SAPHARI

SAFE AND AUTONOMOUS PHYSICAL HUMAN-AWARE ROBOT INTERACTION

Project funded by the European Community's 7th Framework Programme (FP7-ICT-2011-7)
Grant Agreement ICT–287513

Deliverable D3.3.1

Safe dynamic control laws for redundant robots

Deliverable due date: 31 October 2014	**Actual submission date:** 1 January 2015
Start date of project: 1 November 2011	**Duration:** 48 months
Lead beneficiary: UNIROMA1	**Revision:** Final

Nature: R	**Dissemination level:** PU

R = Report
P = Prototype
D = Demonstrator
O = Other
	PU = Public
	PP = Restricted to other programme participants (including the Commission Services)
	RE = Restricted to a group specified by the consortium (including the Commission Services)
	CO = Confidential, only for members of the consortium (including the Commission Services)

Executive Summary

This third deliverable of work package WP3 presents a summary of the most recent research results on safe control of redundant robots obtained by the partners of SAPHARI and developed during the first three years of the project. The presence of robot redundancy with respect to common tasks has been explored in two different ways: *i)* during free motion, for controlling the robot so as to avoid dynamic obstacles (including humans); *ii)* during physical contact, for improving the interaction modality in which safe and accurate control of human-robot exchanged forces is achieved.

The results briefly described in this document are in part individual evolutions of the methods already reported in milestone MS11 “Safe kinematic control laws for redundant robots” reached @M18, and in part they represent the integration of those methods. Activities within the specific task T3.3 of WP3 will continue as planned in the last year of the project, since free motion control results in the presence of humans are now mature for being directly applied to the use cases of WP8. On the other hand, the interaction control methods summarized here will be further developed, integrated, tested, and refined in task T3.4 (the next deliverable D3.4.1 “Intentional human-interaction control of compliant robots” is due @M42).

Table of Contents

1 Introduction

When a robot is employed near humans, the robot must be controlled so as to ensure a desired free motion, but also to physically interact in a safe way with the humans. Undesired collisions should be avoided, unexpected impacts safely handled, and intentional exchange of forces suitably controlled.

Indeed, the safest possible solution is avoiding any undesired contact with humans or environment obstacles. For kinematically redundant manipulators with fixed or mobile base in a shared human-robot workspace, suitable control algorithms allow avoiding collisions with humans or other obstacles, while still performing a task defined in the Cartesian space. At Fraunhofer IOSB, a Nonlinear Model Predictive Control (NMPC) approach has been developed (Sec. 3) to move the end-effector of the robot along a geometric reference path or trajectory to a desired pose (3D position and orientation), while considering obstacles. Due to the underlying general robot model considered, the control algorithm can be applied to both fixed and mobile manipulators, which is illustrated by means of simulations of a 7-DOF fixed manipulator and of a 10-DOF mobile manipulator.

UNIROMA1 has proposed a method that takes into account obstacles by using unilateral constraints within a multi-task redundancy resolution framework (Sec. 4). The introduction of unilateral constraints in the so-called Reverse Priority (RP) redundancy resolution method [15] was presented in [21]. The key feature of the RP method is that contributions of higher priority tasks are added after having computed the solutions for lower priority tasks. This allows checking more efficiently whether one or more unilateral constraints (which have typically the highest priority in the stack of tasks) need to be activated, or if the current solution computed so far is already within these hard inequality constraints. More in general, the method allows to insert tasks associated to unilateral constraints at any priority level, and guarantees the continuity of the joint velocity command during constraint activation/deactivation. To this end, a method that minimizes the variation of the joint velocity commands [16] is used.

The redundancy of a robot in task execution can also be exploited to ensure a compliant behaviour to the robot's body in the presence of physical interaction, so that the assigned interaction task is completed only if safety is ensured. To this aim, two different approaches have been pursued by UNINA. The first approach, presented in Sec. 5), does not use any kind of exteroceptive sensing. It guarantees the correct execution of the main task, while compliance of the robot's body during intentional or accidental interaction is introduced in the null space of the main task. The second approach, presented in Sec. 6, makes use of a new prototype of the flexible artificial skin developed within SAPHARI by the Second University of Naples (SUN). The skin is able to measure both the position of the contact point and the three components of an applied force. The application presented here deals with the control of human-robot interaction through multiple contacts.

UNIROMA1 has developed in parallel a method for estimating interaction forces (in direction and intensity) during dynamic contacts between a robot and a human, with no explicit force/torque sensing [11]. The method is based on the integrated use of model-based residual signals, see eq. (45), that detect the occurrence of a collision [20] with one or more external RGB-D sensors (e.g., Kinects) that approximately localize the contact point on the surface of the robot links. In Sec. 7, we illustrate further the use of the estimated contact forces for the design of two human-robot interaction controllers, which extend respectively the *impedance control* method [13] and the *direct force control* method [14] to the case of a generic contact location on the robot. For this reason, we denote this approach as *generalized contact* motion and force control.

2 Background

In general, a manipulator with fixed or mobile base can be modeled as a chain of rigid links connected by n revolute or prismatic joints. When the commands are set at the velocity level, the following kinematic model can be used

$$\dot{\boldsymbol{q}} = \boldsymbol{v} \tag{1}$$
$$\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{q}), \tag{2}$$

with the revolute or prismatic joint variables $\boldsymbol{q} \in \mathbb{R}^n$, the joint velocities $\boldsymbol{v} \in \mathbb{R}^n$ and the end-effector pose $\boldsymbol{x} \in \mathbb{R}^m$. The nonlinear function $\boldsymbol{f}(\boldsymbol{q})$ maps the joint configuration $\boldsymbol{q}$ into the Cartesian space $\mathbb{R}^m$ and is called the direct kinematics of the robot.

The direct kinematics $\boldsymbol{f}(\boldsymbol{q})$ is derived from the Denavit-Hartenberg convention. This procedure, originally developed for robot manipulators with fixed base, is simply extensible to mobile manipulators by adding two prismatic joints for the translational motion of the platform base and one revolute joint for the platform rotation. This is sufficient for holonomic (omnidirectional) platforms. In the case of a non-holonomic platform, the non-holonomic properties are described by additional constraints. Hence, a generic modeling framework is available that can be applied to both fixed manipulators and mobile manipulators.

At a given robot configuration $\boldsymbol{q}$, differential kinematics of the task is

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}}. \tag{3}$$

Inversion of the differential map (3) provides infinitely many solutions, all of which can be generated as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\#}\dot{\boldsymbol{x}} + \boldsymbol{P}\dot{\boldsymbol{q}}_{\mathcal{N}}, \tag{4}$$

where $\boldsymbol{J}^{\#}$ is the (unique) Moore-Penrose pseudo-inverse of the task Jacobian, and

$$\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{J}^{\#}\boldsymbol{J} \tag{5}$$

is the $n \times n$ orthogonal projector in the Jacobian null space, with $\boldsymbol{I}$ the $n \times n$ identity matrix, and $\dot{\boldsymbol{q}}_{\mathcal{N}} \in \mathbb{R}^n$ is a generic joint velocity.

The dynamic model of a n-link robot manipulator can be expressed by

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{\tau}_{ext} = \boldsymbol{\tau}, \tag{6}$$

with standard notation. In this equation, $\boldsymbol{\tau}_{ext} = -\boldsymbol{J}_c^T\boldsymbol{F}_c$ is the torque resulting from external interaction forces $\boldsymbol{F}_c$, $\boldsymbol{J}_c$ is the Jacobian associated to the contact point, and $\boldsymbol{\tau}$ is the input vector torque that can be computed according to the well-known model-based resolved acceleration control as

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}_c + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}), \tag{7}$$

where $\ddot{\boldsymbol{q}}_c$ is the joint command acceleration to be suitably designed.

In the framework of operational space control [4], it has been shown that using the inertia-weighted pseudoinverse guarantees consistency of the transformation of task forces in the robot dynamics. The inertia-weighted pseudoinverse of a Jacobian $\boldsymbol{J}$ is

$$\boldsymbol{J}^{\#}_{\boldsymbol{M}} = \boldsymbol{M}^{-1}\boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{M}^{-1}\boldsymbol{J}^T)^{-1}. \tag{8}$$

3 Cartesian Control of Redundant Robots in Presence of Obstacles

A Nonlinear Model Predictive Control approach has been followed for the control design of redundant robots in order to avoid Cartesian obstacles. This control approach fits both to the use of a kinematic robot model or of a dynamic robot model, with joint torques as actuating variables. However, since the OmniRob platform, which is the target robot in the WP8.2 use case, cannot be directly controlled by the user using torques and forces, a control design based on dynamic model of the robot is not applicable to the OmniRob platform. Therefore, the control approach is presented here based on the kinematic robot model.

3.1 Model Predictive Control

Nonlinear Model Predictive Control (NMPC) is well suited to exploit the robot redundancy of a given Cartesian task, in order to realize additionally desired robot behaviours like collision avoidance. The NMPC approach optimizes the system input and state sequences

$$\boldsymbol{V} = \{\boldsymbol{v}(0), \dots, \boldsymbol{v}(N_\mathsf{p}-1)\} \tag{9}$$
$$\boldsymbol{Q} = \{\boldsymbol{q}(1), \dots, \boldsymbol{q}(N_\mathsf{p})\} \tag{10}$$

for the next N_p time steps by minimizing a cost function containing the trajectory following error, based on the current measured joint positions $\boldsymbol{q}(0) = \boldsymbol{q}_0$. The optimized input $\boldsymbol{v}^*(0)$ is applied to the system. Then, the optimization is repeated using the next measurement.

For following the desired Cartesian trajectory $\boldsymbol{x}_\mathsf{t}(k) \in \mathbb{R}^m$, the cost function

$$\sum_{k=0}^{N_\mathsf{p}-1} F(\boldsymbol{v}(k), \boldsymbol{q}(k+1)) + E(\boldsymbol{q}(N_\mathsf{p})) \rightarrow \min_{\boldsymbol{V},\boldsymbol{Q}} \tag{11}$$

is minimized subject to the equality constraints

$$\boldsymbol{q}(k+1) - \boldsymbol{q}(k) - T_\mathsf{s}\boldsymbol{v}(k) = 0 \tag{12}$$

and the inequality constraints

$$-\boldsymbol{v}_\mathsf{max} \leq \boldsymbol{v}(k) \leq \boldsymbol{v}_\mathsf{max} \tag{13}$$
$$\boldsymbol{q}_\mathsf{min} \leq \boldsymbol{q}(k+1) \leq \boldsymbol{q}_\mathsf{max}. \tag{14}$$

for $k = 0, \dots, N_\mathsf{p} - 1$. The equality constraints (12) contain the system model (1). The inequality constraints (13)–(14) guarantee that the joint velocity and position limits are met.

The terms of the cost function are

$$F(\boldsymbol{v}, \boldsymbol{q}) = \boldsymbol{v}^T\boldsymbol{Q}_v\boldsymbol{v} + \boldsymbol{q}^T\boldsymbol{Q}_q\boldsymbol{q} + \boldsymbol{e}_\mathsf{t}^T\boldsymbol{Q}_e\boldsymbol{e}_\mathsf{t} \tag{15}$$
$$E(\boldsymbol{q}) = \boldsymbol{q}^T\boldsymbol{R}_q\boldsymbol{q} + \boldsymbol{e}_\mathsf{t}^T\boldsymbol{R}_e\boldsymbol{e}_\mathsf{t} \tag{16}$$

with the trajectory following error

$$\boldsymbol{e}_\mathsf{t}(k) = \boldsymbol{f}(\boldsymbol{q}(k)) - \boldsymbol{x}_\mathsf{t}(k). \tag{17}$$

$\boldsymbol{Q}_v$, $\boldsymbol{Q}_q$, $\boldsymbol{Q}_e$, $\boldsymbol{R}_q$ and $\boldsymbol{R}_e$ are positive semi-definite diagonal matrices. With $F(\boldsymbol{v}, \boldsymbol{q})$, high joint velocities, deviations of the joint positions from the home position and the trajectory following error are penalized. Penalizing the trajectory following error is needed to fulfill the trajectory following task. Penalizing the joint velocities avoids unnecessary high input energy and oscillations. Avoiding large joint position values keeps the joints away from the joint limits and results in a more natural motion. The term $E(\boldsymbol{q})$ is only applied to the last prediction step and is added to improve stability and accuracy.

