cost-effective way. However, this requires challenging intervention operations to be carried out persistently, thus minimizing human supervision and ensuring a reliable vehicle behaviour under unexpected perturbances and failures. This paper describes a system to perform autonomous intervention—in particular valve-turning—using the concept of persistent autonomy. To achieve this goal, we build a framework that integrates different disciplines, involving mechatronics, localization, control, machine learning and planning techniques, bearing in mind robustness in the implementation of all of them. We present experiments in a water tank, conducted with Girona 500 I-AUV in the context of a multiple intervention mission. Results show how the vehicle sets several valve panel configurations throughout the experiment while handling different errors, either spontaneous or induced. Finally, we report the insights gained from our experience and we discuss the main aspects that must be matured and refined in order to promote the future development of intervention autonomous vehicles that can operate, persistently, in subsea facilities.

✉ Narcís Palomeras
npalomer@eia.udg.edu

1 Computer Vision and Robotics Group (VICOROB), University of Girona, 17071 Gerona, Spain

2 School of Mechanical Engineering, National Technical University of Athens, Athens 15780, Greece

3 King's College London, London WC2R 2LS, UK

4 Department of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genoa, Italy

Keywords Autonomous underwater vehicles · Underwater intervention · Persistent autonomy

1 Introduction

Long-term autonomy is a primary goal of ongoing robotics research. The ability of a robot to operate robustly for an extended period of time (hours, days or even weeks), with reduced human supervision and in a real environment, is still a major challenge for today's autonomous robots. Real world operations are hard as the environment is often not completely known due to its dynamic nature and its inherent complexities. In addition, sensors used to perceive the environment and to self-locate often produce data that are noisy and incomplete. As a result, the effects of actions taken by the robot are not deterministic but uncertain. The robot must therefore have the ability to recognize failures and respond to them at all levels of abstraction.

These difficulties are further exacerbated when operations are carried out in marine environments, characterized by highly dynamic conditions and a physical medium that heavily restricts sensor capabilities. Due to these complexities, underwater operations have long relied on the use of remotely operated vehicle (ROVs), which require a dedicated pilot and an expensive support vessel that increases the overall operational cost. To overcome this main issue, research efforts in the last few decades have pushed towards enhancing the autonomy of these vehicles through the so-called autonomous underwater vehicle (AUVs). Free from the limitation of a physical connection to a surface ship, AUVs allow for extended operations at lower costs, providing stand-alone platforms that can gather data without human supervision and avoiding the risks associated with the umbilical cable. To date, AUVs have been deployed successfully for various forms of seabed (Singh et al. 2004; Caress et al. 2008; Gracias et al. 2013) and water column (Kunz et al. 2009; Camilli et al. 2010) transit surveys, on occasions exhibiting persistent autonomy capabilities in missions that have lasted many hours and, in the case of glider vehicles, even weeks or months (Smith et al. 2011; Jones 2012).

However, there is a large number of underwater applications that go beyond survey capabilities. The maintenance of permanent observatories, submerged oil wells, cabled sensor networks, pipes, and the deployment and recovery of benthic stations are just some of them. In general, companies are seeking improved ways to cost effectively and safely carry out more frequent inspection, repair and maintenance tasks on their subsea infrastructures. This is particularly challenging in deep water and is precisely where the development of hover-capable AUVs for subsea inspection and intervention can represent a major technology breakthrough.

Several successful attempts at fully autonomous underwater interventions have been demonstrated in the last few years, in the context of the ALIVE (Evans et al. 2003), SAUVIM (Marani et al. 2009) and TRIDENT (Sanz et al. 2012) research projects. Besides, first commercial units are on the way to be applied to simple hovering inspection tasks (SUBSEA7), with future units expected to address much harder intervention where contact is made to turn a valve or replace a component. However, despite these first demonstrations, and in order to be successful commercially, these intervention AUVs (I-AUVs) must be able to operate for extended periods without the continual presence of a surface vessel. They must therefore demonstrate persistent autonomy in a challenging environment.

This is precisely the goal of the Persistent Autonomy through Learning, Adaptation, Observation and Re-planning (PANDORA) project (Lane 2014) and the topic of this paper. PANDORA aims to demonstrate persistent autonomy by endowing underwater vehicles with advanced capabilities to describe the environment, act robustly, and adapt their plans as required over long periods of operation without human intervention. This requires that long-term missions, involving a variety of tasks that must be executed under time and resource constraints, can be planned at a strategic level, with tactical and operational replanning being performed when the environment does not behave as expected. This means that, even when considering a single sub-mission, planning of the required activities is done in the context of overall mission constraints and has impact on the execution of later tasks in the mission. Research in PANDORA focuses on three application areas: intervention operations, structure inspections, and chain mooring inspection and cleaning. This paper is framed in the first of these scenarios. Therefore, the work presented in this paper aims to demonstrate autonomous intervention missions, in particular autonomous valve-turning, in a persistent way.

We believe that this is a good example of a potential target application, representative of a relevant field of application—the offshore oil industry—that would certainly benefit from such capability. Common operations in the offshore oil production facilities involve the actuation of valves in their subsea structures, either for controlling the flow out of the wells or providing additional functions (injection of chemicals, pressure relief, well interventions, etc).

Our test scenario is defined as follows. An I-AUV is deployed in a partially unknown environment, i.e., at the approximate location of an underwater site where there are some known intervention panels that should be operated. The mission to be conducted consists of manipulating several panel valves to achieve different predefined configurations. This can include time constraints (e.g., a panel's valve must be in a given position at a given time or for a certain amount of time) and must take into account possible failures (such as the possibility of a stuck valve, difficulties in perceiving the panel or unexpected current perturbations). The vehicle must navigate towards several predefined inspection points and, using the on-board vision system, find the precise location of the panel. Then, it must analyze the panel's state and proceed to perform the intervention by autonomously turning the required valves to achieve the desired configuration. The accomplishment of the operation is evaluated again using the perception sensors, and the vehicle's planning system decides—according to the achieved state, the set goals, and the defined constraints—whether to continue with the current plan or re-plan accordingly. The vehicle should remain operating on the site for several hours, performing the actions that are required along the mission time (checking the state of a given panel, actuating the necessary valves, transiting between different panels, etc).

As there are a number of challenges in achieving persistent autonomy in this scenario, we believe the solution must come also from the combination of a number of core disciplines. This include, but may not be limited to, mechatronics, image perception, localization, control, machine learning, and planning. In this paper, we build on these core technologies and we seamlessly integrate them, achieving a single system that embodies the necessary capabilities to perform subsea valve-turning operations in a persistent way.

We begin in Sect. 2 with a description of the mechatronics elements that compose the system, involving the main vehicle platform, Girona 500 AUV, as well as other physical components that have been installed to perform the mission at hand. Section 3 addresses the localization and mapping in a partially-known environment. We implement a visual simultaneous localization and mapping (SLAM) algorithm that detects the panel's location and incorporates it within an extended Kalman filter (EKF) framework, enabling drift-free navigation and real-time localization of the vehicle with respect to the intervention panel. Section 4 introduces

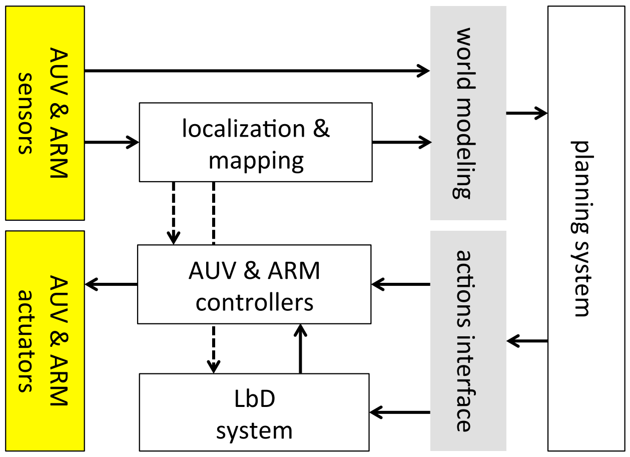


Fig. 1 Relations between the technologies involved to complete the proposed intervention task. Data from the AUV navigation sensors, camera images and manipulator joint states are sent to the localization and mapping algorithms in order to estimate the pose and velocities of the vehicle and the end-effector. These estimates are shared with the high and low level controllers as well as with the world modeller, which transforms the received information into symbols used by the planning system. The planner requests actions to the LbD system and to the AUV & Arm controllers that are executed by the real actuators

the AUV and manipulator low-level controllers. They provide an interface to high-level actions (e.g., go to waypoint, turn valve, etc.) based on position or velocity requests, thus decoupling these actions from complex vehicle dynamics. Moreover, these low-level controllers are designed to withstand perturbations so that they provide a first layer of robustness to the system. In Sect. 5 we lay out our approach to address the autonomous intervention tasks (valve-turning in our defined scenario). Instead of coding a given behaviour, we propose to use a learning by demonstration (LbD) technique, that extracts the task knowledge from a set of demonstrations performed by an expert and then generates the commands required to reproduce the task from the learned model. The advantage of such an approach is twofold: first, it increases the flexibility of the system, being easy to reconfigure it to different types of interventions, and second, it offers an extra layer of robustness as the task reproduction is dynamically generated—and therefore adapted—in the event of external perturbations. A final layer that confers a higher degree of autonomy and persistency to the system is given by the planning module, described in Sect. 6. The problem is modelled as a temporal-metric planning problem, where several configurations of the panel valves are set as a goal along the mission time. In order to handle possible failures during the execution, the implemented planner has the ability to re-plan when the observed states do not correspond with the expected ones. The relations between the different modules of the system detailed in all these sections are shown in Fig. 1.

In Sect. 7 we provide a compelling experimental section, showing results that demonstrate the effectiveness of the different system modules as well as the performance of the whole system in the described test scenario. We report an experiment that synthesises all the work, in which Girona 500 I-AUV operates during several hours, demonstrating persistent autonomy in performing intervention tasks. We analize the accomplishment of the goal valve positions and how the system handles different failures, eventual or induced, throughout the mission time. Finally, Sect. 8 reports on the conclusions and lessons learned from this work. Note that, because this paper draws into several different disciplines, relevant background material and further state-of-the-art descriptions are provided where necessary throughout the sections.

2 Mechatronics

A key factor in the pursuit of persistent autonomy is to have a suitable and reliable vehicle platform from the mechatronics point of view. The whole system will set its foundation on the mechatronics design and therefore it is important to choose the adequate vehicle concept, with the required motion capabilities, sensor suite and complementary equipment to facilitate the mission at hand. In this section we describe the vehicle that has been used in this work, Girona 500 AUV, as well as the manipulator and end-effector that have been incorporated to carry out the valve-turning intervention mission.

2.1 Girona 500 intervention-AUV

The Girona 500 AUV (Ribas et al. 2012), shown in Fig. 2, is a reconfigurable AUV designed to operate at depths up to 500 m. The vehicle is composed of an aluminum chassis supporting three torpedo-shaped hulls (0.3 m in diameter and 1.5 m in length) and a variable number of thrusters. The typical thruster configuration, which we used for this work's experiments, consists in five thrusters providing full controllability in the surge (x), sway (y), heave (z), and yaw (ψ) DoFs. The vehicle is naturally stable in roll (Φ) and pitch (θ) but, vertical thrusters can be used to increase pitch stability. The vehicle's design offers good hydrodynamic performance and room for equipment while keeping the vehicle compact. The overall dimensions of the AUV are 1 m height, 1 m width, 1.5 m length, and it weighs under 200 kg. The actual weight depends on the particular vehicle configuration and payload.

Girona 500's standard navigation sensor suite includes a pressure sensor, a doppler velocity log (DVL), an attitude and heading reference system (AHRS), and a global positioning system (GPS) to receive fixes while at the surface. The measurements from these sensors are integrated via an extended

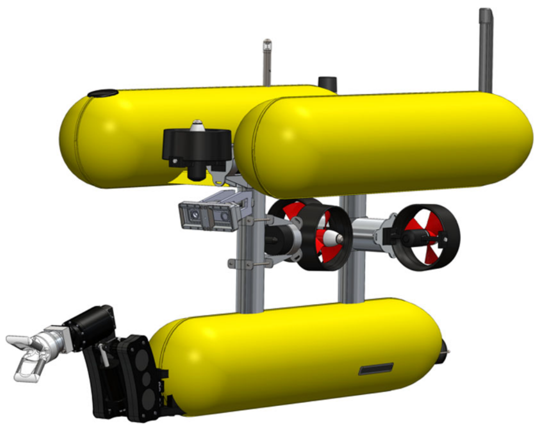


Fig. 2 Girona 500 AUV with the 4 DOFs ECA manipulator equipped with a custom end-effector

Kalman filter as will be explained in Sect. 3. This default sensor suite includes also a stereo camera that can be pointed forward or downward. For the experiments at hand the camera is placed looking forward to detect the intervention panel using a monocular vision-based algorithm.

2.2 Manipulator

Girona 500 can operate as an I-AUV when a manipulator is integrated in its payload area. For the panel intervention task, a 4 rotational DOFs commercial manipulator (ECA ARM 5E Micro), shown in Fig. 2, has been used. The manipulator can control the Cartesian position (x,y,z) and the $roll$ (Φ) of the end-effector. Since the manipulator is under-actuated, $pitch$ and yaw depend on the reached Cartesian position. The manipulator is rated for 300 meters and is one of the few commercial electrical manipulators available today. It maintains the typical mechanical configuration of ROV manipulators, which is useful when teleoperating with visual feedback. However, for autonomous intervention, the manipulator has a reduced workspace and low speed, which have to be compensated by a combined control of the AUV and the manipulator. Also, internal joint sensors do not provide absolute orientation, and forward kinematics must be done with an accurate calibration.

2.3 Custom end-effector

In order to correctly detect and operate the T-bar handles of the panel valve with the 4 DOFs manipulator, a custom end-effector was designed and built, as shown in Fig. 3. The main goal of the end-effector is to compensate the small misalignments in $pitch$ and yaw that cannot be compensated from the manipulator side, due to the reduced DOFs, as described in the previous section. Also, there are always some inaccuracies in the calibration or the detection of the valves which generate some inherent error in the position of the end-effector. Therefore, the external part of the end-effector is a flexible V-shape structure which passively adapts the end-effector position and orientation to the T-bar valve handle driving it to the center of the end-effector.

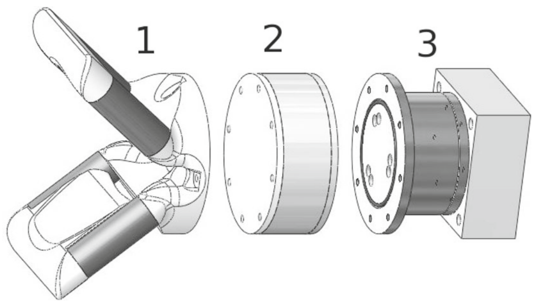


Fig. 3 3D model of the disassembled customized end-effector, in which three blocks can be distinguished: *1* passive gripper, *2* camera in-hand and *3* F/T sensor

The second goal of this end-effector is to integrate some sensors to perceive the intervention area from a closer point of view. A first module contains a small analog camera, in the center of the passive gripper, to provide visual feedback to a human operator while performing a teleoperated intrervention. After the camera module, a force/torque (F/T) sensor measures the contact forces and torques between the manipulator and the valve. This sensor is used to detect if the contact with the valve handle has been established, by measuring an axial force, and also, to measure the torque when rotating the valve. The forces measured by the sensor must be compensated for the depth of the end-effector, to discount the force generated by the water pressure in the F/T housing. The complete end-effector is mounted in the 4 DOFs manipulator, and will rotate according to the manipulator's $roll$ DOF.

3 Localization and mapping

An accurate localization and mapping system is paramount to achieve long-term autonomy in a semi-structured underwater environment like an offshore subsea facility or a marine observatory. This system must facilitate navigation within the environment as well as localize key elements for conducting the required intervention operations (e.g., a panel to turn valves, a structure to dock, a canister to collect objects, etc.). Therefore, a computational mechanism to enable the vehicle to maintain a geometric description of its environment and its location is required. This geometric description will be the basis for a higher-level semantic description used to plan and execute autonomous actions in real-time.

If a precise map of the area is available, the problem can be faced in a computationally cheap way using a map-based localization algorithm. However, this restricts the flexibility of the system to operate in dynamic environments as it does not allow the position of existing objects to be modified or new ones to be added into the map (Maurelli et al. 2008). Another solution is to localize the AUV relative to a known object, like a panel or a docking station. This approach requires *a priori* knowledge of the reference object to apply a model-based pose estimation technique, that relies on a list of detected object features (Palmer et al. 2009). Despite the simplicity and robustness of this method, it does not provide localization when the object is out of the field of view, nor a geometric map of the environment that may be useful for an on-board planner. Moreover, it becomes more difficult to combine georeferenced data that can be used to reach the intervention area (i.e., from GPS or ultra short base line (USBL)) with more accurate relative data (i.e., features detected on the landmark of interest) used to perform the intervention.

