2018 26th Mediterranean Conference on Control and Automation (MED)
Zadar, Croatia, June 19-22, 2018

Assistive control framework for Remotely Operated Vehicles

Daniele Di Vito, Paolo Di Lillo, Filippo Arrichiello, Gianluca Antonelli

Abstract— **The paper presents a control framework aimed at supporting a human operator in the guidance and in the coordination of an underwater Remotely Operated Vehicle (ROV). Due to environmental conditions and operational scenarios, underwater tele-operations can be very demanding. Thus, a very high experience can be required to the operator that, by providing commands to each actuator, has to successfully perform the assigned task without damaging the vehicle. In this sense, a control algorithm that can help the operator in performing the assigned task may represent a system support element able to ensure the vehicle safety. Based on a survey realized with professional ROV pilots, in this paper we propose a framework that allows the human operator to send position commands to the ROV by a joystick and to set different tasks through a dedicated user-friendly interface. The tasks are managed via a control algorithm represented by a closed loop inverse kinematic algorithm able to manage multiple both equality-based and set-based tasks. The overall framework has been developed in the Robot Operating System (ROS) environment and numerically tested using Gazebo environment.**

I. Introduction

Through the years increasingly advanced underwater systems have been developed to physically substitute the human in performing dangerous tasks such as, e.g., maintenance of submarine structures [1]. In particular, underwater robots are mainly divided into two types: Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs).

ROVs are underwater systems physically linked via tether to a human operator whose working station can be either on a submarine or on a surface vessel, as shown in Fig. 1. The tether represents a crucial element for this kind of vehicle since it is in charge of giving power to the ROV as well as closing the manned control loop, i.e., by providing sensor information to the operator and by transmitting motion directives to the ROV. However, its presence makes more difficult the navigation and manoeuvring tasks; indeed, the operator has to take into account the tether inertia, and to reduce the twisting and to avoid entanglements. Thus man continues to have an essential role.

AUVs, on the other side, are systems supposed to be completely autonomous with onboard power and unmanned control loop. However, the presence of technological issues, first among all the limited autonomy of the power system, makes them not widely used inside industry that prefers to rely on ROVs or on a combined use of both of them [2], [3].

ROV guidance and coordination requires a very high-experienced human operator, and a concentration level that grows depending on the environmental conditions, e.g., visibility and ocean currents [4]. On average, in case of bad work conditions, an operator is able to drive the system for 30 minutes at most. Contrarily, in case of good operative conditions, he can work without pauses for one hour and half. Thus, the presence of automatism, e.g., station-keeping, auto-heading, way-point navigation and auto-depth, that helps the operator in performing some tasks, can improve the mission performances. Furthermore control tasks as obstacle avoidance and auto-altitude can ensure to avoid vehicle collisions even when, in case of bad operative conditions, the human operator could fail.

Fig. 1. ROV pilots, Comex, France.

Different solutions aimed at supporting the operator in guiding ROVs exist. In particular, they focus on detecting/tracking pipelines and vehicle positioning. Indeed, vehicles equipped with multi-beam echo-sounder and side-scan sonar systems [5], [6] may allow to perform this kind of operations. Furthermore, more recently, vision based systems that implement image processing algorithms [7] are knowing an increased spread thanks to the more powerful and cheaper processing units. Different algorithms have also been proposed for depth control [8], [9] and more in general for vehicle positioning control [10].

In this work, within the assumptions of having a fully actuated vehicle and not knowing the vehicle dynamic model, and by considering real requirements that arose from interviews with professional ROV pilots, a control framework to give support to human operator in guiding and manoeuvring underwater ROV is presented. More in detail, the framework is represented by the control algorithm and a user-friendly interface. The control algorithm, that receives operator's command inputs by a joystick and by a graphical interface, is based on a multi-task-priority inverse kinematics framework which handles both equality-based and set-based tasks, i.e. tasks in which the control objective is to keep specific value in an interval rather than an exact value. Then, the controller outputs are sent to the Gazebo simulator [11] showing a realistic underwater environment, where the cad model of Girona 500 [12] is used to simulate the vehicle. Tasks can

Authors are with the Department of Electrical and Information Engineering of the University of Cassino and Southern Lazio, via Di Biasio 43, 03043 Cassino (FR), Italy {d.divito, pa.dilillo, f.arrichiello, antonelli}@unicas.it

be enabled/disabled through the graphic interface directly by operator, who can also monitor some telemetry data from the ROV, e.g, its altitude and depth.

