Copyright Information

A Task Priority Approach to Cooperative Mobile Manipulation: Theory and Experiments

E. Simetti[a,*], G. Casalino[a], F. Wanderlingh[a], M. Aicardi[a]

[a] *DIBRIS, University of Genova,*
Via Opera Pia 13, 16145 Genova, Italy

Abstract

Cooperative manipulation and transportation by means of multi-robot systems is a subject that has received an increased interest in the last few years. In this work, a task priority approach is first recalled from the authors previous works as framework for the control of a single mobile manipulator, to manage all its control objectives, including set membership ones and a proper coordination between the manipulator and its supporting vehicle. The approach is then extended, through a novel coordination policy, to execute a cooperative transportation of a common load by means of two (or more) mobile manipulators, via an explicit but limited information exchange, without modifying the individual controllers. Experimental results with two YouBot mobile manipulators are shown to demonstrate the effectiveness of the approach.

Keywords: cooperative mobile manipulation, task priority framework, mobile manipulators.

1. Introduction

In the last few years, cooperative manipulation and transportation has become one of the most important research topics on multi-robot systems [1]. Seminal work on cooperative mobile manipulation was presented in [2], using the experimental platform detailed in [3]. The mobile base was dynamically coordinated with the arm using a potential function aimed at maintaining the manipulator joint positions close to their midrange. In another seminal work on cooperative transportation [4], a team of mobile robots push an object cooperatively toward a desired position. Such an idea has inspired many of the following works. Indeed, [5] follows a similar approach and focuses on the problem of motion planning, dividing the planning problems in a global path planner and a local manipulation one. Manipulation is performed only by changing the height of the end-effector, i.e. only 1 degree of freedom (DOF). A similar approach is presented in [6]. The work [7] proposes a leader-follower strategy, in which the object is assumed as the virtual leader, while the robots carrying the object are considered followers. A leader-follower scheme is also exploited in [8], where the leader (either a robot or human) can guide the whole group towards the destination by applying a relatively small force, whose effect is amplified by the follower robots as they align their forces with the leader's one. The authors of [9] propose a feedback control law for a cooperative transportation system comprising two car-like vehicles, by assuming that the object is held in two manipulation points, which are revolute joints coupling the two vehicles to a carrier.

Except [2], in none of the previously mentioned works the transportation exploits manipulators, which are needed to pick-up and release objects that are outside the reach of the mobile bases. The presence of articulated arms, while significantly increasing the cooperative transportation potentialities, it also increases the number of DOF to be controlled, making the cooperative control problem much more complex. Indeed, the challenges lie not only in the object motion control, but also in the control of the interactions exchanged between the object and manipulators' end-effectors (i.e. the so-called motion and internal wrenches), and in the load balancing (static and dynamic) among the agents.

As it is deemed widely known, once the load bal-

*Corresponding author
Email address: enrico.simetti@unige.it (E. Simetti)

ancing problem is solved through a suitable distribution of the object dynamics among the agents, the object motion and its internal wrenches control problems can be solved separately, provided that at each time instant the following conditions are satisfied: the Cartesian reference velocities for the end-effectors belong to the *end-effectors constrained motion subspace* allowed by the grasping type and the kinematics of each agent, while the Cartesian internal wrenches references belong to the *end-effectors internal-wrenches subspace*, orthogonal to the constrained motion one. This result was established, for instance, in the early work [10] and more recently in [11, 12].

The two aforementioned orthogonal Cartesian spaces are the output spaces of two corresponding Jacobian transformations, constrained by the grasping condition, from the velocity space of the overall multi-agent system. Hence, the basic idea is to exploit these Jacobian transformations to embed the above mentioned conditions within the establishment of the reference system velocities. The overall system velocity vector will be therefore split in two parts: the first one complaint with the grasping constraint, accounting for the object motion and other system objectives (e.g. joint limits and obstacle avoidance, maintenance of adequate manipulability levels, etc.), and (if needed) a second part, acting against the grasping constraint, consequently accounting for the desired internal wrenches.

Following the above remarks, the overall control architecture naturally splits into two hierarchical levels, namely: a Kinematic Control Layer (KCL), in charge of generating the reference system velocities, taking into account all the control objectives of the system, including the desired motion of the object and internal wrenches; and a Dynamic Control Layer (DCL), in charge of tracking the reference system velocity vector, generating the actuation commands.

As a consequence of the hierarchical decomposition, if the load balancing and the reference system velocities are performed and generated as specified above, then the DCL can also work in a decentralized way. This means that it can be composed by separate dynamic controllers, one for each agent. They can be realized as exact or almost-exact computed torque controllers, as done in [10], as sliding mode controllers [13, 14], even if simpler, well tuned, proportional-integrative controllers can generally suffice [15]. Consequently, the KCL is left in charge of guaranteeing an adequate coordination of the cooperating agents, motivating why the present work will focus solely on the kinematic level of the hierarchy within a non-trivial cooperative manipulation/transportation context.

However, some additional remarks are deemed necessary. The first one is that, independently of the structure of the KCL to be designed, the "gains" of its control loops must be tuned on the performances achieved by each underlying DCL controller. The development of suitable tools for the automatic tuning of these kinematic loop gains is currently a research item, and, since it does not impact on the KCL algorithmic architecture, it will not be considered here.

The second remark is that the KCL should be designed not only for achieving the object manipulation/transportation (and for imposing its internal wrenches), but also for managing other control objectives, typically concerning the safety of the individual agents and their global ensemble. For example, the KCL should guarantee the joint limits and obstacle avoidance for each agent, and it should maintain adequate individual manipulability levels. This remark naturally suggests the adoption, at the KCL level of each agent, of the so-called task-priority based kinematic control techniques (see for example [16, 17, 18, 19]), which have now been widely adopted for the control of individually operating agents (see for example [20, 21, 22, 23, 24]).

Hence, it becomes interesting and important from a practical point of view to devise cooperative solutions that could fully exploit the task-priority based KCLs already embedded within each agent. The idea is to suitably coordinating them through a minimal data exchange, allowing an efficient execution of the cooperative manipulation/transportation task, while maintaining the safety needs of each individual agent and of the global constrained system at the highest priority. Such an idea, which will be developed within this paper, differentiates this work with respect to event-based approaches such as [25, 26]. In those works, the "motion reference" variable s is constrained on the assigned path and the object path tracking is permanently located at the highest priority. Hence, any deliberate deviation from staying along the object programmed path actually are never allowed, even if they might be needed, for instance, for improving the obstacle avoidance capability by part of the whole ensemble, or for allowing each agent to maintain its safety and/or operative

conditions.

Succeeding in achieving the aforementioned goals (without resorting to a centralized KCL) will certainly allow for high flexibility within environments where the manipulative agents typically have to be employed in different operational ways, i.e. individually and, when required, in a cooperative fashion. Scope of the present work will be developing such a cooperative decentralized KCL. To this aim, the main contributions of this work are the following ones:

C_1 The task-priority framework, which each agent employs at the kinematic level, is extended to handle cooperative manipulation between multiple mobile manipulators, allowing the agents to tackle several control objectives with their priorities, while satisfying the kinematic constraints imposed by the rigid grasp of the common transported object. In particular, each agent maintains its own KCL, on top of which the coordination policy is used to generate the object velocity to enforce. This solution is an advancement compared to other approaches [2, 27, 28], which focused on the transportation task but neglected other objectives such as maintaining a minimum manipulability measure to avoid kinematic singularities, avoiding joint limits or obstacles, and so on. Conversely to [25, 26], sensible deviations from the originally required object motion direction are deliberately allowed whenever needed for accomplishing agent-internal higher priority needs. This is evidenced by the experimental trials presented in this work, where in the presence of an obstacle, the entire ensemble sensibly deviates itself from the motion that had been otherwise obtained without the obstacle.

C_2 Conversely to many other approaches, the proposed strategy is not based on a rigid leader/follower scheme. Instead, during the cooperative manipulation the involved systems tune themselves toward helping those who have the most difficulty in tracking the desired object velocity. Therefore, during the cooperation, the role of the leader is automatically, for the time required, shifted toward the agent that has the highest tracking error.

While the above original contributions of this manuscript can be applied to single agents controlled through most of the aforementioned task-priority based kinematic control techniques, in this paper we will focus on the specific Task Priority Inverse Kinematic (TPIK) control framework [19, 15] developed in the past few years by the authors. Such a control framework is characterized by some important features such as the following ones:

F_1 It handles inequality control objectives by smoothly activating and deactivating their corresponding control tasks. This feature allows safety tasks to be at an higher priority with respect to other tasks such as reaching the desired position, without overconstraining the system.

F_2 Conversely to the pioneering works [29], where the base was controlled by the user and the arm was just treating its motion as a disturbance, or [2], where the base was just controlled through a potential function aimed at maintaining the manipulator joint positions close to their midrange (a similar approach later used in [28]), in the here proposed framework the vehicle supports the manipulator in executing all its tasks, and at the same time it still executes its own (e.g. avoiding an obstacle with the vehicle base).

The paper is structured as follows. For the reader's benefit, Section 2 recalls the basic elements of the proposed task priority framework for a single agent. Section 3 explains the proposed coordination policy for cooperative transportation. Section 4 presents the experimental results achieved with two YouBot mobile manipulators. Finally, conclusions are given.

