2020 28th Mediterranean Conference on Control and Automation (MED)
15-18 September, 2020 - Saint-Raphaël, France

# Dynamic Configuration for an Autonomous Underwater Robot

Huu-Tho Dang, Lionel Lapierre, Rene Zapata, Pascal Lepinay, Benoit Ropars

*Abstract*— **This paper presents an Autonomous Underwater Vehicle (AUV) with a dynamic configuration of its actuation. The AUV is able to modify its configuration online (at each sampling time) during its missions. A procedure to optimize the dynamic configuration with respect to energy-like criterion is introduced. The simulation results are shown to prove the efficiency of a dynamic management of the actuation configuration.**

## I. Introduction

Nowadays, ocean researches have tremendously progressed because of new technologies, such as sensor techniques, electronic devices, machine learning algorithms. Many models of Remotely Operated Vehicle (ROV), Autonomous Underwater Vehicle (AUV) have been developed in order to discover underwater environments [1] [2], from under-actuated vehicles, i.e. torpedo shape for long-range missions, to over-actuated vehicles [3] [4], i.e. symmetrical shape for station keeping or local environment observation. However, all these underwater robots have fixed configurations and their controllers have been designed to follow specified configurations. To increase the versatility of underwater vehicles, the idea of reconfigurable robots has appeared. The early studies on dynamically reconfigurable system on mobile robot were introduced in [5]. These reconfigurable robots were designed as modular parts which can connect or disconnect to main body with respect to mission's requirements. Readers can found a brief review and challenges of reconfigurable mobile robots in [6] [7].

In the domain of underwater robots, a guidance and control approach for an Unmanned Underwater Vehicle (UUV), with a dynamic configuration, was introduced in [8]. However, the dynamic configuration has been not analyzed clearly. In [9], a robotic fish with undulating fins was developed, nevertheless, it does not have a reconfigurable capability during its operation, just varying design parameters to achieve an another version of the robot. Another reconfigurable robotic fish was introduced in [10] and it was designed as waterproofed modules combination separately. This can help to build robotic fish in different morphologies. But, this is just also a static reconfigurable configuration. An Autonomous Underwater Vehicle for Intervention (RAUVI project) was presented in [11]. This is an AUV equipped with one manipulator which allows to perform manipulation tasks. The robot, inherited from Girona 500 AUV [12], is statically reconfigured with respect to different tasks. A prototype of a reconfigurable underwater robot with bioinspired electric sense was introduced in [13]. A static reconfigurable underwater robot, namely *SeaDrone*, was proposed in [14]. Four configurations of the robot corresponding with four underwater tasks were shown. However, this configuration is chosen before and remains static during the mission. An AUV/ROV for man-robot underwater cooperation was depicted in [15]. This is actually a ROV and is possible to change into an AUV by dropping the cable and it is also a static actuation system. Moreover, it can be mechanically changed to six possible layouts. The studies to find static optimal configuration by heuristic optimization was introduced in [16] [17]. A genetic algorithm was used to search a static optimal configuration which takes into account specified missions. A ROV from SubseaTech [18] can modify the actuator's angles but this mechanism is not shown clearly. Bio-inspired robots can be also considered as self-reconfigruable robots, i.e. snake robots [19]. In order to increase the versatility and reduce the cost of building underwater vehicles (one robot can carry out several tasks with different configurations), a dynamically reconfigrable configuration of robots is needed and attractive.

In general, the control diagram of an underwater robot is shown in Figure 1(a). The separation of control law and control allocation is useful to exploit the advantages of actuator redundancy [20]. For dynamically reconfigurable robots, challenging questions arise and concern: control allocation, control law adaptation and adaptation to modification of dynamic properties (e.g. modification of the meta-center). For simplicity, in this paper, we assume that centers of mass and buoyancy are not changing during the mission. The control system outputs a desired force, also called actuation demand ($\mathbf{F}_B^d$, expressed in the body-frame) which explicitly considers system dynamics and kinematic properties. It is then the role of control allocation to compute actuators inputs ($\mathbf{F}_m$) whose resulting action ($\mathbf{F}_B$) should realize the actuation demand ($\mathbf{F}_B = \mathbf{F}_B^d$). The relation between controller's output ($\mathbf{F}_B^d$) and the actuation inputs ($\mathbf{F}_m$) of a system is described by a mapping, called control allocation. This mapping is described by a *configuration matrix*. Our paper focuses on the question of the optimal adaptation of the actuation configurations with respect to energy-like criterion.

The contributions of the paper are as followings:

1) Investigate the effects of dynamic configuration matrix on control allocation methods

2) Propose an optimal algorithm to reduce the energy of dynamically reconfigurable robot during its operations

The paper is organized as follows. The notations is shown in section II. Backgrounds of underwater robots and reconfig-

Huu-Tho Dang, Lionel Lappierre, Rene Zapata, Pascal Lepinay, and Benoit Ropars are with LIRMM institute, University of Montpellier, France. email:{tho.dang-huu, lapierre, zapata, lepinay, ropars}@lirmm.fr

The work in this paper was supported by Labex NUMEV, Region Occitanie, FEDER, and MUSE

978-1-7281-5742-9/20/$31.00 ©2020 IEEE

urable ability of our robot is presented in section III. Section IV depicts the control allocation problem with dynamic configuration. The optimal reconfigurable configuration problem is deliberated in section V. Optimal online algorithm is presented in section VI . Simulation results are shown in section VII. Finally, conclusion is in section VIII.