During task execution, the robot redundancy can be exploited to avoid collisions with obstacles. Therefore, in WP3.1 the above presented cost function is extended by a term that penalizes the squared distances between the robot and the obstacle points,

$$\sum_{k=0}^{N_\mathsf{p}-1} F_\mathsf{O}(\boldsymbol{q}(k+1)) + E_\mathsf{O}(\boldsymbol{q}(N_\mathsf{p})). \tag{18}$$

in order to additionally increase the distance between the robot and the obstacles. For smooth distance computations, distances between test points on the robot $\boldsymbol{p}_{\mathsf{R},i}(\boldsymbol{q}) \in \mathbb{R}^3$ ($i = 1, \ldots, n_{\mathsf{R}}$) and obstacle points $\boldsymbol{p}_{\mathsf{O},j} \in \mathbb{R}^3$ ($j = 1, \ldots, n_{\mathsf{O}}$) are considered,

$$d_{i,j}^2(\boldsymbol{q}) = \left\|\boldsymbol{p}_{\mathsf{R},i}(\boldsymbol{q}) - \boldsymbol{p}_{O,j}\right\|^2 . \tag{19}$$

For F_{O} and E_{O} different functions are possible. In the present case, the terms are chosen to be

$$F_{\mathsf{O}}(\boldsymbol{q}) = \sum_{i=1}^{n_{\mathsf{R}}} \sum_{j=1}^{n_{\mathsf{O}}} \frac{c_i}{d_{i,j}^2(\boldsymbol{q}) - d_{i,\mathsf{min}}^2} \tag{20}$$

$$E_{\mathsf{O}}(\boldsymbol{q}) = r_{\mathsf{O}} F_{\mathsf{O}}(\boldsymbol{q}) \tag{21}$$

If $d_{i,j}^2 < d_{i,\mathsf{min}}^2$, the term F_{O} is set to infinity.

Considering the quadratic distances between the robot and obstacles in the objective function results in pushing the robot away from obstacles while at the same time the robot end-effector is controlled to follow the reference pose as close as possible. Since at this stage there is still no guarantee that the optimized robot configurations will not be in collision with an obstacle, the following additional constraints are introduced:

$$d_{i,j}^2(\boldsymbol{q}(k)) - d_{i,\mathsf{min}}^2 \geq 0 \tag{22}$$

for $k = 1, \ldots, N_{\mathsf{p}}$, $i = 1, \ldots, n_{\mathsf{R}}$ and $j = 1, \ldots, n_{\mathsf{O}}$. Equation (22) forces the distance between the robot and the obstacle points to be always greater or equal than a specified safety distance $d_{i,\mathsf{min}}$.

Using only the obstacle constraints (22) without the cost function terms (18) is also possible. But then, the robot may move too close to the obstacles, keeping only the requested minimum distance, even if motions at larger distances from the obstacles are possible or more convenient. In any event, especially in the case of dynamic obstacles, it is desirable to increase the distance between the robot and the obstacles as far as the task execution is not impeded.

3.2 Results

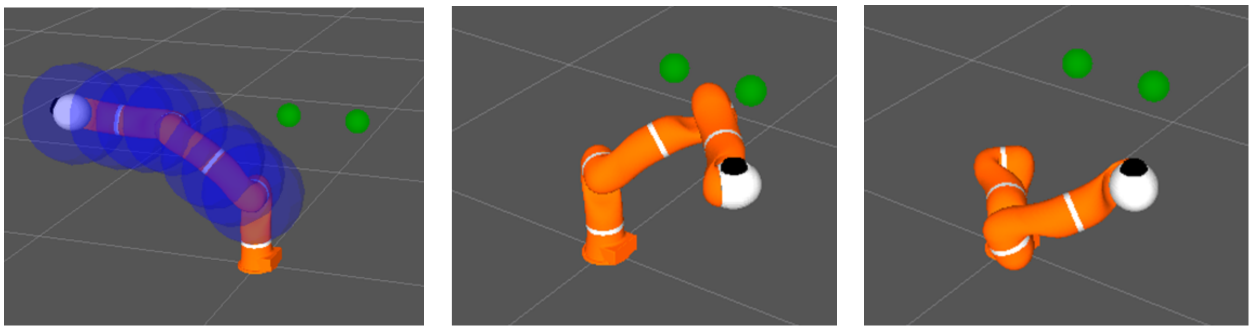
In the first example, the presented algorithm is applied to a KUKA LWR4 robot with seven revolute joints. The control interface of the real robot (KUKA Sunrise) as well as of the simulated robot delivers the measured joint angles and accepts joint velocity commands.

For the consideration of obstacles, eight test points on the robot are chosen with a minimum distance to obstacle points of 0.2 m. These robot points are visualized in Fig. 1(a) as blue spheres with radii according to the minimum distances.

The robot manipulator is controlled to move from a start configuration to a reference pose by using the joint velocities as actuating variables. A sample time of $T_{\mathsf{s}} = 0.1$ s and $N_{\mathsf{p}} = 10$ prediction steps are used, which leads to a prediction horizon of 1 s. In Fig. 1(a) the start configuration of the example motion is shown. Without considering the presence of obstacles. the manipulator successfully reaches the reference pose with the joint configuration shown in Fig. 1(b).

If the two obstacle points visualized in Fig. 1 as green spheres are considered in the control algorithm, the manipulator also reaches the deired end-effector pose but with a robot configuration that results in a larger distance to the obstacle points (see Fig. 1(c)).

In the second example, the developed method is applied to an omni-directional mobile platform equipped with a 7-DOF manipulator (KUKA OmniRob with LWR4). The mobile manipulator has 10-DOF and is highly redundant with regard to Cartesian reference poses. The robot is modeled according to the Denavit-Hartenberg convention as a chain of revolute and prismatic joints. Two prismatic joints describe the platform translation



(a) Start pose (b) Final pose without consideration of obstacles (c) Final pose considering obstacles

Figure 1: Example of a manipulator motion to a reference pose with and without consideration of obstacles (green spheres); the blue spheres represent the robot test points with minimum distances as radii.

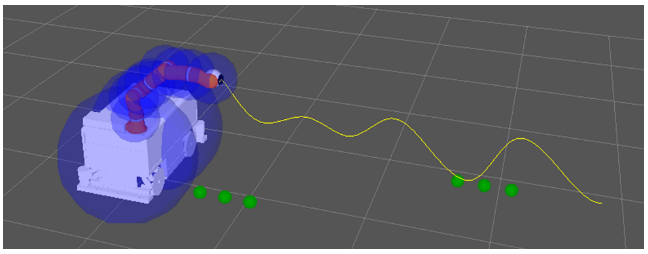


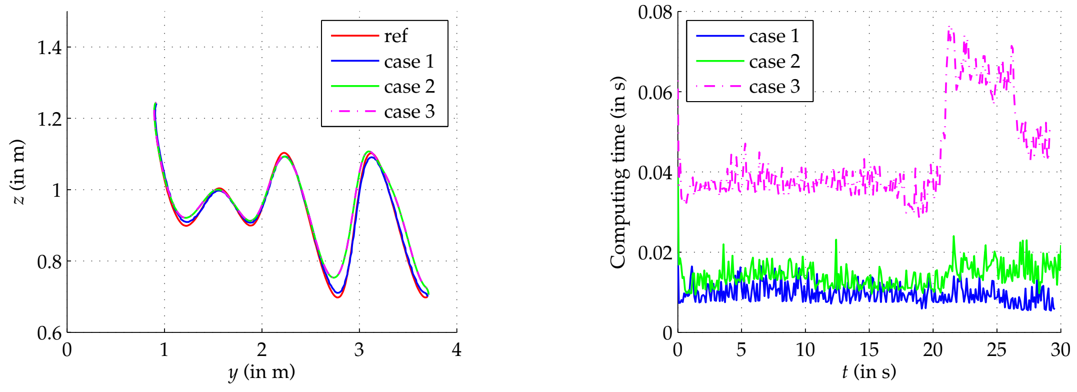
Figure 2: Mobile manipulator in start configuration with reference trajectory (yellow) and obstacles (green). The blue spheres represent the robot test points with minimum distances as radii.

in x- and y-direction with respect to a global frame. A revolute joint models the platform rotation about the vertical robot z-axis.

For collision avoidance, ten test points are chosen on the mobile manipulator. The minimum distance for the two points describing the platform is set to 0.6 m and the points on the manipulator arm have a minimum distance of 0.2 m. In Fig. 2, the corresponding volume approximation is visualized by blue spheres. The mobile manipulator is controlled to follow the end-effector pose trajectory shown as yellow line in Fig. 2. Again, a sample time of $T_\mathrm{s} = 0.1$ s and $N_\mathrm{p} = 10$ prediction steps are used. The control parameters are chosen in order to achieve a good collision avoidance behaviour. The trajectory following accuracy is of less importance.

The resulting end-effector position in the y-z-plane can be seen in Fig. 3(a). In case 1 (blue line), trajectory control is executed without obstacles. The end-effector follows the reference trajectory well. The computing time for one control step is below 20 ms except for the first optimization, when no appropriate start values are available (see Fig. 3(b)). In case 2 (green line), the six obstacles visualized as green spheres in Fig. 2 are considered by the objective function, while in case 3 (magenta line) these are considered by the objective function and in the constraints. In both cases the deviation of the resulting end-effector pose from the reference trajectory increases compared to case 1 (see Fig. 3(a)), especially near the obstacles for $y \in [2.5\,\mathrm{m}, 3.2\,\mathrm{m}]$. Nonetheless, the mobile manipulator successfully moves through the obstacles without any collision and reaches the end of the trajectory with acceptable deviations. The computing time in case 2 increases only slightly in comparison to case 1. In case 3, the computing time is significantly higher due to the large number of constraints to be considered. Still, even when moving closer to the obstacles ($t > 20$ s), the computing time remains acceptable with values below 80 ms.

The detailed algorithm and the simulation results have been submitted to ICIT 2015 [1].



(a) End-effector position (b) Computing time for one control step

Figure 3: Example of a mobile manipulator executing a trajectory following task without obstacles (case 1), with obstacles considered only in the objective function (case 2), and with obstacles considered in the objective function and in the constraints (case 3).

4 Redundancy Resolution with Unilateral Constraints

A task is usually described as a desired reference trajectory for some geometrical components of the robot (the end-effector pose, the Cartesian position of a point on the robot, a joint angle, etc.). Namely, at each instant a desired value for these components is specified (*bilateral* constraints). Even if very effective, this task description lacks sometimes of expressivity. In many applications, it is not necessary to bring such robot components to some specific values, but it is more convenient just to specify regions to which these components should belong. This requirement can be coded with *unilateral* constraints. There are many situations where the use of unilateral constraints is preferable or mandatory for a safe and reliable robot behaviour: for example, when considering hard limits for the joint ranges, or limited robot motion capabilities due maximum joint velocities or accelerations. Unilateral constraint can be associated to the presence of obstacles that have to be avoided, to simulate virtual fixtures, or to accomplish particular tasks, such as keeping visual features in the FOV of a camera, or the projection of the center of mass of a humanoid within the support polygon.

4.1 Redundancy resolution with the Reverse Priority method

The redundancy resolution scheme (4) can be extended to the case of l multiple tasks at time $t = t_k = kT_c$

$$\dot{\boldsymbol{x}}_k^{\{p\}} = \boldsymbol{J}_k^{\{p\}} \dot{\boldsymbol{q}}_k \qquad p = 1, \dots, l, \tag{23}$$

each of dimension m_p (typically, with $\sum_{p=1}^{l} m_p \leq n$), that are ordered by their priority, i.e., task i has higher priority than task j if $i < j$. The execution of a task of lower priority should not interfere with the execution of tasks having higher priority, and this hierarchy is obtained by projecting the p-th task in the null space of all higher priority tasks.

In [15], redundancy is used for addressing the execution of multiple tasks with priority based on the idea of computing first the solution for the lowest priority task, and then adding iteratively the contributions of higher priority tasks. For this reason, the method has been called Reverse Priority (RP). Preservation of the correct hierarchy of task execution is obtained if the portion of lower level tasks that is linearly independent from a higher priority task is kept undeformed once the contribution for the execution of the higher priority task is added. On the other hand, the higher priority task will dominate the portion which is in conflict. Such a result is obtained with the RP recursive equation

$$\dot{\boldsymbol{q}}_k^{\{p\}} = \dot{\boldsymbol{q}}_k^{\{p+1\}} + \boldsymbol{T}_k^{\{p\}} \left(\boldsymbol{J}_k^{\{p\}} \boldsymbol{T}_k^{\{p\}} \right)^{\#} \left(\dot{\boldsymbol{x}}_k^{\{p\}} - \boldsymbol{J}_k^{\{p\}} \dot{\boldsymbol{q}}_k^{\{p+1\}} \right), \tag{24}$$

where $\dot{\boldsymbol{q}}_k^{\{p\}}$ is the solution that takes into account all *low* priority tasks up to p. The $n \times m_p$ matrix $\boldsymbol{T}_k^{\{p\}}$ is related to augmentation by $\boldsymbol{J}_k^{\{p\}}$ of the *reverse* augmented Jacobian $\boldsymbol{J}_{RA,k}^{\{p+1\}}$

$$\boldsymbol{J}_{RA,k}^{\{p\}} = \begin{pmatrix} \boldsymbol{J}_k^{\{p\}^T} & \boldsymbol{J}_k^{\{p+1\}^T} & \dots & \boldsymbol{J}_k^{\{l\}^T} \end{pmatrix}^T = \begin{pmatrix} \boldsymbol{J}_k^{\{p\}} \\ \boldsymbol{J}_{RA,k}^{\{p+1\}} \end{pmatrix} \tag{25}$$

as

$$\boldsymbol{J}_{RA,k}^{\{p\}\#} = \begin{pmatrix} \boldsymbol{T}_k^{\{p\}} & \dots \end{pmatrix}. \tag{26}$$

It is important to note that only the indicated submatrix $\boldsymbol{T}_k^{\{p\}}$ is needed, which can be efficiently obtained as update of $\boldsymbol{J}_{RA,k}^{\{p+1\}\#}$ [17]. For more details please refer to [15].