2. Single Agent Control Architecture

Let us first briefly review the single agent control architecture depicted in Fig. 1, which is based on the hierarchical approach mentioned in the introduction. In particular, the architecture is constituted by three main blocks:

1. The Mission Manager is in charge of supervising the execution of the current *mission*, and generates the corresponding *actions* to be executed by the Kinematic Control Layer. As it will be explained later in section 2.4, an action is any prioritized list of control objectives to be concurrently achieved, and a mission is a sequence (or graph) of actions. As an example, a flexible architecture for planning control

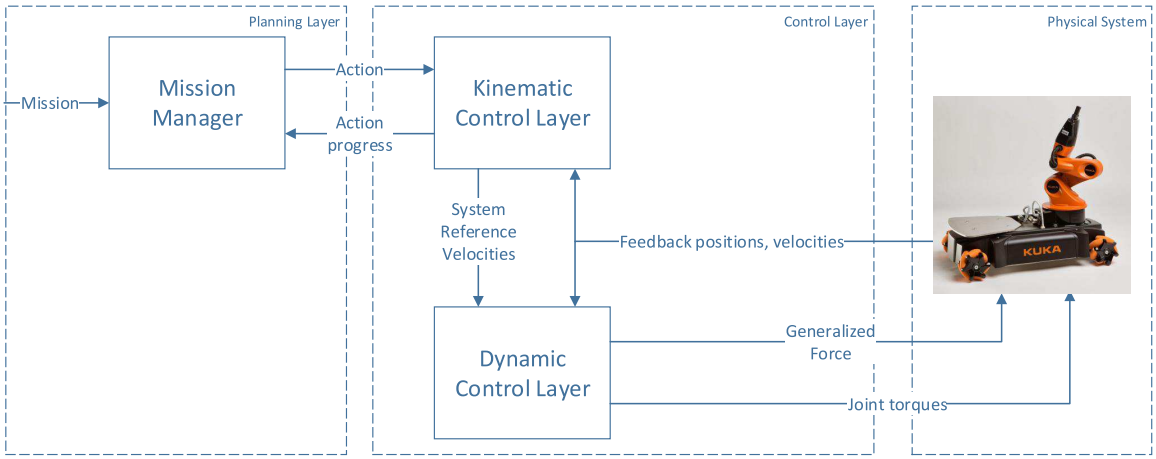

Figure 1: The overall architecture: the Kinematic Control Layer is the one implementing the proposed task priority approach, executing the current action scheduled by a Mission Manager; the output of the Kinematic Control Layer are the system velocities, to be tracked by the underlying Dynamic Control Layer.

actions is proposed in [30], but its discussion falls outside the scope of the present work.

2. The Kinematic Control Layer (KCL) implements the proposed task priority control framework, and is in charge of reactively accomplishing the *control objectives* that make up the current action to be executed, generating the desired system velocity vector.
3. The Dynamic Control Layer (DCL) tracks the desired system velocity vector by generating appropriate force/torques commands for the vehicle and the manipulator.

The paper focuses on the Kinematic Control Layer: the basic, single-agent task priority framework is reviewed in this section, while its extension to the cooperative manipulation case is tackled in the successive section 3. Finally, note that the YouBot mobile manipulator platform, used in the experiments, comes with predefined DCLs both for the vehicle and for the manipulators.

2.1. Basic Definitions

In the following paragraphs, the robot configuration vector is referred to as $\boldsymbol{c} \triangleq \begin{bmatrix}\boldsymbol{q}^\top & \boldsymbol{\eta}^\top\end{bmatrix}^\top \in \mathbb{R}^n$ and contains the n robot DOF, i.e., the l arm joint positions $\boldsymbol{q} \in \mathbb{R}^l$ and the vehicle position and orientation vector $\boldsymbol{\eta} = \begin{bmatrix}x & y & z & \phi & \theta & \psi\end{bmatrix}^\top \in \mathbb{R}^6$, where the latter is expressed in terms of the roll, pitch and yaw angles representation. The robot velocity vector is named $\dot{\boldsymbol{y}} \triangleq \begin{bmatrix}\dot{\boldsymbol{q}}^\top & \boldsymbol{v}^\top\end{bmatrix}^\top \in \mathbb{R}^n$, and represents the controls to actuate the robot, i.e., the joint velocities $\dot{\boldsymbol{q}} \in \mathbb{R}^l$ and the vehicle linear and angular velocities $\boldsymbol{v} \in \mathbb{R}^6$. If the base is constrained, e.g. a ground vehicle moving on the horizontal plane, then the corresponding definitions of $\boldsymbol{\eta}$ and $\boldsymbol{v}$ can be reduced accordingly removing the constrained DOF.

2.2. Control Objectives

Consider a scalar variable $x_o(\boldsymbol{c})$ related to a control objective o, then:

- the requirement, for $t \to \infty$, that $x_o(\boldsymbol{c}) = x_{o,0}$ is called a *scalar equality control objective*,
- the requirement, for $t \to \infty$, that $x_o(\boldsymbol{c}) < x_{o,max}$ or $x_o(\boldsymbol{c}) > x_{o,min}$ is called a *scalar inequality control objective*,

where $x_{o,0}$ is a given reference value, whereas $x_{o,min}$ and $x_{o,max}$ serve as lower and upper thresholds for the values that the scalar variables should assume. Note that if m different variables $x_i(\boldsymbol{c})$ are considered, each of them corresponding to the $i-$th component of a vector $\boldsymbol{p} \in \mathbb{R}^m$, then it is possible to control the vector to any desired value. Thus limiting the discussion to scalar objectives does not influence the generality of the approach. Furthermore, if $x(\boldsymbol{c})$ is the modulus of a certain vector $\boldsymbol{p}$, then it can be used to require a particular value for the norm of $\boldsymbol{p}$ (e.g. to nullify it), or to be below or above a given threshold.

The above requirements establish the objectives that the system should eventually reach, e.g. avoiding obstacles, respecting joint limits, reaching a desired position with the end-effector. In the following, the dependency of x on $\boldsymbol{c}$ will be dropped to ease the notation.

2.3. Control Tasks

The time behaviour of each scalar variable x_o is related to the system velocity vector through a Jacobian relationship $\dot{x}_o = \boldsymbol{J}_o(\boldsymbol{c})\dot{\boldsymbol{y}}$, where $\boldsymbol{J}_o(\boldsymbol{c}) \in \mathbb{R}^{1\times n}$ is the Jacobian matrix of the task. Given this link, a *feedback reference rate* $\dot{\bar{x}}_o$ is defined as

$$\dot{\bar{x}}_o(x_o) \triangleq \gamma(x_o^* - x_o),\ \gamma > 0, \tag{1}$$

where γ is a positive gain proportional to the desired convergence rate for the considered variable, and x_o^* is a point inside the state region where o is satisfied. The *reactive control task* τ_o associated with the objective o is defined as the need of minimizing the difference between the actual task velocity $\dot{x}_o$ and the feedback reference rate $\dot{\bar{x}}_o$.

There are cases where a control task is defined directly in a certain task velocity space, without having an associated control objective. As an example, consider the case where a human operator wants to control the end-effector by generating velocity references through a joystick. In such a case, there is no explicit control objective specified to the system. Therefore, the system sees such a reference rate as a *non-reactive control task* (i.e. not associated to any internal feedback loop).

2.4. Control Actions

From the robot control standpoint, an action $\mathcal{A}$ can be defined as a prioritized list of m control objectives $o_1, \ldots, o_m$ and the associated control tasks $\tau_1, \ldots, \tau_m$, to be managed *concurrently*. As an example, a manipulation action $\mathcal{A}_m$ for a single mobile manipulator is described in terms of its list of control objectives, in order of priority:

o_1) arm joint limits,

o_2) obstacle avoidance,

o_3) arm manipulability,

o_4) end-effector position control,

o_5) end-effector attitude control,

o_6) arm preferred pose.

In such a hierarchy, it is natural to see *safety* objectives such as *arm joint limits* and *obstacle avoidance* at the highest priority followed by the end-effector position and attitude control, finally followed by an optimization objective such as trying to maintain a preferred pose. Maintaining a minimum manipulability in this example is put above the control of the end-effector, although it could be also considered at a lower priority. Indeed, how to devise such hierarchies depends on the specific behaviour that we want to assign to the robot. For the time being, let us assume to have a given list of prioritized control tasks.

To sum up the concepts presented so far: control objectives specify desired target values (or regions) for variables of interest; control tasks are the means to ensure a closed-loop convergence of such variables to the desired values, as they specify a desired time behaviour for the variable's time derivative, linking it to the system velocity vector through a Jacobian relationship; finally, control actions are hierarchies of such control objectives, implemented through their associated control tasks. Different control actions give rise to different complex behaviours of the system, and can be seen as building blocks for higher level mission planning and supervision tools.

2.5. Activation Functions

Control objectives, and their associated reactive control tasks, are relevant depending on the current value of the system configuration vector $\boldsymbol{c}$. For example, the avoidance of a joint limit is relevant only in the vicinities of the joint limit. Whenever the joint is sufficiently far away, the control task should not overconstraint the system. This allows, for example, to place the joint limits avoidance objective at the top of the aforementioned action $\mathcal{A}_m$, knowing that the task will constrain the system only when necessary. To this end, let us define an *activation function*

$$a_o^i(x_o) \in [0, 1] \tag{2}$$

as a continuous sigmoid function of a scalar objective variable x_o, whose value is zero within the validity region of the associated control objective o.

For non-reactive control tasks, the variable $x(\boldsymbol{c})$ is not defined and the activation function (2) reduces to be $a_o^i(x_o) \equiv 1$. From now on, the distinction between reactive and non-reactive control tasks will be dropped, and the generic term control task will be used, unless otherwise specified.

2.6. Task Priority Inverse Kinematics

Given an action $\mathcal{A}$, for each priority level k, the following matrices and vectors are consequently defined:

- $\dot{\tilde{\boldsymbol{x}}}_k \triangleq \left[\dot{\tilde{x}}_{1,k}, \ldots, \dot{\tilde{x}}_{m,k}\right]^\top$ is the stacked vector of all the reference rates, where the first index indicates control task $\tau_1, \ldots, \tau_m$ placed at the priority level k.
- $\boldsymbol{J}_k$ is the Jacobian relationship expressing the current rate of change of the k-th task vector $\left[\dot{x}_{1,k}, \ldots, \dot{x}_{m,k}\right]^\top$ with respect to the system velocity vector $\dot{\boldsymbol{y}}$.
- $\boldsymbol{A}_k \triangleq \mathrm{diag}(a_{1,k}, \ldots, a_{m,k})$ is the diagonal matrix of all the activation functions in the form of (2).

At this point, the control problem is to find the system velocity reference vector $\dot{\tilde{\boldsymbol{y}}}$ that satisfies at best the aforementioned requirements. The Task Priority Inverse Kinematics (TPIK) procedure proposed in [19] is expressed by the following sequence of minimization problems:

$$S_k \triangleq \left\{ \arg\text{R-}\min_{\dot{\tilde{\boldsymbol{y}}} \in S_{k-1}} \left\| \boldsymbol{A}_k (\dot{\tilde{\boldsymbol{x}}}_k - \boldsymbol{J}_k \dot{\tilde{\boldsymbol{y}}}) \right\|^2 \right\}, \quad (3)$$

where S_{k-1} is the manifold of solutions of all the previous tasks in the hierarchy, initialized with $S_0 \triangleq \mathbb{R}^n$. The notation R-min highlights that the minimization process is performed in a special regularized way, as explained formally in [19]. Details of the pseudo-inversion are omitted for sake of brevity.

