Abstract—In this paper we describe the algorithm developed for the motion control of an UVMS (Underwater Vehicle-Manipulator System) within DexROV, a funded EC Horizon 2020 project that proposes to enable the far distance teleoperation of an UVMS via satellite communication. The main use case scenario is the maintenance of Oil&Gas underwater platforms. During subsea oil operations, the robotic system has to be able to navigate to the panel location and manipulate handles, rotate valves, plug and unplug connectors. The DexROV system has two operational modes: fixed-base and floating-base manipulation. In floating-base manipulation the system is not clamped to any structure and it can move both the manipulator and the vehicle at the same time in order to perform the needed tasks. The control relies on a Task Priority Inverse Kinematics framework that has been extended in order to handle both equality-based and set-based tasks. Simulations performed in the Gazebo simulator show the effectiveness of the proposed approach.

I. INTRODUCTION

Underwater robotics is of great interest because it allows to perform tasks that are very dangerous for the man, e.g., oil and gas submarine pipelines maintenance [1], [2]. Underwater Vehicle-Manipulator Systems (UVMSs) can be divided essentially in two main categories: Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs).

ROVs are systems physically linked via theter, used both for power supply and communication, to a surface vessel or a submarine where a human operator has to control and send commands to the vehicle itself. In particular a high level of expertise is required to the human operator in controlling and manoeuvring the vehicle made more difficult by the theter inertia. Thus the man has a fundamental role because he is the control loop.

AUVs are systems with onboard power modules avoiding the theter. The absence of a physical link makes these vehicles more suitable to survey missions. Furthermore they present unmanned control loops contrarily to the ROVS. Thus they are supposed to perform the task for which they are designed without the presence of the man in the loop. However AUVs present a very low power autonomy. Therefore they are not widely used for maintenance-manipulation operations especially inside the Industry but, however, in several scenarios they can be used in combination with ROVs exploiting both ones properties [3].

In this paper we present the algorithm developed for the motion control of an UVMS within the scope of the DexROV project [4] (see Fig. 1).

DexROV is a funded EC Horizon 2020 project that proposes to implement novel operation strategies for underwater semi-autonomous interventions. Currently these operations are performed tele-operating ROVs (Remotely Operated Vehicles) from a support vessel, and require the presence of a significant amount of personnel needed for the effective accomplishment of the intervention. The main goal of DexROV is to move the supervision of the mission onshore, allowing the usage of a much smaller support vessel with a reduced crew, resulting in a major decrease of the costs of the entire operation. The control center is placed onshore, and it is connected to the support vessel via a satellite link. Given the time latency introduced by the communication channel, it is not possible for the pilots to directly teleoperate the system anymore, and it has to exhibit some semi-autonomous capabilities. The control relies on a multi-task priority algorithm that allows the operator to focus only on the operational tasks, while the safety-related and the optimization ones are autonomously handled by the system.

The DexROV system has two operational modes: fixed-base and floating-base manipulation. In floating-base manipulation the system is not clamped to any structure and it can move both the manipulator and the vehicle at the same time in order to perform the needed tasks in a more dexterous way. In this configuration, the system has 10 controllable DOF (Degrees Of Freedom): 4DOF are related to the moving base (the 3 components of the linear velocity and the angular velocity along the z axis), and 6 DOF are the manipulator’s joints.

This redundancy has to be exploited to coordinate the...
motion of the arm and the vehicle in order to perform the manipulation tasks taking into account all the kinematic constraints both in joint space and operational space [5]. For instance the whole system has to move toward the desired point while avoiding singular configurations [6] and collisions with the other arm or with the vehicle itself. In order to do that, it is first necessary to obtain the kinematic model of the vehicle-manipulator system.

This work is organized as follows. In Section II the direct and differential kinematics of an UVMS is briefly reviewed. Section III introduces the Task-Priority Inverse-Kinematics and set-based control for taking into consideration both equality and inequality tasks. In Section IV the simulation setup for the DexROV project is described. Finally, Section V presents the obtained results for two different cases: in the first case a constrained motion of the manipulator is considered; in the second one the manipulator is asked to reach a target valve on a panel.