The paper is organized as follows. Section III reports the control algorithm and the implemented tasks. Section IV presents the overall assistive control framework architecture, developed within ROS [13]. Simulation results of the proposed control framework are finally reported in section V.

II. Motivation

The assistive control framework proposed in this work has been developed on the basis of the real needs that arose from a survey realized with different professional ROV pilots. For a better understanding of identified requirements, let us distinguish between a body-fixed frame $\mathcal{F}_B$ relating to the vehicle and an inertial frame $\mathcal{F}_\mathbf{0}$. In particular, the survey results remarked the following situations:

1) *Unbalanced vehicle*. In several ROV designs, the vehicle is under actuated leaving roll and pitch to be stabilized by a proper weight distribution. However, moving masses such as manipulators or a wrong payload positioning may cause the vehicle to exhibit non-null roll and/or pitch at steady state. For example, this is what has happened within the framework of the H2020 European Project DexROV [14] where an existing under actuated vehicle has been customized, and the presence of two manipulators and of the perception system has caused the system to operate with a non-null pitch angle depending on the arm configuration. In such a case, in order to perform a movement along the earth-fixed frame surge direction (the green dashed line in Fig. 2), the operator, having control of the body-fixed variables, naturally follows a saw tooth shaped path (the red dashed line in Fig. 2) by alternating velocity commands to the horizontal and vertical thrusters respectively. In this sense, a control task allowing the operator to send velocity commands along the effective desired direction might improve the mission efficiency.

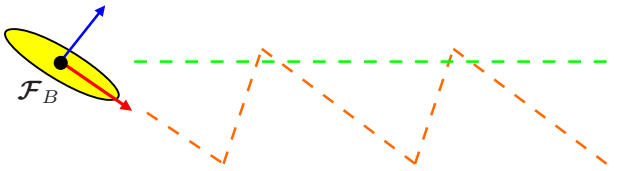

Fig. 2. Vehicle with non-null pitch angle: the green dashed line represents the desired path; the red dashed one is the path followed by an operator acting on the body-fixed variables.

Even when the vehicle is fully actuated, it might be appropriate to leave the roll and pitch angles not actuated to save energy during long missions. In such a case, the situation is analogous to the above and this feature might help the operator. Moreover, the possibility to enable/disable the control task according to the operator needs during the mission is required.

2) *Altitude/depth control*. Controlling depth and altitude is intrinsically a conflicting requirement. The operator is often required to travel at constant depth (orange dashed line in Fig. 3), however, for safety reasons a minimum altitude needs to be maintained. Auto-depth is an existing control feature in most of the vehicles, however, the operator needs to keep track of the current altitude to avoid of impacting the sea bottom. A required feature is an automatic switching between auto-depth and auto-altitude without human intervention.

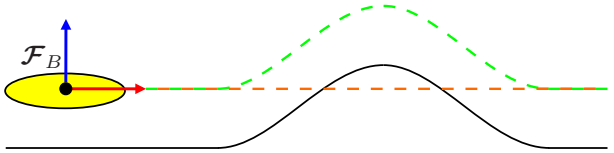

Fig. 3. The green dashed line is the path followed through an auto-altitude task; the orange dashed one is the path followed through an auto-depth task. Operators might be helped by an automatic switching between the two tasks.

The need for this feature is emphasized also by considering again the unbalanced vehicle case.

3) *Reference Frames*. Depending on the specific operation to perform, it could be very helpful for the operator to have the possibility to choose to command the ROV in inertial frame or in the body-fixed frame, e.g., for intervention operation as manoeuvring valves of a panel (see Fig. 4).

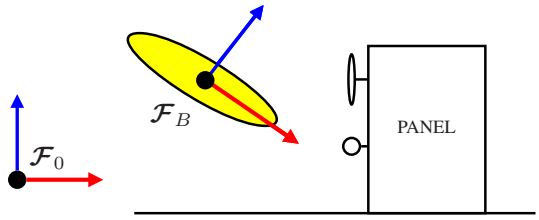

Fig. 4. Reference frames: possibility for the operator to provide commands in inertial or body-fixed frame depending on the manoeuvring in front of the panel.