Note that, so far, this framework controls both the manipulator(s) and the vehicle in the same manner.

2.7. Transition between Actions

Let us consider the following two actions $\mathcal{A}_1 = A \prec B, C, D$ and $\mathcal{A}_2 = A \prec D \prec C, E$, composed by objectives abstractly labelled with alphabetic letters, where $A \prec B$ denotes that A has higher priority than B, and where the comma separates objectives with the same priority. Now consider the following unified list $\mathcal{U} : A \prec D_1 \prec B, C, D_2, E$ where D_1 and D_2 represent the same objective D, but with a different priority. It is clear that, through insertion/deletion of some of the entries, the two original lists can be reconstructed.

Toward the aforementioned aim, the activation function (2) is modified to become

$$a_o(x_o, \boldsymbol{p}) = a_o^i(x) a_o^p(\boldsymbol{p}), \quad (4)$$

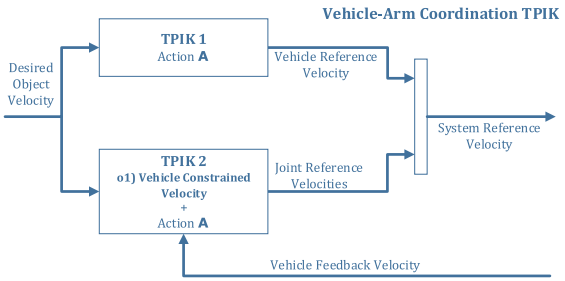

Figure 2: A detailed view of the Vehicle-Arm Coordination scheme, with the example of action $\mathcal{A}$ task hierarchy.

where $a_o^p(\boldsymbol{p}) \in [0, 1]$ is an additional continuous sigmoidal function of a vector of parameters $\boldsymbol{p}$ external to the control task itself, which can be can be conveniently parametrized to obtain the desired activation/deactivation smooth transition.

2.8. Vehicle - Arm Coordination

Until now, whole body Jacobians are employed to control the end-effectors. As a matter of fact, unavoidable minor accuracies and disturbances affecting the mobile platform, such as delayed or imprecise wheel actuation, would propagate through the coupled kinematics immediately to the end effectors of both manipulators.

To cope with this problem, the idea (F_2) is to have two TPIK procedures running in parallel as depicted in Fig. 2 and outlined in Algorithm 1:

1. The first optimization (TPIK 1) considers the vehicle together with the manipulator. Of the whole result $\dot{\tilde{\boldsymbol{y}}}$, only the vehicle reference velocity is used (line 1 in Alg. 1), while the manipulator part is discarded;
2. The second optimization (TPIK 2) considers the vehicle as an *instantaneously non-controllable* entity. A non-reactive control task (line 2) is added as the *highest* priority task in the hierarchy (line 3), whose role is to constrain the desired vehicle velocities equal to the *current measured* ones $\boldsymbol{v}$. More in detail, the vehicle constrained velocity task consists in the following set of quantities (for an unconstrained 6 DOF vehicle):

$$\dot{\tilde{\boldsymbol{x}}}_{vcv} = \boldsymbol{v}, \quad (5)$$

$$\boldsymbol{J}_{vcv} = \begin{bmatrix} \boldsymbol{0}_{6\times l} & \boldsymbol{I}_{6\times 6} \end{bmatrix}, \quad (6)$$

which, once placed at the highest priority level, clearly constraints the successive tasks to work

Algorithm 1 CoordinatedTPIK($\mathcal{A}$, $\dot{\tilde{\boldsymbol{x}}}$, $\boldsymbol{v}$)

Require: An action $\mathcal{A}$, a desired velocity for the end-effector $\dot{\tilde{\boldsymbol{x}}}$, the vehicle actual velocity $\boldsymbol{v}$

Ensure: The reference system velocity vector $\dot{\boldsymbol{y}}$ through arm-vehicle coordination

1: $\dot{\boldsymbol{y}}_1 \triangleq \begin{bmatrix}\dot{\boldsymbol{q}}_1^\top & \bar{\boldsymbol{v}}_1^\top\end{bmatrix}^\top \leftarrow \text{TPIK}(\mathcal{A}, \dot{\tilde{\boldsymbol{x}}})$
2: $\tau_{vcv} \leftarrow \text{VehicleConstrainedVelocityTask}(\boldsymbol{v})$
3: $\mathcal{A}_{vcv} \leftarrow \text{Stack}(\tau_{vcv}, \mathcal{A})$
4: $\dot{\boldsymbol{y}}_2 \triangleq \begin{bmatrix}\dot{\boldsymbol{q}}_2^\top & \bar{\boldsymbol{v}}_2^\top\end{bmatrix}^\top \leftarrow \text{TPIK}(\mathcal{A}_{vcv}, \dot{\tilde{\boldsymbol{x}}})$
5: $\dot{\boldsymbol{y}} \leftarrow \begin{bmatrix}\dot{\boldsymbol{q}}_2^\top & \bar{\boldsymbol{v}}_1^\top\end{bmatrix}^\top$
6: **return** $\dot{\boldsymbol{y}}$

with the vehicle velocities fixed to the feedback value $\boldsymbol{v}$. In such a task hierarchy optimization, de-facto only the manipulator variables are subject to be optimized. In fact, the outputs of this procedure are the *optimal* joint velocities in correspondence of the *measured* vehicle velocity (line 4).

The outputs of the two TPIK procedures are then forwarded to the underlying DCL for tracking (lines 5 and 6). Therefore, the vehicle will do its best to track the output of the centralized arm-vehicle optimization (TPIK 1), properly supporting the arm in all the tasks. However, thanks to the TPIK 2 optimization, the arm reference joint velocities are always the optimal ones based on the current vehicle velocities, independently of any vehicle inaccuracy in tracking the desired ones generated by TPIK 1. A detailed experimental campaign, showing the advantages of the proposed coordination strategy for a single underwater vehicle manipulator agent, can be found in [22] and is outside the scope of the present paper.

Finally, note how this parallel technique can easily adapted to a multi-rate control scenario. Indeed, usually manipulators can be controlled at a frequency higher than those of the vehicles. Therefore, the algorithm can be easily modified to execute TPIK 2 at the arm control frequency, while executing TPIK 1 only at the lower vehicle control frequency.

3. Task Priority Based Cooperative Mobile Manipulation

After the single agent control architecture has been recalled in the previous section, let us now focus on its extension to the case of multiple cooperative mobile manipulators. For simplicity of discussion, the section will focus on two robots, as done in the experiments, but the approach can be scaled to multiple agents.

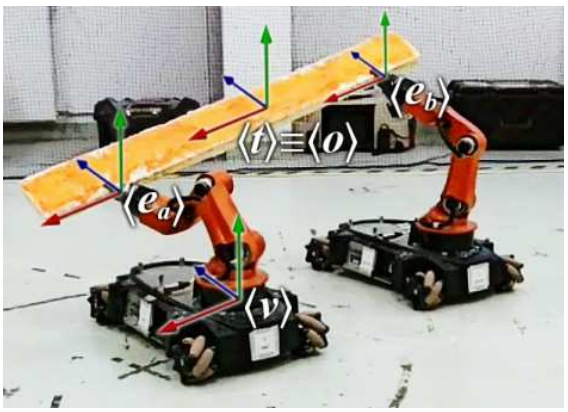

Figure 3: Frames involved in the cooperative mobile manipulation: the vehicle frame $\langle v \rangle$, the end-effector frames $\langle e \rangle$ and the tool control point $\langle t \rangle$ which coincides with object frame $\langle o \rangle$ for both agents.

3.1. Introduction

Assuming a firm object grasping by part of two agents a, b, the tool control points $\langle t_a \rangle$, $\langle t_b \rangle$ are assigned by the agents to coincide with the shared object fixed frame $\langle o \rangle$, that is $\langle t_a \rangle \equiv \langle t_b \rangle \equiv \langle o \rangle \triangleq \langle t \rangle$, as exemplified in Fig. 3. In the following we will only consider the tool-frame velocities, and whenever the term end-effector is used is to be intended prolonged at the tool control point $\langle t \rangle$.

In these conditions, the following differential constraints are imposed as consequence of the geometric ones

$$\dot{\boldsymbol{x}}_t = \boldsymbol{J}_{t,a}\dot{\boldsymbol{y}}_a = \boldsymbol{J}_{t,b}\dot{\boldsymbol{y}}_b, \tag{7}$$

with $\dot{\boldsymbol{x}}_t$ the object velocity with components on $\langle t \rangle$ and $\boldsymbol{J}_{t,a}$, $\boldsymbol{J}_{t,b}$ the system absolute Jacobians (i.e. w.r.t. an inertial observer) of frame $\langle t \rangle$, with output components on $\langle t \rangle$ itself, which is the sole frame common to both agents. Let us rewrite the second equation in (7) as

$$\begin{bmatrix}\boldsymbol{J}_{t,a} & -\boldsymbol{J}_{t,b}\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{y}}_a \\ \dot{\boldsymbol{y}}_b\end{bmatrix} \triangleq \boldsymbol{G}\dot{\boldsymbol{y}}_{ab} = \boldsymbol{0}, \iff \dot{\boldsymbol{y}}_{ab} \in \ker(\boldsymbol{G}), \tag{8}$$

which represents the subspace where $\dot{\boldsymbol{y}}_{ab}$ is constrained to lay as a consequence of the firm grasp assumption. The definition of the constrained space $\boldsymbol{G}$ of the overall system velocities of both mobile manipulators clearly couples the separate system