In this sense, a more complete mechanism are the popular SLAM algorithms. Several SLAM alternatives have been proposed in the underwater domain (Newman and Leonard 2003; Ribas et al. 2010; Salvi et al. 2008; Nagappa et al. 2013; Ozog and Eustice 2014). Some of these solutions are difficult to implement in real-time due to its high computational cost, mainly because they attempt to address very broad localization problems in completely unknown environments. However, we can exploit the fact that in our target scenario we do have an *a priori* knowledge of the intervention objects (i.e., the panel) to overcome that problem. In this section we present a low-computational cost SLAM system designed for real-time AUV operation in semi-structured environments with *a priori* known objects. The system is divided in two main modules: an EKF filter able to merge different sources of navigation data and a vision-based template detection algorithm capable to detect and estimate the position of a known object in the environment.

3.1 EKF localization filter

An EKF is used to estimate the vehicle's position ($[x\ y\ z]$) with respect to an inertial frame **I** and the vehicle's linear velocity ($[u\ v\ w]$) with respect to the vehicle's frame **V** (see Welch and Bishop 1995 for an introduction to EKF). Vehicle orientation ($[\phi\ \theta\ \psi]$) with respect to **I** and angular velocity ($[p\ q\ r]$) with respect to **V** are not estimated but directly measured by an AHRS. The vehicle attitude is not estimated for several reasons: (i) we only have one source that provides absolute attitude information (i.e., an AHRS); (ii) the AHRS has a high frequency and a small error when properly calibrated; and (iii) although the attitude was estimated, given the simplicity of the constant velocity model used in the prediction, its state would be driven almost exclusively by the measurements from the AHRS.

This filter is also able to map the pose of several landmarks in **I**, thus, working as a SLAM algorithm. Even though the filter is designed to deal with several landmarks, only one -the intervention panel- is used for the valve-turning task. A vision-based system, which will be explained later on, identifies the presence of the intervention panel in the environment and computes its relative position with respect to the vehicle. This information is introduced in the localization filter as a landmark and is updated with successive observations.

The information to be estimated by the SLAM algorithm is stored in the following state vector:

$$\mathbf{x}_k = [x\ y\ z\ u\ v\ w\ l_1\ \dots\ l_n]^T \tag{1}$$

where ($[x\ y\ z\ u\ v\ w]$) is the vehicle position and linear velocity and ($l_i = [lx_i\ ly_i\ lz_i\ l\phi_i\ l\theta_i\ l\psi_i]$) is the pose (position and orientation) of a landmark in **I**.

A constant velocity kinematics model is used to determine how the vehicle state will evolve from time $k-1$ to k. Detected landmarks are expected to be static, which is a reasonable assumption in the case of an intervention panel. The predicted state at time k, $\mathbf{x}_k^-$ follows the equations:

$$\mathbf{x}_k^- = f(\mathbf{x}_{k-1}, \mathbf{n}_{k-1}, \mathbf{u}_k, t). \tag{2}$$

$$\mathbf{x}_k^- = \begin{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \end{bmatrix} + \mathbf{R}(\phi_k\theta_k\psi_k)\left(\begin{bmatrix} u_{k-1} \\ v_{k-1} \\ w_{k-1} \end{bmatrix} t + \begin{bmatrix} n_{u_{k-1}} \\ n_{v_{k-1}} \\ n_{w_{k-1}} \end{bmatrix} \frac{t^2}{2}\right) \\ u_{k-1} + n_{u_{k-1}} t \\ v_{k-1} + n_{v_{k-1}} t \\ w_{k-1} + n_{w_{k-1}} t \\ l_{1_{k-1}} \\ \dots \\ l_{n_{k-1}} \end{bmatrix} \tag{3}$$

where t is the time period, $\mathbf{u} = [\phi\ \theta\ \psi]$ is the control input determining the current vehicle orientation in **I** and $\mathbf{n} = [n_u\ n_v\ n_w]$ is a vector of zero-mean white Gaussian acceleration noise whose covariance, represented by the system noise matrix **Q**, has been set empirically:

$$\mathbf{Q} = \begin{bmatrix} \sigma_{n_u}^2 & 0 & 0 \\ 0 & \sigma_{n_v}^2 & 0 \\ 0 & 0 & \sigma_{n_w}^2 \end{bmatrix} \tag{4}$$

Associated with the state vector $\mathbf{x}_k$ there is the covariance matrix $\mathbf{P}_k$. Following standard EKF operations, the covariance of the prediction at time k is obtained as:

$$\mathbf{P}_k^- = \mathbf{A}_k \mathbf{P}_{k-1} \mathbf{A}_k^T + \mathbf{W}_k \mathbf{Q}_{k-1} \mathbf{W}_k^T, \tag{5}$$

where $\mathbf{A}_k$ is the Jacobian matrix of partial derivatives of f with respect to the state (1) and $\mathbf{W}_k$ is the Jacobian matrix of partial derivatives of f with respect to the process noise **n**.

Three linear measurement updates are applied in the filter: pose, velocity, and landmark updates. Each sensor measurement is modelled as:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{s}_k, \tag{6}$$

where $\mathbf{z}_k$ is the measurement itself, $\mathbf{H}$ is the observation matrix that relates the state vector with the sensor measurement, and $\mathbf{s}_k$ is the sensor noise. Updates are applied by means of the following equations:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}^T (\mathbf{H}\mathbf{P}_k^- \mathbf{H}^T + \mathbf{R})^{-1}, \tag{7}$$

$$\mathbf{x}_k = \mathbf{x}_k^- + \mathbf{K}_k (\mathbf{z}_k - \mathbf{H}\mathbf{x}_k^-), \tag{8}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H})\mathbf{P}_k^-, \tag{9}$$

where $\mathbf{K}_k$ is the Kalman gain, $\mathbf{R}$ the measurement noise covariance matrix and $\mathbf{I}$ an identity matrix.

Several sensors provide position information, which can be used to initialize the vehicle position and bound dead-reckoning errors. A GPS receiver measures vehicle position in the plane $(x,\ y)$ while the vehicle is at the surface, a pressure sensor transforms pressure values into depth (z) and a USBL device measures vehicle position $(x,\ y,\ z)$ while submerged. To integrate any of these measurements, three $\mathbf{z}_k$ and $\mathbf{H}$ equation pairs have been defined for the GPS (10), the depth sensor (11), and the USBL (12):

$$\mathbf{z}_k = [x\ y],\ \ \mathbf{H} = \begin{bmatrix} \mathbf{I}_{2\times 2} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 3} & \mathbf{0}_{2\times 6n} \end{bmatrix}, \tag{10}$$

$$\mathbf{z}_k = [z],\ \ \mathbf{H} = \begin{bmatrix} 0 & 0 & 1 & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 6n} \end{bmatrix}, \tag{11}$$

$$\mathbf{z}_k = [x\ y\ z],\ \ \mathbf{H} = \begin{bmatrix} \mathbf{I}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 6n} \end{bmatrix}, \tag{12}$$

where $\mathbf{I}_{m\times m}$ denotes an $m \times m$ identity matrix and $\mathbf{0}_{m\times 6n}$ is an $m \times 6n$ zero matrix, with n being the number of landmarks.

In the Girona 500 AUV, velocity updates are provided by a DVL sensor that measures linear velocities with respect to the sea bottom or the water below the vehicle. These measurements are integrated using:

$$\mathbf{z}_k = [u\ v\ w], \tag{13}$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{0}_{3\times 3} & \mathbf{I}_{3\times 3} & \mathbf{0}_{3\times 6n} \end{bmatrix}. \tag{14}$$

When only velocity updates are available, the filter behaves as a dead-reckoning algorithm that drifts over time. However, if position updates or landmarks are detected, the localization filter is able to keep its error bounded.

To identify the valve panel and compute its pose a vision-based algorithm is used. This algorithm computes the panel position with respect to $\mathbf{V}$ as well as its orientation with respect to $\mathbf{I}$. This information is introduced in the localization filter to improve both vehicle and panel localization:

$$\mathbf{z}_k = [L_x\ L_y\ L_z\ L_\phi\ L_\theta\ L_\psi], \tag{15}$$

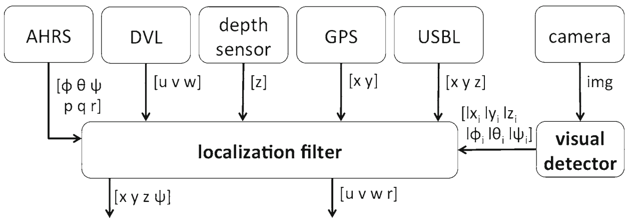


Fig. 4 Set of nodes that take part in the localization and mapping system for the Girona 500 I-AUV

$$\mathbf{H} = \begin{bmatrix} -\mathbf{Rot}^T & \mathbf{0}_{3\times 3} & \mathbf{Rot}^T & \mathbf{0}_{3\times 3} & \dots \\ \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{I}_{3\times 3} & \dots \end{bmatrix}, \tag{16}$$

where $[L_x\ L_y\ L_z]$ is the relative position of the landmark with respect to the vehicle, $[\ L_\phi\ L_\theta\ L_\psi]$ is the landmark orientation with respect to the inertial frame and $\mathbf{Rot}$ is the vehicle orientation rotation matrix.

The block diagram in Fig. 4 shows how each navigation sensor is connected with the localization filter. The state vector is initialized either with the GPS and pressure sensor measurements or according to the USBL, in case it is available and the vehicle is submerged. Linear velocities are set to zero and no landmarks are present when the filter is initialized. To introduce a new landmark the following procedure is applied: (i) on the event of a new detected landmark, its pose in the world is computed by composing its relative pose with respect to the AUV and the AUV pose contained in the state vector; (ii) the obtained $[l_x\ l_y\ l_z\ l_\phi\ l_\theta\ l_\psi]$ pose is stored into a first in first out (FIFO) queue of size 5 and the mean and standard variation of all the elements in this queue is computed; (iii) when the standard deviation of all the elements is below a threshold the landmark is introduced in the filter extending the state vector and the covariance matrix as shown in Eqs. (17) and (18).

$$\mathbf{x}_k = [\mathbf{x}_{k-1}\ \bar{l}_x\ \bar{l}_y\ \bar{l}_z\ \bar{l}_\phi\ \bar{l}_\theta\ \bar{l}_\psi]^T, \tag{17}$$

$$\mathbf{P}_k = \begin{bmatrix} \multicolumn{2}{c}{\mathbf{P}_{k-1}} & \mathbf{P}_{k-1}[0:3,0:3] & \mathbf{0}_{3\times 3} \\ & & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} \\ \mathbf{P}_{k-1}[0:3,0:3] & \mathbf{0}_{3\times 3} & \mathbf{M1} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{M2} \end{bmatrix}, \tag{18}$$

where $\mathbf{P}_{k-1}[0:3,0:3]$ is the first 3×3 submatrix of $\mathbf{P}_{k-1}$ and $\mathbf{M1}$ and $\mathbf{M2}$ are defined as follows.

$$\mathbf{M1} = [\mathbf{Rot} \times \mathbf{L}[0:3,0:3] \times \mathbf{Rot}^T + \mathbf{P}_{k-1}[0:3,0:3]], \tag{19}$$

$$\mathbf{M2} = [\mathbf{Rot} \times \mathbf{L}[3:6,3:6] \times \mathbf{Rot}^T], \tag{20}$$

where $\mathbf{L}$ is the covariance matrix for the detected landmark.

3.2 Vision-based template detection algorithm

To compute the position of a known landmark (e.g., the subsea valve panel) we use a vision-based algorithm. The images gathered by the on-board camera are compared against an *a priori* known template and salient features are extracted in both the camera images and the template. If a sufficient number of features can be matched across the two representations it is possible to detect the presence of the landmark as well as accurately estimate its pose.

In order to extract keypoints in the image we use the oriented FAST and rotated BRIEF (ORB) feature extractor that relies on features from accelerated segment test (FAST) corner detection (Rosten and Drummond 2006) and a binary descriptor vector based on binary robust independent elementary features (BRIEF) (Calonder et al. 2010). These kind of features are present on man-made structures like a valve panel and can be quickly detected.

With this approach, differences between descriptors can be calculated rapidly, allowing real-time matching of keypoints at higher image frame-rates when compared to other commonly used feature extractors such as scale invariant feature transform (SIFT) (Lowe 2004) and speeded-up robust features (SURF) (Bay et al. 2008).

Figure 5 illustrates the matching procedure between the *a priori* known template and an image received from the camera. A minimum number of keypoints (i.e., 25–40) must be matched between the template and the camera image to satisfy the landmark detection requirement. The detected correspondences are used to compute the transformation that relates the template image to the detected landmark. Then, using the camera parameters and the known dimensions of the landmark (i.e., the panel), the landmark's pose can be determined in the camera frame and therefore also in the vehicle frame **V**.

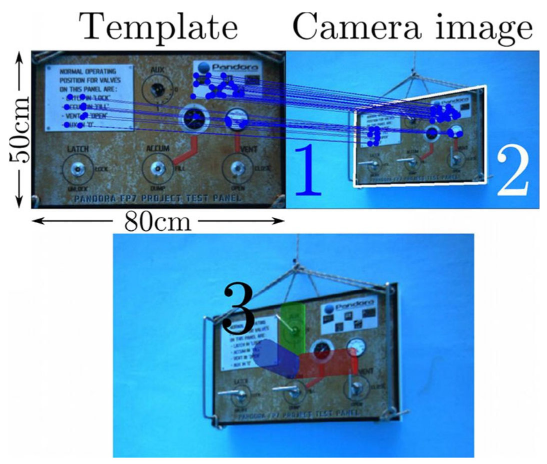


Fig. 5 Steps of the landmark detection: *1* Matching of keypoints between the template and camera image. *2* Estimation of the panel corners in the camera image. *3* Estimation of the template's position and rotation in the image by using the camera parameters and the known geometry of the landmark

Note that the performance of vision-based algorithms can be significantly degraded in operational scenarios due to water turbidity. In this situation, the visibility range can decrease from a few meters to less than 10 cm and a template detection algorithm like the one presented here can fail to accurately estimate a landmark pose. To overcome this problem it is possible to replace or combine this visual-based detector with other detectors that are more resilient to low visibility conditions. From the filter point of view, this change is straightforward as the landmark detector is seen as a black box that provides a relative position between the AUV and a specific landmark. A widely accepted solution to detect a target under poor visibility conditions is to equip it with active beacons. While this solution is outside the scope of this present paper, two approaches have already been explored and preliminary results have been obtained with Girona 500 AUV. In the first solution, an acoustic pinger is attached to the target panel. In order to localize it, a sum of Gaussians (SoG) filter together with an active localization method are used to minimize the uncertainty of the beacon position (Vallicrosa et al. 2014). The second proposed solution, uses a set of light beacons to localize static or moving objects (Gracias et al. 2015). Although this method also relies on vision, it has been shown capable of estimating the pose of a target from up to 2–3 m in low visibility conditions. Hence, these methods can be a good solution to complement or substitute the visual-based detector presented in this article when moving to real sea scenarios.

4 Control

The motion control problem for autonomous underwater vehicles has been an active research field for the past two decades and continues to pose considerable challenges to control designers especially when the vehicles are affected by severe environmental disturbances and exhibit large model uncertainties. The particular case of the vehicle used in this work (Girona 500 I-AUV) is a clear example of a vehicle that targets complex operations and dexterous tasks (i.e., monitoring of underwater structures or manipulation of underwater equipment such as control valves) while it might be subject to the influence of strong external disturbances caused from ocean currents and waves.

The main motion control modes of AUVs are categorized in: a) velocity and b) pose (position/orientation) tracking control schemes. The objective of velocity control schemes is to minimize the error between a commanded velocity setpoint and the actual velocity of the vehicle. Velocity controllers are commonly used in teleoperation schemes where the commanded velocity is directly given by the human operator through a joystick device. On the other hand, the objective of pose tracking controllers is to minimize the error between a desired pose setpoint and the actual pose of the vehicle. Pose controllers are usually employed for way-point reaching and station keeping missions.