III. Control Algorithm

A. Kinematic model

The ROV state is defined by the position and the orientation of the body fixed-frame $\mathcal{F}_B$ with respect to the inertial frame $\mathcal{F}_\mathbf{0}$ [15]. Thus, defining the vector $\boldsymbol{\eta}_1 = \begin{bmatrix} x & y & z \end{bmatrix}^\mathrm{T}$ and $\boldsymbol{\eta}_2 = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^\mathrm{T}$ representing the ROV position and orientation with respect to $\mathcal{F}_\mathbf{0}$, respectively, the entire state can be described by the vector $\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_1 & \boldsymbol{\eta}_2 \end{bmatrix}^\mathrm{T}$.

Defining $\boldsymbol{\nu}_1 = \begin{bmatrix} u & v & w \end{bmatrix}^\mathrm{T}$ as the vector representing the linear velocity of the body-fixed frame $\mathcal{F}_B$ with respect to $\mathcal{F}_0$ expressed in $\mathcal{F}_B$, its relation with the time derivative $\dot{\boldsymbol{\eta}}_1$ can be written as

$$\boldsymbol{\nu_1} = \boldsymbol{R}_0^B \dot{\boldsymbol{\eta}}_1 \tag{1}$$

where $\boldsymbol{R}_0^B$ is the rotation matrix representing the transformation between $\mathcal{F}_0$ and $\mathcal{F}_B$ [16].

Defining $\boldsymbol{\nu}_2 = \begin{bmatrix} p & q & r \end{bmatrix}^\mathrm{T}$ as the angular velocity of $\mathcal{F}_B$ with respect to $\mathcal{F}_0$, expressed in $\mathcal{F}_B$, the relation with the time derivative $\dot{\boldsymbol{\eta}}_2$ is given by

$$\boldsymbol{\nu}_2 = \boldsymbol{J}_o(\boldsymbol{\eta}_2)\dot{\boldsymbol{\eta}}_2 \tag{2}$$

where $\boldsymbol{J}_o(\boldsymbol{\eta}_2)$ represents the jacobian matrix relating the angular velocity components [15]. Collecting Eq. (1) and (2) in matrix form, the kinematic model of the ROV can be expressed as

$$\boldsymbol{\nu} = \boldsymbol{J}(\boldsymbol{\eta}_2)\dot{\boldsymbol{\eta}} \tag{3}$$

where $\boldsymbol{J}(\boldsymbol{\eta}_2)$ is the Jacobian matrix consisting of $\boldsymbol{R}_0^B, \boldsymbol{J}_o(\boldsymbol{\eta}_2)$ and $\boldsymbol{\nu} = \begin{bmatrix}\boldsymbol{\nu}_1 & \boldsymbol{\nu}_2\end{bmatrix}^{\mathrm{T}}$.

In the following, matrix dependencies will be dropped out to increase readability.

B. Inverse kinematics control

A generic control objective, referred in the following as *task*, is a function of the system state defined as $\boldsymbol{\sigma}(\boldsymbol{\eta}) \in \mathbb{R}^m$ with m the function dimension. In particular, it is possible to divide these tasks in two main groups: *equality-based* tasks and *set-based* tasks [17]. In an equality-based task the control objective is to bring the task value to a desired one, for instance to move the ROV in a specific position; in a set-based task, on the other side, the control objective is to keep the task value within an interval, for instance to maintain the ROV beyond a certain distance from a potential obstacle. Thus, such kind of task is mono-dimensional by definition.

Given a generic equality-based task $\boldsymbol{\sigma}$, the system velocity, that fulfills it, can be computed by resorting to the Closed-Loop-Inverse-Kinematics algorithm [18]:

$$\boldsymbol{\nu} = \boldsymbol{J}^{\dagger}\boldsymbol{K}\tilde{\boldsymbol{\sigma}} \tag{4}$$

where $\boldsymbol{K}$ is a positive-definite matrix of gains and $\tilde{\boldsymbol{\sigma}} = \boldsymbol{\sigma_d} - \boldsymbol{\sigma}$ is the task error. Furthermore, multiple tasks can be simultaneously performed setting a priority to each one and then projecting the velocity components of the lower priority tasks into the null space of the higher priority ones. Thus the fulfillment of the higher priority tasks is guaranteed, as the velocity component coming from a low priority task, that would influence it, is filtered out. In particular, given a hierarchy composed by n prioritized tasks, the system velocity can be computed by resorting to the Null-Space-Based Inverse Kinematics control [19], [20]:

$$\boldsymbol{\nu} = \boldsymbol{J}_1^{\dagger}\boldsymbol{K}_1\tilde{\boldsymbol{\sigma}}_1 + \boldsymbol{N}_1\boldsymbol{J}_2^{\dagger}\boldsymbol{K}_2\tilde{\boldsymbol{\sigma}}_2 + \cdots + \boldsymbol{N}_{1,n-1}\boldsymbol{J}_n^{\dagger}\boldsymbol{K}_n\tilde{\boldsymbol{\sigma}}_n \tag{5}$$

where $\boldsymbol{N}_{1,i}$ is the null space of the augmented Jacobian $\boldsymbol{J}_{1,i}$ matrix obtained by stacking all the tasks Jacobian matrix from $\boldsymbol{\sigma}_1$ to $\boldsymbol{\sigma}_i$.

