Cooperative Object Transportation through an Unstructured Environment

TORIELLI Davide FUSARO Fabio

EMARO - European Master on Advanced Robotics
DIBRIS - Dip. Informatica, Bioingegneria, Robotica, Ing. Dei Sistemi
Università Degli Studi Di Genova, 16th November 2018

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. This work is focused on cooperation of two mobile robots (KUKA YouBot) in a difficult enviroment, each one with a 5-DOF arm on it, carrying a common load with their end-effectors. In particular we focused on the obstacle avoidance task.

1 Introduction

The cooperation of multi-robot systems is a hot topic, important for many applications. There are situations in which a load is too big or too heavy to be carried by a single agent; so a collaboration among multiple agents for a safe and/or a physically possible transportation is necessary.
In this work we focused on two mobile robots with arms, but the theory which we based on is applicable in many others fields (for example, submarine vehicles coordination [1]).

The first step is to control the single robot. As seen in other works [2], this is done through a task priority inverse kinematic (TPIK) approach. This strategy permits the robot to accomplish various tasks in order of priority and it allows also to not consider one (or more) of them that are not important in a particular moment (for example, it is not necessary to avoid an obstacle if this is far away).
Then this approach is extended to include the cooperation between the two robots, as in [3] [4]. The chosen approach is not a leader-follower one; here the agents help each other after understanding which one is in a more difficult situation (for example, a near obstacle). Starting from here, we added a new task: the obstacle avoidance.
The work was done in EMAROLAB of University of Genova, with two KUKA youBot running in a Motion Capture (MoCap) enviroment.

The paper is structured as follows.
Section 2 describes the aim of the project. Section 3 recalls the theory of the used task priority framework. Section 4 explains the development of the obstacle avoidance task and section 5 presents the achieved experimental results. Finally, conclusions in section 6 are given.

2 Scope of work

We will now briefly explain the problem we have faced. As can be seen in Fig. 1 the two robots are approaching an obstacle. Obviously the aim is to avoid it with both robots. This new task is added together with the other ones, like strictly keeping the load or avoiding singular positions, in the TPIK list.

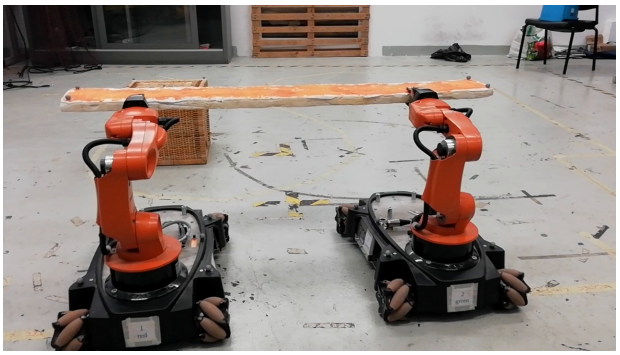


Figure 1: *The youBots approaching the obstacle*

As said previously, the robot used is the KUKA youBot that is a mobile robot composed by two parts: a mobile platform and an arm. The platform is an omnidirectional base with four mecanum wheels; the arm is a five degrees of freedoms manipulator with two-finger gripper [5].

3 Theory behind the work

Here we summarize the theory explained deeper in [3] and [4].

3.1 Control Architecture

The control architecture for the single robot is represented in Fig. 2. There are three main blocks:

1. The Mission Manager is the supervisor of the current *mission* (i.e. sequence of actions $\mathcal{A}_j$) and generates the actions to be executed by the Kinematic Control Layer.

2. The Kinematic Control Layer (KCL) is in charge of reactively accomplishing the *control objectives* generating the desired system velocity vector.

3. The Dynamic Control Layer (DCL) generates appropriate forces or torques to track the desired system velocity vector.

This work focuses on the KCL where the task priority approach is implemented.