In this section, we address the velocity and pose tracking control problems for the Girona 500 I-AUV as well as the controller for its manipulator. The proposed control schemes have been fully integrated in its architecture and were developed in accordance with the control specifications and requirements induced by the high-level planning modules and tasks. Both velocity and pose control schemes were designed based on the dynamic model parameters of the Girona 500 I-AUV model that were obtained via an off-line identification process detailed in Karras et al. (2013). Their purpose is twofold: (i) achieve the desired pose or velocity set-points respectively and (ii) compensate for external disturbances and parametric model uncertainties. Moreover, the stability of the unactuated degrees of freedom (DoF) (roll) is secured.

The rest of this section is divided in two parts. First, the corresponding velocity and pose control schemes are presented along with the thruster control matrix (TCM) design that maps the controller's output (i.e., body forces and torques) into thrust commands. Second, the controller implemented for the I-AUV manipulator is described.

4.1 AUV controller

Figure 6 shows the velocity and pose control schemes for the Girona 500 I-AUV along with the TCM module that maps the controllers output into thruster commands. This figure shows how the proposed low-level control system can equally deal with pose, velocity or force requests.

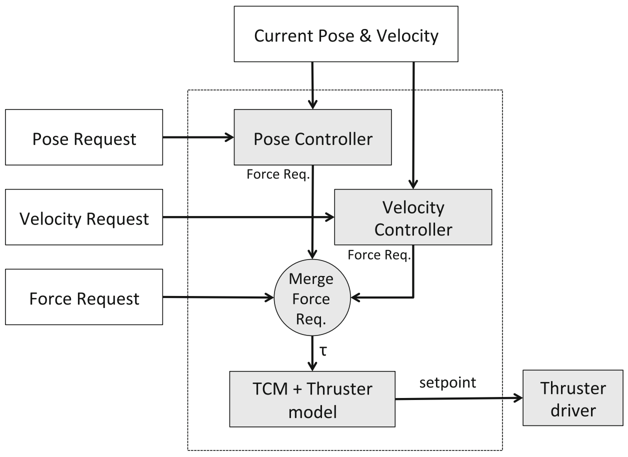


Fig. 6 Girona 500 I-AUV control scheme

4.1.1 Velocity control

The velocity control scheme is based on the already identified vehicle dynamic model parameters. It tries to compensate for external disturbances and residual parametric uncertainty while maintaining the stability of the unactuated roll DoF.

Let $u_d(t)$, $v_d(t)$, $w_d(t)$ and $q_d(t)$, $r_d(t)$ denote the desired linear and angular velocities respectively. Then, the linear and angular velocity errors are defined as:

$$e_u = u - u_d,\ e_v = v - v_d,\ e_w = w - w_d,\\ e_q = q - q_d,\ e_r = r - r_d \tag{21}$$

As a result, we design the external forces in the surge (X), sway (Y) and heave (Z) as well as the external torques in pitch (M) and yaw (N) as:

$$\begin{bmatrix} X \\ Y \\ Z \\ M \\ N \end{bmatrix} = \hat{M} \begin{bmatrix} \dot{u}_d \\ \dot{v}_d \\ \dot{w}_d \\ \dot{q}_d \\ \dot{r}_d \end{bmatrix} + \left(\hat{C} + \hat{D}\right) \begin{bmatrix} u_d \\ v_d \\ w_d \\ q_d \\ r_d \end{bmatrix} + \hat{G} - K_v \begin{bmatrix} e_u \\ e_v \\ e_w \\ e_q \\ e_r \end{bmatrix} (I + K_r F_r) - K_I \int_0^t \begin{bmatrix} e_u(\tau) \\ e_v(\tau) \\ e_w(\tau) \\ e_q(\tau) \\ e_r(\tau) \end{bmatrix} d\tau. \tag{22}$$

where K_v, K_I, K_r are positive definite gain matrices, $\hat{M}$, $\hat{C}$, $\hat{D}$, $\hat{G}$ denote the estimate of the dynamic model of the underwater vehicle (following the nomenclature established in Fossen 1994) and F_r is a robustifying term, that compensates for residual parametric uncertainty, given as follows:

$$F_r = \left\| \begin{bmatrix} \dot{u}_d \\ \dot{v}_d \\ \dot{w}_d \\ \dot{q}_d \\ \dot{r}_d \end{bmatrix} \right\|^2 + \left\| \begin{bmatrix} u_d \\ v_d \\ w_d \\ q_d \\ r_d \end{bmatrix} \right\|^2 \tag{23}$$

4.1.2 Position and orientation tracking control

A smooth pose tracking control scheme is designed based on the dynamic parameters of the AUV model, compensating for external disturbances as well as for residual parametric uncertainty and securing the stability of the unactuated roll DoF. Let x_d, y_d, z_d, θ_d, and ψ_d denote the desired way points and orientations respectively. Then, the position and orientation errors are defined as follows:

$$e_x = x - x_d,\ e_y = y - y_d,\ e_z = z - z_d,$$
$$e_\theta = \theta - \theta_d,\ e_\psi = \psi - \psi_d \tag{24}$$

As a result, we design the desired velocities:

$$\begin{bmatrix} u_d \\ v_d \\ w_d \end{bmatrix} = -K_p J_v^{-1} \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}$$
$$\begin{bmatrix} q_d \\ r_d \end{bmatrix} = -K_o J_\omega^{-1} \begin{bmatrix} e_\theta \\ e_\psi \end{bmatrix} \tag{25}$$

with positive definite gain matrices K_p, K_o, where the terms J_v and J_ω denote the transformation matrices from linear (u, v, w) and angular (q, r) body velocities to position (x,y,z) and Euler angle (θ, ψ) rates respectively. Moreover, we define the velocity errors $e_u = u - u_d$, $e_v = v - v_d$, $e_w = w - w_d$, $e_q = q - q_d$ and $e_r = r - r_d$ and design the external forces in the surge, sway and heave as well as the external torques in pitch and yaw as:

$$\begin{bmatrix} X \\ Y \\ Z \\ M \\ N \end{bmatrix} = \hat{M} \begin{bmatrix} \dot{u}_d \\ \dot{v}_d \\ \dot{w}_d \\ \dot{q}_d \\ \dot{r}_d \end{bmatrix} + \left(\hat{C} + \hat{D}\right) \begin{bmatrix} u_d \\ v_d \\ w_d \\ q_d \\ r_d \end{bmatrix}$$
$$+ \hat{G} + \begin{bmatrix} J_v^T \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} \\ J_\omega^T \begin{bmatrix} e_\theta \\ e_\psi \end{bmatrix} \end{bmatrix} - K_v \begin{bmatrix} e_u \\ e_v \\ e_w \\ e_q \\ e_r \end{bmatrix} (1 + K_r F_r)$$
$$- K_I \begin{bmatrix} \int_0^t J_v^T \begin{bmatrix} e_x(\tau) \\ e_y(\tau) \\ e_z(\tau) \end{bmatrix} d\tau \\ \int_0^t J_\omega^T \begin{bmatrix} e_\theta(\tau) \\ e_\psi(\tau) \end{bmatrix} d\tau \end{bmatrix} \tag{26}$$

where K_v, K_I, K_r are positive definite gain matrices, $\hat{M}, \hat{C}, \hat{D}, \hat{G}$ denote the estimate of the dynamic model of the underwater vehicle and F_r denotes the robustifying term as described in Eq. 23.

4.1.3 Performance properties

It can be easily verified, following standard elements of Lyapunov theory, combined with the backstepping technique for the pose tracking control (see for example Fossen 1994 and Spooner et al. 2002), that the error performance is given as follows:

$$e_u,\ e_v,\ e_w,\ e_q,\ e_r \sim \left(\frac{\left\|\tilde{M}\right\|^2 + \left\|\tilde{C}\right\|^2 + \left\|\tilde{D}\right\|^2 + \left\|\tilde{G}\right\|^2 + \left\|\tilde{d(t)}\right\|_\infty^2}{\lambda_{\min}\left(K_p\right) \lambda_{\min}\left(K_v K_r\right)} \right) \tag{27}$$

and

$$e_x,\ e_y,\ e_z,\ e_\theta,\ e_\psi \sim \left(\frac{\left\|\tilde{M}\right\|^2 + \left\|\tilde{C}\right\|^2 + \left\|\tilde{D}\right\|^2 + \left\|\tilde{G}\right\|^2}{\lambda_{\min}\left(K_p\right) \lambda_{\min}\left(K_v K_r\right)} \right) \tag{28}$$

where $\tilde{M}, \tilde{C}, \tilde{D}, \tilde{G}$ denote the residual parametric modelling errors. We can therefore improve the performance of the controller via increasing the control gains K_p, K_v, K_r or alternatively by forcing small parametric errors via more extensive and well-organized on-line identification processes.

4.1.4 Thruster allocation design

Both pose and velocity controllers generate as output a force request (FR) that is merged with other possible FR coming from high-level controllers. The resulting force (τ_d) must be then generated by the vehicle thrusters. To compute the thrust (thr) that each propeller should yield, the combined FR τ_d is multiplied by the inverse of the TCM:

$$thr = \tau_d \times TCM^{-1},$$
$$command_i = p_1 \times thr_i^n + \cdots + p_n \times thr_i^1 + p_{n+1}. \tag{29}$$

The TCM codifies the amount of force that each thruster produces per DoF. Then, to compute the command for each thruster we use a model for the SeaEye MCT1 thrusters in polynomial form ($p_1 + \cdots + p_{n+1}$) that has been found by comparing issued commands with the generated thrust measured with a dynamometer. The whole control loop is executed at 20Hz.

4.2 Manipulator controller

Given that the manipulator is underactuated (see Sect. 2.2), it is only able to reach a Cartesian position (x, y, z) with a desired *roll* (Φ) orientation. This means that pitch (θ) and yaw (ψ) can not be specified, being implicitly defined by the other 4 DoFs.

The manipulator's low-level controller receives velocity requests for the end-effector in the Cartesian space. In order to reach these velocities, it controls the velocity for each joint ($\dot{q} \in \Re^4$). To this end, the desired velocity is transformed to an increment in the Cartesian space (**x**), and using the pseudo-inverse Jacobian (J^+) of the manipulator, $\dot{q}$ is obtained as follows:

$$\dot{q} = J^+\dot{x}. \tag{30}$$

Furthermore, there is also a pose controller in cascade with this velocity controller that allows the end-effector to be moved to a particular position.

5 Learning by demonstration

In order to program the intervention tasks in a flexible and intuitive way we propose to use a machine learning algorithm known as learning by demonstration (LbD). Rather than analytically decomposing the problem and manually programming a desired behaviour, the LbD infers the knowledge from a set of user demonstrations. This kind of algorithm consist of three sequential phases: *Demonstration* where a batch of task examples performed by a human pilot are recorded; *Learning*, where a model is created by generalizing all the demonstrations; and *Reproduction*, where the model is used to accomplish new instantiations of the learned task. Using such a technique, the framework becomes easily extensible and new intervention tasks (involving both the AUV and manipulator motions) can be added effortlessly just from operator demonstrations.

Different LbD algorithms have been proposed along the literature to encode and learn a trajectory. In the work of Calinon (2010) a humanoid robot learned different human skills using a representation based on Gaussian mixture models (GMM). This representation was later extended by Kruger (2012) using incremental GMM to automatically set the number of Gaussians to control a robotic arm. Similar to the GMM, a hidden Markov model (HMM) (Hovland et al. 1996) has also been used to represent a trajectory. Both GMM and HMM representations require the use of a regression algorithm like the Gaussian mixture regression (GMR) to generate a desired trajectory with an associated density distribution. A different possibility consists of using dynamic movement primitives (DMP) (Hoffmann et al. 2009; Pastor et al. 2009). DMP encapsulates the learned skill in a superposition of basis motion fields. Unlike GMM and HMM, DMP uses the learned model to dynamically generate the commands required to perform the reproduction of the trajectory. This inherently gives the approach certain robustness to external perturbations and the capability to adapt in different domains. DMP has been also parametrized by Matsubara et al. (2011) and extended by Kormushev et al. (2011) to include a force associated with the trajectory.

Therefore, given the simplicity of the representation and its flexibility, DMP is more suitable than GMM or HMM in the context of this work and has been chosen as the base of our learning framework.

5.1 Dynamic movement primitives

As introduced before, DMP encapsulates a learned skill as a superposition of basis motion fields. In this work we have used and extended the DMP approach proposed by Kormushev et al. (2011). To better understand the DMP encoding, we can imagine a mass attached to different damped strings. These strings attract the mass changing their forces throughout the experiment execution, thus moving the mass along the desired trajectory (see Fig. 7).

To generate the proper superposition each attractor has an associated weight which changes along the time defined by the $h_i(t)$ function (31). The weight of each attractor is represented with Gaussians, whose centers μ_i^T are equally distributed in time, and whose variance parameters $\Sigma_i^T = total\_time/K$ are set to a constant value inversely proportional to the number of Gaussians (K).

$$h_i(t) = \frac{\mathcal{N}\left(t; \mu_i^T, \Sigma_i^T\right)}{\sum_{k=1}^{K} \mathcal{N}\left(t; \mu_k^T, \Sigma_k^T\right)}, \tag{31}$$

Instead of using the real time a decay term is used, to obtain a time invariant model:

$$t = \frac{ln(s)}{\alpha}, \tag{32}$$

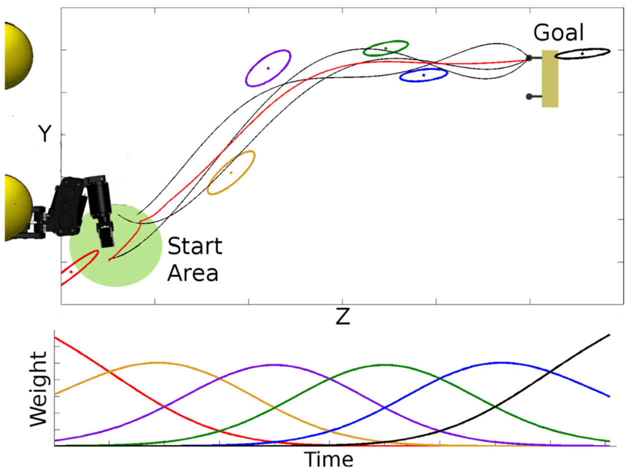


Fig. 7 *Top figure* shows a set of 2D demonstrated trajectories (*black*) and one reproduction (*red*). In this case, the demonstrated trajectory must grasp the valve aligning the fore arm of the manipulator with the valve. Below, the *h* function is represented. The encoding of the trajectory using a DMP algorithm has been done using 6 Gaussians adequately weighted over time (Color figure online)

where s is a canonical system:

$$\dot{s} = s - \alpha s,$$

and the α value is selected by the user depending on the duration of the demonstrated task.

The number of attractors (represented by Gaussians) is preselected by the user, according to the complexity of the task. The position of the attractor is determined by the center of the Gaussian (μ_i^x) and its stiffness (K_i^P) is represented by the covariance. The values are learned from the observed data through least-squares regressions. All the data from the demonstrations is concatenated in a matrix $Y = [\ddot{x}\frac{1}{K^P} + \dot{x}\frac{K^V}{K^P} + x]$, where x, $\dot{x}$ and $\ddot{x}$ are the position, velocity and acceleration recorded at each time instant of the demonstrations. Also the weights at each time instant are concatenated to obtain matrix H. With these two matrices, the linear equation $Y = H\mu^x$ can be written. The least-square solution to estimate the attractor center is then given by $\mu^x = H^{\dagger}Y$, where $H^{\dagger} = (H^T H)^{-1}H^T$ is the pseudo-inverse of H.

The user needs to establish a minimum K_{min}^P, and maximum K_{max}^P to define the limits of the stiffness and to estimate the damping as follows:

$$K^P = K_{min}^P + \frac{K_{max}^P - K_{min}^P}{2},\ K^V = 2\sqrt{K^P}. \tag{33}$$

To take into account variability and correlation along the movement and among the different demonstrations, the residual errors of the least-squares estimations are computed in the form of covariance matrices, for each Gaussian ($i \in \{1, \ldots K\}$).

$$\Sigma_i^X = \frac{1}{N}\sum_{j=1}^{N}\left(Y'_{j,i} - \bar{Y}_i{}'\right)\left(Y'_{j,i} - \bar{Y}_i{}'\right)^T, \quad \forall_i \in \{1, \ldots K\}, \tag{34}$$

where:

$$Y'_{j,i} = H_{j,i}\left(Y_j - \mu_i^x\right). \tag{35}$$

In (34), the $\bar{Y}_i{}'$ is the mean of Y'_i over the N data points.