The NSB framework has been extended to handle also set-based tasks [17], [21]. This is possible by considering each set-based task as an equality-based one that can be activated and deactivated in function of the operating conditions. It is worth noticing that the activation/deactivation of a set based task is in general different from enabling/disabling a generic task through the graphical interface. In particular, a set-based task has to be activated when its value exceeds the desired lower (upper) threshold $\sigma_{a,l}$ ($\sigma_{a,u}$), adding it to the hierarchy as a new equality-based task with $\sigma_{s,l}$ ($\sigma_{s,u}$) as desired value (see Fig. 5). Then it can be deactivated when the solution of the hierarchy that contains only the other tasks would push its value toward the valid set.

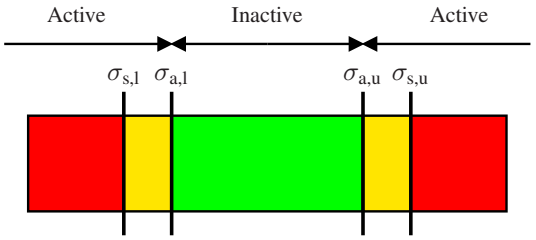

Fig. 5. Activation and safety thresholds of a set-based task, with the green (red) section representing the valid (invalid) set.

C. Implemented tasks

With the aim to support the operator in guiding and manoeuvring underwater vehicles, different tasks have been implemented [15]. In detail, tasks are divided into equality-based and set-based, respectively. Starting from the equality-based ones there are:

- **vehicle position/orientation** ($m = 3$). It consists in the control of vehicle position/orientation.
- **vehicle pose** ($m = 6$). This task allows to control the entire vehicle configuration merging the previous two tasks.
- **vehicle attitude** ($m = 2$). It consists in the control of two components of orientation, in detail the roll and pitch ones. It results very useful if the operator needs to work with specific values of roll and pitch angles or in case the ROV is not well balanced as mentioned in section II.
- **vehicle relative field of view** ($m = 1$). This task allows to direct the vehicle towards a desired target $\boldsymbol{p}_O$, e.g., a panel on the seabed. Thus if the human operator knows the target position a priori, he can activate this task without manually manoeuvring the vehicle. From the control perspective, it is not necessary to control all orientation DOFs but only the desired outgoing vector [22].
- **vehicle heading** ($m = 2$). This task, contrarily to the vehicle attitude, controls the pitch and yaw components allowing to maintain the ROV along a desired straight trajectory.
- **vehicle depth** ($m = 1$). Most of underwater vehicles is equipped with depth sensors. Indeed they are pressure-based and therefore they are more reliable than altitude ones that are instead based on acoustic systems. Thus a control on ROV depth is implemented giving the possibility to the operator to set a determined depth value.

The set-based tasks, on the other side, are the following:

- **vehicle roll, pitch and yaw limit**, respectively ($m = 1$). ROVs generally have small variance ranges for roll and pitch components. This is essentially due to hardware that presents physical limits for the orientation. Contrarily, the yaw component can present a large variance range or also it can have no limits. Indeed, the only constraint is represented by the presence of the tether. The limits can be set directly by the operator.
- **vehicle altitude limit** ($m = 1$). The following task allows to set a minimum altitude value from seabed and eventual objects preventing possible damages to the vehicle as mentioned in section II.
- **obstacle avoidance** ($m = 1$). The obstacle avoidance ensures of avoiding eventual obstacles along the trajectory

without the operator intervention who has just to set the safety distance. This task can be implemented controlling the ROV distance from the obstacle position $\boldsymbol{p}_{ob} \in \mathbb{R}^3$.

All described tasks are dynamically combined through the priority hierarchy resorting to the NSB control.

IV. Control framework

The assistive control framework, shown in Fig. 6, has been developed inside the well-known Robot Operating System [13]. Thus, the implemented software is structured in a graph architecture where processes are able to communicate among them like nodes of the same network. In particular, the framework architecture is divided into two main blocks: the command station and the vehicle simulator.