## II. Notations

This section depicts the notations used in the whole paper. However, further notations will be introduced when needed.

| | |
|---|---|
| $\mathbf{A}$ | Configuration matrix |
| $\mathbf{u}_i^B$ | $(3 \times 1)$- unit vector of direction of the $i^{th}$ thruster w.r.t body-frame |
| $\mathbf{r}_i^B$ | $(3 \times 1)$- unit vector of position of the $i^{th}$ thruster w.r.t body-frame |
| $\mathbf{F}_m$ | $(m \times 1)$- Force vector of m thrusters |
| $F_{m,i}$ | Force magnitude of the $i^{th}$ thruster |
| $\mathbf{F}_B^d$ | $(6 \times 1)$- Desired force (force and torque) w.r.t body frame |
| $\mathbf{F}_B = \binom{\mathbf{f}}{\boldsymbol{\tau}}$ | $(6 \times 1)$- actuator resulting force (force and torque) w.r.t body frame |
| $\otimes$ | Cross product |
| $\lVert \cdot \rVert$ | Euclidian norm |
| $m$ | the number of thrusters |
| $n$ | the number of degree of freedoms (DOFs) |

## III. Background

An autonomous underwater robot with dynamic configuration is being developed at LIRMM and Polytech, Montpellier university, France, called Umbrella Robot (UmRobot) for karst exploration, as in Figrue 1(b). The robot can change configurations corresponding with its missions, i.e. path following, surface survey. The UmRobot carries 7 thrusters with 3 in foreward branch and 4 in backward branch. We can modify the positions and directions of these thrusters by changing two angles ($\alpha_F$ and $\alpha_B$) as in Figure 1(c). Therefore, the actuation configuration of UmRobot is dynamic and its versatility can be achieved by the dynamical configuration (e.g. from an under-actuated system to an over-actuated system and vice versa).

### A. Robot model

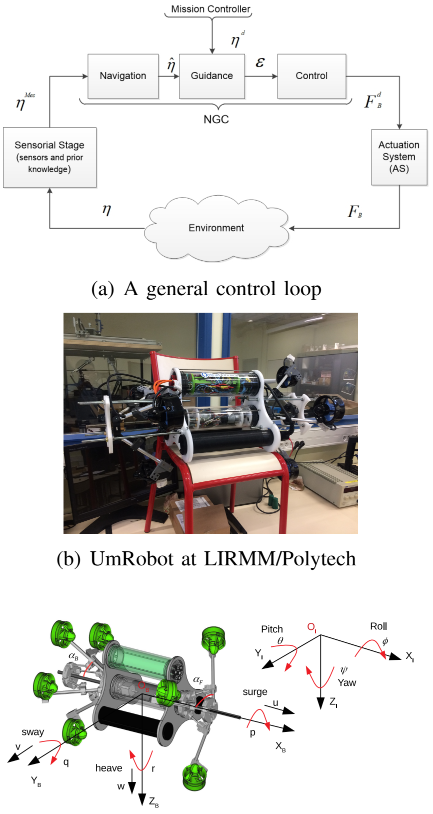
In general, the kinematic and dynamic model of a marine vehicle are as follows [21]:

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{1}$$

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\nu + \mathbf{g}(\boldsymbol{\eta}) = \mathbf{F}_B \tag{2}$$

where state vector $\boldsymbol{\eta} = [x \quad y \quad z \quad \phi \quad \theta \quad \psi]^T$, velocity vector $\boldsymbol{\nu} = [u \quad v \quad w \quad p \quad q \quad r]^T$ as in Figure 1(c), and $\mathbf{M}$, $\mathbf{C}$, $\mathbf{D}$, $\mathbf{g}$ are Mass, Coriolis-Centrifuge, Damping, Gravity-buoyancy matrices respectively. $\mathbf{F}_B$ consists of forces/torques applied on robot by its actuation system (external factors such as wind or/and wave are neglected).

The dynamic parameters of the robot can be approximately computed by a numerical software, i.e. ANSYS and real tests.



(a) A general control loop

(b) UmRobot at LIRMM/Polytech

(c) General notations

Fig. 1: General control loop and Umbrella robot

### B. Model of dynamic configuration matrix

The relation between forces vector, including force and torque elements, in body-frame $\mathbf{F}_B$ and force vector applied on each thrusters is described through the *configuration matrix*, denoted as $\mathbf{A}$, which describes the geometric organization of thrusters in body-frame:

$$\mathbf{F}_B = \mathbf{A}(\alpha_F, \alpha_B)\mathbf{F}_m = \begin{pmatrix} \mathbf{f} \\ \boldsymbol{\tau} \end{pmatrix} \tag{3}$$

where $\mathbf{F}_B = [F_u \quad F_v \quad F_w \quad \tau_p \quad \tau_q \quad \tau_r]^T \in \mathbb{R}^6$, $\mathbf{A} \in \mathbb{R}^{6 \times m}$, and $\mathbf{F}_m = [F_{m,1} \quad F_{m,2} \quad \dots \quad F_{m,m}]^T \in \mathbb{R}^m$, $m$ is the number of thrusters, $m = 7 > 6$, since the system has 6 DOF with 7 actuators, the actuation system is said to be redundant. From the scheme of the Umbrella Robot (UmRobot), the configuration matrix is as follows:

$$\begin{aligned} \mathbf{A} &= \begin{pmatrix} \mathbf{u}_1^B & \mathbf{u}_2^B & \cdots & \mathbf{u}_m^B \\ \mathbf{r}_1^B \otimes \mathbf{u}_1^B & \mathbf{r}_2^B \otimes \mathbf{u}_2^B & \cdots & \mathbf{r}_m^B \otimes \mathbf{u}_m^B \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{u}_1^B & \mathbf{u}_2^B & \cdots & \mathbf{u}_m^B \\ \boldsymbol{\tau}_1^B & \boldsymbol{\tau}_2^B & \cdots & \boldsymbol{\tau}_m^B \end{pmatrix} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix} \end{aligned} \tag{4}$$

where $m = 7$ and $\mathbf{u}_1^B, \dots, \mathbf{u}_7^B$ and $\mathbf{r}_1^B, \dots, \mathbf{r}_7^B$ are shown in the Appendix. The basic idea for computing matrix $\mathbf{A}$ is to use transformation matrices between coordinate systems. Because of page limitation, this computation is not shown in this paper. When two angles, $\alpha_F$ and $\alpha_B$ (Figure 1(c)), are varied, the UmRobot's configuration is changed.

### C. Control allocation methods

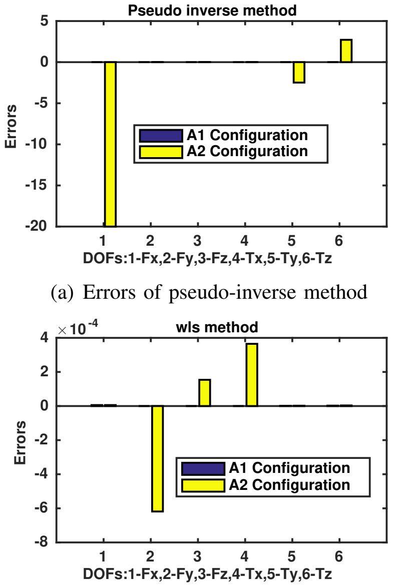
With a given configuration matrix, $\mathbf{A}$, and desired actuation demand $\mathbf{F}_B^d$, how to find the control forces of actuators $\mathbf{F}_m$ which satisfy Equation (3) and constraints of actuators (saturation and dead-zone). That is control allocation problem.

Normally, there are two classes of control allocation (CA) methods. The first one is based on the Moore-Penrose pseudo inverse including direct method, daisy chain method [22], cascade general inverse (CGI) [23], and the second one is based on optimization techniques such as sequential least square, minimal least square, and fixed-point method, nonlinear programming method [24]. However, one class of CA methods based on neural network [25] has been proposed recently. Many studies have been published to solve the control allocation problem. Readers can refer to [20] and references therein for more details.

## IV. Dynamic Control allocation-The singularities

As aforementioned, there are many approaches to solve CA problem. However, in all these cases, the configuration matrix is constant and remains unchanged during robot's operation. For our robot, the configuration matrix is varying and the properties of this matrix has to be studied carefully, because it yields to a singular configuration or nearly-singular configuration, i.e. the actuation is redundant but results in an under-actuated system (some DOFs are not controllable). The case of nearly-singularity, the minimum singular value of configuration matrix is too small. This yields that the pseudo-inverse is too big (it is easy to see with the SVD decomposition of the matrix $\mathbf{A}$) and causes the big error if the pseudo-inverse based CA methods are used. One solution is that we can avoid the nearly-singular situation by neglecting too small singular values. However, this causes the error and not suitable for dynamic configuration because configuration matrix can move from singular to nearly-singular configuration. We can verify this problem by an example of our robot. At $\alpha_F = \alpha_B = 45^0$, the configuration matrix $\mathbf{A}_1$ is singular and the robot can only go along x-axis, and at $\alpha_F = \alpha_B = 45.02^0$ the configuration matrix changes to $\mathbf{A}_2$, a nearly-singular matrix, and the robot is also controlled along x-axis. We investigate the errors of control allocation problem of two configurations with different control allocation methods including pseudo inverse based and nonlinear programming based approaches [24].

From Figure 2, when two angles $\alpha_F$ and $\alpha_B$ change a small value, the errors grow significantly for control allocation methods which based on pseudo-inverse. The errors also remain for CA methods based on nonlinear programming, however, these values are quite small because these methods avoid nearly-singularity of configuration matrix. It is easy to see that the nonlinear programming based control allocation methods are suitable for the reconfigurable Umbrella Robot.