Lastly, the residual terms of the regression process are used to estimate the K_i^P through the eigen components decomposition.

$$K_i^P = V_i D_i V_i^{-1}, \tag{36}$$

where:

$$D_i = k_{min}^P + \left(k_{max}^P - k_{min}^P\right)\frac{\lambda_i - \lambda_{min}}{\lambda_{max} - \lambda_{min}}. \tag{37}$$

In (37), the λ_i and the V_i are the concatenated eigenvalues and eigenvector for the inverse covariance matrix $(\Sigma_i^x)^{-1}$. The underlying idea is to determine a stiffness matrix proportional to the inverse of the observed covariance.

Therefore, the model for a given task will be composed by: the k_i^P matrices and μ_i^x centers representing the Gaussians; $h_i(t)$ representing the influence of each matrix functions; K^V representing the damping; and α, which is assigned according to the duration of the sample. Figure 7 shows a simple example where the learned data are represented.

Finally, to reproduce the learned skill, the desired acceleration is generated with

$$\hat{\ddot{x}} = \sum_{i=1}^{K} h_i(t)\left[K_i^P\left(\mu_i^X - x\right) - K^v\dot{x}\right], \tag{38}$$

where x and $\dot{x}$ are the current position and velocity.

5.2 Demonstrations with teleoperation

The teleoperation phase is a key step for the proper functioning of the presented framework, as the quality of the learning will depend on the quality of the acquired demonstrations. For this reason, the controls used to teleoperate the AUV and the manipulator must be easy to use by the human operator that performs the demonstrations.

In our implementation, the whole system (i.e., AUV and manipulator) is controlled using an Omega 7 haptic device (Fig. 8). This device provides an intuitive control of 7 DOFs using only one hand. Furthermore, its force-feedback capability has been connected with the data measured by the F/T sensor so that the operator perceives the contact with the valve handle.

The operator uses a Graphical User Interface (GUI) to see the vehicle's cameras as well as a 3D representation of the AUV, the end-effector, and the target of interest.

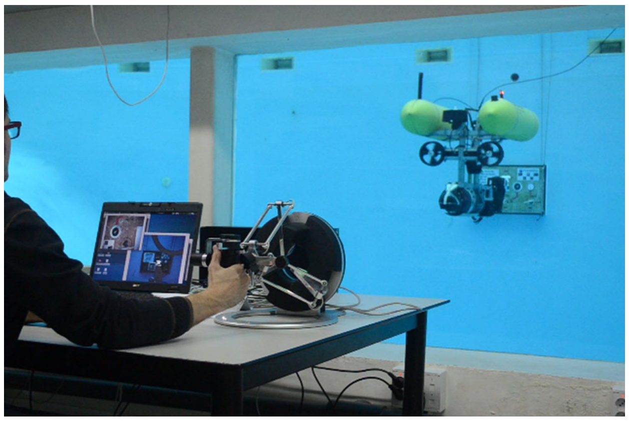


Fig. 8 A user performing a demonstration using the Omega 7 haptic device and the GUI for the Girona 500 AUV

During the teleoperation process the AUV and the end-effector trajectories are recorded and stored in a demonstration file. The main coordinate frame is defined at the target valve position, but with the panel's orientation. The measured valve orientation is only used to adjust the end-effector alignment in order to grasp the valve. This allows us to use the same learned model for any valve in any configuration.

5.3 Learning the DMP trajectory

The LbD loads the set of recorded trajectories and uses them to generate a model with the previously described formulation (38). The DMP algorithm has been implemented to control a total of 8 DoFs, 4 DoF to represent the trajectory of the AUV and 4 DoF for the manipulator. Since DMP uses an n-dimensional state vector, the number of DoFs to control does not affect the underlying formulation. However, to represent the AUV orientation *yaw* (ψ) and the end-effector *roll* (Φ) the reproduction formula (38) has been modified to properly handle angle values by normalizing the difference ($\mu_i^X - x$) between π and $-\pi$.

5.3.1 LbD reproducer

The LbD reproducer uses the created model to generate the AUV and manipulator requested velocities to perform the task. The requested velocities sent to the manipulator are obtained by subtracting the AUV requested velocities to the desired end-effector velocities. To generate these requests the model needs to know the current pose of the AUV, the manipulator, and the target valve.

A specific procedure has been added to turn the valve to a desired orientation while controlling the applied torque to avoid breaking the valve in case it gets stuck. This procedure is activated when the F/T sensor detects that the valve is correctly grasped. The LbD reproducer is forced to keep the grasping position and applies, at the same time, a force to keep a constant contact. Finally, when the turn is completed or aborted, the AUV moves backward to a safe distance from the panel.

To maintain the grasping position, the DMP has been modified to alter the advance of the time (t) (32) associated to the reproduction. This time is generated by a canonical system ($\dot{s} = s - \alpha s i$) that accepts as an input (i) an external continuous value ($[-1, 1]$). In our case, the input value will be set to 1 if we want the reproduction to take the same time than the demonstration. However, when the valve is grasped the input will be set to 0 forcing the AUV and the end-effector to stop thus maintaining the current position. Note that with this modification we can also slow down the reproduction or even follow the trajectory backwards. This could be convenient to roll back the approaching maneuver once it has been initiated and some unexpected disturbance takes place (e.g., the panel is occluded) (Ahmadzadeh et al. 2014).

6 Planning

Planning is a technology explored in Artificial Intelligence that seeks to organize activities in order to achieve specific goals over time and under resource constraints (Ghallab et al. 2004). It begins with a *domain model* describing the actions available to the planner, and a description of the *problem instance* specifying the current state, of both the executive and its environment, and the goals to be achieved. The domain model is fixed, while different planning instances can be constructed in response to changes in the environment. The planner constructs a plan by organizing instances of the actions into a partially ordered network of activity that is causally valid and is predicted to achieve the goals, while optimizing a cost function.

Planning is needed for strategic problem-solving, giving structure and purpose to the execution of specific activities. Planning is essential when acting with constrained resources; the consideration of how future goals will be achieved can place constraints on the ways in which current tasks should be best achieved that are not apparent when viewing the current task in isolation.

Several previous works in AUV control have explored the use of offline AI Planning. Examples include coordinating autonomous behaviour in an underwater maintanence vehicle (Cashmore et al. 2013, 2014), generating a coarse plan to initialise the inspection of an unknown hull (Englot and Hover 2010) and using a plan-based policy to guide an AUV for autonomously tracking the boundary of the surface of a partially submerged harmful algal bloom (Fox et al. 2012).

Because of the need for an agile response to execution conditions, offline methods are not the most robust for mission planning (Rajan et al. 2007). The Teleo-Reactive Executive (T-REX), currently deployed on the Dorado-class AUV at MBARI (McGann et al. 2007), has been proposed as an architecture to support online planning and plan execution. In PANDORA, an on-board planner is used to address the need for agility. In the event of the failure of an action under execution, the system is able to reformulate the model of the environment and replan to achieve new goals as they arise.

In PANDORA we consider long-term missions involving valve-turning and inspection tasks to be carried out in a large undersea area. The actions available in this PANDORA application are shown in Table 1 and an example plan is shown in Fig. 9. Details about the modelling of these actions and the structure of the plan are presented in Sect. 6.2.

There might be many valve-turning tasks involved in a mission, as well as other maintenance and inspection tasks, to be carried out at different locations, interacting with different

Table 1 The set of action schemas used in PANDORA

do_hover_fast	Fast but inaccurate navigation between waypoints
do_hover_controlled	Slow and acccurate navigation between waypoints
turn_valve	A manipulation action that turns a valve to a desired angle
observe_inspection_point	Observation action
correct_position	Localization action following navigation
recalibrate_arm	Recalibrates the arm
examine_panel	An action to check valve positions after a turn action

```
0.000: (correct_position auv wp0)  [10.000]
10.001: (do_hover_controlled auv wp0 wp_p0)
[33.532]
43.534: (turn_valve auv wp_p0 p0 v0)  [120.000]
163.535: (correct_position auv wp_p0)  [10.000]
173.536: (turn_valve auv wp_p0 p0 v1)  [120.000]
293.537: (correct_position auv wp_p0)  [10.000]
293.537: (recalibrate_arm auv wp0)  [180.000]
473.538: (turn_valve auv wp_p0 p0 v2)  [120.000]
593.539: (correct_position auv wp_p0)  [10.000]
603.540: (turn_valve auv wp_p0 p0 v3)  [120.000]
```

Fig. 9 First part of plan for the three missions, showing the actions for the first valve-turning mission

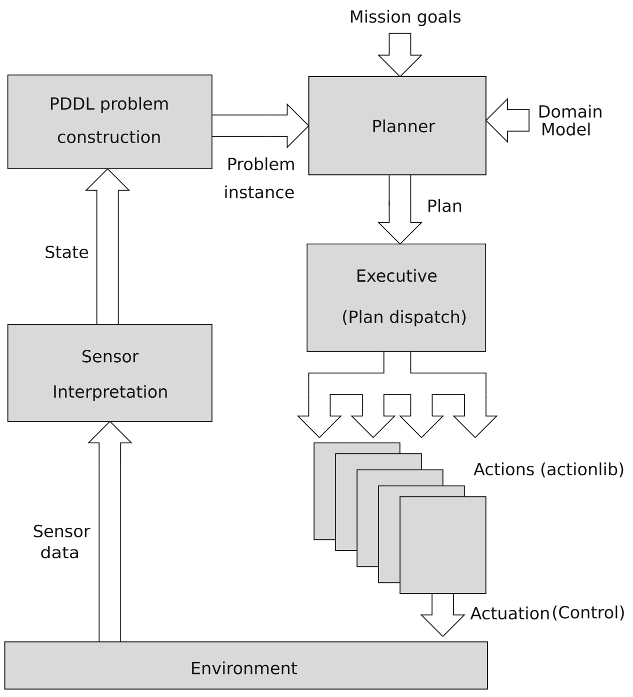


Fig. 10 General overview of the planning system framework. Sensor data is passed continuously to the planning system, and used to construct models. Actions are dispatched as ROS actions and are carried out by lower-level controllers. These controllers respond reactively to immediate events and provide feedback

structures and with different deadlines. The planner is used to decide the order in which to attempt these tasks, how the tasks are to be carried out, and within what resource and time constraints. For example, when addressing a valve-turning task as part of a longer mission, planning is used to determine how much time and resource to commit to the task, and to coordinate the search for the valve panel while respecting the duration and energy constraints of the entire mission.

The actions that are relevant to the valve-turning task are described in Sect. 6.2. In addition to determining how to achieve the task, the planning system is also responsible for deciding how to respond to execution-time failures, such as losing the panel position due to drift in the navigation, or failing to turn a valve because it is blocked. While a single valve-turning task could be attempted using a scripted controller, when considered in the context of a longer persistent mission, planning is required.

6.1 Execution-time planning system

The planning system framework shown in Fig. 10 illustrates how the planning system is supported by the other components required to enable on-line planning. The framework comprises three parts: sensor data interpretation to enable the construction of a planning problem instance describing the mission, the automated construction of a plan from the domain model and problem instance, and the dispatch of the planned actions to the controllers. When actions are executed by the controllers, errors might be encountered so that a replan is requested.

The planning domain and problem instances are modelled in the planning domain description language, PDDL2.1 (Fox and Long 2003). This language allows the definition of domains and problems with temporal and numeric relationships and constraints. PDDL2.1 is a member of the PDDL family of languages, and we refer to it throughout as PDDL. The problem instance is built dynamically, by applying a state estimation process to the sensor data. An example problem file is shown in Fig. 11. This construction is performed at the outset of the mission, and then each time a replan is requested. The problem is then solved using the planner POPF (Coles et al. 2010), using the anytime planning feature of its OPTIC variant (Benton et al. 2012). POPF is a temporal metric planner that is widely used in the AI Planning community and can cope with many features that are common in applications (Piacentini et al. 2013; Fox et al. 2011). The plan produced by POPF includes estimated durations and dispatch times for each action. During dispatch, these times can be interpreted with some flexibility depending on deadlines in the plan.

```
(define (problem valve_task)
(:domain valve_turning)
(:objects
    auv - vehicle
    wp1 ... wp20 - waypoint
    p1 - panel
    v1 - valve)
(:init
    (at auv wp1)
    (canreach auv wp8 p1)
    (on v1 p1) (valve_blocked v1)
    (= (valve_goal v1) 0.014)
    (= (valve_state v1) 0.014)
    (= (valve_goal_completed v1) 0)

    ;; VALVE TIME WINDOW
    (at 1 (not (valve_blocked v1)))
    (at 1 (valve_free v1))
    (at 1 (= (valve_goal v1) 1.57))
    (at 251 (valve_blocked v1))
    (at 251 (not (valve_free v1)))
    ...
    (connected wp1 wp2)
        (= (distance wp1 wp2) 5.072)
    (connected wp2 wp1)
        (= (distance wp2 wp1) 5.072)
    ...)
(:goal (and
    (>= (valve_goal_completed v1) 5))))
```

Fig. 11 Partial PDDL problem instance for a valve-turning task. The problem is built dynamically from the sensed environment and the mission instructions. The existence of valve panels and the angles of valves are sensed, while the desired angles for those valves are prescribed. In this problem there is a valve *v1*, which must be turned to 1.57 radians within the first 251 s of the mission

When the plan is passed to the executive for dispatch, the actions are converted into predefined action-goal messages, using the ROS actionlib library (WillowGarage 2007). A controller is responsible for achieving the actionlib goal and providing feedback. The feedback can be either *success*, in which case the executive passes the next action to the controller, or *failure*, with a request for replanning.

The idea of translating planned actions into control programs has been previously explored many times, for example (Simmons et al. 1997; Beetz et al. 2012; McGann et al. 2008). In contrast to previous approaches, we use a *temporal* planning system, and focus on the coordination of activity around the deadlines by which tasks must be achieved. Furthermore, rather than building an in-house plan execution architecture, we use ROS and open standards. We also address the problem of updating the knowledge available to the planner by the use of run-time remodelling and replanning.

There are four possible reasons for replanning:

1. *Action failure*: an action execution reports failure, using the ROS action feedback;
2. *Change of environment*: the state estimation process identifies a change in the environment that invalidates the plan, or new information pertinent to mission goals is discovered;
3. *Budget deficit*: the estimated cost (time or energy usage) differs from that measured during execution, and the executive calculates that the plan is invalidated (the measured cost was high).
4. *Budget surplus*: the estimated cost (time or energy usage) differs from that measured during execution, and the executive calculates that there is room to perform extra tasks (the measured cost was low).

The first three situations initiate a replan request to which the planner can respond by replanning the current task within global mission constraints. In situations where the plan is invalidated by the new state or resource availability, the planner must construct a new plan to work with the new conditions. In the last of the four sitiations, if the existing plan remains valid, the planner is free to leave it unchanged.

Two significant situations are relevant to the valve-turning task considered in this paper: first, when the precise location of the valve panel is unknown, the planner will construct a plan to search for the panel according to its most likely location and, second, once the valve position is known a plan is constructed to turn the valve and then to confirm that it is set to the correct angle. The first plan can lead to a replanning request being triggered if the valve panel is found during the search: this is a positive outcome, but requires a new plan to interact with the valve. The second plan can fail if the valve-turning action fails to achieve the desired angle and will lead to a new plan to attempt to interact with the valve again. If resource constraints prevent a new plan from being constructed, for the purposes of the current work the planner simply abandons this task and reverts to a safe state plan.

6.2 Planning-execution interface

An interface has been developed to link the controllers of the vehicle to the planning module. The significant details of the actions and their implementations are described below.

6.2.1 Go to waypoint

The planner is used to determine the order in which to visit the inspection locations when searching for the panel, minimizing the duration and energy cost of the mission. Once a waypoint position (x, y, z, ψ) is computed, the vehicle can be asked to move there by one of two possible modes of motion.