II. KINEMATICS OF UVMS

A. Direct kinematics

The direct kinematics in case of a vehicle-manipulator system is the process of computing the end-effector position and orientation in inertial frame knowing the \( n \) joint positions and the vehicle state, defined as:

\[
\eta = \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix}
\]  

(1)

where \( \eta_1 = [x \ y \ z]^T \) is the vector of the 3D coordinates of the vehicle position in inertial frame and \( \eta_2 = [\alpha \ \beta \ \gamma]^T \) is its orientation expressed in Euler angles [7]. Defining the end-effector position in the inertial frame \( \eta_{ee,1} \) and its orientation \( \eta_{ee,2} \), the entire pose can be written as:

\[
\eta_{ee} = \begin{bmatrix} \eta_{ee,1} \\ \eta_{ee,2} \end{bmatrix} \in \mathbb{R}^6
\]

and its relation with the system configuration can be expressed as:

\[
\eta_{ee} = k(\eta, q)
\]

(2)

or in matrix form, computing the transformation matrix \( T^{ee}_{e} \) between the inertial frame and the end-effector frame as successive rototranslations as:

\[
T^{I}_{e} = T^{I}_{e}T^{v}_{0}T^{0}_{ee}
\]

(3)

where \( T^{I}_{e} \) is the transformation matrix between the inertial frame and the vehicle frame, \( T^{v}_{0} \) is the constant transformation between the vehicle frame and the arm base frame and \( T^{0}_{ee} \) is the transformation between the arm base frame and the end-effector. Figure 2 shows all the involved frames.

B. Differential kinematics

The differential kinematics is the process of obtaining the end-effector linear and angular velocity \( \dot{\eta}_{ee} \) given the system velocity:

\[
\xi = \begin{bmatrix} \nu_1 \\ \nu_2 \\ \dot{q} \end{bmatrix} \in \mathbb{R}^{6+n}.
\]

(4)

The mapping between the operational space and the system variables space is given by a proper Jacobian matrix:

\[
\dot{\eta}_{ee} = J\xi = \begin{bmatrix} \dot{\eta}_{ee,1} \\ \dot{\eta}_{ee,2} \end{bmatrix} = \begin{bmatrix} J_{pos} \\ J_{ori} \end{bmatrix} \xi,
\]

(5)

with \( J_{pos} \) and \( J_{ori} \) the position and orientation Jacobians of the UVMS, respectively [8], defined as

\[
J_{pos} = \begin{bmatrix} R^{I}_{v} \ - (S(r^{v}_{0}r^{e}_{0}) + S(r^{0}_{0}q_{0,ee})R^{I}_{v} J_{pos,1} \end{bmatrix}
\]

(6)

\[
J_{ori} = \begin{bmatrix} O \ R^{I}_{v} J_{ori,1} \end{bmatrix}
\]

(7)

where:

- \( R^{I}_{v} \) is the rotation matrix between the inertial frame and the vehicle frame
- \( r^{v}_{0} \) is the vector connecting the origin of the vehicle frame and the origin of the arm base frame, expressed in vehicle frame
- \( S \) is the skew symmetric matrix
- \( q_{0,ee} \) is the end effector position expressed in the arm base frame
- \( J_{pos,1} \) is the position Jacobian of the arm with respect to the inertial frame, expressed as:

\[
J_{pos,1} = R^{I}_{0}J_{pos,0}
\]

- \( J_{ori,1} \) is the orientation Jacobian of the arm with respect to the inertial frame, expressed as:

\[
J_{ori,1} = R^{I}_{0}J_{ori,0}
\]

Figure 3 shows the variables needed for the Jacobian computation.

III. SET-BASED CONTROL

For an UVMS the state is described by the vector \( s = [\eta_1, \eta_2, q]^T \in \mathbb{R}^{6+n} \) in which \( \eta_1 \) is the vehicle position, \( \eta_2 \) is the vehicle orientation and \( q \) is the vector stacking the joint positions. Defining a task as a generic \( m \)-dimensional control objective as a function of the system state \( \sigma(s) \in \mathbb{R}^m \), the reference velocity that brings the task value \( \sigma \) to a desired
\( \sigma_d \) can be computed resorting to the Closed-Loop Inverse-Kinematics algorithm [9]:

\[
\zeta = J^\dagger (\sigma_d + K \sigma),
\]

where \( K \in \mathbb{R}^{m \times m} \) is a positive-definite matrix of gains, \( \dot{\sigma} = \sigma_d - \sigma \) is the task error and \( J^\dagger \) is the Moore-Penrose pseudoinverse of the Jacobian matrix \( J \), defined as:

\[
J^\dagger = J^T (J J^T)^{-1}
\]