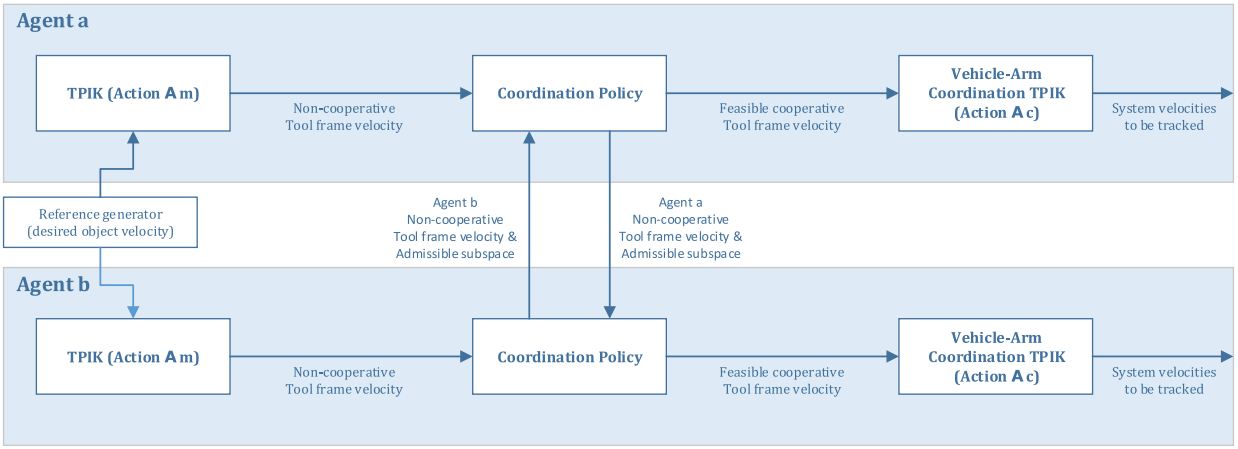

Figure 4: Block diagram of the cooperation. 1) A TPIK (action $\mathcal{A}_m$) is first solved, acting as if the agent was alone, using the desired object velocity $\dot{\boldsymbol{x}}_t$ generated by the reference generator. 2) The system velocity vector output of the TPIK is used to compute the *non-cooperative* tool-frame velocity. 3) The non-cooperative tool frame velocity, together with the admissible Cartesian subspace is exchanged with the other agent. Then, the *feasible cooperative* velocity is computed through (17) and (18). 4) Each agent now solves a coordinated TPIK (action $\mathcal{A}_c$) constrained to the feasible cooperative tool-frame velocity

velocities of the two agents, and it could be exploited to find a *centralized solution* satisfying the constraints at the kinematic level. However, such a possibility is discarded due to its required high rate of information exchange between the systems and the required context switch between separate controllers and a centralized one. Instead, the coordination policy proposed here is based on a reduced set of information that qualifies the evolution of the cooperation objectives, i.e. to comply with the grasp condition and to transfer the object to the desired location, and does not require changing the individual controllers.

Before proceeding with the coordination policy, let us compute the subspace of the combined end-effector velocities that are compatible with the grasp constraints. Let us first consider the motion space of a single agent, i.e.

$$\boldsymbol{H}_i \triangleq \boldsymbol{J}_{t,i}\boldsymbol{J}_{t,i}^{\#} \tag{9}$$

where the notation $\boldsymbol{J}_{t,i}^{\#}$ indicates the Moore-Penrose pseudo inverse of $\boldsymbol{J}_{t,i}$, computed using the singular value decomposition (see [31] for a review on pseudo-inverses and [19] for details on the regularized inversion algorithm). Then, since $\dot{\boldsymbol{x}}_t$ belongs to the motion spaces of both agents, (7) can be written as follows

$$\dot{\boldsymbol{x}}_t = \boldsymbol{H}_a\dot{\boldsymbol{x}}_t = \boldsymbol{H}_b\dot{\boldsymbol{x}}_t, \tag{10}$$

which in matrix form becomes

$$\begin{bmatrix}\boldsymbol{H}_a & -\boldsymbol{H}_b\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{x}}_t \\ \dot{\boldsymbol{x}}_t\end{bmatrix} \triangleq \boldsymbol{C}\dot{\boldsymbol{x}}_{t,ab} = \boldsymbol{0}, \tag{11}$$

which is satisfied iff $\dot{\boldsymbol{x}}_{t,ab} \in \ker(\boldsymbol{C}) = \mathrm{Span}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})$. During the cooperation, the actual end-effector velocities $\dot{\boldsymbol{x}}_{t,ab}$ necessarily lie in such a subspace, as result of the constraint. From a control point of view, if one is interested in avoiding interaction between the mobile manipulators and the object, the *desired* end-effector velocities must be forced to lie within this subspace. Hence, projecting the ideally desired, even possibly different, tool-frame velocities $\dot{\tilde{\boldsymbol{x}}}_{t,a}$,$\dot{\tilde{\boldsymbol{x}}}_{t,b}$ as follows

$$\dot{\tilde{\boldsymbol{x}}}_{t,ab} = (\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})\begin{bmatrix}\dot{\tilde{\boldsymbol{x}}}_{t,a} \\ \dot{\tilde{\boldsymbol{x}}}_{t,b}\end{bmatrix} \tag{12}$$

forces $\dot{\tilde{\boldsymbol{x}}}_{t,ab} \in \ker(\boldsymbol{C})$, and hence makes it complaint with the constraints. Note that this does not imply that the first six components of the projected reference velocities $\dot{\tilde{\boldsymbol{x}}}_{t,ab}$ are equal to the second six ones. Indeed, there might be components outside of the motion spaces of each agent (i.e. each one respectively belonging to the separate infeasible output spaces $\ker(\boldsymbol{J}_{a,t}^{\top})$ and $\ker(\boldsymbol{J}_{b,t}^{\top})$ of each agent) that are naturally complaint with the constraints, since they do not correspond to any tool frame $\langle t \rangle$ motion.

However, despite the possible differences, if these projected references velocities $\dot{\tilde{\boldsymbol{x}}}_{t,ab}$ are respectively

given to the corresponding agent and provided each one has the non-reactive tool-frame velocity tracking task located at the *highest priority*, then the actual tool-frame velocities $\dot{\boldsymbol{x}}_{t,ab} \in \mathbb{R}^{12}$ result to be

$$\dot{\boldsymbol{x}}_{t,ab} = \begin{bmatrix} \dot{\boldsymbol{x}}_t \\ \dot{\boldsymbol{x}}_t \end{bmatrix} = \begin{bmatrix} \boldsymbol{H}_a & \boldsymbol{0}_{6\times6} \\ \boldsymbol{0}_{6\times6} & \boldsymbol{H}_b \end{bmatrix} (\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}) \begin{bmatrix} \dot{\tilde{\boldsymbol{x}}}_{t,a} \\ \dot{\tilde{\boldsymbol{x}}}_{t,b} \end{bmatrix} \triangleq \boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}) \begin{bmatrix} \dot{\tilde{\boldsymbol{x}}}_{t,a} \\ \dot{\tilde{\boldsymbol{x}}}_{t,b} \end{bmatrix} \tag{13}$$

which are consequently composed by identical six-dimensional partitions complaint with the grasp constraints.

From the above equation we can see that $\boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})$ maps from the full $\mathbb{R}^{12}$ space of tool-frame velocities to the subspace of tool-frame velocities feasible by *both* mobile manipulators *and* complaint with the object kinematic constraint. To better understand the role of the subspace $\mathrm{Span}(\boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}))$, let us remark that the whole $\mathbb{R}^{12}$ space can be divided in two orthogonal subspaces:

- $\mathrm{Span}(\boldsymbol{H}_{ab}) \subseteq \mathbb{R}^{12}$ represents the *end-effectors unconstrained motion subspace* and is the direct sum of the unconstrained motion spaces of each single agent when the object is not yet grasped.
- $\ker(\boldsymbol{H}_{ab}) = \mathrm{Span}(\boldsymbol{I} - \boldsymbol{H}_{ab}^{\#}\boldsymbol{H}_{ab}) \subseteq \mathbb{R}^{12}$ represents the *end-effectors infeasible motion subspace*, i.e. all the velocities that, even if separately commanded to both agents when the object is not yet grasped, are infeasible for both agents.

Clearly, to simultaneously control the object motion and its internal wrenches, it is useful to consider reference velocities belonging only to $\mathrm{Span}(\boldsymbol{H}_{ab})$, as those lying on $\ker(\boldsymbol{H}_{ab})$ do not generate neither motion nor interaction forces and can consequently be neglected. Once an object has been grasped by both agents, the original unconstrained subspace $\mathrm{Span}(\boldsymbol{H}_{ab})$ can be further subdivided in two orthogonal subspaces, namely:

- $\mathrm{Span}(\boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}))$ represents the subspace of tool-frame velocities which are complaint with the object kinematic constraints, and are feasible by both agents. Hence, it is termed the *end-effectors constrained motion subspace*. Since the tool frames are prolonged to the same point $\langle t \rangle$, this subspace is only composed by vectors $\dot{\boldsymbol{x}}_{t,ab} \in \mathbb{R}^{12}$ whose first six and second six components are equal and feasible by both agents.
- $\mathrm{Span}(\boldsymbol{H}_{ab}\boldsymbol{C}^{\#}\boldsymbol{C})$ instead represents the subspace of tool-frame velocities which violate the object kinematic constraints, and therefore would generate internal wrenches. Hence it is termed the *end-effectors internal-wrenches subspace*.

Therefore, to ensure a pure motion of the object (i.e., with no internal wrenches), $(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})$ must be formerly used to project any non-complaint set of desired tool-frame velocities into complaint ones:

$$\begin{bmatrix} \dot{\tilde{\boldsymbol{x}}}_{t,a} \\ \dot{\tilde{\boldsymbol{x}}}_{t,b} \end{bmatrix} = (\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}) \begin{bmatrix} \dot{\tilde{\boldsymbol{x}}}_{t,a} \\ \dot{\tilde{\boldsymbol{x}}}_{t,b} \end{bmatrix}. \tag{14}$$

Of course, one will start with desired velocities which are equal, i.e. $\dot{\tilde{\boldsymbol{x}}}_t = \dot{\tilde{\boldsymbol{x}}}_{t,a} = \dot{\tilde{\boldsymbol{x}}}_{t,b}$, however the above projection process is anyway necessary to cope with possible defectiveness of each agent, while still satisfying the grasp constraints.

While the use of the sole projection matrix $(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})$ is sufficient to ensure that the resulting desired velocities $\dot{\tilde{\boldsymbol{x}}}_{t,a}, \dot{\tilde{\boldsymbol{x}}}_{t,b}$ are complaint with the grasp constraints, they might still contain components belonging to the separate infeasible output spaces $\ker(\boldsymbol{J}_{a,t}^{\top})$ and $\ker(\boldsymbol{J}_{b,t}^{\top})$ of each agent. For clarity purposes, the successive projection in the feasible subspace is also performed, as follows:

$$\begin{bmatrix} \dot{\tilde{\boldsymbol{x}}}_t \\ \dot{\tilde{\boldsymbol{x}}}_t \end{bmatrix} = \boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}) \begin{bmatrix} \dot{\tilde{\boldsymbol{x}}}_{t,a} \\ \dot{\tilde{\boldsymbol{x}}}_{t,b} \end{bmatrix}. \tag{15}$$

In this way, the projected velocities are also the same ones, since they are now constituted only by feasible components.