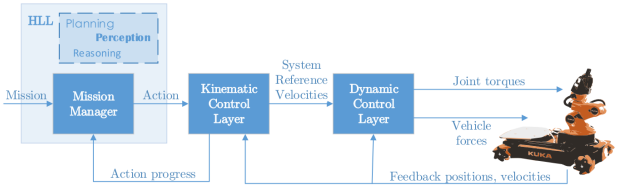


Figure 2: *The overall control architecture*

3.2 Single Agent

In the following paragraphs, the robot configuration vector is referred to as $\boldsymbol{c} \triangleq [\boldsymbol{q} \quad \boldsymbol{\eta}]^T \in \mathbb{R}^n$ where $\boldsymbol{q} \in \mathbb{R}^l$ is the arm joint position vector and $\boldsymbol{\eta} \in \mathbb{R}^6$ is the vehicle position and orientation vector. The robot velocity vector is named $\dot{\boldsymbol{y}} \triangleq [\dot{\boldsymbol{q}} \quad \boldsymbol{v}]^T \in \mathbb{R}^n$, and represents the controls to actuate the robot, e.g., joint velocities and vehicle linear and angular velocities. For the youBot mobile manipulators used in the experiments, both $\boldsymbol{\eta} \in \mathbb{R}^3$ and $\boldsymbol{v} \in \mathbb{R}^3$ consist only in the horizontal position and velocity, plus the yaw angle and its derivative.

3.2.1 Control Objective

Considering a scalar variable $x_o(\mathbf{c})$ 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 and $x_{o,min}$ and $x_{o,max}$ are lower and upper thresholds among which the variable must stay. These requirements model the objectives that the system should satisfy, for example avoiding obstacle, respecting joint limits, reaching desired position.

3.2.2 Control Tasks

Remembering that 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), the feedback reference rate $\dot{\bar{x}}_o$ is defined as

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

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_0 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. In such cases, there is no explicit control objective specified to the system and so we talk about *non-reactive control task*.

3.2.3 Control Actions

From the robot control standpoint, an action $\mathcal{A}$ can be defined as a prioritized list of m control objectives $o_1, ..., o_m$ each one with associated control task $\tau_1, ..., \tau_m$. These tasks have to be managed *concurrently* which means that we must satisfy the most prioritized objectives first and then, if possible, the others.

3.2.4 Activation Functions

Control objectives and their associated reactive control task are relevant depending on the current value of the system configuration vector $\mathbf{c}$. For example, it is useless to overconstrain the system adding a task that avoid an obstacle that is far away. So, an *activation function* is used

$$a_o^i(x_o) \in [0, 1] \qquad (2)$$

to *activate / deactivate* the task. This is a continuous sigmoid function of the scalar objective variable x_o, whose value is zero when we are in the validity region of the associated control objective o.
For non-reactive control tasks, the variable $x(\boldsymbol{c})$ is not defined and the (2) is simply $a_o^i(x_o) \equiv 1$.
When we talk about different actions $\mathcal{A}_j$, the (2) is modified to become

$$a_o(x_o, \boldsymbol{p}) = a_o^i(x_o)a_o^p(\boldsymbol{p}) \qquad (3)$$

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

3.2.5 Task Priority Inverse Kinematics

For an action $\mathcal{A}$ we define:

- $\dot{\bar{\boldsymbol{x}}}_k \triangleq [\dot{\bar{x}}_{1,k}, ..., \dot{\bar{x}}_{m,k}]^T$ is the stacked vector of all the reference rates, where the first index indicates control task $\tau_1, ..., \tau_m$ 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 $[\dot{x}_{1,k}, ..., \dot{x}_{m,k}]^T$ with respect to the system velocity vector $\dot{\boldsymbol{y}}$.

- $\boldsymbol{A}_k \triangleq diag(a_{1,k}, ..., a_{m,k})$ is the diagonal matrix of all the activation functions in the form of (2).

Now, the control problem is to find the system velocity reference vector $\dot{\bar{\boldsymbol{y}}}$ that satisfies *at best* the requirements. As in [3], this is expressed by the following minimization problems:

$$S_k \triangleq \left\{ arg\, R\text{-}\min_{\dot{\bar{\boldsymbol{y}}} \in S_{k-1}} \left\| \boldsymbol{A}_k(\dot{\bar{\boldsymbol{x}}}_k - \boldsymbol{J}_k\dot{\bar{\boldsymbol{y}}}) \right\|^2 \right\} \qquad (4)$$

where S_{k-1} is the manifold of solutions of all the previous tasks in the hierarchy, which means that $S_0 \triangleq \mathbb{R}$.