If the system is redundant \((n > m)\) it is possible to perform multiple tasks simultaneously, setting a priority level among them and then projecting the velocity components coming from a lower priority task into the null space of the higher priority ones. In this way the fulfillment of the primary task is guaranteed. Thus for a hierarchy composed by \( h \) tasks, the reference system velocity can be computed as [10]:

\[
\zeta = \zeta_1 + N_1 \zeta_2 + \cdots + N_{1,h-1} \zeta_h,
\]

where \( \zeta_i \) is the reference velocity that fulfills the \( i \)-th task and \( N_{1,i} \) is the null space of the augmented Jacobian:

\[
J_{1,i} = \begin{bmatrix}
J_1 \\
J_2 \\
\vdots \\
J_i 
\end{bmatrix}
\]

This kind of control framework has been developed to handle equality-based tasks, in which the control objective is to bring the task value to a specific one. It needs to be extended in order to handle also set-based tasks, in which the task value has to be kept above a lower threshold and below an upper threshold [11].

The key idea is that a set-based task can be seen as an equality-based one which gets active or inactive depending on its current value. In particular, it is necessary to set different thresholds for each set-based task \( \sigma \): physical thresholds \( \sigma_M (\sigma_m) \), safety thresholds \( \sigma_{s,u} < \sigma_M (\sigma_{s,l} > \sigma_m) \), and activation thresholds \( \sigma_{a,u} = \sigma_{a,u} - \varepsilon (\sigma_{a,l} = \sigma_{s,l} + \varepsilon) \) (see Figure 4). When the task value reaches \( \sigma_{a,u} \) or \( \sigma_{a,l} \), it has to be added to the task hierarchy as a new equality-based task with desired value:

\[
\sigma_d = \begin{cases}
\sigma_{s,u} & \text{if } \sigma \geq \sigma_{a,u} \\
\sigma_{s,l} & \text{if } \sigma \leq \sigma_{a,l}
\end{cases}
\]

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. More in particular, this solution is not a priori known and therefore it results necessary to compute the solutions of all hierarchies obtained removing/inserting the set-based tasks (see Fig. 5). For further details about the activation/deactivation algorithm, see [12].

The UVMS is characterized by two twin 6-DoF arms. Table I and Table II describe their kinematics by the Denavit-Hartenberg convention together with the joint mechanical and velocity limits.

The simulations have been performed in Gazebo [13] and ROS (Robotic Operating System) [14], using the framework specifically for the DexROV project as described in [15]. Figure 6 shows the 3D model of the UVMS.

The proposed algorithm for the motion control of the system has been implemented in a ROS node that receives from topic the desired end-effector trajectory and publishes the computed system velocity \( \zeta \) on another topic. It is then read by two Gazebo plugins that implement the low-level controller for the vehicle and the arm joints. These controllers
TABLE II
ARM DATA: JOINT MECHANICAL AND VELOCITY LIMITS.

<table>
<thead>
<tr>
<th>joint</th>
<th>min angle (deg)</th>
<th>max angle (deg)</th>
<th>max velocity (deg/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-119</td>
<td>119</td>
<td>12.17</td>
</tr>
<tr>
<td>2</td>
<td>-110</td>
<td>110</td>
<td>13.02</td>
</tr>
<tr>
<td>3</td>
<td>-110</td>
<td>110</td>
<td>11.7</td>
</tr>
<tr>
<td>4</td>
<td>-170</td>
<td>170</td>
<td>12.3</td>
</tr>
<tr>
<td>5</td>
<td>-110</td>
<td>110</td>
<td>11.9</td>
</tr>
<tr>
<td>6</td>
<td>-170</td>
<td>170</td>
<td>17.8</td>
</tr>
</tbody>
</table>

Fig. 6. Screenshot of the DexROV system in the Gazebo simulator.

are purely kinematic, meaning that the computed velocity is instantaneously applied to the model in simulation without taking into account the dynamics of the system. The current vehicle position and orientation needed for computing the direct kinematics is taken directly from Gazebo, as well as the the pose of all the other objects in the scene.

V. SIMULATIONS RESULTS

Several simulations have been run with different task hierarchies, in order to give the system a behavior that allows to fulfill a trajectory tracking task in the most efficient way.