The first mode (which corresponds to the action do_hover _controlled) moves the AUV holonomically by sending waypoint requests to the pose controller (see Sect. 4). However, if the waypoint to be reached is far from the current position of the vehicle, this motion mode might be too slow to be appropriate. In this case a variation of the line-of-sight (LOS) pure pursuit guidance algorithm described in Fossen (2011) may be used. This is the do_hover_fast action. The orientation error (ψ_e) between the current pose aof the vehicle nd the desired waypoint is computed (39) and sent to the pose controller together with the desired depth (z_d).

$$\begin{aligned} \Delta_x &= x_d(t) - x(t), \\ \Delta_y &= y_d(t) - y(t), \\ \psi_e(t) &= atan2(\Delta_y, \Delta_x). \end{aligned} \tag{39}$$

When, $\psi_e(t)$ is smaller than a defined error ($angle\_error$), the desired surge (u_d) is computed and sent to the velocity controller following:

$$u_d(t) = min\left(\frac{\sqrt{\Delta_x^2 + \Delta_y^2}}{approach\_factor}, 1\right) \cdot \left(1 - \frac{\mid \psi_e(t) \mid}{angle\_error}\right) \cdot max\_surge, \tag{40}$$

where $angle\_error$, $approach\_factor$ and max_surge are user-defined constants. In our case, they take the following values based on our experience: 0.3 rad, 4 and 0.6 m/s, respectively. It is worth noting that using this second motion mode, the final position of the vehicle is subject to a larger error than using the first mode and also, the final orientation of the AUV, ψ, does not correspond to the desired orientation, ψ_d, at the waypoint. The planner determines which of the holonomic and the LOS motion modes to apply in any situation. The PDDL representation of these actions is shown in Fig. 12.

```
(:durative-action do_hover_controlled
    :parameters
     (?v - vehicle ?from ?to - waypoint)
    :duration
     ( = ?duration (* (distance ?from ?to) 10))
    :condition (and
        (at start (at ?v ?from))
        (at start (connected ?from ?to)))
    :effect (and
        (at start (not (at ?v ?from)))
        (at end (at ?v ?to)))
)

(:durative-action do_hover_fast
    :parameters
     (?v - vehicle ?from ?to - waypoint)
    :duration
     ( = ?duration (* (distance ?from ?to) 5))
    :condition (and
        (at start (at ?v ?from))
        (at start (connected ?from ?to)))
    :effect (and
        (at start (not (at ?v ?from)))
        (at end (near ?v ?to)))
)
```

Fig. 12 PDDL representation of the *Go to waypoint* control. The two operators require that there is a connection—a collision free path—between the two waypoints, and have different durations. In addition, the fast motion moves the vehicle *near* the destination, requiring the position to be corrected before continuing with the mission

```
(:durative-action observe_inspection_point
 :parameters
  (?v - vehicle ?wp - waypoint
   ?ip - inspectionpoint)
 :duration ( = ?duration 10)
 :condition (and
  (at start (at ?v ?wp))
  (at start (cansee ?v ?wp ?ip)))
 :effect (and
  (at start (not (cansee ?v ?wp ?ip)))
  (at end (increase (observed ?ip)(obs ?ip ?wp)))
  (at start (not (at ?v ?wp)))
  (at end (near ?v ?wp)))
)
```

Fig. 13 The observe_inspection_point action

6.2.2 *Observe inspection point*

When this action is used, the vision-based system described in Sect. 3.2 must identify whether the inspection panel is within the field of view of the camera. Thus, the action succeeds if the panel is identified and fails otherwise. The PDDL representation is shown in Fig. 13.

6.2.3 *Examine panel*

The geometry of the panel is known at the point of invocation of this action. Therefore, the positions of the valves within the panel are also known. A region of interest (RoI) centered at each valve is extracted. The Hough line transform is used to detect the orientation of the principal line in this RoI. Outliers are limited by constraining the length of the lines as well as possible orientations. This action returns the orientation for each valve or *failure* if the panel is out of the field of view of the vehicle.

6.2.4 *Turn valve*

The controller for turning a valve has been detailed in Sect. 5. It defines the whole intervention with respect to the valve of interest. However, the rotation of the valve is not learned by the LbD algorithm but executed by a simple additional controller. If the task cannot be completed before a timeout or the rotation controller detects that the valve is blocked (i.e., using the Force/Torque sensor) a failure is generated as feedback. However, if the task finalizes with a success this is not sufficient to ensure that the correct valve has been properly turned. It is the responsibility of the planner to examine the panel after each intervention to check its current state. It is also important to note that the turn_valve action, shown in

```
(:durative-action turn_valve
    :parameters
     (?v - vehicle ?wp - waypoint
      ?p - panel ?a - valve)
    :duration ( = ?duration 120)
    :condition (and
        (at start (at ?v ?wp))
        (at start (examined ?p))
        (at start (canreach ?v ?wp ?p))
        (at start (on ?a ?p))
        (at start
              (<= (manipulator_calibration ?v) 1))
        (at start (valve_free ?a)))
    :effect (and
        (at start (not (valve_free ?a)))
        (at end
         (assign (valve_state ?a) (valve_goal ?a)))
        (at start (not (at ?v ?wp)))
        (at end (near ?v ?wp))
        (at end
         (increase (valve_goal_completed ?a) 1))
        (at start (not (examined ?p)))
        (at end (increase
             (manipulator_calibration ?v) 1))
        (at end (valve_blocked ?a)))
)
```

Fig. 14 PDDL representation of the turn_valve action

Fig. 14, can be aborted by the planner at any time. This allows the planner to avoid depleting resources on a repeatedly failing task, beyond the time allowed for the completion of that task.

6.3 Mission plan execution example

Consider a mission involving three tasks: two valve-turning tasks ($V1$ and $V2$) and an inspection task (I). The valve-turning tasks are at separate panels, with deadlines $t(V1)$ and $t(V2)$. The inspection task does not have a deadline, and requires the AUV to visit points around the entire structure. The vehicle begins the mission at the panel to be manipulated for valves in $V1$.

A plan for this mission is to turn the valves for $V1$, perform the inspection mission en-route to the second panel, and then complete $V2$. The timeline for this mission is shown in Fig. 16. The initial plan for turning the valves for $V1$ is shown in Fig. 9. An example output from the dispatcher during execution of this mission is shown in Fig. 15.

The first turn_valve action failed to complete (Fig. 15, line 3–4). This is shown by the red box in Fig. 16. If the

```
 1.Dispatching action [46, turn_valve, 815.759...
 2.Feedback received [46,action enabled]
 3.Feedback received [46,action failed]
 4.Valve 0 detected as blocked 1 time(s)
 5.Generating PDDL problem file
 6.Run: timeout 10 run planning_system popf...
 7.Planning complete
 8.Dispatching action [46, turn_valve, 815.759...
 9.Action [46] is 100.476033 second(s) late
10.Feedback received [46,action enabled]
11.Feedback received [46,action failed]
12.Valve 0 detected as blocked 2 time(s)
```

Fig. 15 Part of the plan dispatch log

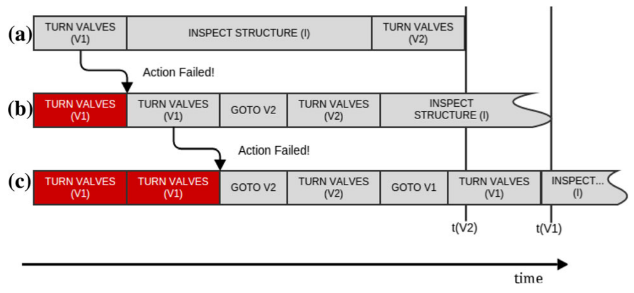


Fig. 16 Execution trace with repairs, for part of a mission

action is immediately repeated, then task $V2$ can no longer be completed before $t(V2)$. The planning system generates a new plan (Fig. 15, lines 5–7) that can be seen in the second row (B) of Fig. 16.

The inspection task is moved later in the timeline, as it does not have a deadline. The turn_valve action is immediately retried (Fig. 15, lines 8–10), as can be seen in Fig. 16, and the new plan still achieves task $V2$ before $t(V2)$.

The turn_valve action fails a second time (Fig. 15, lines 11–12). This time it is not possible to retry the action and still accomplish task $V2$ before $t(V2)$. The planning system generates a new plan, shown in the third row (C) of Fig. 16. The valve-turning tasks are re-ordered and the inspection task pushed even later. Following this plan will still accomplish both tasks $V1$ and $V2$ before their respective deadlines.

From this example it can be seen that, when taken in the context of a longer mission, the reaction to failure, or to new discoveries, requires reasoning about how the surrounding tasks can best be organized in the construction of the current repair, and how long those tasks might take to complete. Moreover, the durations of these tasks depend upon the order in which they are attempted. For example, in Fig. 16 it can be seen that moving the inspection mission necessitated the introduction of new movement actions to go to the site of $V2$ and then on to the site of $V1$. A planner is an ideal tool for this reasoning.

7 Experiments and results

In this article we have presented a framework to perform persistent autonomous interventions in a subsea facility with an underwater vehicle. In order to evaluate it, we have defined a long-term mission that has been carried out in a water tank of $16 \times 8 \times 5$ meters using the Girona 500 I-AUV. A mock-up panel of 0.8×0.5 meters with 4 manipulable valves has been used to emulate a subsea panel from an offshore facility. To make the environment more challenging, two Seaeye MCT1 thrusters able to generate up to 14 Kg of thrust each, have been placed close to the panel in order to generate water currents when enabled (see Fig. 17). The experiments were

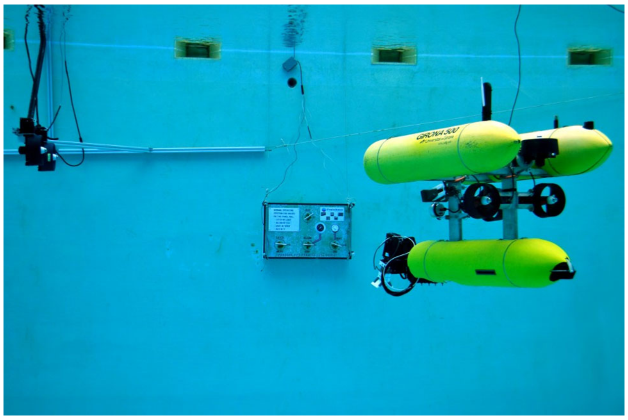


Fig. 17 Girona 500 I-AUV in the water tank with the mock-up subsea panel in the background and the water current system at its left

performed in a completely autonomous mode (i.e., the vehicle ran on its own batteries and all the required processing was performed with the on-board computers). However, there was a connection through an umbilical cable to a base station for monitoring and safety purposes during the execution of the mission. The same connection was used for conducting the demonstrations of the turn valve LbD action.

7.1 Mission definition

One of the main advantages of using a planning system lies in the simplicity of defining a new mission. In our case, the mission is defined by describing the location of several inspection points where the panel can be and the desired configurations for each valve, each configuration corresponding to a specified time interval (see Fig. 18).

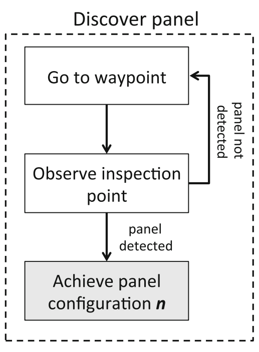
As described previously, the planner decides how best to act in the event of failure, by replanning the mission from a revised initial state. A plan is built on the assumption that the world will behave as expected. However, many things can go wrong. Some of the issues that can occur are detailed next:

- *Turn action failure*: Although the control and perception systems have been programmed to ensure a high degree of robustness, failure in grasping or turning a valve can still occur due to strong external perturbations, an incorrect manipulator calibration, problems in detection of the valve orientation, etc. In those cases, it is the responsibility of the planner to ensure that the accomplishment of the `turn_valve` action is checked (by scheduling an `examine_panel` action just after). If the valve is not in the expected position a replan will be triggered, otherwise, the plan will continue.
- *Blocked Valve*: A common problem that we might encounter while operating a subsea panel is that valves can get stuck because of accumulated biofouling or corrosion. In this situation, the vehicle must avoid applying an excessive torque as this could result in breaking the valve. Moreover, the blockage should be recorded thus acknowledging that the desired configuration will never be achieved in the current mission. The execution of the `turn_valve` action is able to detect this situation thanks to the F/T sensor and notify it to the planning system. The task is retried at least once, to avoid false detections, but if the valve appears again as blocked the executive will report this problem, the blockage will be recorded in the knowledge base and will appear in the revised world model used by the planner.
- *Panel not in previously discovered place*: Initially, the vehicle knows possible locations in which the panel can be but not its actual position. After an inspection procedure, if the panel is discovered, the vehicle maps the panel location with respect to its reference frame. Because the AUV localization can drift when the panel is not in its field of view, it is possible that between two interventions estimation of the vehicle position becomes less accurate. From the AUV point of view this localization issue is interpreted as a change in the position of the panel. In this event, the panel location is deleted from the planner model and the inspection procedure is started again. For safety reasons, in order to avoid a potential risk of collision of the vehicle due to the limited space of the environment, the previous location of the panel is also removed from the localization filter to avoid large position corrections.
- *Manipulator calibration problem*: As described before, the calibration of the manipulator encoders can lose accuracy along the different interventions. To avoid problems when turning a valve due to miscalibration, the arm calibration procedure is periodically run during the mission.

```
inspection points [[-3.5, 0.0, 2.0, 3.141],
                   [3.5, 0.0, 2.0, 0.0]]
valve goal angle [0, 12:00, 30, 1.57]
valve goal angle [1, 12:00, 30, -1.57]
valve goal angle [0, 12:20, 15, 0.0]
```

Fig. 18 Mission definition. *Inspection points* contains a list of places (x, y, z, ψ) in which the panel can be located and *valve goal angle* specifies a valve id, a time interval (start time and interval in minutes), and the desired angle

Although these are the most probable failures, there are other circumstances that might not be contemplated here that can also prevent the defined mission goals from being achieved. In these cases, the planner will attempt to find a solution—normally retrying the failed action—or, in the worst case, aborting the whole mission. Therefore, although the mission that has been defined contains only the possible panel locations and the state of each valve in a time period, it is automatically modified online according to the errors



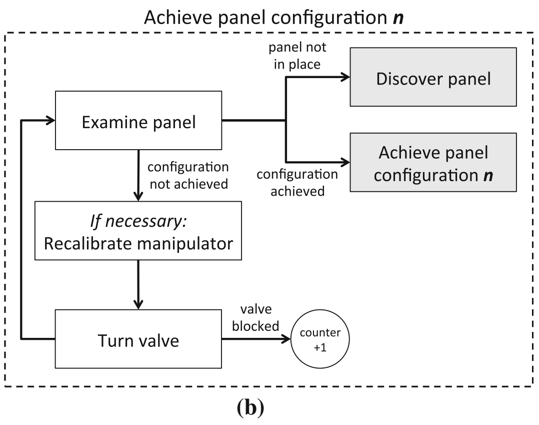

Achieve panel configuration *n*
panel not in place
Discover panel
Examine panel
configuration not achieved
configuration achieved
Achieve panel configuration *n*
If necessary: Recalibrate manipulator
Turn valve
valve blocked
counter +1
(b)

Fig. 19 Normal planning actions execution to **a** discover the valve panel location and **b** achieve a specific panel configuration

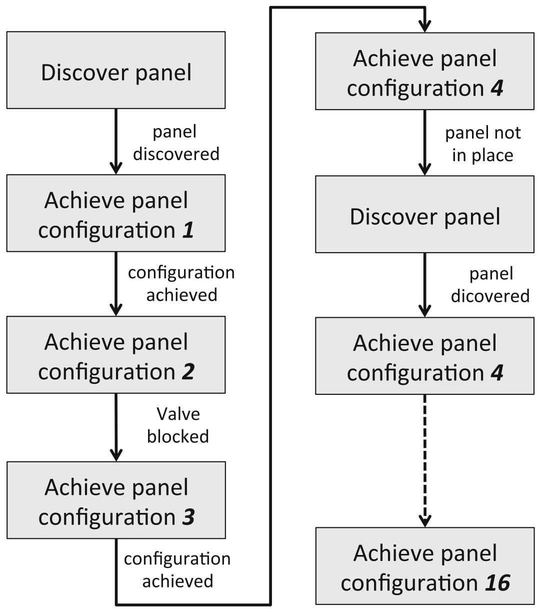

Discover panel
panel discovered
Achieve panel configuration *1*
configuration achieved
Achieve panel configuration *2*
Valve blocked
Achieve panel configuration *3*
configuration achieved
Achieve panel configuration *4*
panel not in place
Discover panel
panel dicovered
Achieve panel configuration *4*
Achieve panel configuration *16*

Fig. 20 Portion of the persistent plan created by the on-board planner while running the intervention mission

(either spontaneous or induced for the sake of the demonstration) that the vehicle faces.

To understand the mission flow we have grouped the planning actions into two groups. The first one, Fig. 19a, gathers the actions executed to discover the panel location and the second, Fig. 19b, involves the mechanism to turn valves in a panel until a desired configuration is reached. An example of the long term mission plan created online by the planning system is presented in Fig. 20 using the previously introduced blocks. The vehicle starts looking for the panel and once this is localized it tries to achieve the different valve configurations defined in the mission plan during the different time slots. It is worth noting that while some failures in the `turn_valve` action occurred spontaneously during the mission execution, additional errors were intentionally introduced by moving the panel from its location and manually blocking and unblocking some valves.