The Command Station is represented by the graphical user interface that allows the operator to send commands by joystick and to enable/disable the different tasks. As mentioned above, the task enabling/disabling is different from the set-based task activation/deactivation. Indeed, when a task is enabled/disabled by the operator it is added/removed to/from the control task hierarchy. On the other side, the task activation/deactivation, as described in section III, is a crucial step of the set-based control algorithm, and therefore it is not managed by the operator.

The Vehicle Simulator block, developed in Gazebo environment, consists of the implemented high and low level controllers, and the vehicle kinematic/dynamic model.

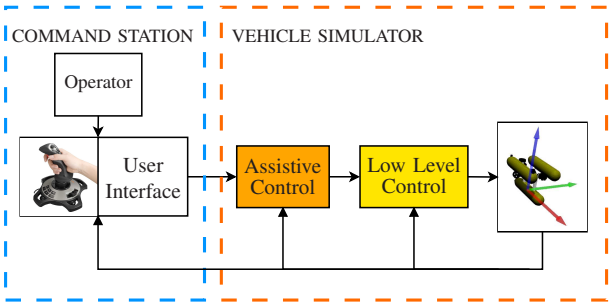

Fig. 6. Assistive control framework architecture.

A. Vehicle Simulator

The simulations described in the following section have been performed using the Gazebo 3D simulator for the visualization of the ROV motion and the simulation of the needed sensors, while ROS has been used for interfacing it with the control algorithm and the graphical user interface. The Girona 500 [12] CAD model has been used, and it has been equipped with several sensors provided by Gazebo as, e.g., laser scanners and cameras.

The motion controller has been developed as two nested control loops: the Assistive Control is kinematic and it implements the algorithm described in section III giving as output the reference vehicle velocities that fulfill all the tasks, while the Low Level Control computes the torque commands for the motors. The latter has been developed as two interchangeable Gazebo plugins, one kinematic and one dynamic. The kinematic one instantaneously applies the desired velocity coming from the Assistive Control to the ROV in the simulator. It is useful to debug the high-level kinematic control. The dynamic low-level controller is implemented as a standard PID that simulates a second order dynamics on the ROV velocity, making the simulation more realistic.

B. Graphical interface

Figure 7 shows two sub-blocks of the graphical interface that has been developed exploiting Qt, which is a cross-platform application framework used for application software development. In detail, the interface displays the two videos

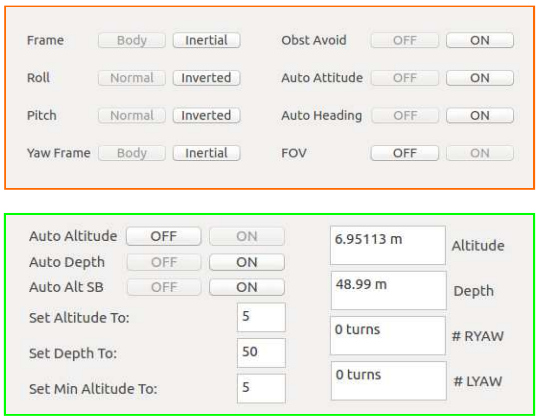

Fig. 7. Graphical interface: vision panel (two cameras), control panel (divided into two sub-blocks to increase the readability).

taken from the cameras on-board the vehicle and some important telemetry data, such as altitude, depth and number of left/right spins of the ROV. Furthermore the operator can enable/disable a series of tasks: obstacle avoidance (*SB*), auto-attitude (*EB*), auto-heading (*EB*), field of view (*EB*), auto-altitude (*EB* or *SB*), and auto-depth (*EB*), where *EB* and *SB* mean *Equality-Based* and *Set-Based*, respectively. For the auto-depth (*EB*) and the auto-altitude (*EB*) the operator can set the relative desired value; instead, for the auto-altitude (*SB*) he can set the minimum altitude from the seabed which represents the safety distance to prevent damages. It is worth noticing that the task enabling/disabling corresponds to add/remove it to the task hierarchy of the inverse kinematics control changing also the relative priorities. Indeed, the dynamic priority change is managed by the high level controller. The interface gives the operator also the possibility to switch between the inertial and body-frame, for the reasons explained in section II. Within the objective to help the operator as much as possible, he has the further option to switch only the yaw components between the inertial and body-frame. Finally, the interface presents other two options for the roll and pitch components that allow to set the orientation convention clockwise or counter-clockwise.