Therefore, if the feasible tool-frame velocity $\dot{\tilde{\boldsymbol{x}}}_t$, lying on $\mathrm{Span}(\boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}))$, is separately commanded to both agents *and* the corresponding non-reactive tool-frame velocity tracking task is located at the *highest priority* in both agents task priority lists, then the object kinematic constraint is satisfied at the kinematic level.

At this point, accordingly with the above conclusion and the expressed general requirements, the following coordination policy development can be therefore proposed.

3.2. Coordination Policy

For the development of the coordination policy the following assumptions are made:

- A *reference generator* software computes the desired object velocity and broadcasts it to all the involved agents. This desired velocity could be generated to asymptotically transfer the object to a given goal location (e.g. the velocity is the output of a position loop), or be directly generated by a user in case of teleoperation. Note that this reference generator could be hosted by one of the involved agents, if required.

- The coordination policy is *distributed* between all the agents. In case of a perfect wireless network (no packet loss, infinite bandwidth, no delay), this corresponds to having a centralized coordinator software doing the computation and broadcasting the result. However, the first solution is clearly preferable from a practical implementation point of view.

Then, during each sampling interval, the following sequential steps are executed, as sketched in Fig. 4 and reported in Algorithm 2 (as seen from a generic agent i):

1. The latest velocity $\dot{\tilde{\boldsymbol{x}}}_t$ generated by the reference generator is acquired (line 1).
2. Each agent runs the TPIK procedure detailed in section 2 as if it were the sole one acting on the object, with its original task hierarchy $\mathcal{A}_m$ having safety and prerequisite tasks located at higher priority than the tool control point motion. The two TPIK procedures separately provide the vectors $\dot{\boldsymbol{y}}_a, \dot{\boldsymbol{y}}_b$ (line 2).
3. Each agent evaluates the Cartesian velocity that it would impose to the object if it were alone and it were applying $\dot{\boldsymbol{y}}_a$ or $\dot{\boldsymbol{y}}_b$ respectively, i.e. the following *non-cooperative* tool-frame velocities (line 3)

$$\dot{\boldsymbol{x}}_{t,i} = \boldsymbol{J}_{t,i}\dot{\boldsymbol{y}}_i, \quad i = a, b. \tag{16}$$

   Note that the separate Cartesian velocities in (16) might not satisfy the kinematic constraint (7), i.e. $\dot{\boldsymbol{x}}_{t,a} \neq \dot{\boldsymbol{x}}_{t,b}$, due to higher priority tasks within each agent task hierarchy. Furthermore, each agent evaluates its corresponding matrix $\boldsymbol{H}_i$ (line 4) as defined in (9), which represents the admissible tool-frame velocity space of each agent whenever standalone acting.
4. Both agents transfer their computed quantities $\boldsymbol{H}_i$ and $\dot{\boldsymbol{x}}_{t,i}$ to the other involved agents (lines 5-6).
5. Each agent, in a *distributed way*, evaluates the *cooperative* tool-frame velocity vector (lines 7-9)

$$\dot{\hat{\boldsymbol{x}}}_t = \frac{1}{\mu_a + \mu_b}\left(\mu_a \dot{\boldsymbol{x}}_{t,a} + \mu_b \dot{\boldsymbol{x}}_{t,b}\right), \quad \mu_a, \mu_b > 0, \tag{17}$$

   which corresponds to a weighted compromise between the two output velocities $\dot{\boldsymbol{x}}_{t,a}, \dot{\boldsymbol{x}}_{t,b}$. The rationale and the details underlying such a weighting policy will be given at the end of the description of this procedure.
   If both agents have their safety objectives satisfied, and their current tool-frame Jacobians $\boldsymbol{J}_{t,i}$ are non-defective, i.e. $\boldsymbol{H}_i = \boldsymbol{I}$ for both $i = a, b$, then $\dot{\hat{\boldsymbol{x}}}_t = \dot{\boldsymbol{x}}_{t,a} = \dot{\boldsymbol{x}}_{t,b}$ independently from the weights, and the kinematic constraints are clearly satisfied.
   Instead, in the general case, when $\dot{\boldsymbol{x}}_{t,a} \neq \dot{\boldsymbol{x}}_{t,b}$ their weighted sum (17) might not lay in the end-effectors constrained motion subspace, therefore it must be projected on such a subspace, as performed in the next two steps.
6. Each agent evaluates the Cartesian constraint matrix as defined in (11) (line 10).
7. Each agent considers the vector $\dot{\hat{\boldsymbol{x}}}_t$ as its reference velocity, i.e. $\dot{\hat{\boldsymbol{x}}}_{t,i} = \dot{\hat{\boldsymbol{x}}}_t$ and projects it on the end-effectors constrained motion subspace $\mathrm{Span}(\boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}))$, obtaining the so-called *feasible cooperative* velocity vector (line 11)

$$\begin{bmatrix} \dot{\tilde{\boldsymbol{x}}}_t \\ \dot{\hat{\boldsymbol{x}}}_t \end{bmatrix} \triangleq \boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}) \begin{bmatrix} \dot{\tilde{\boldsymbol{x}}}_t \\ \dot{\hat{\boldsymbol{x}}}_t \end{bmatrix}. \tag{18}$$

8. Both agents separately run a new TPIK procedure, implementing the vehicle arm coordination of section 2.8, with the original task-priority hierarchy now modified into the one having the non-reactive tool-frame velocity tracking of $\dot{\tilde{\boldsymbol{x}}}_t$ at the *highest priority* (line 12):

   o_1) constrained tool velocity tracking;

   o_2) arm joint limits;

   o_3) obstacle avoidance;

   o_4) arm manipulability;

   o_5) arm preferred pose.

   In this way, both systems independently optimize their safety tasks under the imposed constraint of the common tool frame velocity. Such a new hierarchy is termed action $\mathcal{A}_c$ in Fig. 4 and Algorithm 2.

Algorithm 2 CooperativeTPIK($\mathcal{A}_m$)

Require: The manipulation action $\mathcal{A}_m$, the constrained manipulation action $\mathcal{A}_c$

Ensure: The reference system velocity vector $\dot{\tilde{\boldsymbol{y}}}_i$

1: $\dot{\tilde{\boldsymbol{x}}}_t \leftarrow$ GetDesiredObjectVelocity()
2: $\dot{\boldsymbol{y}}_i \leftarrow$ TPIK($\mathcal{A}_m$, $\dot{\tilde{\boldsymbol{x}}}_t$)
3: $\dot{\boldsymbol{x}}_{t,i} \leftarrow \boldsymbol{J}_{t,i}\dot{\boldsymbol{y}}_i$
4: $\boldsymbol{H}_i \leftarrow \boldsymbol{J}_{t,i}\boldsymbol{J}_{t,i}^{\#}$
5: SendDataToAgent(j, $\dot{\boldsymbol{x}}_{t,i}$, $\boldsymbol{H}_i$)
6: $\dot{\boldsymbol{x}}_{t,j}, \boldsymbol{H}_j \leftarrow$ GetLatestDataFromAgent(j)
7: $\mu_i \leftarrow \mu_0 + \|\dot{\tilde{\boldsymbol{x}}}_t - \dot{\boldsymbol{x}}_{t,i}\|$
8: $\mu_j \leftarrow \mu_0 + \|\dot{\tilde{\boldsymbol{x}}}_t - \dot{\boldsymbol{x}}_{t,j}\|$
9: $\dot{\hat{\boldsymbol{x}}}_t \leftarrow \frac{1}{\mu_i+\mu_j}\left(\mu_i\dot{\boldsymbol{x}}_{t,i} + \mu_j\dot{\boldsymbol{x}}_{t,j}\right)$
10: $\boldsymbol{C} \leftarrow \begin{bmatrix}\boldsymbol{H}_i & -\boldsymbol{H}_j\end{bmatrix}$
11: $\dot{\tilde{\boldsymbol{x}}}_t \leftarrow \boldsymbol{H}_{ij}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})\begin{bmatrix}\dot{\hat{\boldsymbol{x}}}_t^{\top} & \dot{\hat{\boldsymbol{x}}}_t^{\top}\end{bmatrix}^{\top}$
12: $\dot{\tilde{\boldsymbol{y}}}_i \leftarrow$ CoordinatedTPIK($\mathcal{A}_c$, $\dot{\tilde{\boldsymbol{x}}}_t$, $\boldsymbol{v}_i$)
13: **return** $\dot{\tilde{\boldsymbol{y}}}_i$

9. Each agent actuates the output of the second TPIK procedure (line 13).

In order to explain the rationale supporting the proposed coordination procedure, first of all note that, as regard the proposed choice for the assignment of weights μ_a, μ_b within (17) the following choice is actually made

$$\begin{aligned}\mu_a &= \mu_0 + \|\dot{\tilde{\boldsymbol{x}}}_t - \dot{\boldsymbol{x}}_{t,a}\| \triangleq \mu_0 + \|\boldsymbol{e}_a\|,\\ \mu_b &= \mu_0 + \|\dot{\tilde{\boldsymbol{x}}}_t - \dot{\boldsymbol{x}}_{t,b}\| \triangleq \mu_0 + \|\boldsymbol{e}_b\|,\end{aligned} \tag{19}$$

with $\mu_0 > 0$ and where the norms $\|\boldsymbol{e}_i\|$ serve as a measure of the difficulties that agent i has in tracking the original object reference velocity, due to higher priority safety and prerequisite tasks. Thus, with the above weighting choice, the resulting cooperative velocity $\dot{\hat{\boldsymbol{x}}}_t$ will be closer to the one evaluated by the agent exhibiting the greatest difficulty in tracking the original desired object velocity, than to the one evaluated by the other agent.

For example, in case $\|\boldsymbol{e}_a\|/\|\boldsymbol{e}_b\| \to \infty$, then $\dot{\hat{\boldsymbol{x}}}_t \to \dot{\boldsymbol{x}}_{t,a}$, which implies that agent a progressively imposes its non-cooperative tool-frame velocity as the cooperative one. Consequently, agent b will progressively follow what is imposed by agent a at the Cartesian level. Such a situation, and its opposite one, correspond to the extreme cases. In all the other situations, the weighting rule (19) provides an adequate compromise.

The above discussion is strongly based on the implicitly assumed property that nothing can change, for an agent, whenever an equality task is transferred to the highest priority, provided that its velocity is maintained equal to the result it provided when it was at a lower priority level. Such an assumed property, despite appearing as an evident one at a first glance, is proven in the Appendix.