In this framework, it is possible to introduce an auxiliary joint velocity command $\dot{\boldsymbol{q}}_{\mathbb{N},k}$ that has to be projected in the null space of all tasks, by simply setting $\dot{\boldsymbol{q}}_k^{\{l+1\}} = \dot{\boldsymbol{q}}_{\mathbb{N},k}$. This auxiliary joint velocity can be used, e.g., to implement a projected gradient task [18], or to obtain the minimum variation of the joint velocity command when incrementing time from t_{k-1} to t_k, by setting

$$\dot{\boldsymbol{q}}_k^{\{l+1\}} = \lambda \dot{\boldsymbol{q}}_{k-1}^{\{1\}} \tag{27}$$

for some positive scalar $\lambda \leq 1$ (typically, equal or close to 1).

4.2 Unilateral constraints

A unilateral constraint can be represented in the form of an upper bound

$$x_C \leq \bar{x}_C, \tag{28}$$

where $x_C = x_C(\boldsymbol{q})$ is some geometrical component of the robot expressed as a function of its configuration $\boldsymbol{q}$. If $\dot{x}_{C,d}$ is the nominal velocity of the chosen robot component that results from using redundancy to impose all other tasks, the additional presence of the unilateral constraint (28) is transformed into the differential kinematic map

$$\dot{x}_C = \begin{cases} \text{free to move}, & \text{if } x_C < \bar{x}_C \text{ or } \dot{x}_{C,d} < 0, \\ 0, & \text{if } x_C \geq \bar{x}_C \text{ and } \dot{x}_{C,d} \geq 0. \end{cases} \tag{29}$$

In words, the feature x_C is not allowed to have a velocity that will bring to exceeding the bound. Then, the unilateral constraint can be represented as a equality task with Jacobian $\boldsymbol{J}_C(\boldsymbol{q}) = \frac{\partial x_C(\boldsymbol{q})}{\partial \boldsymbol{q}}$ and desired velocity $\dot{x}_C = 0$ that needs to be activated only when $x_C \geq \bar{x}_C$ and $\dot{x}_{C,d} \geq 0$.

Moreover, in many applications and especially when multiple tasks should be executed, it is important to distinguish between *soft* and *hard* constraints. A unilateral constraint is handled as soft with respect to a task, if the constraint will be fulfilled only when it is not in contrast with the task. It is considered as hard when it should not be relaxed in any case. Therefore, any task in contrast with a hard constraint will be deformed.

One of the main features of the RP framework with respect to standard task priority schemes is the possibility to insert a high priority task after the evaluation of the solution of all lower priority tasks. Assume that the task associated to the unilateral constraint is inserted as the i-th task in the hierarchy, namely it is a soft constraint for all other tasks j with $j < i$ and it is a hard constraint for all other tasks with $j > i$. The nominal velocity for the constrained robot component up to task $i+1$ can be obtained as

$$\dot{x}_{C,d} = \boldsymbol{J}_C \dot{\boldsymbol{q}}_k^{\{i+1\}}. \qquad (30)$$

This velocity can be used to check whether the constraining task has to be activated or not, according to the rule (29). Introducing anactivation variable h_C, we will have $h_C = 1$ when the constraint is active and $h_C = 0$ when it is inactive, or

$$h_C = \begin{cases} 0 & \text{if } x_C < \bar{x}_C \text{ or } \dot{x}_{C,d} < 0, \\ 1 & \text{if } x_C \geq \bar{x}_C \text{ and } \dot{x}_{C,d} \geq 0. \end{cases} \qquad (31)$$

Thus, the unilateral constraint will be added to the RP method simply as

$$\dot{\boldsymbol{q}}_k^{\{i\}} = \dot{\boldsymbol{q}}_k^{\{i+1\}} - h_C \boldsymbol{T}_k^{\{i\}} \left(\boldsymbol{J}_{C,k} \boldsymbol{T}_k^{\{i\}} \right)^{\#} \boldsymbol{J}_{C,k} \dot{\boldsymbol{q}}_k^{\{i+1\}}, \qquad (32)$$

obtained from (24) by setting $\boldsymbol{J}_k^{\{i\}} = \boldsymbol{J}_{C,k}$ and $\dot{\boldsymbol{x}}_k^{\{i\}} = \dot{\boldsymbol{x}}_C = 0$.

To obtain a continuous insertion/removal of a constraint, the task activation variable h_C should be varied smoothly. To this end, the distance to the constraint and the velocity toward the constraint (given by the solution computed so far) can be used to smoothly change the activation variable. For more details please refer to [21].

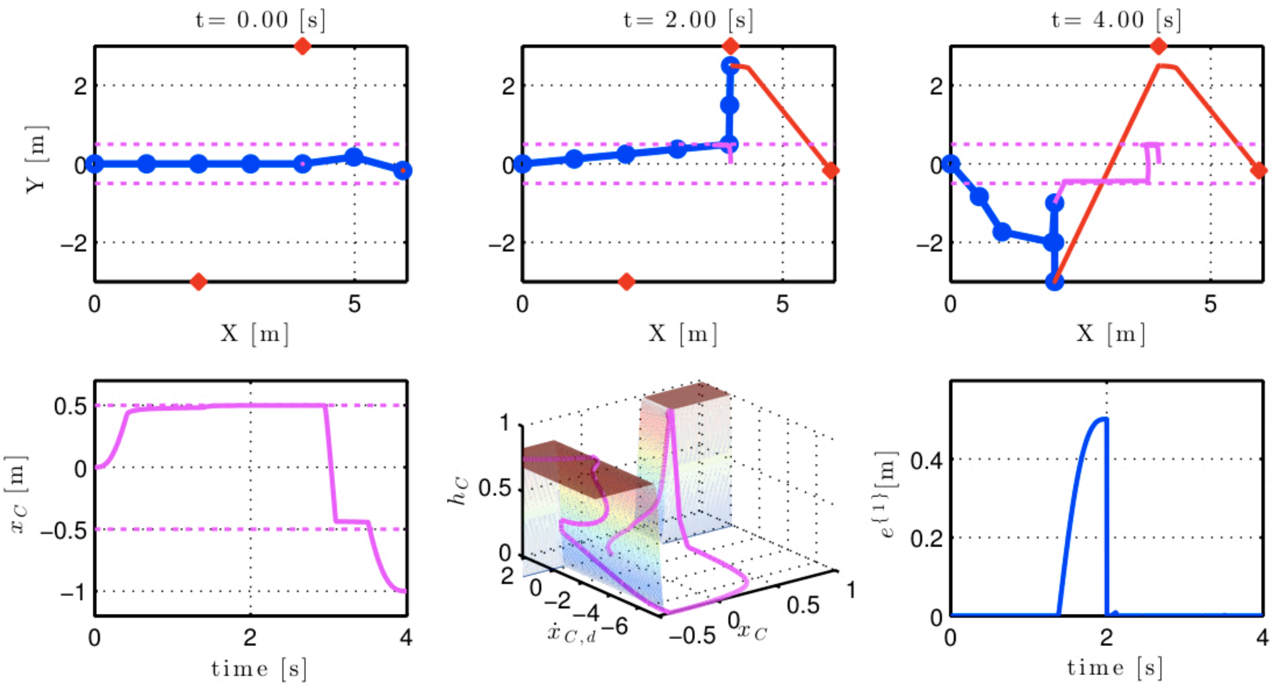


Figure 4: [Top] Snapshots of the resulting motion, with the robot represented in blue, the executed end-effector trajectory in red, and the trajectory of the robot geometric component subject to unilateral constraints in magenta. [Bottom] Time evolution of the constrained component (left), of the activation value (middle), and of the norm of the primary task error (right).

4.3 Results

To show the effectiveness of the proposed method, we present a simulation performed in Matlab. We have considered a planar robot with six revolute joints and links of unitary length. The primary (and single) task requires the robot end-effector position $\boldsymbol{x}^{\{1\}}$ to move between three Cartesian points connected by linear paths, starting from $\boldsymbol{q}_0 = (0, 0, 0, 0, 10, -30)$ [deg]. The two unilateral constraints are an upper bound $\bar{x}_C = 0.5$ [m] and a lower bound $\underline{x}_C = -0.5$ [m] for the y-component of the position of the fifth joint (= the end of the fourth link). The upper bound $\bar{x}_C$ is considered as a hard constraint for the primary task, while the lower bound $\underline{x}_C$ is a soft constraint for the primary task.

From the starting feasible configuration, the primary task requires the end-effector to move first to point $[4, 3]$, and then to point $[2, -3]$. Both these points cannot be reached in the presence of the two unilateral constraints on the y-coordinate of the fifth joint. The results in Fig. 4. show clearly the hard and soft nature of the two unilateral constraints. The upper bound (hard constraint) is never exceeded, and thus the first desired point will not be reached; on the other hand, the second desired point is reached with a relaxation of the lower bound (soft constraint).

5 Task Space Control of Manipulators with Null Space Compliance

In this section we consider the problem of controlling a robot manipulator in the task space while ensuring a compliant behaviour for the redundant degrees of freedom in the joint space. An example of application scenario is depicted in Fig. 5, where a robot working on a table experiences a contact with a human. This contact may produce errors on the main task of the robot if active compliance is used to achieve a safe interaction. Our goal is to minimise the error of the main task and, at the same time, to ensure safe interaction through active compliance in the null space of the main task.

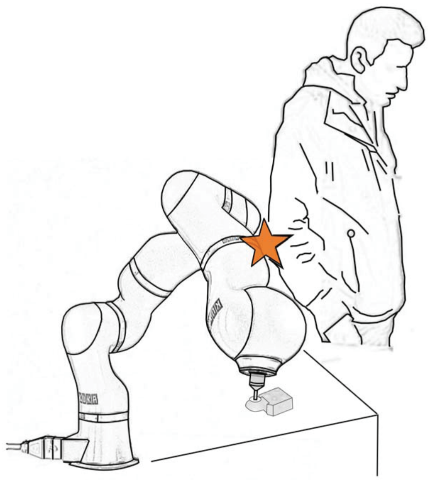


Figure 5: Robot working close to a human

To this purpose, two control schemes which do not require direct joint torque measurements are briefly presented. More details can be found in [2]. The first scheme is based on a disturbance observer which estimates the external forces acting on the main task variables on the basis of the task space error. The second scheme relies on the momentum-based observer [19]. In both cases, the overall stability of the system, with asymptotic convergence of the main task and a desired impedance behaviour in the null space of the main task, is proven through a rigorous analysis. A number of experiments are presented for a 7-DOF KUKA LWR4 robot.

5.1 Null space impedance

For a redundant manipulator, redundancy lets us to have some kind of joint impedance and task space control simultaneously. The so-called null-space impedance can be realized in the null space of the main task to control the interaction on the robot's body. The corresponding command joint acceleration in (7) is given by

$$\ddot{\boldsymbol{q}}_c = \boldsymbol{J}^{\dagger}(\ddot{\boldsymbol{x}}_c - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}) + \boldsymbol{N}(\ddot{\boldsymbol{q}}_d + \boldsymbol{M}_d^{-1}(\boldsymbol{K}_d\dot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_p\tilde{\boldsymbol{q}})), \tag{33}$$

which produces the task space and null-space closed-loop behaviour respectively as follows

$$\ddot{\boldsymbol{x}}_c - \ddot{\boldsymbol{x}} = \boldsymbol{J}\boldsymbol{M}^{-1}\boldsymbol{\tau}_{ext}, \tag{34}$$
$$\boldsymbol{N}(\ddot{\tilde{\boldsymbol{q}}} + \boldsymbol{M}_d^{-1}(\boldsymbol{K}_d\dot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_p\tilde{\boldsymbol{q}}) - \boldsymbol{M}^{-1}\boldsymbol{\tau}_{ext}) = \boldsymbol{0}. \tag{35}$$

Here $\ddot{\boldsymbol{x}}_c$ is a m-vector representing the command acceleration in the task space of dimension $m < n$, $\tilde{\boldsymbol{q}} = \boldsymbol{q}_d - \boldsymbol{q}$ where $\boldsymbol{q}_d$ is the desired trajectory or a rest configuration in the joint space, $\boldsymbol{J}^{\dagger}$ is any (weighted) right pseudo-inverse of the task space Jacobian matrix $\boldsymbol{J}$, $\boldsymbol{N} = (\boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{J})$ is the null-space projection matrix, and $\boldsymbol{M}_d > 0$ the desired inertia matrix, $\boldsymbol{K}_d > 0$ the desired damping matrix, $\boldsymbol{K}_p > 0$ the desired stiffness matrix (very often, these matrices are taken as diagonal).