3.2.6 Vehicle-Arm Coordination

If the whole body Jacobian were employed to control the end-effectors, disturbances of the mobile platform would propagate immediately through the coupled kinematics to the end effectors. To cope with this problem, the idea of [3] is to have two TPIK procedures running in parallel as depicted in Fig. 3.

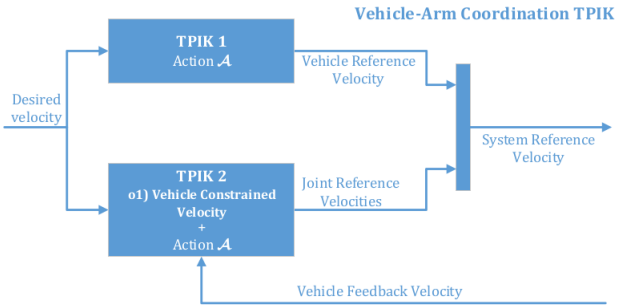


Figure 3: *Vehicle-Arm coordination scheme for action* $\mathcal{A}$

1. **TPIK 1** considers the vehicle together with the manipulator. Of the whole result $\dot{\boldsymbol{y}}$, only the vehicle reference velocity is used, while the manipulator part is discarded.

2. **TPIK 2** considers the vehicle as a *non-controllable* entity. A *non-reactive* control task is added with the highest priority. Its role is to constrain the desired vehicle velocities equal to the *current measured* ones. The outputs of this procedure are the *optimal* joint velocities in correspondence of the *measured* vehicle velocity.

Thanks to the TPIK 2 optimization, the arm joint velocities are always the optimal ones based on the current vehicle velocity, independently of any vehicle inaccuracy in tracking the desired one generated by TPIK 1. Also, this method copes with multi-rate control requirements of the two different subsystems.

3.3 Cooperative Agents

We now focus on the case of multiple cooperative mobile manipulators; for simplicity we will consider two robots but the approach can be applied to multiple agents.

3.3.1 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. 4.

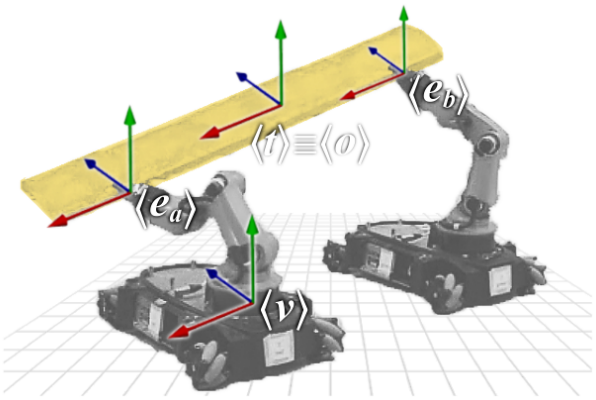


Figure 4: *Frames involved in the cooperative mobile manipulation*

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

$$\dot{\boldsymbol{x}}_t = \boldsymbol{J}_{t,a}\dot{\boldsymbol{y}}_a = \boldsymbol{J}_{t,b}\dot{\boldsymbol{y}}_b, \tag{5}$$

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 Jacobians with respect to $\langle t \rangle$.
Let us rewrite the second equation in (5) as

$$\begin{bmatrix} \boldsymbol{J}_{t,a} & -\boldsymbol{J}_{t,b} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{y}}_a \\ \dot{\boldsymbol{y}}_b \end{bmatrix} \triangleq \boldsymbol{G}\dot{\boldsymbol{y}}_{ab} = 0 \iff \dot{\boldsymbol{y}}_{ab} \in ker(\boldsymbol{G}), \tag{6}$$

which represents the subspace where $\dot{\boldsymbol{y}}_{ab}$ is constrained to lay as a consequence of the firm grasp assumption.
Before proceeding with the coordination policy, let us consider the subspace of admissible Cartesian velocities of both systems which

derived from (6) by eliminating the internal motion components within each subsystems

$$(\boldsymbol{J}_{t,a}\boldsymbol{J}_{t,a}^{\#} - \boldsymbol{J}_{t,b}\boldsymbol{J}_{t,b}^{\#})\dot{\boldsymbol{x}}_t \triangleq \boldsymbol{C}\dot{\boldsymbol{x}}_t = \boldsymbol{0} \qquad (7)$$

thus implying $\dot{\boldsymbol{x}}_t \in ker(\boldsymbol{C}) = Span(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})$, where the kernel of the matrix $\boldsymbol{C}$ expresses the space of achievable object velocities at the current configuration. Therefore, if a desired object velocity $\dot{\tilde{\boldsymbol{x}}}_t$ lying in this subspace is commanded separately to both agents and the corresponding non-reactive end-effector velocity tracking task is located at the highest priority in both agents task priority lists, then the object kinematic constraint is satisfied at kinematic level.
We now proceed with the description of the coordination policy.