(a) Errors of pseudo-inverse method

(b) Errors of wls method

Fig. 2: Errors of pseudo-based CA method and nonlinear programming based one

## V. Dynamic configuration problem

In this section, we consider a dynamic configuration problem with respect to an energy-like criterion which is defined as the norm of force vector, $\mathbf{F}_m$, applied on thrusters. This is reasonable thanks to the nearly-linear characteristics of motors. The problem is formulated as:

$$\min_{\alpha_F,\alpha_B,\mathbf{F}_m} \quad J = \|\mathbf{F}_m\|^2 \tag{5a}$$

$$s.t \quad 45^0 \leq \alpha_F, \alpha_B \leq 90^0 \tag{5b}$$

$$\mathbf{F}_m \in \mathbb{F} \tag{5c}$$

$$\mathbf{F}_B^d - \mathbf{A}(\alpha_F, \alpha_B)\mathbf{F}_m = 0 \tag{5d}$$

where $\mathbf{F}_B^d$ is desired forces vector (from the controller), $\mathbb{F}$ is a feasible set of thrusters forces. The constraint (5b) is mechanical limitations of the Umbrella robot.

The objective is to find two angles, $\alpha_F, \alpha_B$, and force vector $\mathbf{F}_m$ in order to minimize an energy-like function $J$ and satisfy constraints. This is a nonlinear optimization problem and is solved for each sampling time (online) because the desired force vector $\mathbf{F}_B^d$ is changed in each time step in general case. In our problem, the configuration matrix $\mathbf{A}$ is dynamic and belongs to two angles, $\alpha_F$ and $\alpha_B$.

## VI. Proposed solution

This section proposes an approach to solve online the problem (5). The constraint (5d) is strictly and not easy to solve, therefore we relax this constraint by adding it to the objective function with a weighting matrix. This problem can be rewritten as:

$$\min_{\alpha_F,\alpha_B,\mathbf{F}_m} J = w_1\|\mathbf{F}_m\|^2 + w_2\|(\mathbf{F}_B^d - \mathbf{A}(\alpha_F,\alpha_B)\mathbf{F}_m)\|^2 \tag{6a}$$

$$s.t \quad 45^0 \leq \alpha_F, \alpha_B \leq 90^0 \tag{6b}$$

$$\mathbf{F}_m \in \mathbb{F} \tag{6c}$$

where $w_1$ and $w_2$ are weighting scalars.

By denoting a vector $\mathbf{x} = [\alpha_F \quad \alpha_B \quad \mathbf{F}_m]^T$, the problem can be formulated as:

$$\min_{\mathbf{x}} f(\mathbf{x}) = w_1\|\mathbf{x}\|^2 + w_2\|(\overline{\mathbf{F}}_B^d - \overline{\mathbf{A}}(\mathbf{x})\mathbf{x})\|^2 \tag{7a}$$

$$s.t \quad \mathbf{x} \in \mathbb{X} \tag{7b}$$

where $\mathbb{X}$ is a box-constraint including limitations of two angles and saturations of thrusters, $\overline{\mathbf{A}}(\mathbf{x}) = [\mathbf{0}^{n\times 1} \quad \mathbf{0}^{n\times 1} \quad \mathbf{A}]$, $\overline{\mathbf{F}}_B^d = [0 \quad 0 \quad (\mathbf{F}_B^d)^T]^T$.

The problem (7) can be rewritten in a compact form as:

$$\min_{\mathbf{x}} f(\mathbf{x}) \tag{8a}$$

$$s.t \quad \mathbf{c}_i^T\mathbf{x} \leq b_i, i = \overline{1...m} \tag{8b}$$

where $\mathbf{c}_i$ is a proper vector, $m$ is the number of constraints.

In this paper, we propose a A-SQP (Accelerating-Sequential Quadratic Programming) algorithm to solve online our problem. This is based on active-set SQP approach which is an efficient method for small and medium nonlinear problem [26].

*Definition 6.1:* An active set of (8) is a set of constraints indices such that $\mathbf{c}_i^T\mathbf{x} = b_i$. Specifically, $\mathcal{A} = \{i|\mathbf{c}_i^T\mathbf{x} = b_i\}$

Suppose the active set of (8) is given, we consider our problem only with equality constraints as:

$$\min_{\mathbf{x}} f(\mathbf{x}) \tag{9a}$$

$$s.t \quad \mathbf{c}_i^T\mathbf{x} = b_i, i \in \mathcal{A} \tag{9b}$$

The Lagrangian function of problem (9) is $L(\mathbf{x},\boldsymbol{\lambda}) = f(\mathbf{x}) + \sum_{i\in\mathcal{A}}\lambda_i(\mathbf{c}_i^T\mathbf{x} - b_i)$ where $\boldsymbol{\lambda} = [\lambda_i]$ is the vector of corresponding multipliers. The optimal KKT (Karush-Kuhn-Tucker) condition of this problem at local optimal point $(\mathbf{x},\boldsymbol{\lambda})$ can be written as:

$$\nabla_{\mathbf{x}} L(\mathbf{x},\boldsymbol{\lambda}) = \nabla f(\mathbf{x}) + \sum_{i\in\mathcal{A}} \lambda_i \mathbf{c}_i = \mathbf{0} \tag{10a}$$

$$\mathbf{c}_i^T\mathbf{x} = b_i, i \in \mathcal{A} \tag{10b}$$

The problem (10) can be rewritten as a compact form:

$$F(\mathbf{x},\boldsymbol{\lambda}) = \begin{bmatrix} \nabla f(\mathbf{x}) + \mathbf{C}^T\boldsymbol{\lambda} \\ \mathbf{C}\mathbf{x} - \mathbf{b} \end{bmatrix} = \mathbf{0} \tag{11}$$

where $\mathbf{C} = [\mathbf{c}_i^T], \quad \boldsymbol{\lambda} = [\lambda_i] \quad \mathbf{b} = [b_i], \quad i \in \mathcal{A}$.