The above choice of $\ddot{\boldsymbol{q}}_c$ allows the joint space impedance in the null-space of the main task to be realized, provided that the desired inertia matrix is chosen as $\boldsymbol{M}_d = \boldsymbol{M}(\boldsymbol{q})$. Moreover, if the external interaction happens only at the end effector, it can be shown that the null space closed loop dynamics is not affected by these forces only if the dynamically consistent generalised inverse $\boldsymbol{J}_M^{\#}$ is used [4]. Therefore, in the following, we assume that $\boldsymbol{M}_d = \boldsymbol{M}$ and $\boldsymbol{J}^{\dagger} = \boldsymbol{J}_M^{\#}$.

In order to impose a desired task to the end effector together with a impedance behaviour only in the null space of the desired task, two issues must be taken into account.

The first problem is related to the fact that eq. (35) represents the impedance behaviour projected in the null space, with dimension $r = n - m$, through n equations that, therefore, are not all independent. A solution is that of considering a $(n \times r)$ matrix $\boldsymbol{Z}(\boldsymbol{q})$, such that $\boldsymbol{J}\boldsymbol{Z} = \boldsymbol{O}$, and introducing a $(r \times 1)$ velocity vector $\boldsymbol{\nu}$, such that

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\#}\dot{\boldsymbol{x}} + \boldsymbol{Z}\boldsymbol{\nu}. \tag{36}$$

Using these variables, it is possible to rewrite the closed loop dynamics in terms of the $n = m + r$ equations

$$\ddot{\boldsymbol{x}}_c - \ddot{\boldsymbol{x}} = \boldsymbol{J}\boldsymbol{M}^{-1}\boldsymbol{\tau}_{ext}, \tag{37}$$
$$\dot{\boldsymbol{\nu}} = \dot{\boldsymbol{\nu}}_c - \boldsymbol{Z}^{\#}\boldsymbol{M}^{-1}\boldsymbol{\tau}_{ext}, \tag{38}$$

representing the task space and null space dynamics, respectively. The null space velocity vector $\boldsymbol{\nu}$ is, in general, non-integrable [5], and thus a null space position error cannot be easily defined. However, a projected joint space error can be used to define the null space command acceleration

$$\dot{\boldsymbol{\nu}}_c = -\boldsymbol{\Lambda}_\nu^{-1}((\boldsymbol{\mu}_\nu + \boldsymbol{B}_\nu)\boldsymbol{\nu} - \boldsymbol{Z}^T\boldsymbol{K}_p\tilde{\boldsymbol{q}}), \tag{39}$$

with $\boldsymbol{K}_p$, $\boldsymbol{B}_\nu$ symmetric and positive definite matrix, $\tilde{\boldsymbol{\nu}} = \boldsymbol{\nu}_d - \boldsymbol{\nu}$. The configuration dependent quantities $\boldsymbol{\Lambda}_\nu = \boldsymbol{Z}^T\boldsymbol{M}\boldsymbol{Z}$ and $\boldsymbol{\mu}_\nu$ are respectively the inertia matrix and the Coriolis/centrifugal matrix in the null space. The corresponding closed loop equation is

$$-\boldsymbol{\Lambda}_\nu\dot{\boldsymbol{\nu}} - (\boldsymbol{\mu}_\nu + \boldsymbol{B}_\nu)\boldsymbol{\nu} + \boldsymbol{Z}^T\boldsymbol{K}_p\tilde{\boldsymbol{q}} = \boldsymbol{Z}^T\boldsymbol{\tau}_{ext}, \tag{40}$$

where $\boldsymbol{Z}^T\boldsymbol{\tau}_{ext}$ is the projection of the external torque on the null space. Equation (40) can be interpreted as an impedance equation defined in the null space, with inertia $\boldsymbol{\Lambda}_\nu$, damping $\boldsymbol{B}_\nu$ and projected elastic torque $\boldsymbol{Z}^T\boldsymbol{K}_p\tilde{\boldsymbol{q}}$.

The second problem to solve is that the main task, represented by (38), may be affected by errors as a result of the external torques that are applied on the robot's body. These errors, as long as the external torques remain below a safety threshold, can be reduced by estimating and compensating the external torques acting on the task variables. To this aim, two different methods, based on suitable disturbance observers, are proposed in the following.

5.2 Task error based disturbance observer

The following proposition holds:

Proposition 1: Let us denote with $\widehat{\boldsymbol{\tau}}$ the estimated external torque and with $\tilde{\boldsymbol{\tau}} = \boldsymbol{\tau}_{ext} - \widehat{\boldsymbol{\tau}}$ the estimation error. Under the assumption of constant (or slowly time-varying) unknown external torque, the task space command acceleration

$$\ddot{\boldsymbol{x}}_c = -\boldsymbol{P}\dot{\boldsymbol{x}} + \boldsymbol{\Lambda}_x^{-1}((\boldsymbol{\mu}_x + \boldsymbol{K})\boldsymbol{s} + \boldsymbol{J}_M^{\#T}\widehat{\boldsymbol{\tau}}) \tag{41}$$

with the disturbance observer

$$\dot{\widehat{\boldsymbol{\tau}}} = -\boldsymbol{\Gamma}_f^{-1}\boldsymbol{J}^{\#}\boldsymbol{s}, \tag{42}$$

together with the null space command acceleration (39) guarantees that $\tilde{\boldsymbol{x}}$, $\dot{\boldsymbol{x}}$ and $\boldsymbol{\nu}$ go to zero asymptotically while a compliant behaviour is imposed in the null space of the main task. Moreover, $\widehat{\boldsymbol{\tau}}$ remains bounded and the closed-loop system is stable. In the above controller, $\boldsymbol{s} = -\dot{\boldsymbol{x}} + \boldsymbol{P}\tilde{\boldsymbol{x}}$, $\boldsymbol{K}_p$ is a diagonal and positive definite matrix, $\boldsymbol{B}_\nu$ and $\boldsymbol{\Gamma}_f$ are positive definite constant matrices.

In the above equations, the quantities $\boldsymbol{\Lambda}_x$ and $\boldsymbol{\mu}_x$ are the inertia matrix and the Coriolis/centrifugal matrix in task space, respectively. The controller-observer law of Proposition 1 can be modified according to Proposition 2 where the command acceleration is based on the PD+ controller written in the task space:

Proposition 2: The task space command acceleration and disturbance observer in Proposition 1 can be replaced by

$$\ddot{\boldsymbol{x}}_c = \boldsymbol{\Lambda}_x^{-1}(-(\boldsymbol{\mu}_x + \boldsymbol{D})\dot{\boldsymbol{x}} + \boldsymbol{K}\tilde{\boldsymbol{x}} + \boldsymbol{J}_M^{\#T}\widehat{\boldsymbol{\tau}}), \tag{43}$$

and disturbance observer

$$\dot{\widehat{\boldsymbol{\tau}}} = -\boldsymbol{\Gamma}_f^{-1}\boldsymbol{J}^{\#}(-\dot{\boldsymbol{x}} + \gamma\boldsymbol{f}(\tilde{\boldsymbol{x}})), \qquad \boldsymbol{f}(\tilde{\boldsymbol{x}}) = \frac{1}{1 + ||\tilde{\boldsymbol{x}}||}\tilde{\boldsymbol{x}}, \tag{44}$$

where γ is a properly chosen constant positive gain. The stability of whole system, the convergence of $\tilde{\boldsymbol{x}}$, $\dot{\boldsymbol{x}}$ and $\boldsymbol{\nu}$ to zero, and the compliant behaviour in the null space of the main task are preserved.

The proofs of Proposition 1 and 2 are based on the concept of conditional stability and can be found in [2].

5.3 Momentum based observer

An alternative method relies on the momentum based observer introduced in [19]. The basic concepts are the generalized momentum $\boldsymbol{p}(t) = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}$ and the n-dimensional residual vector $\boldsymbol{r}$ defined as

$$\boldsymbol{r}(t) = \boldsymbol{K}_I\left[\boldsymbol{p}(t) - \int_0^t (\boldsymbol{\tau} + \boldsymbol{C}^T(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{r}(t))dt\right], \tag{45}$$

with $\boldsymbol{r}(0) = \mathbf{0}$, $\boldsymbol{K}_I$ a diagonal positive matrix and $\boldsymbol{p}(0) = \mathbf{0}$. These quantities can be computed using the measured signals $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$, and the commanded torque $\boldsymbol{\tau}$. It can be shown that the dynamics of $\boldsymbol{r}$ is

$$\dot{\boldsymbol{r}} = -\boldsymbol{K}_I\boldsymbol{r} - \boldsymbol{K}_I\boldsymbol{\tau}_{ext},$$

thus the residual vector is a filtered version of the real external torque, i.e. $\boldsymbol{r}(t) \approx -\boldsymbol{\tau}_{ext}$.

The idea is to use the residual vector as an estimate of the external torque in the task space control law.

Proposition 3: In the presence of constant (or slowly time-varying) unknown external torque, for positive definite matrices $\boldsymbol{K}_\nu$ and $\boldsymbol{K}_p$, the task space command acceleration

$$\ddot{\boldsymbol{x}}_c = -\boldsymbol{K}_\nu \dot{\boldsymbol{x}} + \boldsymbol{K}_p \tilde{\boldsymbol{x}} - \boldsymbol{\Lambda}_x^{-1} \boldsymbol{J}_M^{\#T} \boldsymbol{r}, \tag{46}$$

together with null space command acceleration (39) and residual dynamics (45), guarantees that $\tilde{\boldsymbol{x}}, \dot{\boldsymbol{x}}$, $\boldsymbol{\nu}$ and estimation error $\tilde{\boldsymbol{r}} = \boldsymbol{r} + \boldsymbol{\tau}_{ext}$, go to zero asymptotically while a desired compliant behaviour is imposed in the null space of the main task.

The proof of Proposition 3 can be found in [2].

5.4 Results

The proposed approaches have been verified experimentally on a 7-DOF KUKA LWR4 ($n = 7$). The control algorithms are executed through Fast Research Interface (FRI) library on a remote PC with Ubuntu operating system. The remote computer is connected to the KUKA Robot Controller (KRC) unit through UDP socket with sampling rate of 2 ms.

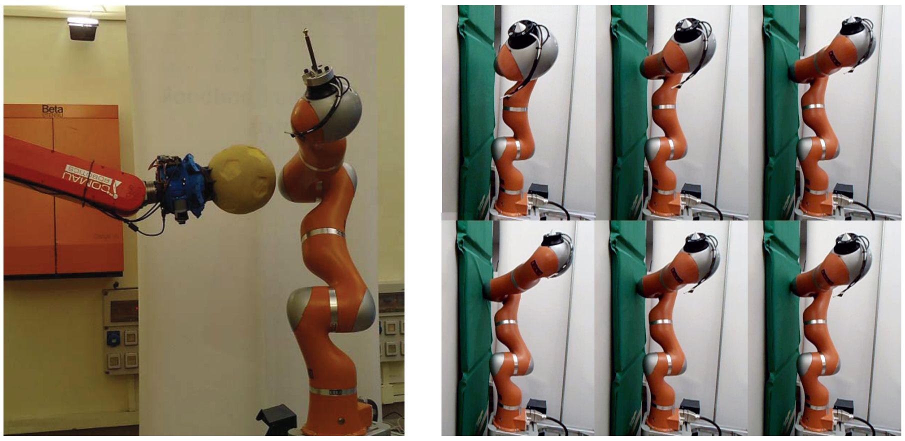


Figure 6: [Left] An elastic ball in interaction with the KUKA LWR4 arm during the first set of experiments. [Right] Snapshot of the KUKA LWR4 robot in contact with a wall during the second set of experiments.

The experiments have been performed for three cases: without observer, with task error based observer, and with momentum based observer. In all cases the position of the end effector is assumed as the main task, $m = 3$. Therefore, the robot has degree of redundancy $r = n - m = 4$.