3.3.2 Coordination Policy

During each sampling interval, the following sequential steps are executed, as sketched in Fig. 5 :

1. Each agent runs the TPIK procedure detailed in subsection 3.2 as if it were the only one acting on the object. The two TPIK procedures separately provide the vectors $\dot{\boldsymbol{y}}_a$, $\dot{\boldsymbol{y}}_b$.

2. Each agent evaluates the Cartesian *non-cooperative* tool-frame velocities

$$\dot{\boldsymbol{x}}_{t,i} = \boldsymbol{J}_{t,i}\dot{\boldsymbol{y}}_i, \quad i = a, b \qquad (8)$$

   Moreover, each one evaluates the matrix $\boldsymbol{J}_{t,i}\boldsymbol{J}_{t,i}^{\#}$ which is the representation of the admissible tool-frame velocity space.

3. Both agents transfer their computed quantities to the Coordinator.

4. The Coordinator performs the following steps:

   a) It evaluates the cooperative tool-frame velocity vector

$$\dot{\hat{\boldsymbol{x}}}_t = \frac{1}{\mu_a + \mu_b}(\mu_a\dot{\boldsymbol{x}}_{t,a} + \mu_b\dot{\boldsymbol{x}}_{t,b}), \quad \mu_a, \mu_b > 0 \qquad (9)$$

   which corresponds to a weighted compromise between the two output velocities $\dot{\boldsymbol{x}}_{t,a}, \dot{\boldsymbol{x}}_{t,b}$. In the general case, i.e. $\dot{\boldsymbol{x}}_{t,a} \neq \dot{\boldsymbol{x}}_{t,b}$, the (9) might not lay in the space of feasible object velocities, therefore it must be projected on this subspace, as performed in the next two steps.

   b) It evaluates the Cartesian constraint matrix

$$\boldsymbol{C} \triangleq \left(\boldsymbol{J}_{t,a}\boldsymbol{J}_{t,a}^{\#} - \boldsymbol{J}_{t,b}\boldsymbol{J}_{t,b}^{\#}\right) \qquad (10)$$

   c) It projects the evaluated cooperative velocity vector $\dot{\hat{\boldsymbol{x}}}_t$ on the feasible velocity space $Span(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C})$ for the constrained object, obtaining the so-called *feasible cooperative* velocity vector

$$\dot{\tilde{\boldsymbol{x}}}_t \triangleq \left(\boldsymbol{I} - \boldsymbol{C}^{\#}\boldsymbol{C}\right)\dot{\hat{\boldsymbol{x}}}_t \qquad (11)$$

   d) It transfers the so-computed Cartesian velocity vector $\dot{\tilde{\boldsymbol{x}}}_t$ to both agents.

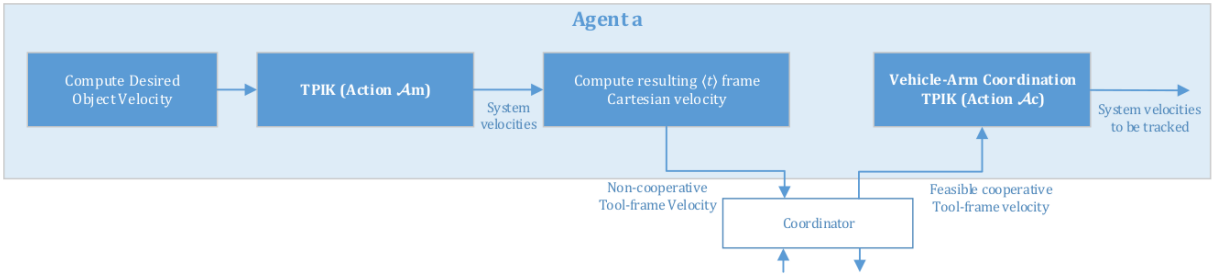


Figure 5: *Block diagram of the cooperation, as seen from agent* a

5. Both agents separately run a new TPIK procedure, implementing the vehicle arm coordination of subsection 3.2.6 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*. Such a new hierarchy is termed action $\mathcal{A}_c$ in Fig. 5.