Furthermore, it should be noted how the proposed coordination policy does not require an explicit, a priori, definition of any leader or follower roles (contribution C_2 highlighted in the introduction), since the tendency of an agent to become leader or follower within the team is governed by its comparative difficulty with the other agent in accomplishing the required object motion task. Such an emerging behaviour is very similar to how we operate in everyday life when transporting a big object with another human, as we naturally adapt to the other person if he/she is in difficulty, e.g. due to an obstacle or to a worse grasp of the object.

Finally, a remark should be made on the possibility of imposing desired internal wrenches. During step 7 (line 11 of Algorithm 2) of the coordination policy, the cooperative object velocity $\begin{bmatrix}\dot{\hat{\boldsymbol{x}}}_t^{\top} & \dot{\hat{\boldsymbol{x}}}_t^{\top}\end{bmatrix}^{\top}$ was projected using the projection matrix $\boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})$, thus guaranteeing that its result was feasible, i.e. compatible with object kinematic constraint, therefore ensuring that the motion of the agents would not generate *unwanted* internal object forces. However, (18) could be modified to take into account *desired* internal wrenches using the orthogonal space projection matrix $\boldsymbol{H}_{ab}\boldsymbol{C}^{\#}\boldsymbol{C}$:

$$\begin{bmatrix}\dot{\tilde{\boldsymbol{x}}}_{t,a}\\ \dot{\tilde{\boldsymbol{x}}}_{t,b}\end{bmatrix} \triangleq \boldsymbol{H}_{ab}(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})\begin{bmatrix}\dot{\hat{\boldsymbol{x}}}_t\\ \dot{\hat{\boldsymbol{x}}}_t\end{bmatrix} + \boldsymbol{H}_{ab}(\boldsymbol{C}^{\#}\boldsymbol{C})\begin{bmatrix}\dot{\tilde{\boldsymbol{x}}}_{w,a}\\ \dot{\tilde{\boldsymbol{x}}}_{w,b}\end{bmatrix}, \tag{20}$$

where $\dot{\tilde{\boldsymbol{x}}}_{w,i}$ are velocity requests that are projected to be against the constraint and therefore, after a small transient [10, 32], will generate wrenches in the same direction. For example, one could generate both $\dot{\tilde{\boldsymbol{x}}}_{w,i}$ to point from the i-th end-effector grasping point to the object's centre, to keep it compressed. However, this possibility could not be tested and validated during the experiments since the YouBot mobile manipulators are not equipped with wrench sensors able to measure the interaction with the object, and therefore is left for future investigations.

4. Experimental Results

In this section we present some experimental results of the proposed cooperative control architec-

ture. The setup used to test the proposed algorithm is composed by two Kuka YouBot mobile manipulators, as already shown in Fig. 3. The YouBot robot is constituted by a platform actuated by 4 omnidirectional wheels and by a 5 DOF manipulator. Therefore, the quantities defined in Section 2 can now be narrowed to the following ones:

- $\boldsymbol{c} \triangleq \begin{bmatrix}\boldsymbol{q}^\top & \boldsymbol{\eta}^\top\end{bmatrix}^\top \in \mathbb{R}^8$ contains the robot DOF, i.e. the arm joint positions $\boldsymbol{q} \in \mathbb{R}^5$ and the vehicle position and orientation vector $\boldsymbol{\eta} = \begin{bmatrix}x & y & \psi\end{bmatrix}^\top \in \mathbb{R}^3$;
- $\dot{\boldsymbol{y}} \triangleq \begin{bmatrix}\dot{\boldsymbol{q}}^\top & \boldsymbol{v}^\top\end{bmatrix}^\top \in \mathbb{R}^8$, and represents the controls to actuate the robot, i.e. the joint velocities $\dot{\boldsymbol{q}} \in \mathbb{R}^5$ and vehicle linear and yaw angular velocities $\boldsymbol{v} \triangleq \begin{bmatrix}\dot{x} & \dot{y} & \dot{\psi}\end{bmatrix}^\top \in \mathbb{R}^3$.

The two agents operate in a motion capture area, which is used to assess their absolute position, and to compute the transformation matrices necessary for setting their tool-frames $\langle t_a \rangle \equiv \langle t_b \rangle \equiv \langle o \rangle \triangleq \langle t \rangle$. The use of the motion capture environment allows to simplify the perception problem and to focus this work on the kinematic control strategy. Recent multi-robot localization and object tracking methodologies, such as [33], could be employed in lieu of the motion capture in a real scenario. During each experiment, the software process running the KCL is executed at 100 Hz. During every loop, the controller sends the computed quantities $\boldsymbol{H}_i$ and $\dot{\boldsymbol{x}}_{t,i}$ to the other involved agents using non-reliable UDP (User Datagram Protocol) packets (line 5 of Algorithm 2), and it uses the latest information received in the same manner from the other agents (line 6). The choice of UDP is made to avoid the queuing messages when only the latest one is actually useful. If one of the agents does not receive the corresponding information for a time longer than a time-out threshold, the cooperation process is stopped. No time synchronization is needed between the agents.

Finally, let us recall how the DCLs of the YouBot arm and vehicle are predefined and could not be modified. Hence, the KCL performances were tuned, acting on the task gains and thresholds, to cope with the underlying DCL performances.

4.1. Teleoperation Experiment

In the first experiment, a user teleoperates the cooperating agents by generating desired object velocities. The outcome of the experiment is shown through some snapshots in Fig. 5, while the user-generated object reference velocity $\dot{\tilde{\boldsymbol{x}}}_t$ is depicted in Fig. 6a.

Figure 6 shows an excerpt of the experiment, which can be seen in the attached video. In particular, at the start of the graphs ($t = 75$ s) the two mobile manipulators are in a configuration similar to the one represented in Fig. 5c, where agent b is close to a singular configuration. This is shown by the fact that the manipulability task's activation function is greater than zero, as depicted in Fig. 6b. From $t = 80$ s to $t = 100$ s the operator specifically gives a desired object reference velocity which agent b cannot track exactly. This fact can be seen comparing Fig. 6a with Fig. 6c and Fig. 6d for agents a and b respectively, and it is highlighted by the error norms $\|\dot{\tilde{\boldsymbol{x}}}_t - \dot{\boldsymbol{x}}_{t,i}\|$ shown in Fig. 6f. As can be seen in Fig. 6e, the *feasible cooperative* tool frame velocity is tuned in favour of agent b. Indeed, components on the linear z-direction are now present, which instead were zero in the original reference velocity. Thanks to the proposed coordination policy, agent b manages to keep its manipulability above the required threshold, as confirmed by the fact that its activation value in Fig. 6b never reaches the value one, while both systems instantaneously generate system velocities complying with the firm grasp object kinematic constraint.

4.2. Obstacle Avoidance Experiment

In this experiment an obstacle has been placed within the motion capture area. As the YouBots are not endowed with perception sensors, its position is known a priori to both agents. Of course, in a real scenario, the position of the obstacle would be perceived in real-time through the use of dedicated sensors and detection algorithms. The desired object velocity is generated to reach a final goal position on the other side of the motion capture area, with respect to the initial position of the YouBots and the object.

The execution of this experiment is depicted in Fig. 7. Roughly at $t = 24$ s, agent b is getting close to the obstacle, and therefore the activation function of the obstacle avoidance task raises, as shown in Fig. 8b. For this reason, the error norm $\|\boldsymbol{e}_b\|$ of the end-effector position and attitude tasks increases, as shown in Fig. 8f. The non-cooperative Cartesian velocity of agent b, shown in Fig. 8d, mainly differs from the desired object velocity (Fig. 8a) in the $\dot{x}$ and $\dot{y}$ linear velocity components. This is due to the fact that the vehicle needs to avoid

(a) (b) (c) (d)

Figure 5: Teleoperation experiment: snapshots of the two YouBots during the teleoperated manipulation.

the obstacle moving in the (x, y) horizontal plane. Instead, agent a has no problem in tracking the desired object velocity, and indeed its non-cooperative Cartesian velocity shown in Fig. 8c is very close to the desired one. Thanks to the proposed coordination policy, the two agents agree on a cooperative tool-frame velocity (Fig. 8e) which is substantially the one of agent b, due to its higher tracking error. In this way, the two YouBots coordinate themselves for avoiding the obstacle, as agent a adapts to the movement imposed by agent b for avoiding the obstacle.

5. Conclusions

The paper has presented a novel task-priority based approach to cooperative mobile manipulation (contribution C_1 highlighted in the introduction). The approach is based on a hierarchical control architecture, and exploits the task-priority based KCL [19] of each single agent. The proposed approach extends the KCL framework of each agent with a cooperative policy that allows the mobile manipulators to automatically tune, during the transportation, toward the agent in most difficulty (C_2). Several experiments have been performed and were presented in section 4, showing the capabilities of the two agents to coordinate themselves autonomously. In particular, the first experiment shows two agents transporting an object, which is teleoperated by a user. The two agents coordinate themselves to follow the desired object velocity, while satisfying their own control objectives. The experiment shows one agent avoid entering a singular posture, despite the user generated object velocity, and the other agent adapting to such a need. In the second experiment, the two agents need to transport the object to a goal position, however agent b has an obstacle in its path. The two agents successfully coordinate themselves to avoid it since, thanks to the proposed coordination policy, agent b automatically becomes the leader and agent a follows it in the avoidance manoeuvre.

In the present work, a method for managing internal wrenches was also briefly presented at the end of section 3.2. Future works will concern a deeper investigation of this possibility through suitable testing and validation campaigns, and possible automatic tuning procedures for the KCL gains.

Appendix

Let us consider an action $\mathcal{A} = \tau_A \prec \ldots \prec \tau_K \prec \ldots \prec \tau_Z$, and where τ_K is an equality task. The TPIK procedure proposed in section 2.6 is used to solve such an hierarchy of tasks, and let us call its output ${}^1\dot{\boldsymbol{y}}^*$, which is the optimal control vector satisfying the priority requirements. To such an optimal control vector the associated optimal task velocities are ${}^1\dot{\boldsymbol{x}}_i^* = \boldsymbol{J}_i {}^1\dot{\boldsymbol{y}}^*, i = A, \ldots, Z$.