In a first set of experiments, the interaction occurs with an elastic ball of 1200 N/m approximate stiffness at a point of the robot arm close to the fourth joint. While the end effector is commanded to be in the desired position, the sphere comes in contact with the robot, stops for 10 s, and finally is removed from robot contact. In order to have the same scenario in all the experiments and guarantee repeatability, the ball is moved by a position controlled industrial robot with constant speed of 4.5 cm/s along a straight line. A snapshot of the experimental setup is depicted in Fig. 6 (left).

In a second set of experiments, performance of the considered control schemes has been tested in a scenario in which the end effector of the robot follows a trajectory in the task space and the body of the robot experiences a contact with a vertical wall in a point close to the fourth joint, as shown in Fig. 6 (right). In this case, both task space control and safety during contact are required. To increase safety and protect the body of the robot from any damage during the experiment, the wall was covered with a soft pad.

These experiments have confirmed the theoretical findings. The experimental results are reported and commented in detail in [2].

6 Human-Robot Collaboration Using a Sensitive Skin

Starting from the solution described in [6], we briefly present next a flexible skin prototype that is able to estimate both the contact point(s) and the force vector(s). This sensor has been installed on a 7-DOF KUKA LWR4 manipulator and tested in a human-robot collaborative strategy for manual guidance with multi-point contact, based on admittance control and the multi-priority control formulation described in [8]. The choice of admittance control is motivated by the safety requirement to ensure a robot motion in the Cartesian space coherent with the direction of the forces applied by humans. More details can be found in the paper [9].

6.1 Sensitive skin

The working principle of the sensitive skin is based on the use of a PCB (Printed Circuit Board), constituted by optoelectronic couples (emitter/detector), to detect the local deformations generated by an external force applied to the deformable silicone cap that covers the optoelectronic layer.

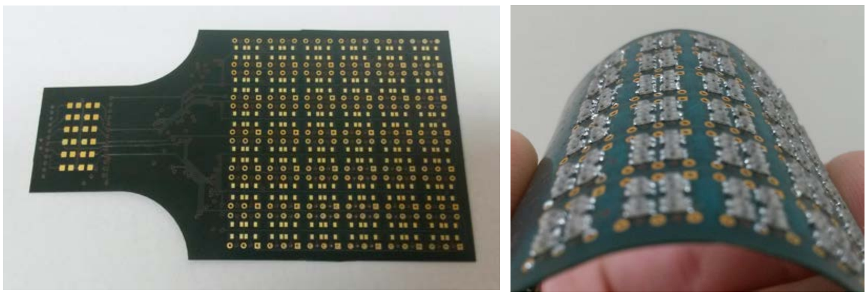


Figure 7: Pictures of the flexible PCB before (left) and after (right) the mounting of optoelectronic components.

Based on this idea, originally presented in [7], the distributed artificial skin detailed in [6] has been developed. It is constituted by a number of identical sensing modules, each capable of measuring the contact force vector acting on it. By exploiting some characteristics of the rigid version of the skin, as described in [6], the optoelectronic layer has been suitably re-designed in order to maximize flexibility of the new version of the skin. The new solution, after soldering of all the components, is able to keep a high flexibility (see Fig. 7), which allows the skin patch to be conformable to a surface with minimum curvature radius of $2-3$ cm, This is found sufficient for applications that require to cover robot surfaces such as arms, legs, or torso.

The flexible skin patch has to be conformed to the surface selected for the final assembly. To this purpose, a mechanical support, designed according to the shape of the surface selected for the final mounting, has been designed and built. Moreover, since the actual assembly of the sensor (e.g., components soldering, bonding of the silicone caps) could introduce differences in the response of the skin modules, a calibration procedure has been set up and applied to each sensing module.

6.2 Human-robot collaboration

The sensing capability of the skin has been tested in a number of control strategies for human-robot collaboration with a 7-DOF KUKA LWR4 robot manipulator. The algorithm presented here allows the manual guidance of the robot using one or two contact points, where force sensing is available.

The first contact point $\boldsymbol{p}_e \in \mathbb{R}^3$ is located on the end effector, where the force is measured using a commercial wrist force sensor, while the second contact point $\boldsymbol{p}_b \in \mathbb{R}^3$ is located on link 3, where a patch of sensitive skin has been bonded.

The robot control law is a model-based position tracking control in the joint space. The control torque $\boldsymbol{\tau}_c \in \mathbb{R}^7$ is computed as:

$$\boldsymbol{\tau}_c = \boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{q}}_c + \boldsymbol{K}_D(\dot{\boldsymbol{q}}_c - \boldsymbol{q}) + \boldsymbol{K}_P(\boldsymbol{q}_c - \boldsymbol{q})) + \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{J}_b^T(\boldsymbol{q})\boldsymbol{F}_b - \boldsymbol{J}_e^T(\boldsymbol{q})\boldsymbol{F}_e \tag{47}$$

being $\boldsymbol{J}_e(\boldsymbol{q}) \in \mathbb{R}^{3\times 7}$ and $\boldsymbol{J}_b(\boldsymbol{q}) \in \mathbb{R}^{3\times 7}$ the (known) Jacobian matrices associated to the points $\boldsymbol{p}_e$ and $\boldsymbol{p}_b$, respectively, where the forces $\boldsymbol{F}_e \in \mathbb{R}^3$ and $\boldsymbol{F}_b \in \mathbb{R}^3$ are applied. It can be shown that the above control law, with $\boldsymbol{K}_P, \boldsymbol{K}_D \in \mathbb{R}^{7\times 7}$ suitable positive definite matrix gains, allows tracking of the commanded joint trajectory $\boldsymbol{q}_c(t)$. A further additional contribution is usually required in (47) to account for the Coriolis and centrifugal effects, which have quadratic dependence on the joint velocities. This term has been neglected here, since joint velocities are usually small in human-robot collaboration tasks.

Note that the control torque (47) ideally rejects the effects of the contact forces $\boldsymbol{F}_e$ and $\boldsymbol{F}_b$. On the other hand, the compliant behaviour required for manual guidance is entrusted to the commanded joint trajectory $\boldsymbol{q}_c(t)$, which is computed on the basis of suitable dynamic relationships, or admittances, between the sensed contact forces and the displacements of the contact points. Such approach allows to implement the human-robot collaboration strategy even on standard robot manipulators equipped with a position control interface, rather than a torque control interface.

For a generic contact point $\boldsymbol{p}_c$, the desired acceleration $\ddot{\boldsymbol{p}}_{c,d}$, velocity $\dot{\boldsymbol{p}}_{c,d}$ and position $\boldsymbol{p}_{c,d}$ can be computed from the contact force $\boldsymbol{F}_c$ using the admittance equation:

$$\ddot{\boldsymbol{p}}_{c,d} = \boldsymbol{M}_c^{-1}(\boldsymbol{F}_c - \boldsymbol{D}_c\dot{\boldsymbol{p}}_{c,d}), \tag{48}$$

where $\boldsymbol{M}_c, \boldsymbol{D}_c \in \mathbb{R}^{3\times 3}$ are suitable positive definite matrix gains, with the meaning of mass and damping, respectively.

Since the two contact points belong to the same kinematic chain, their motion cannot be assigned arbitrarily and conflicting situations may occur. These conflicts can be managed by the control through a suitable task priority strategy. Depending on the specific situation, the motion of one of the two contact points is considered as the main task, while the motion of the other point is considered as a secondary task. Only the motion components of the secondary task that are not conflicting with the main task, i.e., those projected into the null space of the Jacobian of the main task, will be executed.

The main task can be associated, for example, to the point that has been touched as first. Therefore, if the human applies first a force $\boldsymbol{F}_e$ at point $\boldsymbol{p}_e$ on the end effector and then applies a force $\boldsymbol{F}_b$ at point $\boldsymbol{p}_b$ in the robot body, the latter will cause a reconfiguration of the robot that will not affect motion of $\boldsymbol{p}_e$, which depends only on $\boldsymbol{F}_e$. Vice versa, if the human applies first a force $\boldsymbol{F}_b$ at point $\boldsymbol{p}_b$ on the robot body, then the motion of $\boldsymbol{p}_b$ will depend only on $\boldsymbol{F}_b$, no matter whether another force will be applied at the end effector or not.

By adopting the multi-priority control formulation presented in [8], the command acceleration $\ddot{\boldsymbol{q}}_c$ is computed as

$$\ddot{\boldsymbol{q}}_c = \boldsymbol{J}_e^{\dagger}(\ddot{\boldsymbol{x}}_{e,d} - \dot{\boldsymbol{J}}_e\dot{\boldsymbol{q}}) + \bar{\boldsymbol{J}}_b^{\dagger}(\ddot{\boldsymbol{x}}_{b,d} - \dot{\boldsymbol{J}}_b\dot{\boldsymbol{q}} - \boldsymbol{J}_b\boldsymbol{J}_e^{\dagger}(\ddot{\boldsymbol{x}}_{e,d} - \dot{\boldsymbol{J}}_e\dot{\boldsymbol{q}})), \tag{49}$$

where $\boldsymbol{J}_e^{\dagger}$ is a generalized inverse of $\boldsymbol{J}_e$, $\bar{\boldsymbol{J}}_b^{\dagger} = (\boldsymbol{I} - \boldsymbol{J}_e^{\dagger}\boldsymbol{J}_e)\boldsymbol{J}_b^{\dagger}$ is the generalized inverse of $\boldsymbol{J}_b$ projected into the null space of $\boldsymbol{J}_e$, while the command accelerations $\ddot{\boldsymbol{x}}_{c,d}$, with $c = e, b$ are computed as

$$\ddot{\boldsymbol{x}}_{c,d} = \ddot{\boldsymbol{p}}_{c,d} + k_d(\dot{\boldsymbol{p}}_{c,d} - \dot{\boldsymbol{p}}_c) + k_p(\boldsymbol{p}_{c,d} - \boldsymbol{p}_c), \tag{50}$$

being k_d, k_p strictly positive gains. The vectors $\dot{\boldsymbol{p}}_c$ and $\boldsymbol{p}_c$ are the actual velocity and position of the contact point, respectively, and the vectors $\ddot{\boldsymbol{p}}_{c,d}$, $\dot{\boldsymbol{p}}_{c,d}$ and $\boldsymbol{p}_{c,d}$ are computed using the admittance equation (48), with $c = e, b$. Equation (49) assumes that the motion of point $\boldsymbol{p}_e$ has higher priority with respect to the motion of point $\boldsymbol{p}_b$. The change of priority can be achieved using the same equation, by replacing the subscript e with the subscript b and vice versa.

Finally, when the Jacobians are close to a singularity, generalized inverses can be calculated more robustly using damped least squares pseudoinversion.

6.3 Results

A conformable skin patch is installed on the third link of the KUKA LWR4 robot (see Fig. 8). In the performed experiments, the priority of the two tasks is defined by the operator on the basis of the first point touched. In both cases, the desired position of the end effector is computed by the admittance equations (48), that use as input the force $\boldsymbol{F}_e$ applied to the robot tip $\boldsymbol{p}_e$, as measured by an ATI Mini45 F/T sensor installed on the robot wrist, while the position of the contact point on the robot body and the contact force are measured by the sensor skin.

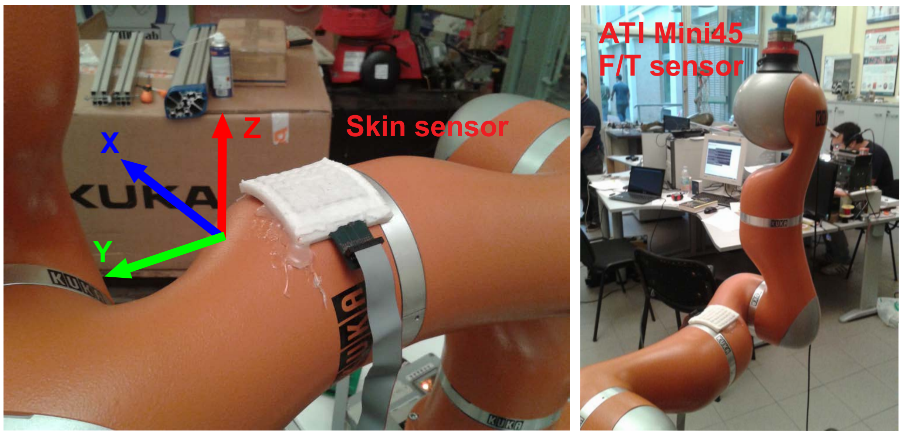


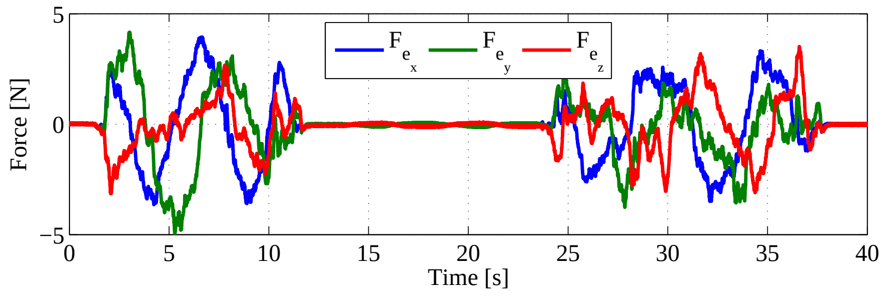
Figure 8: The skin sensor installed on the 3rd link of the KUKA LWR4 robot.