6. Each agent actuates the output of the second TPIK procedure.

As regard of weights μ_a, μ_b within (9) the following choice is actually made

$$\begin{aligned} \mu_a &= \mu_0 + \|\dot{\tilde{\boldsymbol{x}}}_t - \dot{\boldsymbol{x}}_{t,a}\| \triangleq \mu_0 + \|\boldsymbol{e}_a\|, \\ \mu_b &= \mu_0 + \|\dot{\tilde{\boldsymbol{x}}}_t - \dot{\boldsymbol{x}}_{t,b}\| \triangleq \mu_0 + \|\boldsymbol{e}_b\|, \\ &\mu_0 > 0 \end{aligned} \tag{12}$$

Where the norm $\|\boldsymbol{e}_i\|$ can be interpreted as a global measure of the difficulties that agent i has in tracking the original object reference velocity. Thus, with the above weighting choice, the resulting feasible cooperative velocity $\dot{\tilde{\boldsymbol{x}}}_t$ will be closer to the one evaluated by the agent exhibiting the greatest difficulty in tracking the original desired object velocity.

4 Proposed Solution: Obstacle Avoidance Task

As mentioned in the introduction, the contribution of this work regards the addition of the obstacle avoidance task in the TPIK list. The obstacle shape is approximated to a hemisphere which ideally creates a bounding box around the real obstacle. Its position and its dimension (its radius r) are hardcoded, i.e. we do not use sensors or the MoCap cameras to detect it.
We design the task using control objective which controls the distance $\boldsymbol{d}_i$ from each robot frame i (the 5 joints) to the bounding box. This is done individually for both youBots. We will use 'ja' subscript to indicate 'joint avoidance'.
The reference rate $\dot{\tilde{x}}_{ja,i}$ for each robot frame is defined as follows:

$$\begin{aligned} \dot{\tilde{x}}_{ja,i} &= \gamma_{ja}((r + \Delta_{ja}) - \|\boldsymbol{d}_i\|), \quad i = 1, ..., 5, \\ \dot{\tilde{\boldsymbol{x}}}_{ja} &= [\dot{\tilde{x}}_{ja,1}, ..., \dot{\tilde{x}}_{ja,5}]^T \end{aligned} \tag{13}$$

where γ_{ja} is the control gain and Δ_{ja} is a constant that define a safety distance from the obstacle.
Being $\boldsymbol{n}_i = \frac{\boldsymbol{d}_i}{\|\boldsymbol{d}_i\|}$ the distance versor from each robot frame i to the obstacle frame, we define the Jacobian $\boldsymbol{J}_{ja} \in \mathbb{R}^{5x8}$ for this task:

$$\begin{aligned} \boldsymbol{J}_{ja,i} &= \boldsymbol{n}_i^T \boldsymbol{J}_i(\boldsymbol{c}), \quad i = 1, ..., 5 \\ \boldsymbol{J}_{ja} &= [\boldsymbol{J}_{ja,1}, ..., \boldsymbol{J}_{ja,5}]^T \end{aligned} \tag{14}$$

Pre-multiplying the arm Jacobian $\boldsymbol{J}_i(\boldsymbol{c})$ by $\boldsymbol{n}_i^T$ we are requesting to the system to generate a velocity which points away from the obstacle.
As mentioned in section 3.2.4, the activation function has a decreasing sigmoid shape which activates the task only when the joint is approaching the Δ_{ja}-transition zone.
This task is added to the priority list of both youBots and treated as the other tasks. The associated objective, being a system safety one, is placed at the second priority level after the physical constraints objectives (i.e. interacting with the enviroment).
For the point of view of the cooperation, things proceed as described in 3.3.

5 Experimental Results

In this section we present the system architecture, as we can see in Fig. 6 and the results obtained with the proposed kinematic control strategy.