V. Simulation case study

A generic survey mission has been taken into account as simulation case study. In particular, an underwater environment with a non uniform seabed has been rendered adding the presence of a panel.

The validation is made by an operator that drives the vehicle, enabling/disabling the tasks from the graphical interface as explained in the following. In particular, the control framework reduce the complexity of the operation, allowing the operator to focus only on the operational tasks while the safety-related ones are autonomously handled by the system.

The implemented task values and errors have been plotted in real-time during the simulation. In particular, the equality-based task errors and the set-based task values with the corresponding activation thresholds (black lines) are shown in Fig. 8, 9 and Fig. 10, respectively, where the green and pink plot backgrounds represent the enabled and disabled states of the corresponding task.

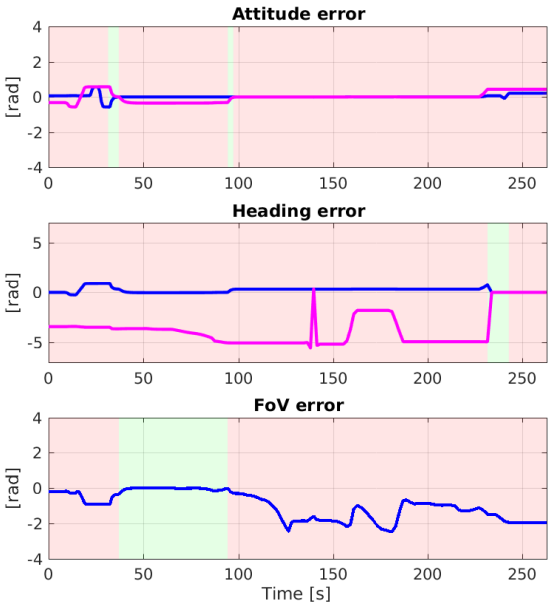

Fig. 8. Task error plots: attitude ($m = 2$), heading ($m = 2$) and field of view ($m = 1$). The green and pink background represents the enabled and disabled state, respectively, of the corresponding task.

In detail, aiming at testing the entire control framework, all tasks have been enabled/disabled approaching the vehicle to the panel. First of all, a series of rotations have been performed keeping still the vehicle to verify the roll and pitch limits respect ($\sigma_{a,l} = -\pi/4$, $\sigma_{a,u} = \pi/4$ for both ones). This can be observed during the first 50 seconds in the third and fourth plot in Fig. 10. Then the attitude task has been enabled just to set roll and pitch angles to zero (see first plot in Fig. 8). With the aim of directing the camera (and therefore also the vehicle) toward the panel, the field of view task has been enabled, as observable in the third plot in Fig. 8 in the green range where the error goes to zero. Furthermore, since both the attitude and field of view tasks are fulfilled controlling the angular components their simultaneous enabling does not make sense for guiding the vehicle. This is implemented in the graphical interface in such a way that enabling one implies disabling the other one. At the same time also the obstacle avoidance has been enabled. It becomes active as soon as the vehicle is close to the panel ($\sigma_{a,l} = 2\,\text{m}$). Then the altitude task has been enabled setting the desired value $\sigma_{i,d} = 2\,\text{m}$ as noticeable from the null error in the first plot in Fig. 9.

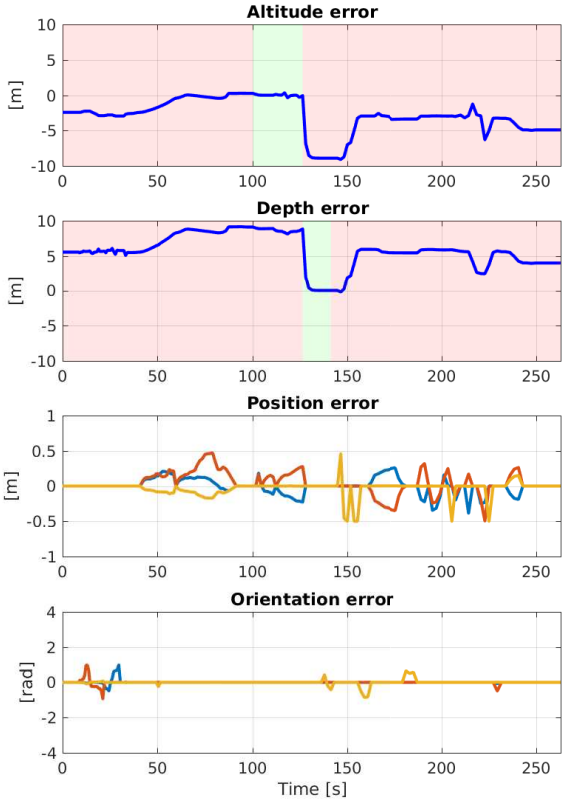

Fig. 9. Task error plots: altitude ($m = 1$), depth ($m = 1$), position ($m = 3$) and orientation ($m = 3$). The green and pink background represents the enabled and disabled state, respectively, of the corresponding task.