Two case studies have been considered to illustrate the robot behaviour with the different priorities.

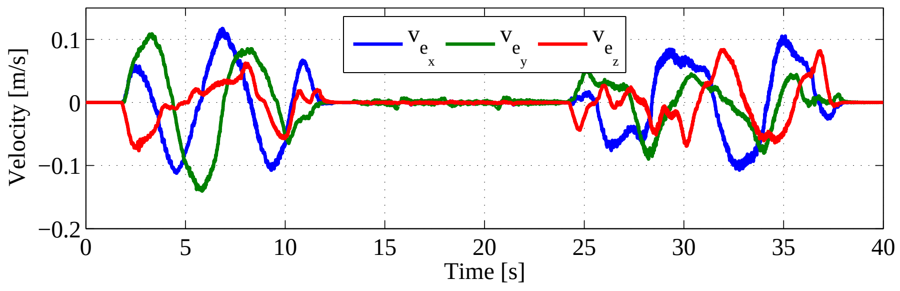
The first case study is aimed at testing the behaviour of the robot when the operator touches first the end-effector point $\boldsymbol{p}_e$ and then the robot body at $\boldsymbol{p}_b$. In this case, the end effector motion is the task with higher priority. The results are reported in Fig. 9. It can be recognized that, during the first time interval of about 12 s the operator touches only point $\boldsymbol{p}_e$. Comparing Figs. 9(a) and 9(b), it can be seen that the end effector moves according to the forces exerted at the tip. Moreover, from Fig. 9(d), it can be recognized that, some motion is induced also on point $\boldsymbol{p}_b$ of the body, where the contact force, reported in Fig. 9(c), is zero.

From Fig. 9(c) it can be seen that, at around $t = 14$ s the operator touches the point on the robot body, since the forces measured by the skin sensor becomes different from zero. These forces produce a motion only in the null space of the main task. In fact, comparing the time histories of the forces applied to the point $\boldsymbol{p}_e$ (Fig. 9(a)) and $\boldsymbol{p}_e$ (Fig. 9(c)) with that of the velocities $\dot{\boldsymbol{p}}_e$ (Fig. 9(b)) and $\dot{\boldsymbol{p}}_b$ (Fig. 9(d)), it is clear that the end effector motion is only affected by the force applied to the tip, also when the configuration of the robot is additionally brought in motion under the action of $\boldsymbol{F}_b$. On the other hand, the velocity $\dot{\boldsymbol{p}}_b$ derives from the composition of the motion allowed in the null space with that produced by the main task.

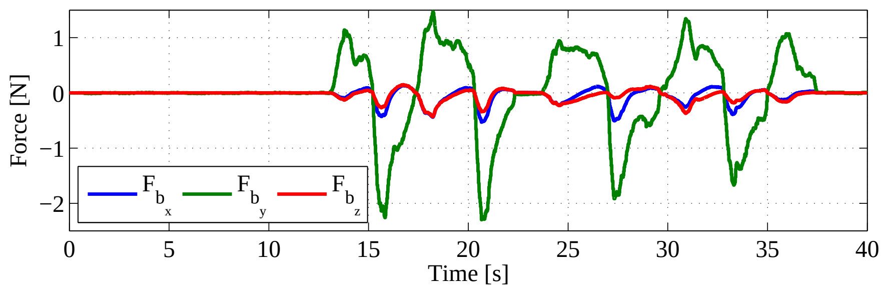
Figure 10 reports a similar analysis for the second case study. The desired position of the body point is selected as the main task by touching first point $\boldsymbol{p}_b$ on the sensor skin and then the robot tip $\boldsymbol{p}_e$. Figures 10(b), 10(c) and 10(d) show that the velocities $\dot{\boldsymbol{p}}_e$ and $\dot{\boldsymbol{p}}_b$ are affected only by the force $\boldsymbol{F}_b$. Instead, Figs. 10(a), 10(b) and 10(c) show that the end-effector motion does not produce any contribution to the velocity $\dot{\boldsymbol{p}}_b$, which is the main task.



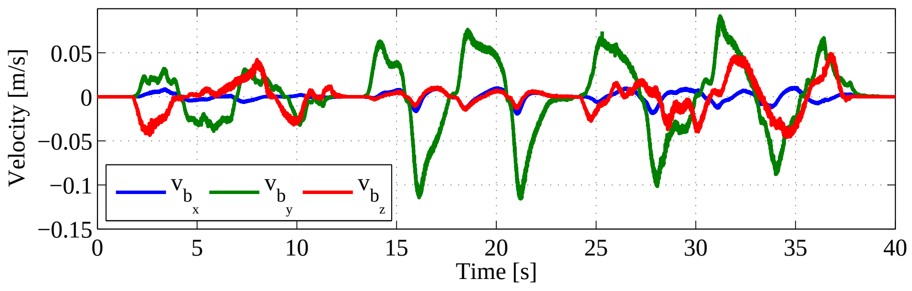
(a) Force components $\boldsymbol{F}_e$ measured with the ATI F/T sensor.



(b) Velocity $\dot{\boldsymbol{p}}_e$ of the end effector.

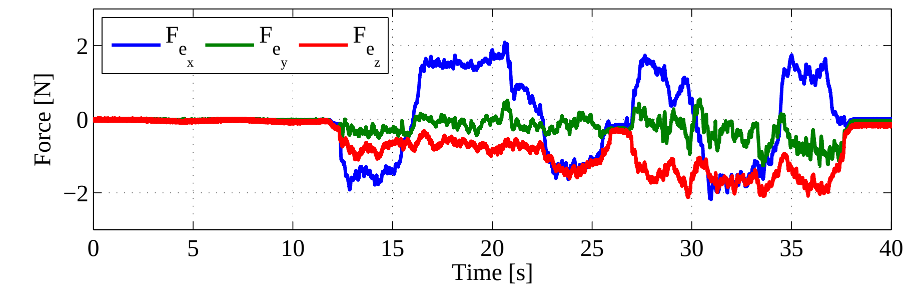


(c) Force components $\boldsymbol{F}_b$ measured with the skin sensor.

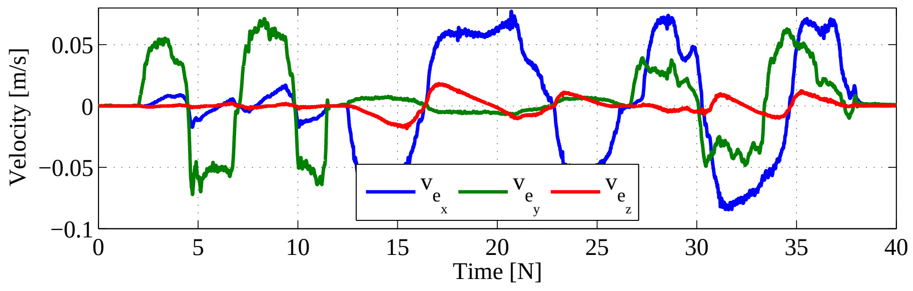


(d) Velocity $\dot{\boldsymbol{p}}_e$ of the contact point on the body.

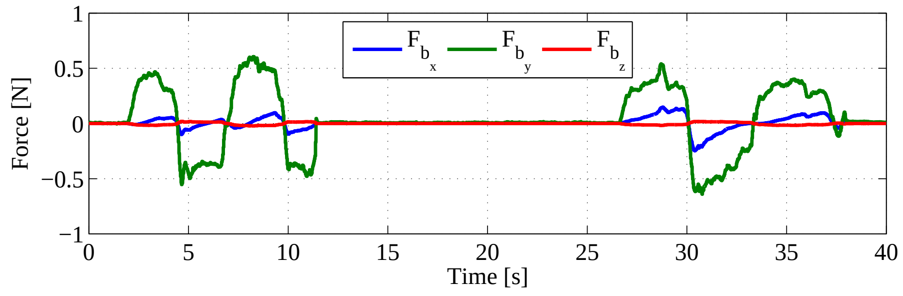
Figure 9: First case study. All measurements are expressed w.r.t. robot base frame.



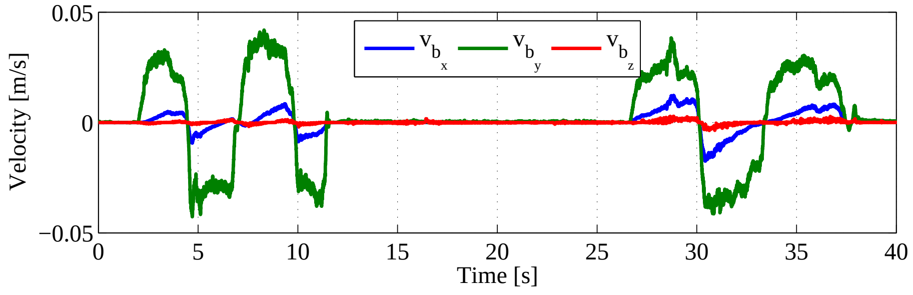
(a) Force components $\boldsymbol{F}_e$ measured with the ATI F/T sensor.



(b) Velocity $\dot{\boldsymbol{p}}_e$ of the end effector.



(c) Force components $\boldsymbol{F}_b$ measured with the skin sensor.



(d) Velocity $\dot{\boldsymbol{p}}_e$ of the contact point on the body.

Figure 10: Second case study. All measurements are expressed w.r.t. robot base frame.

7 Control of Generalized Contact Motion and Force

In [11], we have introduced a method for localizing the contact point(s) between a human and the robot body, as well as for estimating the associated contact force(s), by integrating the information collected by one (or more) RGB-D external sensor with the proprioceptive data coming from the n-dimensional residual vector $\boldsymbol{r}$ defined in eq. (45) —see also Deliverable D.4.1.1. In particular, in the case of a single contact at any point of the robot structure, the Jacobian of the contact point $\boldsymbol{J}_c(\boldsymbol{q})$ is obtained using the RGB-D information, and then the contact force can be estimated by pseudoinversion as

$$\widehat{\boldsymbol{F}}_c = \left(\boldsymbol{J}_c^T(\boldsymbol{q})\right)^{\#}\boldsymbol{r}. \tag{51}$$

We note that this method, which is alternative to the one proposed in Sec. 6, works both in static and dynamic conditions (i.e., when the robot is moving relatively fast while contact occurs). The force estimate (51) can be used then for controlling the human-robot physical collaboration according to paradigmatic schemes.

In fact, classical impedance control and direct force control have been extended in [22] from joint- or Cartesian-based schemes to a generic contact point $\boldsymbol{x}_c$ along the robot manipulator, considering within this generalization the use of the estimate $\widehat{\boldsymbol{F}}_c$ provided by (51) in place of an actual measurement of the contact force, every time this would be needed.

It should be mentioned that direct force control needs always a measure (or an estimate) of the contact force, in order to close a control loop on the force error. Instead, standard joint or Cartesian impedance control laws require a force measure only if we wish to change the *natural* robot inertia (defined at the proper level) into a desired one. When the robot inertia is left unchanged, force sensing is no longer required (the term *compliance control* should be used then). The same situation holds true also for our generalized impedance control laws (replacing measured with estimated quantities).