5.1 System Setup

The odometry of the wheels, in general, is not precise and, for the collaboration, the agents have to know the position of each other and their position respect to the world frame; we need an external view to understand where the robots are located. So the environment where the robots work is fully observed by 8 OptiTrack Flex3 cameras which detect the 6

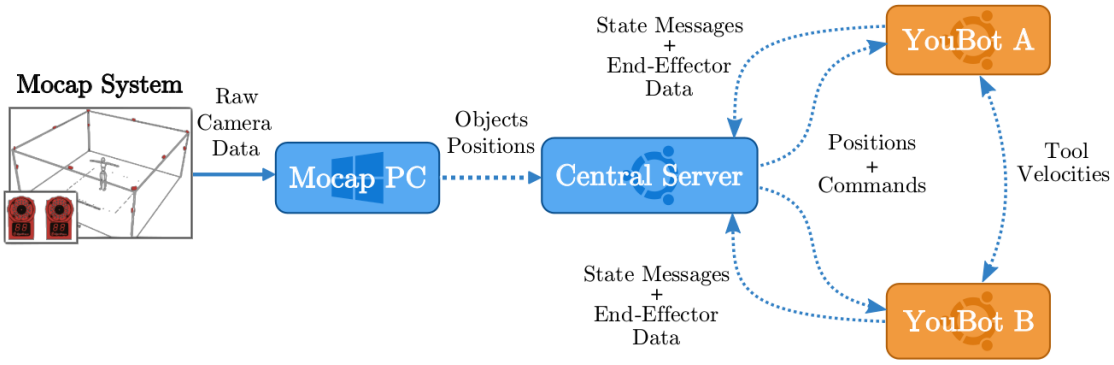
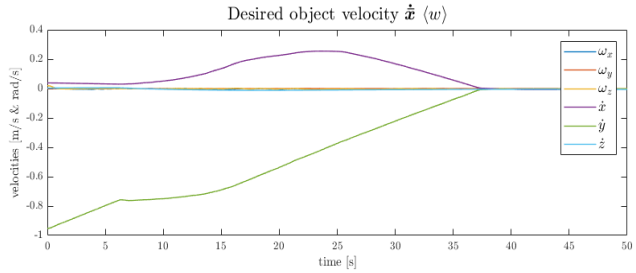


Figure 6: *Overall setup scheme, where the arrows indicate the UDP data flow, wired (solid) and wireless(dotted)*

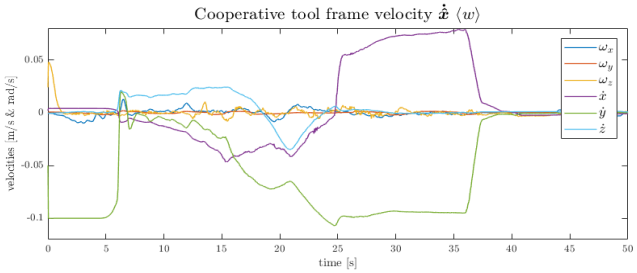
markers putted on each robot. The MoCap PC with Motive software constructs the two rigid bodies of the robots from the 6 markers and sends their position to the central server via UDP.
The central server is the master which sends the relevant data to the two youBots and it serves the purpose of being a unified console for controlling the two agents.

5.2 Obtained Results

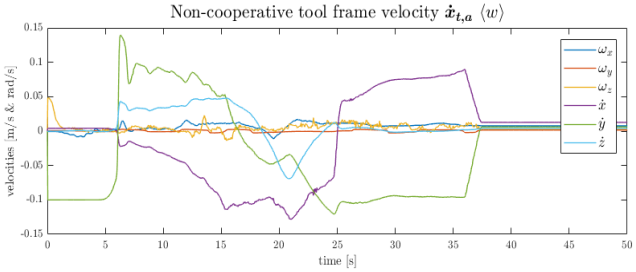
At the beginning of the graphs the two youBots are in a configuration similar to the one represented in Fig. 1. Please note that, due to the MoCap system frame, the final position has negative component on y axes; so the velocity needed on y must be negative.
In the Fig. 7a we can see the desired object velocity $\dot{\tilde{\boldsymbol{x}}}$ calculated to make the tool reaching a desired position without considering the obstacle.
As the robots go on, we can see in Fig. 7c (t = $6s$) that the TPIK procedure of agent a generates a non-cooperative tool frame velocity $\dot{\boldsymbol{x}}_{t,a}$ to make it go around the obstacle. Instead, the non-cooperative tool frame velocity $\dot{\boldsymbol{x}}_{t,b}$ in Fig. 7d is similar to the desired one because the agent b does not meet any difficulties in its action. So, due to the fact