Using Newton's method, the Jacobian of (11) with respect to $(\mathbf{x},\boldsymbol{\lambda})$ is given by:

$$F'(\mathbf{x},\boldsymbol{\lambda}) = \begin{bmatrix} \nabla^2 L(\mathbf{x}) & \mathbf{C}^T \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \tag{12}$$

The direction for the next step of problem (9) is a solution of (11) and is computed as:

$$\begin{bmatrix} \nabla^2 L(\mathbf{x}) & \mathbf{C}^T \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{p}_{x_k} \\ \mathbf{p}_{\lambda_k} \end{bmatrix} = \begin{bmatrix} -\nabla f(\mathbf{x}) - \mathbf{C}^T\boldsymbol{\lambda} \\ -\mathbf{C}\mathbf{x} + \mathbf{b} \end{bmatrix} \tag{13}$$

At each iteration, we have an active set, called a working set $\mathcal{W}_k$. One constraint can be added or eliminated through the working set. This can be done by checking the sign of Lagrange multipliers $\lambda_i \geq 0$ thanks to KKT conditions of (8) which can be referred in [26]. If we correctly identify the optimal active set then our problem will converge rapidly by Newton's method as the aforementioned analysis. However, optimal active set is not easy to determine.

Thanks to the efficient and popular of convex quadratic programming which can converge in milliseconds [27], at step $k$ with $\mathbf{x}_k$, we consider another quadratic problem:

$$\min_{\mathbf{p}} \frac{1}{2}\mathbf{p}^T\nabla^2_{\mathbf{xx}}L\mathbf{p} + \nabla f^T\mathbf{p} \tag{14a}$$

$$s.t \quad \mathbf{C}\mathbf{p} = -\mathbf{C}\mathbf{x}_k + \mathbf{b} \tag{14b}$$

The optimality conditions of (14) with Lagrange multipliers $\boldsymbol{\lambda}_q$:

$$\nabla^2_{\mathbf{xx}}L\mathbf{p} + \nabla f + \mathbf{C}^T\boldsymbol{\lambda}_{\boldsymbol{q}} = \mathbf{0} \tag{15a}$$

$$\mathbf{C}\mathbf{p} - \mathbf{b} + \mathbf{C}\mathbf{x}_k = \mathbf{0} \tag{15b}$$

This is equivalent with:

$$\begin{bmatrix} \nabla^2 L & \mathbf{C}^T \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \boldsymbol{\lambda}_{\boldsymbol{q}} - \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\nabla f - \mathbf{C}^T\boldsymbol{\lambda} \\ -\mathbf{C}\mathbf{x}_k + \mathbf{b} \end{bmatrix} \tag{16}$$

It is easy to see that (13) and (16) are almost the same except that:

$$\mathbf{p}_{\lambda_k} = \boldsymbol{\lambda}_{\boldsymbol{q}} - \boldsymbol{\lambda} \tag{17}$$

In order to find the direction $[\mathbf{p}_x \quad \mathbf{p}_\lambda]^T$, instead of solving (13), we solve quadratic problem (14) and update Lagrange multipliers as (17). Moreover, Hessian matrix $\nabla^2 L$ is approximated by BFGS (Broyden-Fletcher-Goldfarb-Shanno) formula [26] to make sure it is positive definite or semidefinite and problem (14) is quadratic convex problem in which a local optimal solution is also a global optimal one.

The idea is extended to the problem with inequality (8) by solving the sub-quadratic programming as follows to find the direction:

$$\min_{\mathbf{p}} \frac{1}{2}\mathbf{p}^T\nabla^2_{\mathbf{xx}}L\mathbf{p} + \nabla f^T\mathbf{p} \tag{18a}$$

$$s.t \quad \mathbf{C}\mathbf{p} \leq -\mathbf{C}\mathbf{x}_k + \mathbf{b} \tag{18b}$$

This is possible because the set of active constraints $\mathcal{A}_k$ at the solution of (18) constitutes the guess of the active set at the solution of the nonlinear program (8). Algorithm 1 shows our procedure to solve online dynamic configuration.

**Algorithm 1** A-SQP optimal configuration algorithm

```
Input: desired control inputs $\mathbf{F}_B^d$ (from controller)
Output: Optimal angles $\alpha_F$, $\alpha_B$ and thruster forces $\mathbf{F}_m$
Initialization: primal-dual parameters $(\mathbf{x}_0, \boldsymbol{\lambda}_0)$, Hessian approximation $\mathbf{B}_0$
for $k = 1 <= 2$ do
    Evaluate $f$, $\nabla f$ at $\mathbf{x}_k$
    Compute direction $\mathbf{p}_k$ by solving (18) and have corresponding Lagrange multipliers $\boldsymbol{\lambda}_{\boldsymbol{q_k}}$
    Compute direction $\mathbf{p}_{\lambda_k} = \boldsymbol{\lambda}_{\boldsymbol{q_k}} - \boldsymbol{\lambda}$
    Choose step length $\alpha_k = 1$ (maximize the direction)
    Update $\mathbf{x}_k = \mathbf{x}_k + \alpha \mathbf{p}_k$ and $\boldsymbol{\lambda_k} = \boldsymbol{\lambda_k} + \alpha \mathbf{p}_{\lambda_k}$
    Set $\mathbf{s}_k \leftarrow \alpha_k \mathbf{p}_k$ and $\mathbf{y}_k \leftarrow \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}_{k+1}, \boldsymbol{\lambda}_{k+1}) - \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}_k, \boldsymbol{\lambda}_{k+1})$
    Update $\mathbf{B}_{k+1}$ using BFGS formula
end for
```