7.1 Impedance control

Starting from robot model (6), defining $\boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q})$ and dropping from now on dependencies for the sake of compactness, consider the inverse dynamics control law

$$\boldsymbol{\tau} = \boldsymbol{M}\boldsymbol{a} + \boldsymbol{n} - \boldsymbol{J}_c^T\widehat{\boldsymbol{F}}_c. \tag{52}$$

In ideal design conditions, the above control law realizes a feedback linearization, leading to a system of double integrators $\ddot{\boldsymbol{q}} = \boldsymbol{a}$. The auxiliary control input $\boldsymbol{a}$ is chosen so as to impose a mechanical impedance model between the (estimated) contact forces $\widehat{\boldsymbol{F}}_c$ and the motion of the contact point $\boldsymbol{x}_c$, or

$$\boldsymbol{M}_d(\ddot{\boldsymbol{x}}_c - \ddot{\boldsymbol{x}}_d) + \boldsymbol{K}_d(\dot{\boldsymbol{x}}_c - \dot{\boldsymbol{x}}_d) + \boldsymbol{K}_p(\boldsymbol{x}_c - \boldsymbol{x}_d) = \widehat{\boldsymbol{F}}_c, \tag{53}$$

where, as in (33), $\boldsymbol{M}_d > 0$ is the desired inertia matrix, $\boldsymbol{K}_d > 0$ is the desired damping matrix, $\boldsymbol{K}_p > 0$ is the desired stiffness matrix (very often, these matrices are taken as diagonal), and $\boldsymbol{x}_d(t)$ is the smooth desired trajectory of the contact point. Solving for $\ddot{\boldsymbol{x}}_c$ from (53), the control input $\boldsymbol{a}$ is obtained as

$$\boldsymbol{a} = \boldsymbol{J}_{c,M}^{\#}\boldsymbol{M}_d^{-1}\left(\boldsymbol{M}_d\ddot{\boldsymbol{x}}_d + \boldsymbol{K}_d\dot{\boldsymbol{e}} + \boldsymbol{K}_p\boldsymbol{e} - \boldsymbol{M}_d\dot{\boldsymbol{J}}_c\dot{\boldsymbol{q}} + \widehat{\boldsymbol{F}}_c\right) + \boldsymbol{P}_c\ddot{\boldsymbol{q}}_{\mathcal{N}}, \tag{54}$$

with contact position error $\boldsymbol{e} = \boldsymbol{x}_d - \boldsymbol{x}_c$, projection matrix $\boldsymbol{P}_c$ in the null space of $\boldsymbol{J}_c$, and a generic acceleration vector $\ddot{\boldsymbol{q}}_{\mathcal{N}} \in \mathbb{R}^n$ that is used to shape the null space robot motion. Thus, the resulting torque control input is

$$\boldsymbol{\tau} = \boldsymbol{M}\boldsymbol{J}_{c,M}^{\#}\boldsymbol{M}_d^{-1}\left(\boldsymbol{M}_d\ddot{\boldsymbol{x}}_d + \boldsymbol{K}_d\dot{\boldsymbol{e}} + \boldsymbol{K}_p\boldsymbol{e} - \boldsymbol{M}_d\dot{\boldsymbol{J}}_c\dot{\boldsymbol{q}} + \widehat{\boldsymbol{F}}_c\right) + \boldsymbol{M}\boldsymbol{P}_c\ddot{\boldsymbol{q}}_{\mathcal{N}} + \boldsymbol{n} - \boldsymbol{J}_c^T\widehat{\boldsymbol{F}}_c. \tag{55}$$

In (55), observe that the contribution of $\widehat{\boldsymbol{F}}_c$ is given by

$$\boldsymbol{\tau} = \ldots + \left(\boldsymbol{M}\boldsymbol{J}^{\#}_{c,M}\boldsymbol{M}_d^{-1} - \boldsymbol{J}_c^T\right)\widehat{\boldsymbol{F}}_c. \tag{56}$$

As expected, when choosing the desired inertia matrix $\boldsymbol{M}_d$ equal to the *natural Cartesian inertia* of the robot *at the contact point*,

$$\boldsymbol{M}_d = \left(\boldsymbol{J}_c\boldsymbol{M}^{-1}\boldsymbol{J}_c^T\right)^{-1}. \tag{57}$$

also the estimation of the contact forces $\widehat{\boldsymbol{F}}_c$ will no longer be needed, as expected. In fact, by using (8), it is easy to see that $\boldsymbol{M}\boldsymbol{J}^{\#}_{c,M}\boldsymbol{M}_d^{-1} - \boldsymbol{J}_c^T = \boldsymbol{0}$. On the other hand, when we desire to impose a different inertial behaviour at the contact, e.g., so as to achieve different apparent masses in different directions of the contact operational space, we shall make use of the contact force estimate $\widehat{\boldsymbol{F}}_c$.

7.2 Force control

The contact impedance scheme realizes only an *indirect* control of the forces exchanged during the interaction with the human, imposing instead to the position $\boldsymbol{x}_c$ of the contact point a dynamic balance in terms of error with respect to the desired position. The development of a *direct* controller for the contact force is crucial for collaborative tasks that needs accurate execution in uncertain conditions.

With the reference to (52), choose in place of (54) the acceleration

$$\boldsymbol{a} = \boldsymbol{J}^{\#}_{c,M}\boldsymbol{M}_d^{-1}\left(\boldsymbol{K}_f(\boldsymbol{F}_d - \widehat{\boldsymbol{F}}_c) - \boldsymbol{K}_d\dot{\boldsymbol{x}}_c - \boldsymbol{M}_d\dot{\boldsymbol{J}}_c\dot{\boldsymbol{q}}\right) + \boldsymbol{P}_c\ddot{\boldsymbol{q}}_{\mathcal{N}}, \tag{58}$$

where $\boldsymbol{F}_d$ is the desired force at the contact point, the contact force error is $\boldsymbol{e}_f = \boldsymbol{F}_d - \widehat{\boldsymbol{F}}_c$, and $\boldsymbol{K}_f > 0$ is the force error gain matrix. Replacing (58) in (52), the final torque control input is

$$\boldsymbol{\tau} = \boldsymbol{M}\boldsymbol{J}^{\#}_{c,M}\boldsymbol{M}_d^{-1}\left(\boldsymbol{K}_f\boldsymbol{e}_f - \boldsymbol{K}_d\dot{\boldsymbol{x}}_c - \boldsymbol{M}_d\dot{\boldsymbol{J}}_c\dot{\boldsymbol{q}}\right) + \boldsymbol{M}\boldsymbol{P}_c\ddot{\boldsymbol{q}}_{\mathcal{N}} + \boldsymbol{n} - \boldsymbol{J}_c^T\widehat{\boldsymbol{F}}_c. \tag{59}$$

The closed-loop system is then described by

$$\boldsymbol{M}_d\ddot{\boldsymbol{x}}_c + \boldsymbol{K}_d\dot{\boldsymbol{x}}_c = \boldsymbol{K}_f(\boldsymbol{F}_d - \widehat{\boldsymbol{F}}_c), \tag{60}$$

which shows that the force error will go to zero and the contact point will eventually stop whenever a constant contact force is applied by the human.

Note that in the above expression of the control law, the specification of a desired contact force $\boldsymbol{F}_d$ in the human-robot interaction is apparently an unconstrained one. However, there is an issue of interaction task compatibility that remains open. We shall see through the experiment results that a careful choice of $\boldsymbol{F}_d$ should be made so as to avoid task inconsistency, with a resulting drift behaviour of the robot which would seriously affect safety.

7.3 Results

A series of dynamic human-robot interaction experiments with the proposed controllers have been performed on a KUKA LWR4. The workspace is monitored by a Microsoft Kinect depth sensor, positioned at a distance of 1.7 m behind the robot and at a height of 1.1 m w.r.t. the robot base frame. The Kinect provides 640×480 depth images at 30 Hz rate. All algorithms are executed on a quad-core CPU. The complete process of contact force estimation and of motion and/or force control run at 5 ms cycle time.

With reference to the impedance control law (55), define $\boldsymbol{x}_d = \boldsymbol{x}_c(t_c) \in \mathbb{R}^3$ as the initial (and constant) position of the contact point when the interaction with the human begins at $t = t_c$. The desired velocity and acceleration are set then to zero, $\dot{\boldsymbol{x}}_d = \ddot{\boldsymbol{x}}_d = \mathbf{0}$. Due to the redundancy of this robot with respect to many tasks, a null-space acceleration vector has been chosen as $\ddot{\boldsymbol{q}}_{\mathcal{N}} = -\boldsymbol{K}_N\dot{\boldsymbol{q}}$, with $\boldsymbol{K}_N > 0$, in order to damp out self motions of the arm. Recall also that the KUKA LWR4 has a built-in gravity compensation. Thus, the actual impedance control law used to command the robot from the user point of view is

$$\boldsymbol{\tau} = \boldsymbol{M}\boldsymbol{J}^{\#}_{c,M}\boldsymbol{M}_d^{-1}\left(\boldsymbol{K}_p\boldsymbol{e} - \boldsymbol{K}_d\dot{\boldsymbol{x}}_c - \boldsymbol{M}_d\dot{\boldsymbol{J}}_c\dot{\boldsymbol{q}} + \widehat{\boldsymbol{F}}_c\right) - \boldsymbol{M}\boldsymbol{P}\boldsymbol{K}_N\dot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{J}_c^T\widehat{\boldsymbol{F}}_c \tag{61}$$

Similarly, for the force control law it is

$$\boldsymbol{\tau} = \boldsymbol{M}\boldsymbol{J}^{\#}_{c,M}\boldsymbol{M}_d^{-1}\left(\boldsymbol{K}_f\boldsymbol{e}_f - \boldsymbol{K}_d\dot{\boldsymbol{x}}_c - \boldsymbol{M}_d\dot{\boldsymbol{J}}_c\dot{\boldsymbol{q}}\right) - \boldsymbol{M}\boldsymbol{P}\boldsymbol{K}_N\dot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{J}_c^T\widehat{\boldsymbol{F}}_c. \tag{62}$$

Impedance control with natural contact inertia

In the first experiment, the human pushes the robot at different contact points and on different links, as shown in the snapshot of Fig. 11. The control law is given by (61), with the desired inertia matrix at the contact chosen equal to (57). The other impedance parameters were set to $\boldsymbol{K}_d = 80 \cdot \boldsymbol{I}_3$, $\boldsymbol{K}_p = 500 \cdot \boldsymbol{I}_3$, and $\boldsymbol{K}_N = 20 \cdot \boldsymbol{I}_7$, where $\boldsymbol{I}_k$ denotes the $k \times k$ identity matrix. Figure 12 shows the behaviour of the residual vector $\boldsymbol{r}$, of the contact force estimate $\widehat{\boldsymbol{F}}_c$, and of the Cartesian position error $\boldsymbol{e}$ of the contact point with respect to its initial position $\boldsymbol{x}_d = \boldsymbol{x}_c(t_c)$.

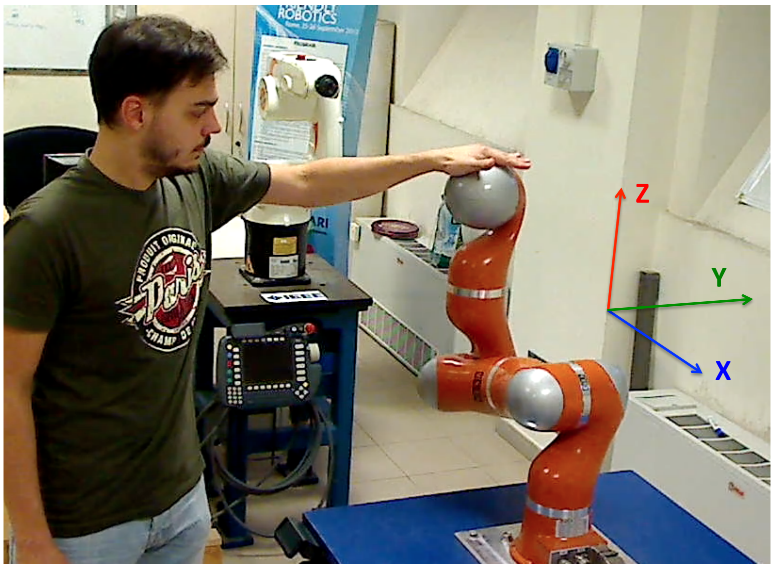


Figure 11: The set-up and the Cartesian reference frame for the contact impedance control experiments. The colors of X, Y, and Z axes are the same used for the associated components in the plots of Fig. 12.

Impedance control with modified contact inertia

In this experiment, the impedance scheme (61) was used again during the human-robot interaction. However, the desired inertia matrix at the contact point was set to

$$\boldsymbol{M}_d = \begin{pmatrix} 20 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 10 \end{pmatrix},$$

i.e., different from the natural one (which was also coupled). Non-uniform values were chosen on the diagonal, so as to obtain different behaviours along the three Cartesian axes –see Fig. 11. To enhance this effect, the human pushes the robot always on link 6, although in different directions. The other impedance parameters used were the same as before. In this case, the estimated contact forces $\widehat{\boldsymbol{F}}_c$ are needed by the control law in order to obtain the desired mass-spring-damper system. Figure 13 shows the same quantities of the previous impedance control experiment. When the contact is initially detected, the robot will move the contact point with a dynamics that depends on the direction of the external force. When the hand is removed, the robot brings back the contact point to the original initial position. As shown in Fig. 13, the error dynamics in returning to zero is faster along the Y axis, where $M_{d,y} = 3$. When increasing the desired inertia (e.g., along x axis), the error dynamics slows down too.