7.2 Results

The results obtained with the proposed framework are summarized here. Because each module (i.e., control, learning, and planning) builds on the previous modules, we follow the same incremental approach to show the results.

7.2.1 Control results

Even though the performance of the proposed control scheme is already verified in Sect. 4 following standard elements of Lyapunov theory, it has been also evaluated in the context of a valve-turning task. In this experiment the LbD task was guiding the vehicle by means of velocity requests in $(u,\ v,\ w,\ r)$ while a second task was sending a pose request in (θ) to keep the vehicle's pitch constant at $-2°$. Moreover, a water current was generated using the two external thrusters placed at 2.0 meters from the panel and running at 35% of their power ($\sim$10 kg thrust). Velocity controller outputs are shown, in blue, in Figs. 21 and 22, proving that the vehicle accurately follows the desired velocities demanded by the LbD task (shown in red). Similarly, Fig. 23 shows the stabilization of the pitch DoF acheived by the pose controller. The fact that both pose and velocity are not directly measured from a sensor but estimated by the filter presented in Sect. 3 provides an smoother feedback to the controllers thus enhancing their overall performance.

7.2.2 Learning by demonstration results

To evaluate the performance of the LbD algorithm, we compare the task demonstrations performed by a human operator against the autonomous reproduction of the task. Figure 24

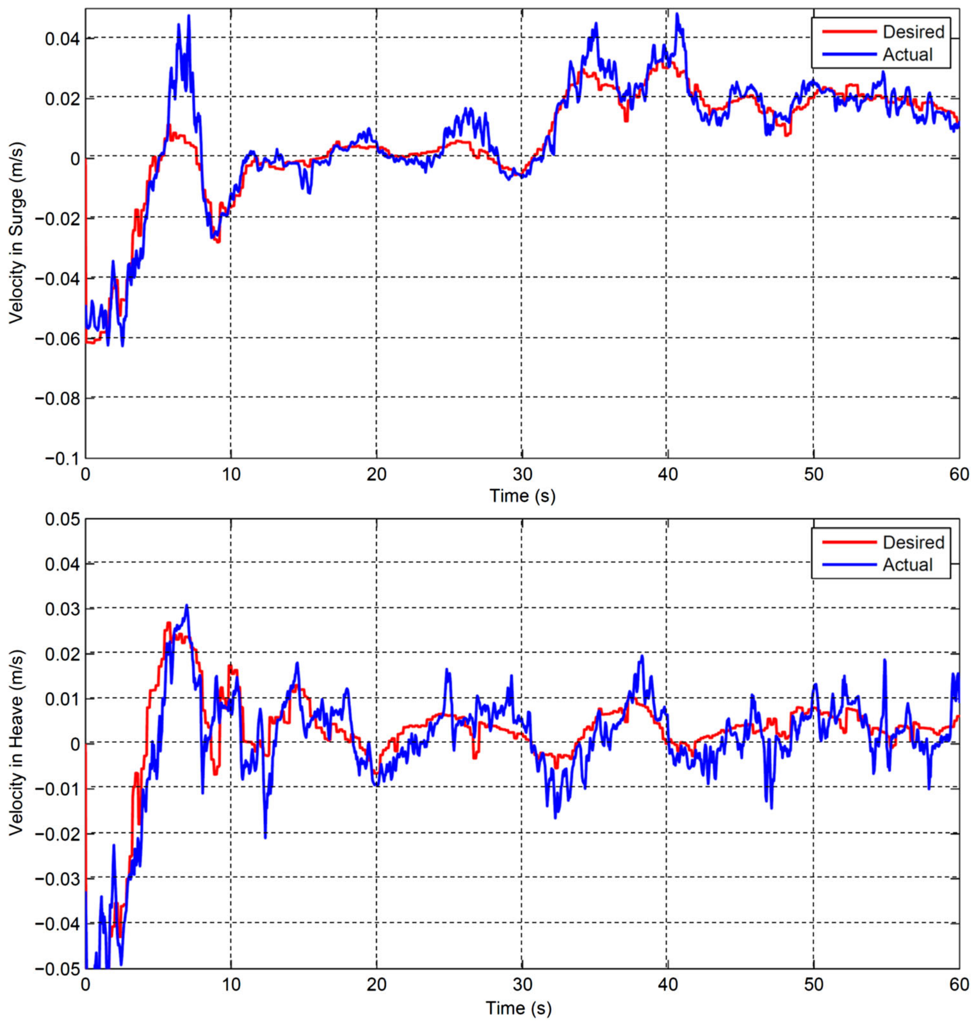


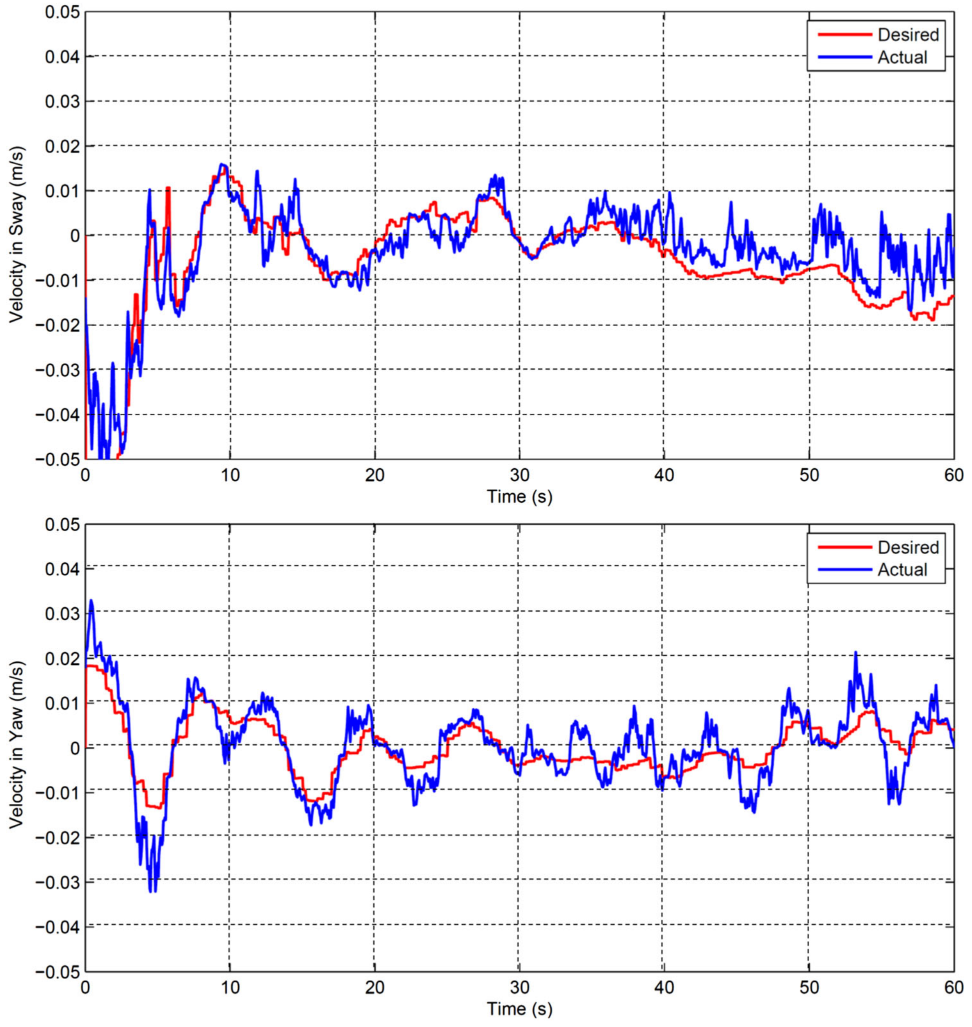
Fig. 21 Vehicle response when following the velocity requests in Surge and Heave generated by the LbD valve-turning task (Color figure online)

shows in blue the demonstrated trajectories for both the AUV and the end-effector. These trajectories have been recorded while teleoperating the vehicle and the manipulator with the haptic device to perform a valve grasping. The number of Gaussians used by the DMP algorithm to describe the model has been empirically set to 20 to ensure an accurate trajectory following given the characteristics of the learned trajectory and the manipulator workspace limitations. K^P_{min} and K^p_{max} have been also adjusted to obtain K^V and K^P values that balance the smoothness of the velocity commands and its stiffness to ensure that the desired trajectory is followed even in the presence of currents. Once the DMP model has been computed, we have used it to reproduce the learned task. Figure 24 shows in red the trajectories of the autonomous reproduction with a plot for each DoF of the AUV and the end-effector. We can observe that both trajectories (AUV and end-effector) are most of the time within the average of all the demonstrations, specially at the end of each execution, where the trajectory converges to the valve in order to grasp it and turn it. When reproducing the learned task in a system without currents, 13 out of 16 reproductions were successful in turning the valve (success rate of 81%).

Table 2 shows the accuracy of the final position computed as the mean error of all the reproduced trajectories with respect to the average demonstration trajectory. Tolerance values for the end-effector position when grasping the valve are estimated based on the mechanical design and the compliance of the gripper. Notice that even though the manipulator has reduced maneuverability and must cope with the underlying errors of the AUV pose, the accuracy achieved by the end-effector is high enough to be within the gripper tolerance and thus succeed to perform the valve-turning most of the times.

When testing the LbD task under the effect of currents, the success rate decreases to 69 % mostly due to the perturbation introduced in the vehicle's *yaw* when the I-AUV is close to the grasping point.

Fig. 22 Vehicle response when following the velocity requests in Sway and Yaw generated by the LbD valve-turning task (Color figure online)



7.2.3 *Planning results*

We ran experiments to test the robustness of the planning method, and the ability of the planner to coordinate its activity in the presence of deadlines. In the mission shown in Fig. 25a, the `turn_valve` action was executed and returned failure; the valve was detected as blocked. The planning system replanned, and retried the turning action. The second plan executed correctly, and both valves were turned before the deadline. In contrast, in Fig. 25b, the deadline of both valves is set to be at the same time, but is closer (in time) to the point at which the initial plan is expected to complete. When the first valve-turning action fails (for `valve_0`) there is not enough time left to turn both valves before the deadline. In this case the planning system does not retry the failed action, but instead moves straight to turning `valve_1`. If it were the case that turning valve_1 depended on having successfully completed the turn action on valve_0, this requirement would be expressed in the precondition of the relevant action, and the planner would be forced to abandon both actions once the available time to complete them had become too short.

Table 3 shows fragments of the plans generated during preliminary valve-turning missions. The table shows the PDDL actions and their corresponding ROS action messages. The `observe` PDDL action does not have a corresponding ROS action, as the visual sensing is continuous and passive. The `observe` action merely causes the AUV to wait for several seconds after orienting itself towards the inspection point—this aids the visual detection.

In this example, replanning was performed for three reasons: when the panel was discovered, when a `turn_valve` action failed, and finally when the panel position was lost due to drift. In each case the planning system reformulated the problem, building a new initial state from the sensed data and mission parameters. In this way, the system was robust to the failures, and the mission was accomplished.

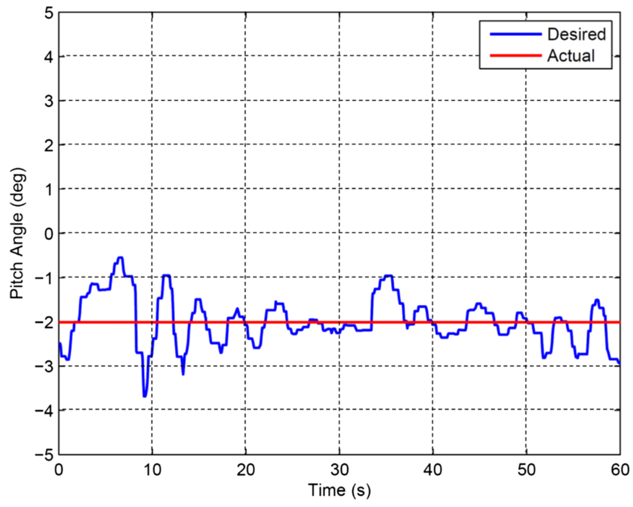


Fig. 23 Pose controller stabilizing the pitch angle to −2° while the LbD valve-turning task is under execution (Color figure online)

7.2.4 Long-term experiment results

We have designed a long experiment in which the vehicle must locate the intervention panel among four different possible locations and set 16 different panel configurations along the mission time. The vehicle can take up to 15 min to achieve a desired configuration, but every 10 min a new configuration is requested. An extract of the mission definition can be seen in Fig. 26.

Several errors have been induced along the mission to observe the response of the system. At the beginning of the mission valve 2 has been blocked. We have moved the panel's location (by removing it from the water tank and setting it to a different place) 3 times, just before configurations 4, 8 and 12. The first time that the panel has been moved, valve 2 has been unblocked, the second time valve 0 has been blocked, and the last time valve 0 has been unblocked. A small current (below 35 % of thrust power) has been applied during half of the experiment. However, while the vehicle was attempting the 6th configuration a higher current (60 % of thrusters power) has been applied.

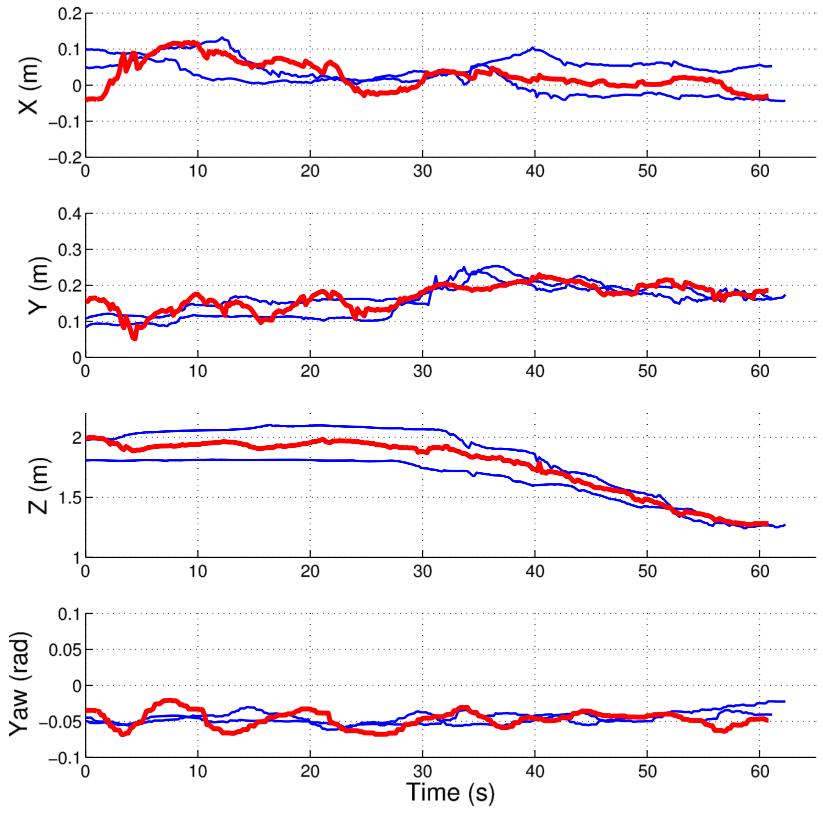
Table 4 summarizes the main results obtained during the long execution. 10 out of the 16 panel configurations have been correctly achieved. From the remaining 6, 4 have been correctly handled by the planning system, by reporting the appropriate errors. Configurations 2 and 9 have not been completed because the valve to be actuated was detected as blocked. Configuration number 6 has not been accomplished during its time slot and has been aborted by the planner due to the difficulties of the vehicle to correctly approach the panel under the strong water currents. Configuration 12 has been as well aborted given that the vehicle had to relocate again the panel and once it was located, the turn_valve action failed twice, thus running out of time to perform the corresponding turns. Besides, due to imprecisions in the vision-based algorithm used by the examine_panel action, the I-AUV has left two valves at incorrect angles, having determined that they were at the correct angles, and has therefore reported a false success. Finally, another failure took place during the panel localization before configuration number 8. When the panel was moved to a new location, the vehicle was unable to detect it from any of the predefined inspection points due to the accumulated drift in the vehicle's position while looking for the panel. Then, the planner triggered a new inspection cycle and we slightly moved the panel again to force its appearance in the camera's field of view.

In addition to Table 4 that summarizes the mission results, we have included the main parameters used during the experiment in Table 5.