Thus the depth task has been enabled with $\sigma_{i,d} = 100\,\text{m}$. Altitude and depth tasks act on the same work space component and therefore, for the same reasons mentioned above, they can not be enabled simultaneously. Then the set-based altitude task has been enabled setting $\sigma_{a,l} = 5\,\text{m}$. In particular, as it is noticeable in the second plot in Fig. 10, there are values under the lower threshold. This is due to the intentional decision to move the ROV over the panel, causing a temporary violation of the minimum altitude. Finally the heading task has been enabled to control the pitch and yaw angles, as shown in the second plot in Fig. 8. It is worth noticing that spikes present in position and orientation errors, shown in Fig. 9 (third and fourth plot), are due to the operator inputs. Indeed he sends position commands that are constant increments with respect to the current ROV pose.

Another aspect taken into consideration during the simulation is the number of twistings. Indeed, within the assumption to have the tether and reading the corresponding values from the graphical interface, the number of clockwise spins has been compensated by the number of counter clockwise ones to prevent twistings.

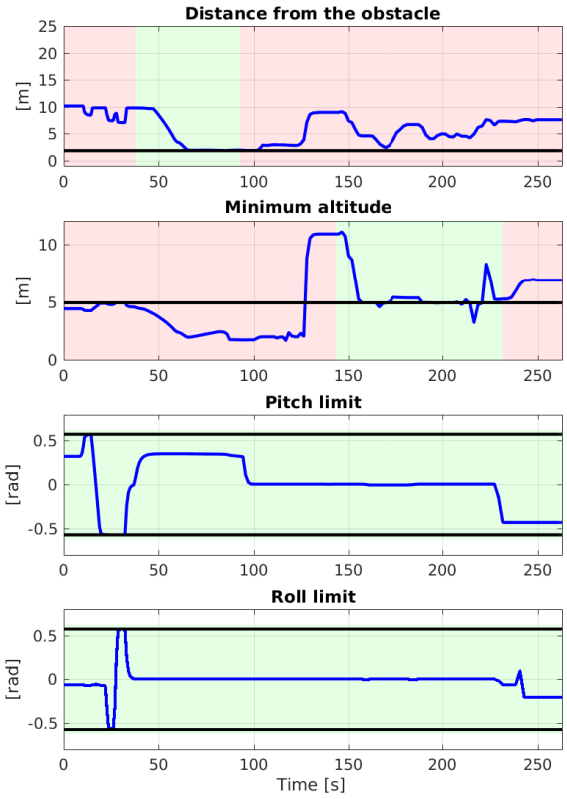

Fig. 10. Task value plots: obstacle avoidance, auto-altitude, pitch limit and roll limit. The green and pink background represents the enabled and disabled state, respectively, and the black lines are the upper/lower activation thresholds of the corresponding set-based task.

VI. Conclusions

In this paper we presented a framework that is focused in giving support to a human operator in guiding and manoeuvring a ROV. All the implemented functionalities have been specifically chosen to address real needs that arose after interviews with professional ROV pilots. We have described in detail all the developed tasks and the control algorithm that coordinates them, together with the simulation framework and the graphical user interface specifically implemented for the application. Finally, simulation results including different task hierarchies suitable for several kind of operations have been shown. Future works will be focused on the experimental implementation of the designed control framework.

Acknowledgments

The Authors are gratefully to the ROV pilots of the CSSN (Centro di Supporto e Sperimentazione Navale) of the Italian Navy and to ROV trainer Gilbert Pachoud, Comex, France.

This work was supported by the European Community through the projects EUMR (H2020-731103-2), ROBUST (H2020-690416) and DexROV (H2020-635491).

References

[1] L. L. Whitcomb, "Underwater robotics: out of the research laboratory and into the field," in *Proceedings of 2000 IEEE International Conference on Robotics and Automation*, vol. 1, 2000, pp. 709–716.