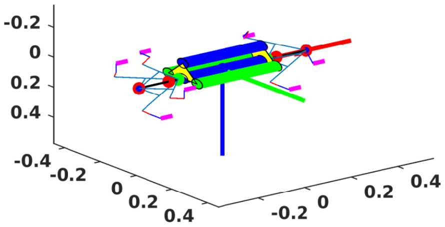


Fig. 3: Simulation robot

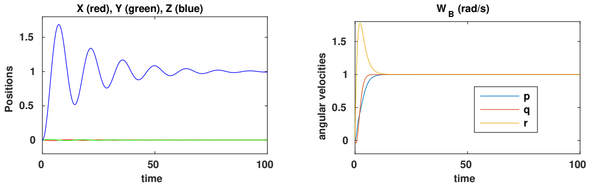
## VII. Simulations

We present the simulation results of the Umbrella robot in which the robot dives to constant depth with desired angular velocities, i.e. $X(m) = [x \quad y \quad z]^T = [0 \quad 0 \quad 1]^T$ and $\omega_B(rad/s) = [p \quad q \quad r]^T = [1 \quad 1 \quad 1]^T$. The model of simulation robot is shown in Figure 3. The controller is designed with quaternion techniques [28]. The simulations include fixed and dynamic configurations. For dynamic cases, we compare our algorithm with *Fmincon* function in Matlab Toolbox. The simulation results of fixed configuration are shown in Figure 4 in which two angles $\alpha_F = \alpha_B = 90^0$. The simulation results of dynamic configuration are depicted in Figure 5 and 6. Note that in simulation, we assume that all states of robot can be measured completely. Moreover, warm start technique is used to accelerate the computational time. To evaluate several approaches in which saturation and dead-zone of thrusters are also taken into account, the energy-like criterion is computed as:

$$E = \int \mathbf{c}_m(t)dt \tag{19}$$

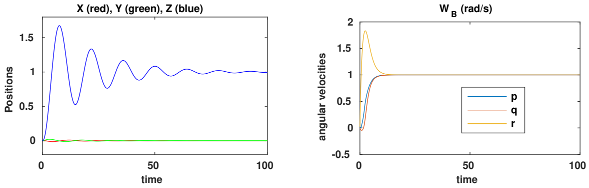
where $\mathbf{c}_m(t)$ is a function of $\mathbf{F}_m$ which is the inverse-characteristic of thrusters.

It is obvious to see that the control performances are guaranteed in static and dynamic configurations. However, the energy-like criterion is different. Specifically, the energy-like criterion of simulations cases which guarantee the control performances is illustrated in Figure 7(a). The dynamic configuration cases (with Fmincon and A-SQP) outperform static configuration one.



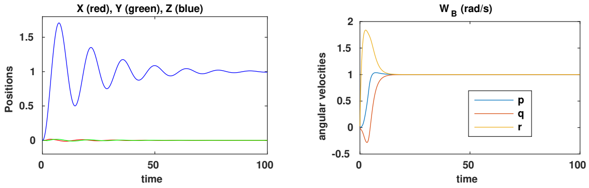
(a) Positions of Robot (b) Angular velocities

Fig. 4: Simulation results with a fixed configuration



(a) Positions of Robot (b) Angular velocities

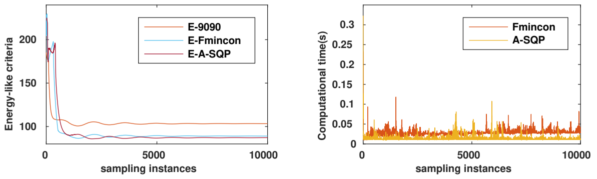
Fig. 5: Simulation results with dynamic configuration (with Fmincon)



(a) Positions of Robot (b) Angular velocities

Fig. 6: Simulation results with dynamic configuration (A-SQP)

Our algorithm shows the same performance in comparison with *Fmincon* function. A comparison of computational time between methods including A-SQP, *Fmincon* function at each sampling time is shown in Figure 7(b). With the same energy-like criterion, we can see that A-SQP outperforms in computational time and is suitable for real test.



(a) Energy-like criterion evolution (b) Computational time

Fig. 7: Energy-like criterion and computational time comparison

## VIII. Conclusions

In this paper, an AUV with a dynamic configuration, called Umbrella robot, is presented. It's an over-actuated system, and then relating control problems, i.e., control allocation and dynamic configuration policy, are investigated. First, dynamic control allocation problem is discussed and compared. Secondly, a fast algorithm based on sequential quadratic programming, called A-SQP, is proposed to modify the configuration of Umrobot. The simulation results show that A-SQP can guarantee good performances and suitable for the real test. In the future researches, we will implement our theory in the real UmRobot which is being developed at Montpellier university. Furthermore, properties of A-SQP algorithm such as contraction and convergence rate are also further studies.