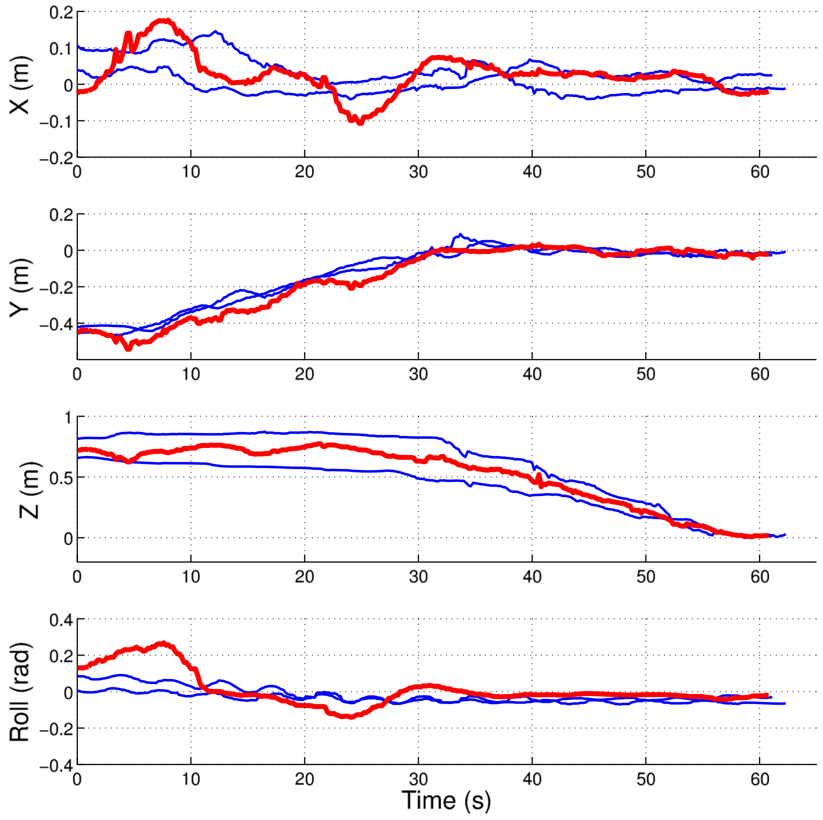
8 Conclusions and lessons learned

This article has described a multidisciplinary system to target the challenging task of persistent intervention in a subsea panel. Four state-of-the-art techniques in the fields of localization, control, machine learning and mission planning have been combined to operate in the always complex underwater environment. Although similar techniques have been used individually in other research contexts, this paper constitutes a system integration effort towards building a framework to perform a challenging task in a persistent way, seeking robustness at all levels. This has led to an experimental demonstration in which an I-AUV has remained for a few hours performing autonomous intervention in free-floating mode, being, to the best of the authors' knowledge, the first demonstration of this kind. Thus, we believe that our main contribution is to report the lessons learned during the development of this framework as well as the future areas for improvement.

As it has been shown throughout the paper, implementing a complex system like the one proposed here requires each step to be consolidated before building up the next one. In this sense, we believe that a big part of the encountered problems are rooted down to the mechatronics, in particular to the employed manipulator. The fact that the manipulator is underactuated, has a very limited workspace and does not provide absolute orientation in the joint sensors, brings in a number of issues that affect also the rest of the system. The inclusion of a *Recalibrate manipulator* action in the planner has mitigated the calibration problems, but the limited workspace or the low speed of the manipulator still complicate the intervention task. Unfortunately, current commercial electrical manipulators for small AUVs are still in their early days and market options are scarce. In this regard, we envision that manipulator limitations will be the Achilles heel



(a)



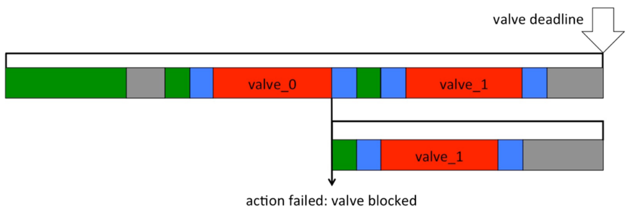
(b)

Fig. 24 Trajectory of the AUV (**a**) and trajectory of the end-effector (**b**) during 2 demonstrations and 1 reproduction. The trajectories are represented in the valve frame and each controlled DoF is shown in one plot. The demonstrations are depicted in *blue* lines and the reproduction in *bold red* (Color figure online)

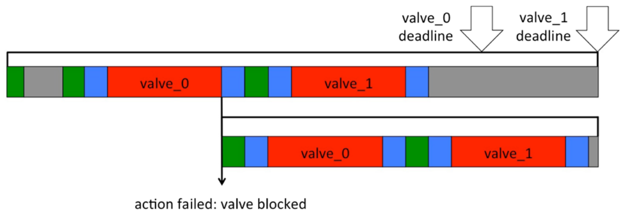
Table 2 Valve grasping accuracy computed as the mean error of the reproduced trajectories with respect the average of the demonstration trajectories for each DoF

	Accuracy	Tolerance
AUV - X (m)	0.021 ± 0.018	
AUV - Y (m)	0.025 ± 0.019	
AUV - Z (m)	0.056 ± 0.038	
AUV - Yaw (rad)	0.016 ± 0.015	
End-effector - X (m)	0.018 ± 0.013	0.04
End-effector - Y (m)	0.016 ± 0.013	0.04
End-effector - Z (m)	0.045 ± 0.037	0.08
End-effector - Roll (rad)	0.15 ± 0.045	0.2

Rightmost column shows estimated tolerances of the end-effector to correctly grasp a valve



(a)



(b)

Fig. 25 Timeline showing a sequence of planned actions. *Green blocks* are *Go to waypoint* actions, *blue boxes* are examine_panel actions, and *red boxes* are turn_valve actions. The timeline shows a new row where execution of the first plan fails, and a new plan is constructed. In this plan, valve_0 was detected as *blocked*. In **a** a new plan was constructed, which included retrying the turn_valve action while in **b** a new plan was constructed, in which valve 0 was ignored and only valve 1 was turned (Color figure online)

Table 3 Fragments of two PDDL plans produced during a valve-turning mission

PDDL action	ROS action message
0.000: (observe_inspection_point auv wp1 ip3) [10.000]	–
10.001: (correct_position auv wp1) [10.000]	Go to waypoint
20.002: (do_hover_fast auv wp1 wp2) [35.848]	Go to waypoint
55.851: (correct_position auv wp2) [10.000]	Go to waypoint
65.852: (observe_inspection_point auv wp2 ip4) [10.000]	–
75.853: (correct_position auv wp2) [10.000]	Go to waypoint
85.854: (do_hover_controlled auv wp2 wp23) [16.710]	Go to waypoint
. . .	. . .
423.199: (do_hover_fast auv wp2 wp36) [10.726]	Go to waypoint
433.926: (correct_position auv wp36) [10.000]	Go to waypoint
443.927: (observe_inspection_point auv wp36 ip9) [10.000]	–
0.000: (turn_valve auv wp1 p0 v1) [100.000]	Turn valve
100.001: (correct_position auv wp1) [10.000]	Go to waypoint
110.002: (turn_valve auv wp1 p0 v3) [100.000]	Turn valve
210.003: (correct_position auv wp1) [10.000]	Go to waypoint
220.004: (turn_valve auv wp1 p1 v2) [100.000]	Turn valve
. . .	. . .
1310.002: (correct_position auv wp1) [10.000]	Go to waypoint
1311.003: (turn_valve auv wp1 02 v4) [100.000]	Turn valve

The first plan fragment shows the beginning and end of an inspection mission, searching for the valve panel. The second fragment shows the plan to correct the valves, once the panel has been found. The expected dispatch time in seconds for the actions (the first number) is relative to the start of the current plan

of any autonomous underwater intervention system until the market evolves and better options become available.

Similarly, navigation capabilities are crucial as both the control and the LbD tasks rely heavily on it. The combination of a good navigation sensor suite together with a SLAM system to be able to remain drift-free when the intervention panel is in the field of view is paramount. A fine-tuning of all the covariances used in the localization filter as well as a correct measurement of the transformations between the sensors and the vehicle frame—specially for the camera—was capital to obtain accurate results in this aspect. With that said, it is also worth noting that the water tank in which the experiments were performed was not the best environment for the navigation sensors (i.e., compass and DVL) due to the presence of concrete walls that can induce magnetic disturbances and undesired acoustic artifacts.

An equally important factor for the system's success is having a well adjusted controller. The time spent to identify the vehicle's dynamic model as well as to adjust the velocity and the pose controllers has been one of our best investments. Initially, we designed a robust model-based velocity control scheme for all actuated degrees of freedom of the vehi-

```
 inspection points [[−3.5, −1.5, 2.0, 3.1416],
                     [3.5, 1.5, 2.0, 0.0],
                     [−3.5, 1.5, 2.0, 3.1416],
                     [3.5, −1.5, 2.0, 0.0]]
% Configuration 1
 valve goal angle [0, 12:00, 15, 1.57]
% Configuration 2
 valve goal angle [2, 12:10, 15, 1.57]
% Configuration 3
 valve goal angle [0, 12:20, 15, 0.0]
 ...
% Configuration 16
 valve goal angle [1, 14:30, 15, 0.0]
 valve goal angle [3, 14:30, 15, 0.0]
```

Fig. 26 Long term mission goals definition

Table 4 Long term intervention experiment summary

Mission time	2 h 37 min
Attempted configurations	16
Achieved configurations	10
Failed configurations	6
... due to valve blocked	2
... due to timeout	2
... due to false positive	2
Attempted *Turn valve* actions	33
... action sucess rate	72 %

Table 5 Parameters summary

Sensors parameters
DVL std.: 0.002 m/s wrt. bottom and 0.02 m/s wrt. water
AHRS std.: 0.02 rad (without magnetic perturbations)
Depth sensor std.: 0.1 m
Camera parameters: Monocular color 1024 × 768p × 4 fps, field of view in water 70°
User parameters
Navigation filter noise matrix: $\mathbf{Q} = [\mathbf{I}_{3\times3}] \times 0.02$
Identified model parameters: see Karras et al. (2013)
LbD Number of gaussians: 11
LbD Stiffness and damping: $K^P_{\min} = 0.5$, $K^P_{\max} = 2.5$
Planner: POPF Coles et al. (2010)
Planning time: 10 s

cle (surge, sway, heave, pitch and yaw). However, because small movements along the vehicle's Y axis produced large movements at the end-effector when the manipulator was extended, we decided to apply the same scheme of the velocity controller and create a pose controller that improves pitch stability. Hence, in the execution of the *Turn valve* action, the velocity controller is employed to realize the velocity requests dictated by the LbD task while the pose controller is employed to have a stable pitch, which is a key factor to perform successful manipulation interventions.

On the other hand, although we have only targeted valve-turning operations, it is worth underlying that the use of a LbD technique allows us to easily reuse the framework for other applications. In this way, future missions comprising different types of interventions (e.g., press a switch, plug a connector, place an object inside a canister, etc.) can be easily implemented, only requiring new demonstrations from an expert operator. However, it would be useful to explore further techniques to aid in the automatic determination of the proper number of Gaussians required to encode a task as well as the more suitable stiffness and damping parameters.

The use of a planning system has proved essential both in planning and replanning complex valve-turning missions with deadlines. The ability to replan online allows us to envisage a different operation paradigm where the vehicle prioritizes goals and coordinates its behaviour autonomously. Furthermore, it is possible to consider taking advantage of the existing bidirectional acoustic modem technologies to allow new goals and mission specifications to be injected during long-term operations.

Overall, we believe that the implemented framework and the experimental scenario presented in this work contribute toward pushing the envelope of persistent autonomy into a new level where missions are more complex and failures more likely, going beyond simple survey capabilities. The positive results encourage the use, in a short-term future, of autonomous robots operating in subsea facilities, performing interventions with reduced costs than teleoperated vehicles. However, there are still fundamental requirements to be addressed such as leveling the capabilities of underwater electrical manipulator's with those of their above-water counterparts. Likewise, there is also room for improvement in each of the integrated disciplines, refining automatisms and adding redundancy mechanisms to enhance robustness in case of unexpected failures that could compromise the operation and/or the safety of the system.

Acknowledgments This work has been supported by the FP7-ICT-2011-7 project PANDORA-Persistent Autonomy through Learning, Adaptation, Observation and Re-planning (Ref. 288273) funded by the European Commission.

References

Ahmadzadeh, S. R., Kormushev, P., Jamisola, R. S., & Caldwell, D. G. (2014). Learning reactive robot behavior for autonomous valve turning. In *2014 IEEE International Conference on Humanoid Robots (Humanoids)*. Madrid: IEEE.

Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, *110*(3), 346–359. doi:10.1016/j.cviu.2007.09.014.

Beetz, M., Jain, D., Mösenlechner, L., Tenorth, M., Kunze, L., Blodow, N., et al. (2012). Cognition-enabled autonomous robot control for the realization of home chore task intelligence. *Proceedings of the IEEE*, *100*(8), 2454–2471.

Benton, J., Coles, A., & Coles, A. (2012). Temporal planning with preferences and time-dependent continuous costs. In *Proceedings of the international conference on automated planning and scheduling (ICAPS)*.

Calinon, S., D'halluin, F., Sauser, E. L., Caldwell, D. G., & Billard, A. (2010). Learning and reproduction of gestures by imitation. *IEEE Robotics and Automation Magazine*, *17*(2), 44–54.

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). BRIEF: Binary robust independent elementary features. In *Proceedings of the 11th European conference on computer vision: Part IV, ECCV'10* (pp. 778–792). Berlin, Heidelberg: Springer-Verlag. http://dl.acm.org/citation.cfm?.id=1888089.1888148.

Camilli, R., Reddy, C. M., Yoerger, D. R., Van Mooy, B. A., Jakuba, M. V., Kinsey, J. C., et al. (2010). Tracking hydrocarbon plume transport and biodegradation at Deepwater Horizon. *Science*, *330*(6001), 201–204.

Caress, D. W., Thomas, H., Kirkwood, W. J., McEwen, R., Henthorn, R., Clague, D. A., Paull, C. K., Paduan, J., & Maier, K. L. (2008). High-resolution multibeam, sidescan, and subbottom surveys using the MBARI AUV D. Allan B. *Marine habitat mapping technology for Alaska* pp. 47–69.

Cashmore, M., Fox, M., Larkworthy, T., Long, D., & Magazzeni, D. (2013). Planning inspection tasks for AUVs. In *Proceedings of the MTS/IEEE oceans 2013 conference, San Diego (OCEANS'13)*.

Cashmore, M., Fox, M., Larkworthy, T., Long, D., & Magazzeni, D. (2014). AUV mission control via temporal planning. In *IEEE international conference on robotics and automation (ICRA'14)*.

Coles, A., Coles, A., Fox, M., & Long, D. (2010). Forward-chaining partial-order planning. In *Proceedings of the 20th international conference on automated planning and scheduling (ICAPS'10)* (pp. 42–49).

Englot, B., & Hover, F. (2010). Inspection planning for sensor coverage of 3D marine structures. In *IEEE/RSJ international conference on intelligent robots and systems*.

Evans, J., Petillot, Y., Redmond, P., Wilson, M., & Lane, D. (2003). AUTOTRACKER: AUV embedded control architecture for autonomous pipeline and cable tracking. In *MTS/IEEE oceans* (pp. 2651–2658). San Diego, CA.

Fossen, T. (1994). *Guidance and control of ocean vehicles*. New York: Wiley.

Fossen, T. I. (1994). *Guidance and control of ocean vehicles*. Chichester, UK: Wiley.

Fossen, T. I. (2011). *Handbook of marine craft hydrodynamics and motion control*. New York: Wiley.

Fox, M., & Long, D. (2003). PDDL2.1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Researsh (JAIR)*, *20*, 61–124.

Fox, M., Long, D., & Magazzeni, D. (2011). Automatic construction of efficient multiple battery usage policies. In *Proceedings of the 21st international conference on automated planning and scheduling (ICAPS)*.

Fox, M., Long, D., & Magazzeni, D. (2012). Plan-Based policy-learning for autonomous feature tracking. In *Proceedings of the 22nd international conference on automated planning and scheduling (ICAPS)*.

Ghallab, M., Nau, D., & Traverso, P. (2004). *Automated planning: Theory and practice*. San Francisco, CA: Morgan Kaufmann.

Gracias, N., Bosch, J., & Karim, E. M. (2015). Pose estimation for underwater vehicles using light beacons. In *Proceedings of the IFAC workshop on navigation, guidance and control of underwater vehicles, IFAC NGCUV 2015*. Spain: Girona.