A. First case study

The first task hierarchy is chosen in order to constraint the motion of the manipulator in a virtual box surrounding the arm base frame. As long as the desired trajectory lies within the virtual box it would be desirable that the system moves both the manipulator and the vehicle to follow it. As soon as the desired trajectory would require the end-effector to leave the virtual box the arm needs to stop the movement, and the system has to rely only on the vehicle in order to make the end-effector follow the desired path. In this perspective, the implemented task hierarchy is:

- **Virtual box**: it is defined by 6 virtual walls expressed in the arm base frame. Their position has been chosen in order to keep the manipulability of the arm sufficiently high (avoiding configurations in which the arm is completely stretched or retracted), and to avoid collisions with the second arm and the vehicle. This box constrains the end-effector to move within these thresholds expressed in the arm base frame:

  \[0.7m < x < 1.1m\]
  \[0.1m < y < 0.5m\]
  \[0.1m < z < 0.5m\]

  The \(i\)-th task function is defined as:

  \[\sigma_i = \hat{n}_i^T (p_e - p_{1,i})\]

  where \(\hat{n}_i\) is the outer normal unit vector from the plane computed as:

  \[\hat{n}_i = \frac{(p_{2,i} - p_{1,i}) \times (p_{3,i} - p_{1,i})}{\| (p_{2,i} - p_{1,i}) \times (p_{3,i} - p_{1,i}) \|}\]

  where \(p_{1,i}\), \(p_{2,i}\) and \(p_{3,i}\) are three points belonging to the \(i\)-th plane.

- **End-effector position/orientation**: the end-effector has to follow the desired trajectory coming from the operator. The task function is:

  \[\sigma = [p_e \quad Q]^T\]

  that is the vector stacking the 3D position and the quaternion of the end-effector.

The desired trajectory is a cube-like shape larger than the chosen virtual box in which the end-effector has to remain. Figure 7 shows the desired end-effector trajectory (red), the initial vehicle position (blue circle) and the initial end-effector position (green circle), while Fig. 8 shows the end-effector trajectory (green) and vehicle trajectory (blue). It is clear that the the end-effector follows effectively the desired trajectory, moving both the vehicle and the arm joints.

Figure 9 shows the coordinates of the end-effector expressed in the arm base frame (blue), together with the limits imposed by the virtual box (red). It can be seen that the end-effector never violates the limits during the motion.
**B. Second case study**

The second case study takes into account a classical operation that the DexROV system is going to perform. In particular, the end-effector is asked to reach a target valve on a panel. The UVMS starts from a position far away from the panel and it has to reach the desired position while respecting all the constraints both in cartesian and joint space. The chosen task hierarchy is:

1) **Mechanical joint limits**: upper and lower limits have been set on the first, second, third and fifth joint of the manipulator in order to avoid self-collisions. The task function is the \( i \)-th joint value:

\[ \sigma = q_i . \]

2) **Arm manipulability**: a lower threshold of 0.029 on the arm manipulability has been set in order to avoid singular configurations of the arm that would lead to undesirable behavior of the system [16]. The task function is [17]:

\[ \sigma = \sqrt{\det(JJ^T)} . \]

3) **Virtual box**: same as the first case study, with different cartesian limits:

\[
\begin{align*}
0.8m &< x < 1.2m \\
0.05m &< y < -0.55m \\
-0.55m &< z < 0.05m .
\end{align*}
\]

4) **End-effector position and orientation**: the valve position and orientation is taken directly from Gazebo, and the desired configuration for the end-effector is computed accordingly.

Figure 10 shows the first, second, third and fiftth joint positions over time, together with the chosen upper and lower limits, while Fig. 11 shows the measure of manipulability. It is clear that both of the tasks are accomplished during the simulation, as they never violate the chosen thresholds.

Figure 12 shows the \( x, y, z \) coordinates of the end-effector expressed in the arm base frame, and they always stay within the virtual box.

Finally, in Fig. 13 it is possible to notice that the end effector reaches the target configuration while respecting all the higher priority tasks.
Fig. 12. $x$, $y$, $z$ coordinates of the end-effector expressed in arm base frame (blue) and limits imposed by the virtual box (red) over time. The end-effector never violates the limits during the motion.

Fig. 13. Position error (top) and orientation error (bottom). The end-effector reaches the target valve with the desired orientation. Figure 14 shows four screenshots of the simulation.

Fig. 14. Screenshots of the simulation. The end-effector reaches the target valve on the panel.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we presented the algorithm developed for the motion control of an UVMS within the DexROV project. The kinematic model of the system, describing the relation between the end-effector velocity and the system velocity composed by the vehicle and the joint velocities, has been computed. Then, the algorithm for handling the simultaneous resolution of task hierarchies composed by equality-based and set-based tasks has been explained. Finally we have shown the results of two simulations, proving the effectiveness of the proposed approach in the accomplishment of a classical use case scenario in which the DexROV system will be used.

Future efforts will concern the identification of further task hierarchies suitable for other kind of use case scenarios that the system will face, and the testing of the proposed algorithm on real hardware and environment.

REFERENCES