## IX. Acknowledgements

The authors thank to Labex NUMEV, Region Occitanie, FEDER, and MUSE for supporting this research.

## Appendix

In this appendix, we will present the elements of configuration matrix, $\mathbf{A}$, in which $d$ and $d_i$ are constant values.

| | |
|---|---|
| $\mathbf{u}_1^B = \begin{pmatrix} \frac{1}{\sqrt{2}}(sin\alpha_F + cos\alpha_F) \\ \frac{\sqrt{3}}{2\sqrt{2}}(-cos\alpha_F + sin\alpha_F) \\ \frac{1}{2\sqrt{2}}(-cos\alpha_F + sin\alpha_F) \end{pmatrix}$ | $\mathbf{r}_\mathbf{1}^\mathbf{B} = \begin{pmatrix} (d_F + L_F) - \sqrt{d_e^2 + d^2}.cos(\alpha_F + \beta_1) \\ -\frac{d_t}{2} - \frac{\sqrt{3(d_e^2 + d^2)}sin(\alpha_F + \beta_1)}{2} \\ \frac{\sqrt{3}}{2}d_t - \frac{\sqrt{d_e^2 + d^2}sin(\alpha_F + \beta_1)}{2} \end{pmatrix}$ |
| $\mathbf{u}_2^B = \begin{pmatrix} \frac{cos\alpha_F + sin\alpha_F}{\sqrt{2}} \\ 0 \\ \frac{cos\alpha_F - sin\alpha_F}{\sqrt{2}} \end{pmatrix}$ | $\mathbf{r}_2^B = \begin{pmatrix} (d_F + L_F) - \sqrt{d^2 + d_e^2}cos(\alpha_F + \beta_2)) \\ d_t \\ \sqrt{d^2 + d_e^2}sin(\alpha_F + \beta_2) \end{pmatrix}$ |
| $\mathbf{u}_3^B = \begin{pmatrix} \frac{1}{\sqrt{2}}(sin\alpha_F + cos\alpha_F) \\ \frac{\sqrt{3}}{2\sqrt{2}}(cos\alpha_F - sin\alpha_F) \\ \frac{1}{2\sqrt{2}}(sin\alpha_F - cos\alpha_F) \end{pmatrix}$ | $\mathbf{r}_3^B = \begin{pmatrix} (d_F + L_F) - \sqrt{d_e^2 + d^2}.cos(\alpha_F + \beta_3) \\ \frac{d_t}{2} + \frac{\sqrt{3(d_e^2 + d^2)}}{2}.sin(\alpha_F + \beta_3) \\ \frac{\sqrt{3}}{2}d_t - \frac{\sqrt{d_e^2 + d^2}}{2}sin(\alpha_F + \beta_3) \end{pmatrix}$ |
| $\mathbf{u}_4^B = \begin{pmatrix} \frac{cos\alpha_B + sin\alpha_B}{\sqrt{2}} \\ \frac{-cos\alpha_B + sin\alpha_B}{\sqrt{2}} \\ 0 \end{pmatrix}$ | $\mathbf{r}_4^B = \begin{pmatrix} -(d_F + \sqrt{d^2 + d_e^2}cos(\alpha_B + \beta_4)) \\ -(\sqrt{d^2 + d_e^2}sin(\alpha_B + \beta_4)) \\ -dt \end{pmatrix}$ |
| $\mathbf{u}_5^B = \begin{pmatrix} \frac{cos\alpha_B + sin\alpha_B}{\sqrt{2}} \\ 0 \\ \frac{cos\alpha_B - sin\alpha_B}{\sqrt{2}} \end{pmatrix}$ | $\mathbf{r}_5^B = \begin{pmatrix} -(d_F + \sqrt{d^2 + d_e^2}cos(\alpha_B + \beta_5)) \\ d_t \\ \sqrt{d^2 + d_e^2}sin(\alpha_B + \beta_5) \end{pmatrix}$ |
| $\mathbf{u}_6^B = \begin{pmatrix} \frac{cos\alpha_B + sin\alpha_B}{\sqrt{2}} \\ \frac{cos\alpha_B - sin\alpha_B}{\sqrt{2}} \\ 0 \end{pmatrix}$ | $\mathbf{r}_6^B = \begin{pmatrix} -(d_F + \sqrt{d^2 + d_e^2}cos(\alpha_B + \beta_6)) \\ \sqrt{d^2 + d_e^2}sin(\alpha_B + \beta_6) \\ -d_t \end{pmatrix}$ |
| $\mathbf{u}_7^B = \begin{pmatrix} \frac{cos\alpha_B + sin\alpha_B}{\sqrt{2}} \\ 0 \\ \frac{-cos\alpha_B + sin\alpha_B}{\sqrt{2}} \end{pmatrix}$ | $\mathbf{r}_7^B = \begin{pmatrix} -(d_F + \sqrt{d^2 + d_e^2}cos(\alpha_B + \beta_7)) \\ -d_t \\ -(\sqrt{d^2 + d_e^2}sin(\alpha_B + \beta_7)) \end{pmatrix}$ |

TABLE I: Elements of **A** matrix

## References

[1] J. Yuh, "Design and control of autonomous underwater robots: A survey," *Autonomous Robots*, vol. 8, no. 1, pp. 7–24, 2000.