[2] M. Ludvigsen, G. Johnsen, A. J. Sørensen, P. A. Lågstad, and Ø. Ødegård, "Scientific operations combining rov and auv in the trondheim fjord," *Marine Technology Society Journal*, vol. 48, no. 2, pp. 59–71, 2014.

[3] J. C. Evans, K. M. Keller, J. S. Smith, P. Marty, and O. V. Rigaud, "Docking techniques and evaluation trials of the swimmer auv: an autonomous deployment auv for work-class rovs," in *Proceedings of MTS/IEEE Oceans 2001*, vol. 1, 2001, pp. 520–528.

[4] N. J. Cooke, "Human factors of remotely operated vehicles," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 50, no. 1, 2006, pp. 166–169.

[5] Y. R. Petillot, S. R. Reed, and J. M. Bell, "Real time auv pipeline detection and tracking using side scan sonar and multi-beam echo-sounder," in *Proceedings of OCEANS '02 MTS/IEEE*, vol. 1, 2002, pp. 217–222.

[6] J. Evans, Y. Petillot, P. Redmond, M. Wilson, and D. Lane, "Auto-tracker: Auv embedded control architecture for autonomous pipeline and cable tracking," in *Proceedings of OCEANS '03 MTS/IEEE*, vol. 5, 2003, pp. 2651–2658.

[7] M. Narimani, S. Nazem, and M. Loueipour, "Robotics vision-based system for an underwater pipeline and cable tracker," in *Proceedings of OCEANS 2009-EUROPE*, 2009, pp. 1–6.

[8] S. M. Zanoli and G. Conte, "Remotely operated vehicle depth control," *Control engineering practice*, vol. 11, no. 4, pp. 453–459, 2003.

[9] W. M. Bessa, M. S. Dutra, and E. Kreuzer, "Depth control of remotely operated underwater vehicles using an adaptive fuzzy sliding mode controller," *Robotics and Autonomous Systems*, vol. 56, no. 8, pp. 670–677, 2008.

[10] J. Javadi-Moghaddam and A. Bagheri, "An adaptive neuro-fuzzy sliding mode based genetic algorithm control system for under water remotely operated vehicle," *Expert Systems with Applications*, vol. 37, no. 1, pp. 647–660, 2010.

[11] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proceedings 2004 IEE/RJS International Conference on Intelligent RObots and Systems*, vol. 3, 2004, pp. 2149–2154.

[12] D. Ribas, P. Ridao, L. Mag, N. Palomeras, and M. Carreras, "The girona 500, a multipurpose autonomous underwater vehicle," in *Proceedings of OCEANS 2011 IEEE - Spain*, 2011.

[13] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 3, no. 2, 2009, p. 5.

[14] P. Di Lillo, E. Simetti, D. De Palma, E. Cataldi, G. Indiveri, G. Antonelli, and G. Casalino, "Advanced ROV autonomy for efficient remote control in the DexROV project," *Marine Technology Society Journal*, vol. 50, no. 4, pp. 67–80, 2016.

[15] G. Antonelli, *Underwater robots*, 4th ed. Heidelberg, D: Springer Tracts in Advanced Robotics, Springer-Verlag, 2018.

[16] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[17] S. Moe, G. Antonelli, A. R. Teel, K. Y. Pettersen, and J. Schrimpf, "Set-based tasks within the singularity-robust multiple task-priority inverse kinematics framework: General formulation, stability analysis, and experimental results," *Frontiers in Robotics and AI*, vol. 3, p. 16, 2016.

[18] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 3, pp. 398–410, 1997.

[19] G. Antonelli, F. Arrichiello, and S. Chiaverini, "The null-space-based behavioral control for autonomous robotic systems," *Intelligent Service Robotics*, vol. 1, no. 1, pp. 27–39, 2008.

[20] ——, "The NSB control: a behavior-based approach for multi-robot systems," *Paladyn Journal of Behavioral Robotics*, vol. 1, no. 1, pp. 48–56, 2010.

[21] F. Arrichiello, P. D. Lillo, D. D. Vito, G. Antonelli, and S. Chiaverini, "Assistive robot operated via p300-based brain computer interface," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 6032–6037.

[22] G. Muscio, F. Pierri, M. A. Trujillo, E. Cataldi, G. Giglio, G. Antonelli, F. Caccavale, A. Viguria, S. Chiaverini, and A. Ollero, "Experiments on coordinated motion of aerial robotic manipulators," in *Proceedings IEEE International Conference onRobotics and Automation*, 2016, pp. 1224–1229.