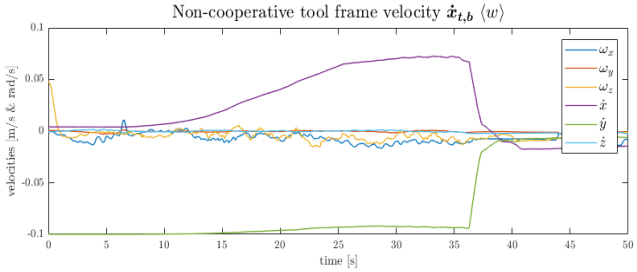
(a)



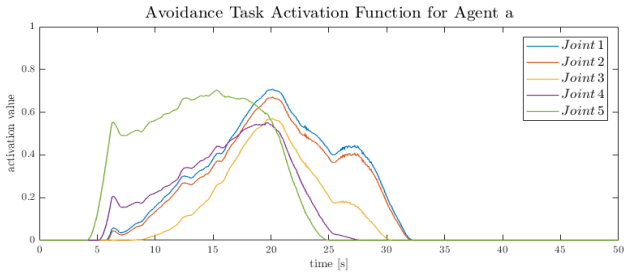
(b)



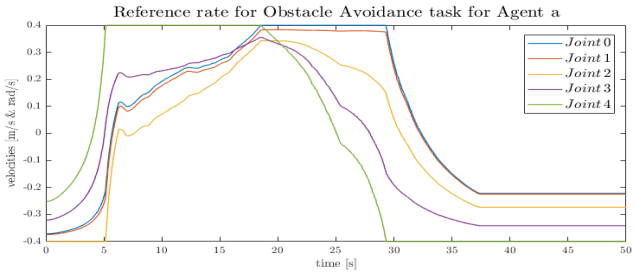
(c)



(d)



(e)



(f)

Figure 7: *Results:* (a) *the desired object velocity,* (b) *cooperative tool-frame velocities (tuned toward agent* a *,* (c) *non-cooperative tool-frame velocities for agent* a *,* (d) *non-cooperative tool-frame velocities for agent* b *,* (e) *avoidance task activation function for agent* a *,* (f) *reference rate for avoidance task for agent* a.

that the non-cooperative velocity of agent a is very different with respect to the desired one, the cooperative tool frame velocity $\dot{\hat{\boldsymbol{x}}}$ in Fig. 7b is tuned toward the agent a which is in more trouble tracking the desired velocity. The presence of the obstacle is shown by the fact that the avoidance task activation function is greater than zero, as depicted in Fig. 7e.
The Fig. 7f shows the agent a reference rate generated by the obstacle avoidance task.

6 Conclusions

We see that the TPIK methods proposed in [2] and in [3] are very efficient to implement the various tasks such as the obstacle avoidance task. The modular architecture developed in [4] makes this work simple and proficient as seen in the results 5.

A further work could consist in the obstacle recognition by the MoCap cameras or by robot sensors which must be added to the agents.

This work is part of a cooperation with another research group which has developed an alignment task. Being both part of a bigger project, the two assignments can be merged to continue the project of the robot cooperative transportation in difficult environments.

Reference

[1] E. Simetti, G. Casalino. *Manipulation and transportation with cooperative underwater vehicle manipulator systems.* IEEE Journal of Oceanic Engineering, 2016.

[2] E. Simetti, G. Casalino. *A novel practical technique to integrate in equality control objectives and task transitions in priority based control.* Journal of Intelligent & Robotic Systems, 2016.

[3] E. Simetti, G. Casalino, F.Wanderlingh. *A novel practical technique to integrate in equality control objectives and task transitions in priority based A Task Priority Approach to Cooperative Mobile Manipulation: Theory and Experiments.* Submitted for publication, 2018.

[4] F.Wanderlingh. *Cooperative Robotic Manipulation for the Smart Factory.* Università degli Studi di Genova, 2018.

[5] KUKA. *KUKA youBot User Manual.* Locomotec, 2012.