Gracias, N., Ridao, P., Garcia, R., Escartin, J., L'Hour, M., Cibecchini, F., Campos, R., Carreras, M., Ribas, D., & Palomeras, N., et al. (2013). Mapping the Moon: Using a lightweight AUV to survey the site of the 17th century ship "La Lune". In *OCEANS-Bergen, 2013 MTS/IEEE* (pp. 1–8). IEEE.

Hoffmann, H., Pastor, P., Park, D. H., & Schaal, S. (2009). Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance. In *ICRA* (pp. 2587–2592). IEEE.

Hovland, G., Sikka, P., & McCarragher, B. (1996). Skill acquisition from human demonstration using a hidden Markov model. In *Proceedings of the 1996 IEEE international conference on robotics and automation* (Vol. 3, pp. 2706–2711). doi:10.1109/ROBOT.1996.506571.

Jones, C. (2012). Slocum glider persistent oceanography. In *2012 IEEE/OES autonomous underwater vehicles (AUV)* (pp. 1–6). doi:10.1109/AUV.2012.6380738.

Karras, G. C., Bechlioulis, C. P., Leonetti, M., Palomeras, N., Kormushev, P., Kyriakopoulos, K. J., & Caldwell, D. G. (2013). On-line identification of autonomous underwater vehicles through global derivative-free optimization. In *IEEE/RSJ international conference on intelligent robots and systems, 2013. IROS 2013*.

Kormushev, P., Calinon, S., & Caldwell, D. G. (2011). Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input. *Advanced Robotics*, *25*(5), 581–603.

Krüger, V., Tikhanoff, V., Natale, L., & Sandini, G. (2012). Imitation learning of non-linear point-to-point robot motions using dirichlet processes. In *ICRA* (pp. 2029–2034). IEEE.

Kunz, C., Murphy, C., Singh, H., Pontbriand, C., Sohn, R. A., Singh, S., et al. (2009). Toward extraplanetary under-ice exploration: Robotic steps in the Arctic. *Journal of Field Robotics*, *26*(4), 411–429.

Lane, D. (2014). PANDORA website. Retrieved December 15, 2014.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110. doi:10.1023/B:VISI.0000029664.99615.94.

Marani, G., Choi, S. K., & Yuh, J. (2009). Underwater autonomous manipulation for intervention missions AUVs. *Ocean Engineering*, *36*(1), 15–23.

Matsubara, T., Hyon, S. H., & Morimoto, J. (2011). Learning parametric dynamic movement primitives from multiple demonstrations. *Neural Networks*, *24*(5), 493–500. doi:10.1016/j.neunet.2011.02.004.

Maurelli, F., Krupinski, S., Petillot, Y., & Salvi, J. (2008). A particle filter approach for AUV localization. In *MTS/IEEE OCEANS* (pp. 1–7).

McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R., & McEwen, R. (2007). T-REX a model-based architecture for AUV control. In *ICAPS, workshop in planning and plan execution for real-world systems*.

McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R., & McEwen, R. (2008). A deliberative architecture for AUV control. In *IEEE international conference on robotics and automation (ICRA)*. Pasadena.

Nagappa, S., Palomeras, N., Lee, C. S., Gracias, N., Clark, D. E., & Salvi, J. (2013). Single cluster PHD SLAM: Application to autonomous underwater vehicles using stereo vision. In *2013 MTS/IEEE OCEANS-Bergen* (pp. 1–9). IEEE.

Newman, P., & Leonard, J. (2003). Pure range-only sub-sea SLAM. In *IEEE international conference on robotics an automation*.

Ozog, P., & Eustice, R. M. (2014). Toward long-term, automated ship hull inspection with visual SLAM, explicit surface optimization, and generic graph-sparsification. In *Proceedings of the IEEE international conference on robotics and automation, Hong Kong, China* (pp. 3832–3839).

Palmer, T., Ribas, D., Ridao, P., & Aggelos, A. (2009). Vision based localization system for AUV docking on subsea intervention panels. In *Oceans 2009-Europe* (pp. 1–10).

Pastor, P., Hoffmann, H., Asfour, T., & Schaal, S. (2009) Learning and generalization of motor skills by learning from demonstration. In *ICRA* (pp. 763–768). IEEE.

Piacentini, C., Alimisis, V., Fox, M., & Long, D. (2013). Combining a temporal planner with an external solver for the power balancing problem in an electricity network. In *Proceedings of the 23rd international conference on automated planning and scheduling (ICAPS'13)*.

Rajan, K., McGann, C., Py, F., & Thomas, H. (2007). Robust mission planning using deliberative autonomy for autonomous underwater vehicles. In *ICRA workshop on robotics in challenging and hazardous environments* (pp. 21–25).

Ribas, D., Palomeras, N., Ridao, P., Carreras, M., & Mallios, A. (2012). Girona 500 AUV, from survey to intervention. *IEEE/ASME Transactions on Mechatronics*, *17*(1), 46–53.

Ribas, D., Ridao, P., & Neira, J. (2010). Underwater SLAM for structured environments using an imaging sonar. In B. Siciliano, O. Khatib, & F. Groen (Eds.), *Springer tracts in advanced robotics*. New York: Springer.

Rosten, E., & Drummond, T. (2006) Machine learning for high-speed corner detection. In *European conference on computer vision* (pp. 430–443).

Salvi, J., Petillot, Y., Thomas, S., & Aulinas, J. (2008). Visual SLAM for underwater vehicles using video velocity log and natural landmarks. In: *Oceans conference*.

Sanz, P. J., Ridao, P., Oliver, B., Casalino, G., Silvestre, C., Melchiorri, C., et al. (2012). TRIDENT: Recent improvements about autonomous underwater intervention missions. In *Navigation, guidance and control of underwater vehicles* (Vol. 3).

Simmons, R. G., Goodwin, R., Haigh, K. Z., Koenig, S., O'Sullivan, J., & Veloso, M. M. (1997). Xavier: Experience with a layered robot architecture. *SIGART Bulletin*, *8*(1–4), 22–33.

Singh, H., Armstrong, R., Gilbes, F., Eustice, R., Roman, C., Pizarro, O., et al. (2004). Imaging coral I: Imaging coral habitats with the SeaBED AUV. *Subsurface Sensing Technologies and Applications*, *5*(1), 25–42.

Smith, R., Schwager, M., Smith, S. L., Rus, D., & Sukhatme, G. (2011). Persistent ocean monitoring with underwater gliders: Towards accurate reconstruction of dynamic ocean processes. In *2011 IEEE international conference on robotics and automation (ICRA)* (pp. 1517–1524). IEEE.

Spooner, J. T., Maggiore, M., Ordonez, R., & Passino, K. M. (2002). *Stable adaptive control and estimation for nonlinear systems-neural and fuzzy approximator techniques*. New York: Wiley.

SUBSEA7: AIV website. Retrieved December 15, 2014.

Vallicrosa, G., Ridao, P., Ribas, D., & Palomer, A. (2014). Active range-only beacon localization for AUV homing. In *2014 IEEE/RSJ international conference on intelligent robots and systems (IROS 2014)* (pp. 2286–2291). IEEE.

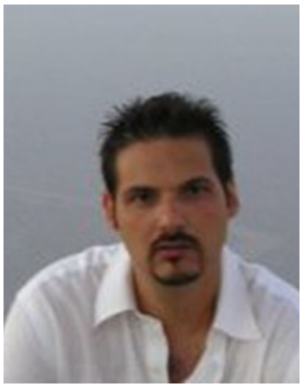
Welch, G., & Bishop, G. (1995). *An introduction to the Kalman Filter*. Technical report, University of North Carolina at Chapel Hill, Chapel Hill, NC.

WillowGarage (2007). Robot operating system. http://www.ros.org Retrieved December, 2014.

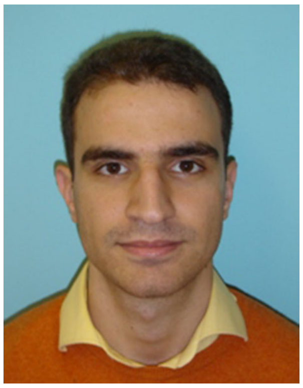
Narcís Palomeras is a Postdoctoral Fellow in the Department of Computer Engineering of University of Girona (UdG), and a member of the Underwater Robotics Laboratory in the Computer Vision and Robotics Group (VICOROB). He holds a B.S. degree in Computer Science (2004) and a PhD in Computer Engineering (2011) from the University of Girona. He has participated in several research projects (both national and European) related with underwater robotics and has taken part in several European AUV competitions. His research activity is mainly focused on underwater robotics in research topics such as intelligent control architectures and mission control.

Arnau Carrera is a PhD student in the Department of Computer Engineering of the Univertisy of Girona (UdG) and a member of the Underwater Robotics Laboratory (CIRS) in the Computer Vision and Robotics Institute (VICOROB). His research interest are focused in applying Learning by Demonstration (LbD) to Intervention Autonomous Underwater Vehicles (I-AUV).

Natàlia Hurtós is a Postdoctoral Fellow in the Department of Computer Engineering of University of Girona (UdG), and a member of the Underwater Robotics Laboratory in the Computer Vision and Robotics Group (VICOROB). She holds a B.S. degree in Computer Science (2007), an European Master in Computer Vision and Robotics (VIBOT, 2009) and a PhD in Computer Engineering (2014) from the University of Girona. Since 2006 she has participated in several research projects (both national and European) revolving around underwater robotics and her interests are mainly focused on mapping of underwater environments using sonar data.



George C. Karras was born in Athens in 1980. He is a Research Scientist at the Control Systems Lab, National Technical University of Athens (NTUA) since 2012. He received a Diploma in Mechanical Engineering in 2003 (NTUA), a Master's Degree in Robotics and Automatic Control in 2006 (NTUA) and a PhD in Mechanical Engineering in 2011 (NTUA). He has more than 10 years of experience working on R&D projects in the field of Robotics and Embedded Control Systems both in Academic Institutes and Industry. He has published 19 papers to journals and fully refereed international conferences.



Charalampos P. Bechlioulis was born in Arta, Greece, in 1983. He is currently a postdoctoral researcher in the Control Systems Laboratory at the School of Mechanical Engineering of the National Technical University of Athens. He received a diploma in electrical and computer engineering in 2006 (first in his class), a bachelor of science in mathematics in 2011 (second in his class) and a Ph.D. in electrical and computer engineering in 2011, all from the Aristotle University of Thessaloniki, Thessaloniki, Greece. His research interests include nonlinear control with prescribed performance, system identification, control of robotic vehicles, multi-agent systems and object grasping. He has authored more than 40 papers in scientific journals and conference proceedings.

Michael Cashmore is a Research Associate in the Department of Informatics at King's College London. There he is a member of the Planning Group, where he works on planning-as-Boolean-formulae and constraint satisfaction. Recently he has worked on embedding planning into robotic systems, maintaining the ROS-Plan system for ROS.narcis

Daniele Magazzeni received a Ph.D. in Computer Science from University of L'Aquila, Italy, in 2009. He is Lecturer in the Department of Informatics at King's College London. His research explores the links between AI planning, controller synthesis and model-checking, with a particular focus on planning in hybrid domains, robotics, hybrid systems control and verification. Daniele is Co-Editor-in-Chief of AI Communications journal, he is serving in the organizing and programme committees of major AI Conferences (AAAI, IJCAI, ICAPS, ECAI) and will be conference chair of ICAPS 2016.

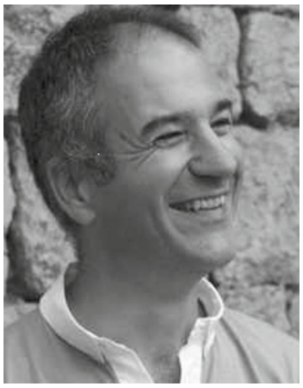
Derek Long Department of Informatic, King's College London. Professor Derek Long is an active researcher in AI Planning, working in metric and temporal planning and its applications. He has helped to develop and promote the use of the Planning Domain Definition Language (PDDL), particularly in representing temporal domains and hybrid continuous-discrete effects of actions. He has been involved in the development of multiple planners capable of working with more expressive variants of PDDL, including POPF and COLIN. Derek has been a member of the Council of the International Conference on Automated Planning and Scheduling (ICAPS) and an Associate Editor of Artificial Intelligence Journal.

Maria Fox Department of Informatic, King's College London. Professor Maria Fox began her career in 1989 as a lecturer at UCL, then moved to Durham University and then the University of Strathclyde before joining KCL in 2011. Her research is in automated planning, a sub-field of artificial intelligence. Within the Planning Group, which she leads, she has helped to develop some of the best-known modern approaches to temporal and metric planning, and has led a number of applications of planning to problems in robotics and industry. She is interested in providing robotic systems with a high-level general problem-solving capability. Maria is an elected member of the Council of the International Conference on Automated Planning and Scheduling (ICAPS), Editor-in-Chief of AI Communications, an Associate Editor of Artificial Intelligence Journal and a member of the Advisory Board of the Journal of AI Research.

Kostas J. Kyriakopoulos received a Diploma in Mechanical Engineering (Honors) from NTUA, in 1985 and the MS & Ph.D. in Electrical, Computer & Systems Engineering (ECSE) from Rensselaer Polytechnic Institute (RPI), Troy, NY in 1987 and 1991, respectively. From 1988 to 1991 he did research at the NASA Center for Intelligent Robotic Systems for Space Exploration. Between 1991-93 he was an Assistant Professor at ECSE - RPI and the New York

State Center for Advanced Technology in Automation and Robotics. Since 1994 he has been with the Control Systems Laboratory of the Mechanical Engineering Department at NTUA, Greece, where he currently serves as a Professor, Director or the Control Systems Lab and Director of the Departmental Computation Lab. His current interests are in the area of Nonlinear Control Systems applications in Sensor Based Motion Planning & Control of multi-Robotic Systems: Manipulators & Vehicles (Mobile, Underwater and Aerial) and Micro- & Bio-Mechatronics. He is a member of IEEE and the Technical Chamber of Greece. He was awarded the G.Samaras award of academic excellence from NTUA, the Bodosakis Foundation Fellowship (1986-1989), the Alexander Onassis Foundation Fellowship (1989-1990) and the Alexander Von Humboldt Foundation Fellowship (1993). Dr. Kyriakopoulos has published close to 280 papers to journals and refereed conferences and serves as a regular reviewer to a number of journals and conferences. He has contributed to a large number of projects funded by the EC and the Greek secretariat for Research & Technology. He served and serves in the editorial committee of a number of (including IEEE) journals and has served as an administrative member of a number of international conferences.

Petar Kormushev is a Research Team Leader of the Robot Learning and Interaction Lab at the Department of Advanced Robotics, Italian Institute of Technology (IIT) in Genoa, Italy. He is also a Visiting Senior Research Fellow at the Centre for Robotics Research (CORE) of King's College London, UK. In 2009 he obtained a PhD in Computational Intelligence from Tokyo Institute of Technology (Tokyo Tech), Japan. He holds two MSc degrees in Artificial Intelligence and Medical Informatics, and a BSc degree in Computer Science from Sofia University, Bulgaria. Dr Kormushev is a co-chair of the IEEE Technical Committee on Robot Learning since 2012. He is the recipient of the 2013 John Atanasoff award by the President of Bulgaria, awarded for scientific achievements in the field of ICT. His research interests include machine learning for robot control and human-robot interaction.



Joaquim Salvi graduated in Computer Science at the Technical University of Catalonia in 1993, received the DEA (MSc) in Computer Science in July 1996 and the PhD in Industrial Engineering in 1998 both at the University of Girona, Spain. He is Professor of Computer Vision at the Computer Architecture and Technology Department and at the Computer Vision and Robotics Group, University of Girona; and he was a visiting professor at the Ocean Systems Lab, Heriot-Watt University (UK). He is involved in some governmental projects and technology transfer contracts to industry. His current interests are in the field of computer vision and mobile robotics, focused on visual SLAM, structured light, stereovision, and camera calibration. He is the leader of the 3D Perception Lab and charter member of the spinoffs AQSense and BonesNotes. Dr. Salvi received the Best Thesis Award in engineering for his PhD. Currently, he is the director of the Polytechnic School of the University of Girona.

Marc Carreras is Associate Professor in the Computer Vision and Robotics Institute of the University of Girona (Spain). He holds a B.Sc. degree in Industrial Engineering (1998) and PhD in Computer Engineering (2003) from the University of Girona. From 1999 until 2014 he has participated in 14 research projects (6 European and 8 National), he is author of more than 90 publications, he has supervised 3 PhDs thesis and has participated in several European AUV competitions (3 times winner). His research activity is mainly focused on underwater robotics in research topics such as intelligent control architectures, robot learning, path planning, AUV design, modelling and identification.