Let us also consider the following orthogonal decomposition of ${}^1\dot{\boldsymbol{y}}^*$:

$$\begin{aligned} {}^1\dot{\boldsymbol{y}}^* &= \boldsymbol{J}_K^{\#} \boldsymbol{J}_K {}^1\dot{\boldsymbol{y}}^* + \left(\boldsymbol{I} - \boldsymbol{J}_K^{\#}\boldsymbol{J}_K\right) {}^1\dot{\boldsymbol{y}}^* \\ &= \boldsymbol{J}_K^{\#} {}^1\dot{\boldsymbol{x}}_K^* + \left(\boldsymbol{I} - \boldsymbol{J}_K^{\#}\boldsymbol{J}_K\right) {}^1\dot{\boldsymbol{y}}^*, \end{aligned} \tag{21}$$

which can be interpreted as follows: not only ${}^1\dot{\boldsymbol{y}}^*$ is the optimal control vector of the whole hierarchy, but it can also be seen as the optimal solution in the null space of task τ_K, constrained to have a task velocity ${}^1\dot{\boldsymbol{x}}_K^*$.

Now, consider a new task hierarchy where τ_K is at the highest priority, therefore $\tau_K \prec \tau_A \prec \ldots \prec \tau_J \prec \tau_L \ldots \prec \tau_Z$. Let us further impose that the reference for the task τ_K is chosen as ${}^1\dot{\boldsymbol{x}}_K^*$, i.e. the best task velocity that could be obtained when τ_K was at its original priority order. The manifold of solutions, after solving the only first task τ_K in the new hierarchy is

$${}^2\dot{\boldsymbol{y}}_1 = \boldsymbol{J}_K^{\#} {}^1\dot{\boldsymbol{x}}_K^* + \left(\boldsymbol{I} - \boldsymbol{J}_K^{\#}\boldsymbol{J}_K\right) \dot{\boldsymbol{z}}, \forall \dot{\boldsymbol{z}}, \tag{22}$$

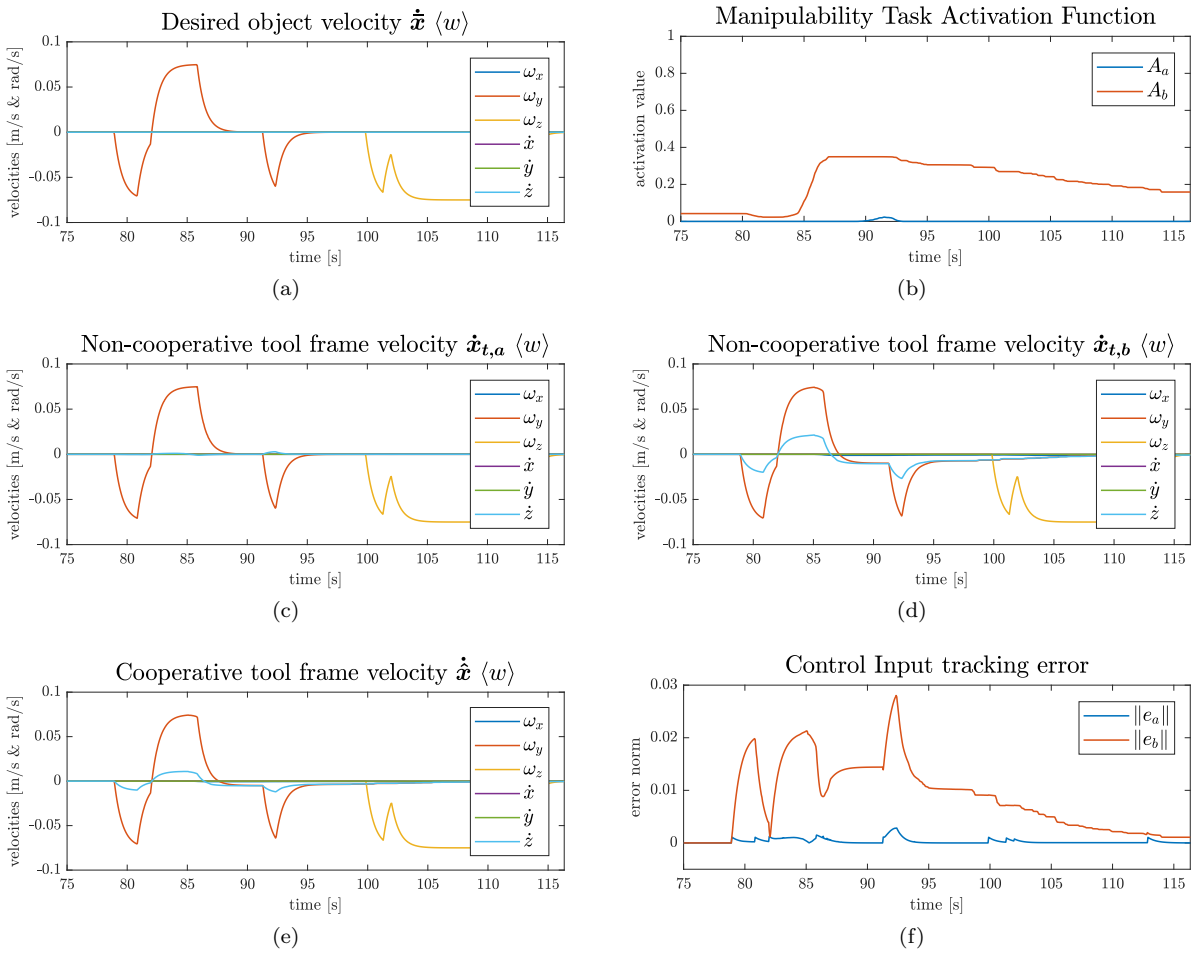

Figure 6: Teleoperation experiment results: (a) the desired object velocity, (b) manipulability task activation function, (c) non-cooperative tool-frame velocities for agent a (d) non-cooperative tool-frame velocities for agent b, (e) cooperative tool-frame velocities (tuned toward agent b), (f) norms of object velocity tracking errors, showing how agent b is in higher difficulty.

since τ_K it is an equality task. Let us immediately notice that the task τ_K velocity is still ${}^1\dot{\boldsymbol{x}}_K^*$:

$$ {}^2\dot{\boldsymbol{x}}_K^* = \boldsymbol{J}_K {}^2\dot{\boldsymbol{y}}_1^* = \boldsymbol{J}_K \boldsymbol{J}_K^{\#} {}^1\dot{\boldsymbol{x}}_K^* = {}^1\dot{\boldsymbol{x}}_K^*, \tag{23} $$

since ${}^1\dot{\boldsymbol{x}}_K^* \in \text{Span}(\boldsymbol{J}_K)$ and $\boldsymbol{J}_K \left(\boldsymbol{I} - \boldsymbol{J}_K^{\#} \boldsymbol{J}_K \right) = \boldsymbol{0}$. Therefore, (22) is the manifold of solutions of task τ_K constrained to have a velocity ${}^1\dot{\boldsymbol{x}}_K^*$, i.e. it is the same as (21).

Let us call the output of this second TPIK procedure as ${}^2\dot{\boldsymbol{y}}^*$. Then, let us express it in the following orthogonal decomposition:

$$ \begin{aligned} {}^2\dot{\boldsymbol{y}}^* &= \boldsymbol{J}_K^{\#} \boldsymbol{J}_K {}^2\dot{\boldsymbol{y}}^* + \left(\boldsymbol{I} - \boldsymbol{J}_K^{\#} \boldsymbol{J}_K \right) {}^2\dot{\boldsymbol{y}}^* \\ &= \boldsymbol{J}_K^{\#} {}^1\dot{\boldsymbol{x}}_K^* + \left(\boldsymbol{I} - \boldsymbol{J}_K^{\#} \boldsymbol{J}_K \right) {}^2\dot{\boldsymbol{y}}^*, \end{aligned} \tag{24} $$

where ${}^1\dot{\boldsymbol{x}}_K^* = \boldsymbol{J}_K {}^2\dot{\boldsymbol{y}}^*$ follows from the fact that the optimization of subsequent tasks $\tau_A \prec \ldots \prec \tau_J \prec \tau_L \ldots \prec \tau_Z$ has been done in the null space of τ_k.

Since the optimal velocity of task τ_k is the same in the two procedures, i.e. ${}^2\dot{\boldsymbol{x}}_K^* = {}^1\dot{\boldsymbol{x}}_K^*$, and the relative priority order of all the other tasks is unchanged, it follows that ${}^2\dot{\boldsymbol{y}}^* = {}^1\dot{\boldsymbol{y}}^*$ and the optimal task velocities will be exactly the same as in the original TPIK procedure. If, absurdly, this was not the case, then it would mean that even if the task velocity ${}^2\dot{\boldsymbol{x}}_K^* = {}^1\dot{\boldsymbol{x}}_K^*$, and the optimization structure is the same in (21) and (24), i.e. in the null space of task τ_K, there are two different sets of optimal task velocities for tasks $i = A, \ldots, Z, i \neq K$.

(a) (b) (c) (d)

Figure 7: Obstacle avoidance experiment: snapshots of the two YouBots as they cooperatively avoid the obstacle and reach the desired final position.

References

[1] F. Caccavale, M. Uchiyama, Cooperative manipulation, in: Springer Handbook of Robotics, Springer, 2016, pp. 989–1006.

[2] O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal, Coordination and decentralized cooperation of multiple mobile manipulators, Journal of Field Robotics 13 (11) (1996) 755–764.

[3] R. Holmberg, O. Khatib, Development and control of a holonomic mobile robot for mobile manipulation tasks, The International Journal of Robotics Research 19 (11) (2000) 1066–1074.

[4] T. G. Sugar, V. Kumar, Control of cooperating mobile manipulators, IEEE Transactions on robotics and automation 18 (1) (2002) 94–103.

[5] A. Yamashita, T. Arai, J. Ota, H. Asama, Motion planning of multiple mobile robots for cooperative manipulation and transportation, IEEE Transactions on Robotics and Automation 19 (2) (2003) 223–237.

[6] N. Miyata, J. Ota, T. Arai, H. Asama, Cooperative transport by multiple mobile robots in unknown static environments associated with real-time task assignment, IEEE transactions on robotics and automation 18 (5) (2002) 769–780.

[7] A. Yufka, M. Ozkan, Formation-based control scheme for cooperative transportation by multiple mobile robots, International Journal of Advanced Robotic Systems 12 (9) (2015) 120.

[8] Z. Wang, M. Schwager, Force-amplifying n-robot transport system (force-ants) for cooperative planar manipulation without communication, The International Journal of Robotics Research 35 (13) (2016) 1564–1586.