[2] E. Zereik, M. Bibuli, N. Mišković, P. Ridao, and A. Pascoal, "Challenges and future trends in marine robotics," *Annual Reviews in Control*, 2018.

[3] B. Ropars, L. Lapierre, A. Lasbouygues, D. Andreu, and R. Zapata, "Redundant actuation system of an underwater vehicle," *Ocean Engineering*, vol. 151, pp. 276 – 289, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0029801817307473

[4] H.-T. Dang, L. Lapierre, R. Zapata, P. Lepinay, and B. Ropars, "Configuration matrix design of over-actuated marine systems," in *OCEANS 2019-Marseille*. IEEE, 2019, pp. 1–10.

[5] D. Schmitz, "The cmu reconfigurable modular manipulator system," 1988.

[6] M. Yim, W.-M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian, "Modular self-reconfigurable robot systems [grand challenges of robotics]," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 43–52, 2007.

[7] K. Stoy, D. Brandt, and D. J. Christensen, *Self-reconfigurable robots: an introduction*. Mit Press, 2010.

[8] M. Caccia and G. Veruggio, "Guidance and control of a reconfigurable unmanned underwater vehicle," *Control engineering practice*, vol. 8, no. 1, pp. 21–37, 2000.

[9] K. Low and J. Yu, "Development of modular and reconfigurable biomimetic robotic fish with undulating fin," in *2007 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2007, pp. 274–279.

[10] Y. Hu, L. Wang, W. Zhao, Q. Wang, and L. Zhang, "Modular design and motion control of reconfigurable robotic fish," in *2007 46th IEEE Conference on Decision and Control*. IEEE, 2007, pp. 5156–5161.

[11] M. Prats, D. Ribas, N. Palomeras, J. C. García, V. Nannen, S. Wirth, J. J. Fernández, J. P. Beltrán, R. Campos, P. Ridao *et al.*, "Reconfigurable auv for intervention missions: a case study on underwater object recovery," *Intelligent Service Robotics*, vol. 5, no. 1, pp. 19–31, 2012.

[12] D. Ribas, P. Ridao, L. Magí, N. Palomeras, and M. Carreras, "The girona 500, a multipurpose autonomous underwater vehicle," in *Oceans 2011 IEEE-Spain*. IEEE, 2011, pp. 1–5.

[13] S. Mintchev, C. Stefanini, A. Girin, S. Marrazza, S. Orofino, V. Lebastard, L. Manfredi, P. Dario, and F. Boyer, "An underwater reconfigurable robot with bioinspired electric sense," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 1149–1154.

[14] E. Moreno and S.-Y. Chung, "Seadrone: A modular and reconfigurable underwater robot for task optimization," in *OCEANS 2014-TAIPEI*. IEEE, 2014, pp. 1–7.

[15] A. Odetti, M. Bibuli, G. Bruzzone, M. Caccia, E. Spirandelli, and G. Bruzzone, "e-urope: a reconfgurable auv/rov for man-robot underwater cooperation," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 11 203–11 208, 2017.

[16] O. Chocron, E. P. Vega, and M. Benbouzid, "Dynamic reconfiguration of autonomous underwater vehicles propulsion system using genetic optimization," *Ocean Engineering*, vol. 156, pp. 564–579, 2018.

[17] E. P. Vega, "Task-based design and optimization of reconfigurable propulsion systems for autonomous underwater vehicles," Ph.D. dissertation, Université de Bretagne occidentale-Brest, 2016.

[18] SubseaTech, "Tortuga robot." [Online]. Available: https://www.subseatech.com/tortuga/

[19] A. A. Transeth, R. I. Leine, C. Glocker, K. Y. Pettersen, and P. Liljebäck, "Snake robot obstacle-aided locomotion: Modeling, simulations, and experiments," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 88–104, 2008.

[20] T. A. Johansen and T. I. Fossen, "Control allocationa survey," *Automatica*, vol. 49, no. 5, pp. 1087 – 1103, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109813000368

[21] T. I. Fossen, *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2011.

[22] W. C. Durham, "Constrained control allocation," *Journal of Guidance, Control, and Dynamics*, vol. 16, no. 4, pp. 717–725, 1993.

[23] K. A. Bordignon, "Constrained control allocation for systems with redundant control effectors," Ph.D. dissertation, Virginia Polytechnic Institute and State University Blacksburg, 1996.

[24] O. Härkegård, "Backstepping and control allocation with applications to flight control," Ph.D. dissertation, Linköpings universitet, 2003.

[25] R. Skulstad, G. Li, H. Zhang, and T. I. Fossen, "A neural network approach to control allocation of ships for dynamic positioning," *IFAC-PapersOnLine*, vol. 51, no. 29, pp. 128–133, 2018.

[26] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[27] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpoases: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[28] S. Louis, L. Lapierre, K. Godary-Dejean, Y. Onmek, T. Claverie, and S. Villéger, "Quaternion based control for robotic observation of marine diversity," in *OCEANS*, 2017.