[9] H. Yamaguchi, A. Nishijima, A. Kawakami, Control of two manipulation points of a cooperative transportation system with two car-like vehicles following parametric curve paths, Robotics and Autonomous Systems 63 (2015) 165–178.

[10] G. Casalino, G. Cannata, G. Panin, A. Caffaz, On a two-level hierarchical structure for the dynamic control of multifingered manipulation, in: Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, Vol. 1, IEEE, 2001, pp. 77–84.

[11] S. Erhart, S. Hirche, Internal force analysis and load distribution for cooperative multi-robot manipulation, IEEE Transactions on Robotics 31 (5) (2015) 1238–1243.

[12] S. Erhart, S. Hirche, Model and analysis of the interaction dynamics in cooperative manipulation tasks, IEEE Transactions on Robotics 32 (3) (2016) 672–683.

[13] J.-M. Yang, J.-H. Kim, Sliding mode control for trajectory tracking of nonholonomic wheeled mobile robots, IEEE Transactions on robotics and automation 15 (3) (1999) 578–587.

[14] D. Xu, D. Zhao, J. Yi, X. Tan, Trajectory tracking control of omnidirectional wheeled mobile manipulators: robust neural network-based sliding mode approach, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 (3) (2009) 788–799.

[15] E. Simetti, G. Casalino, F. Wanderlingh, M. Aicardi, Task priority control of underwater intervention systems: Theory and applications, Ocean Engineering 164 (2018) 40–54.

[16] O. Kanoun, F. Lamiraux, P. B. Wieber, Kinematic control of redundant manipulators: generalizing the task-priority framework to inequality task, IEEE Transactions on Robotics 27 (4) (2011) 785–792. doi:10.1109/tro.2011.2142450.

[17] A. Escande, N. Mansard, P.-B. Wieber, Hierarchical quadratic programming: Fast online humanoid-robot motion generation, The International Journal of Robotics Research 33 (7) (2014) 1006–1028. doi:10.1177/0278364914521306.

[18] S. Moe, G. Antonelli, A. R. Teel, K. Y. Pettersen, 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 3 (2016) 16. doi:10.3389/frobt.2016.00016.

[19] E. Simetti, G. Casalino, A novel practical technique to integrate inequality control objectives and task transitions in priority based control, Journal of Intelligent & Robotic Systems 84 (1) (2016) 877–902. doi:10.1007/s10846-016-0368-6.

[20] H. Sadeghian, L. Villani, M. Keshmiri, B. Siciliano, Task-space control of robot manipulators with null-space compliance, IEEE Transactions on Robotics 30 (2) (2014) 493–506.

[21] E. Simetti, G. Casalino, S. Torelli, A. Sperindé, A. Turetta, Floating underwater manipulation: Developed control methodology and experimental validation within the trident project, Journal of Field Robotics 31 (3) (2014) 364–385. doi:10.1002/rob.21497.

[22] E. Simetti, F. Wanderlingh, S. Torelli, M. Bibuli, A. Odetti, G. Bruzzone, D. Lodi Rizzini, J. Aleotti, G. Palli, L. Moriello, U. Scarcia, Autonomous underwater intervention: Experimental results of the MARIS project, IEEE Journal of Oceanic Engineering (2017) 1–20doi:10.1109/JOE.2017.2733878.

[23] K. Baizid, G. Giglio, F. Pierri, M. A. Trujillo, G. Antonelli, F. Caccavale, A. Viguria, S. Chiaverini, A. Ollero, Behavioral control of unmanned aerial vehicle manipulator systems, Autonomous Robots 41 (5)

(2017) 1203–1220. doi:10.1007/s10514-016-9590-0.

[24] N. G. Tsagarakis, D. G. Caldwell, F. Negrello, W. Choi, L. Baccelliere, V. Loc, J. Noorden, L. Muratore, A. Margan, A. Cardellino, L. Natale, E. Mingo Hoffman, H. Dallali, N. Kashiri, J. Malzahn, J. Lee, P. Kryczka, D. Kanoulas, M. Garabini, M. Catalano, M. Ferrati, V. Varricchio, L. Pallottino, C. Pavan, A. Bicchi, A. Settimi, A. Rocchi, A. Ajoudani, Walkman: A high-performance humanoid platform for realistic environments, Journal of Field Robotics 34 (7) (2017) 1225–1259.

[25] T. Tarn, A. Bejczy, N. Xi, Intelligent motion planning and control for robot arms, IFAC Proceedings Volumes 26 (2, Part 3) (1993) 677 – 680, 12th Triennal Wold Congress of the International Federation of Automatic control. Volume 3 Applications I, Sydney, Australia, 18-23 July. doi:10.1016/S1474-6670(17)48813-6.

[26] N. Xi, T.-J. Tarn, A. K. Bejczy, Intelligent planning and control for multirobot coordination: An event-based approach, IEEE transactions on robotics and automation 12 (3) (1996) 439–452.

[27] Z. Li, P. Y. Tao, S. S. Ge, M. Adams, W. S. Wijesoma, Robust adaptive control of cooperating mobile manipulators with relative motion, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 (1) (2009) 103–116.

[28] S. Erhart, D. Sieber, S. Hirche, An impedance-based control architecture for multi-robot cooperative dual-arm mobile manipulation, in: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, IEEE, 2013, pp. 315–322.

[29] R. Jamisola, M. H. Ang, D. Oetomo, O. Khatib, T. M. Lim, S. Y. Lim, The operational space formulation implementation to aircraft canopy polishing using a mobile manipulator, in: Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), Vol. 1, IEEE, 2002, pp. 400–405.

[30] K. Darvish, F. Wanderlingh, B. Bruno, E. Simetti, F. Mastrogiovanni, G. Casalino, Flexible human–robot cooperation models for assisted shop-floor tasks, Mechatronics 51 (2018) 97–114. doi:10.1016/j.mechatronics.2018.03.006.

[31] A. Ben-Israel, T. Greville, Generalized inverses: theory and applications, Vol. 15, Springer Verlag, 2003.

[32] M. Aicardi, G. Casalino, G. Cannata, Contact force canonical decomposition and the role of internal forces in robust grasp planning problems, The International journal of robotics research 15 (4) (1996) 351–364.

[33] A. Ahmad, G. Lawless, P. Lima, An online scalable approach to unified multirobot cooperative localization and object tracking, IEEE Transactions on Robotics 33 (5) (2017) 1184–1199.

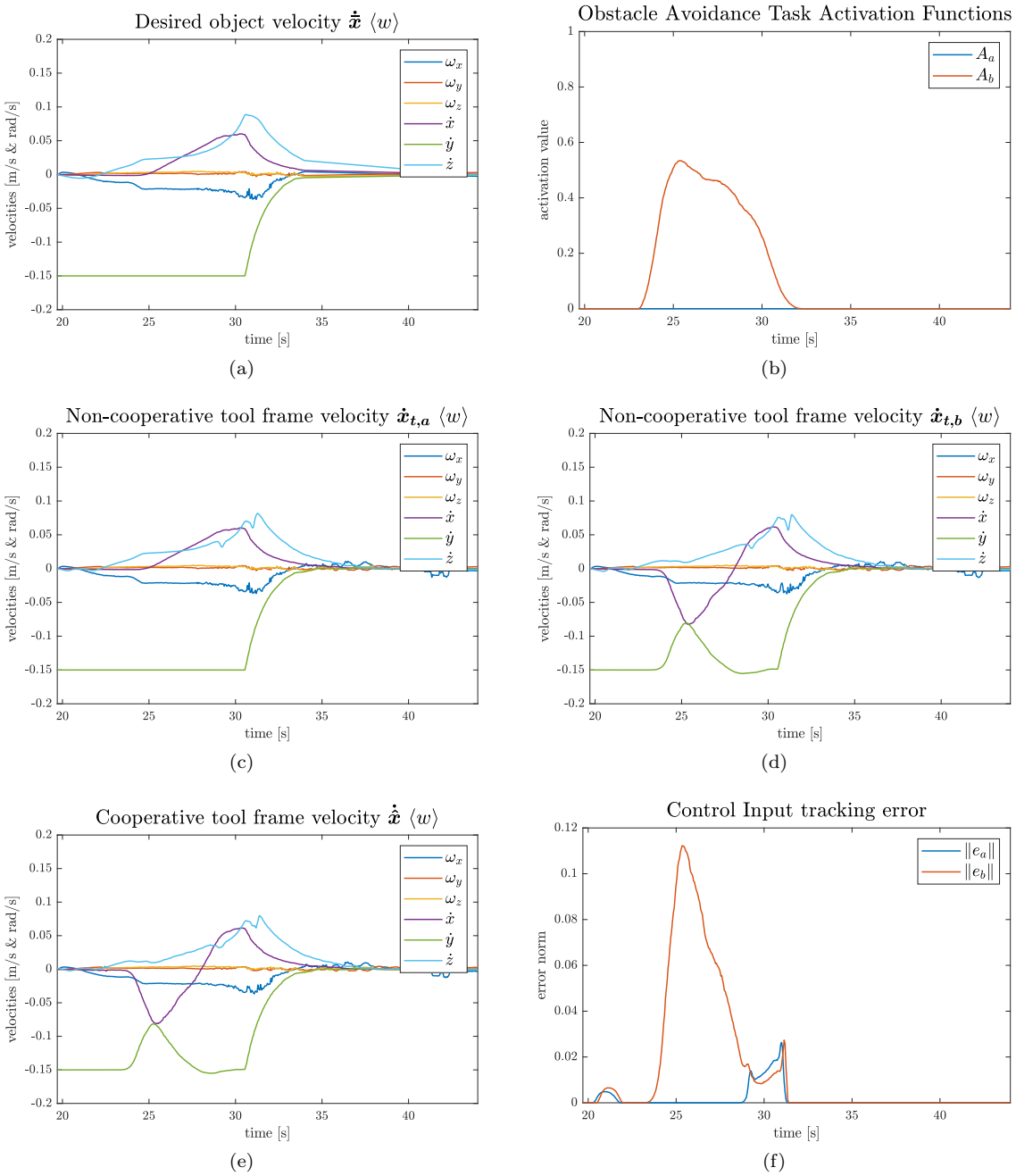

Figure 8: Obstacle avoidance experiment results: (a) the desired object velocity, (b) obstacle avoidance task activation function, (c) non-cooperative tool-frame velocities for agent a (d) non-cooperative tool-frame velocities for agent b, (e) cooperative tool-frame velocities (tuned toward agent b), (f) norms of object velocity tracking errors, showing how agent b is in higher difficulty.