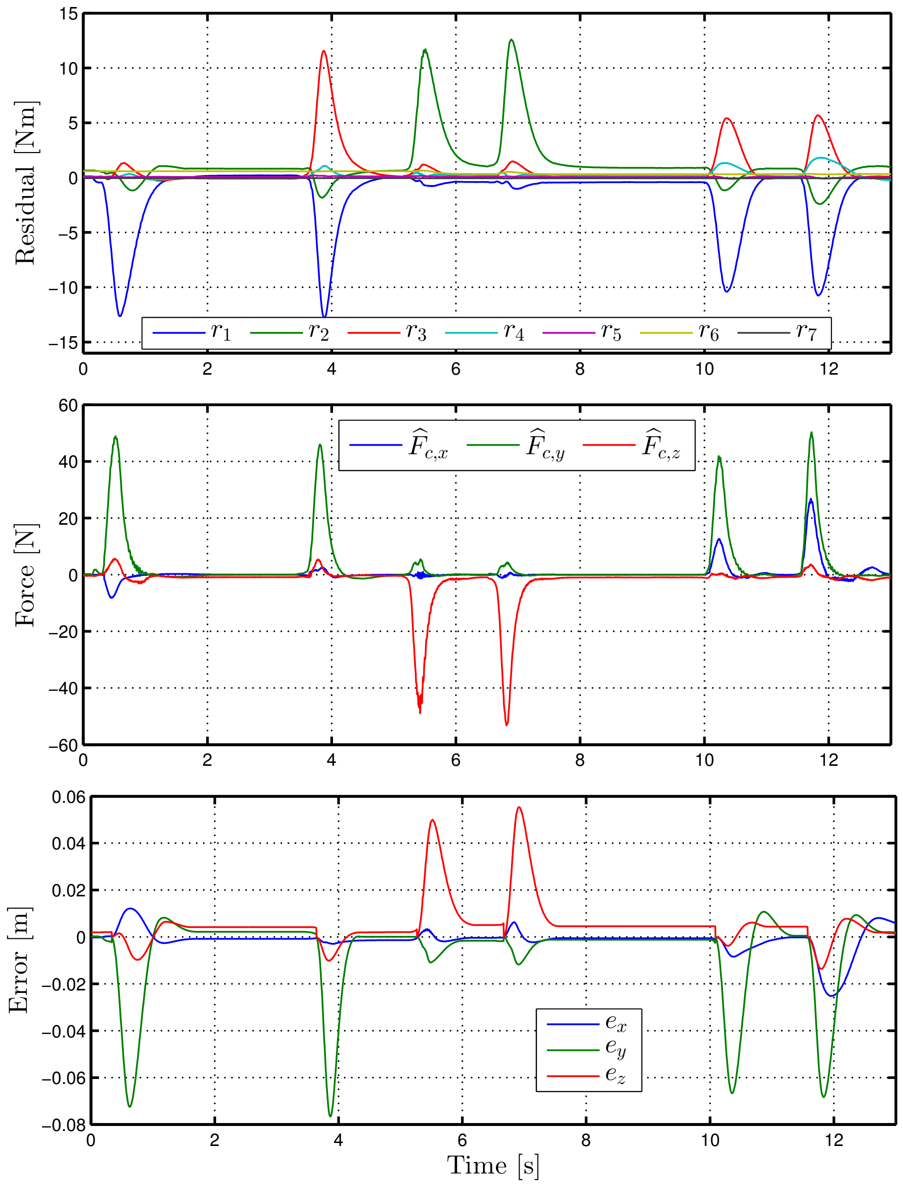


Figure 12: Contact impedance control with desired inertia matrix at the contact equal to the natural one. [Top] Residual vector components. [Center] Estimated contact force components [Bottom] Contact position error components.

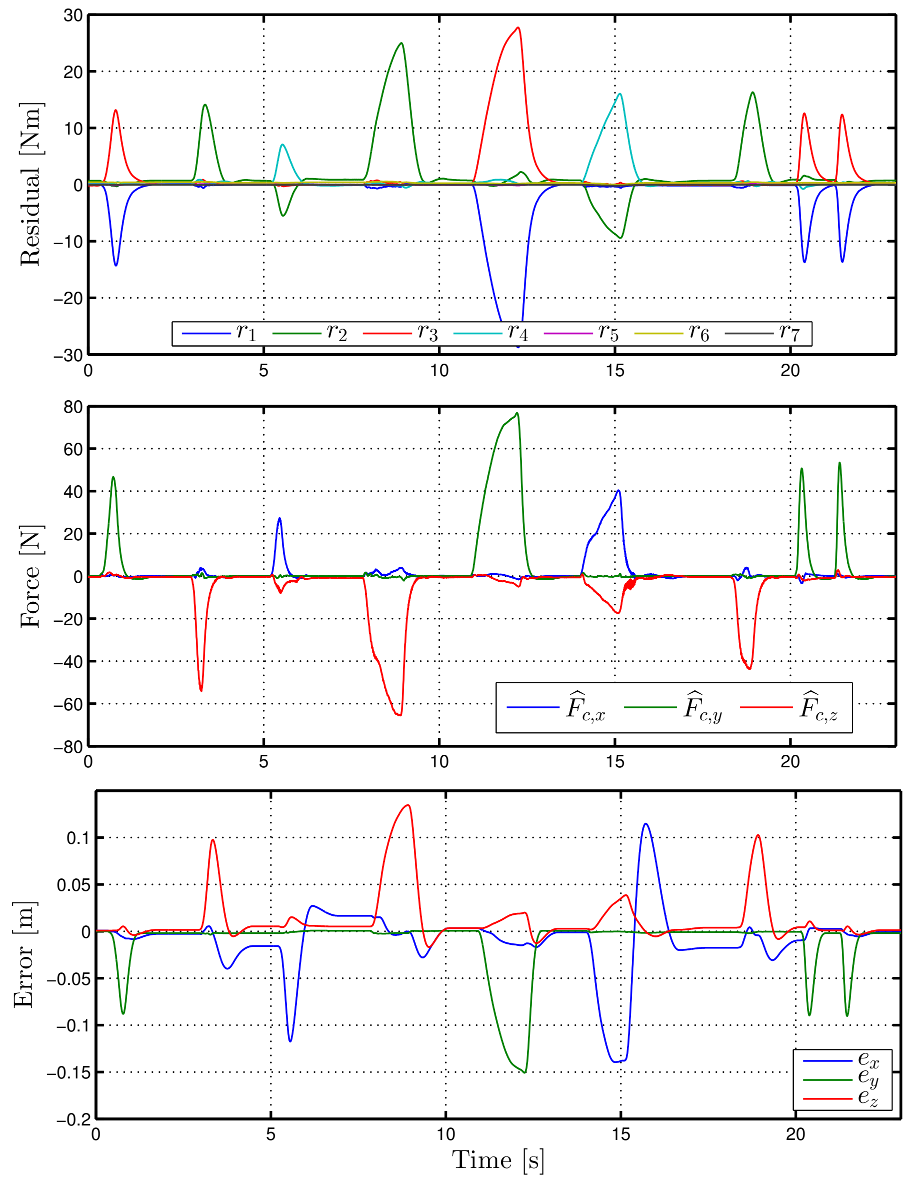


Figure 13: Contact impedance control with modified inertia matrix at the contact. [Top] Residual vector components. [Center] Estimated contact force components [Bottom] Contact position error components.

Contact force control with possible drift effects

The purpose of this experiment was to show that it is highly not recommended to try to regulate human-robot contact forces at arbitrary values and in multiple fixed directions of the Cartesian space. When the human does not push or resist along a direction where a non-zero force reference has been specified, the robot typically drifts in space in the attempt to regulate the incompatible desired force. This could be dangerous because an unexpected movement occurs, through which the robot may collide with the human.

With this in mind, the parameters in the control law (62) were set to $\boldsymbol{K}_d = 10.5 \cdot \boldsymbol{I}_3$, $\boldsymbol{K}_f = 4.1 \cdot \boldsymbol{I}_3$, $\boldsymbol{K}_N = 15 \cdot \boldsymbol{I}_7$, and the desired contact force was chosen as $\boldsymbol{F}_d = \begin{pmatrix} 0 & 15 & 0 \end{pmatrix}^T$, i.e., only in the Y direction. Figure 14 shows the set upo for this experiment. The behaviour of the residual vector $\boldsymbol{r}$, of the contact force estimate $\widehat{\boldsymbol{F}}_c$, and of the Cartesian velocity $\dot{\boldsymbol{x}}_c$ of the contact point are shown in Fig. 15. When the human pushes the robot on link 6 and along the Y axis, the robot reacts properly and regulates the force components to the desired level. However, when the human pushes the robot along the X direction, a drift occurs along the Y direction, as shown in Fig. 15, see, e.g., at $t = 14.8$ s.

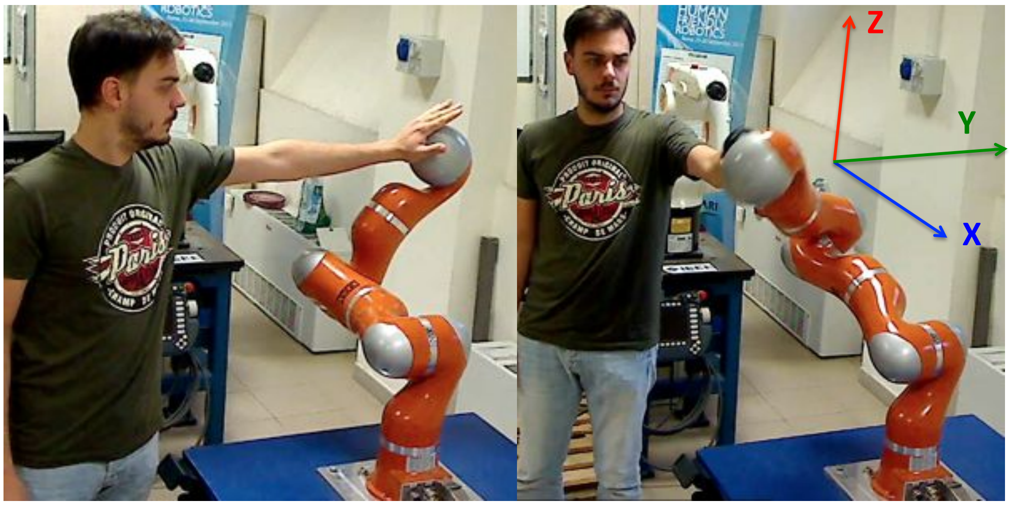


Figure 14: The set-up and the Cartesian reference frame for the contact force control experiments. [Left] The human is pushing along the Y direction, where a non-zero desired force is specified ($F_{y,d} = 15$ N), and regulation occurs (see the first 12 s in Fig. 15). [Right] The human pushing along the X direction is accommodated by the robot at the desired zero value ($F_{x,d} = 0$ N), but the absence of a human contrast in the Y direction does not allow regulation to a non-zero force value in that direction, and generating instead a large velocity drift (see the last 6 s in the third plot of Fig. 15).

Contact force control with task compatibility

The contact force control scheme (62) was used in this interaction control experiment, where the human pushes the robot successively on link 3, link 5, and then link 6 (see the associated behaviours of the residuals at the top of Fig. 16). To avoid task incompatibility in the human-robot interaction, only the norm of the desired contact force $\boldsymbol{F}_d$ is specified, here at a constant value $\|\boldsymbol{F}_d\| = 15$ N. However, the desired vector $\boldsymbol{F}_d$ will change direction according to the human-robot contact type, being always aligned with the estimated contact force vector $\hat{\boldsymbol{F}}_c$, and may now vary without any restriction. We will have thus,

$$F_{d,x} = 15\frac{\widehat{F}_{c,x}}{\|\widehat{\boldsymbol{F}}_c\|}, \quad F_{d,y} = 15\frac{\widehat{F}_{c,y}}{\|\widehat{\boldsymbol{F}}_c\|}, \quad F_{d,z} = 15\frac{\widehat{F}_{c,z}}{\|\widehat{\boldsymbol{F}}_c\|}.$$

The other force control parameters were the same used before.

Figure 16 shows the same quantities of the previous force control experiment. The controller starts operating after the first detection of a contact (around $t = 1.5$ s). When the human pushes with his hand on the mechanical structure, the robot gets compliant in order to regulate the contact force at the desired value. When the human tries to recede, the robot pushes against the human hand so as to maintain the desired $\widehat{\boldsymbol{F}}_c = \boldsymbol{F}_d$. The fact that the human may abandon the contact and then resume it in a different point and/or along a different direction is no longer a problem for this correctly defined contact force controller. On the other hand, this force control law is able to regulate contact forces only in quasi-static conditions (i.e., when $\dot{\boldsymbol{x}}_c = 0$). As long as there is a joint human-robot motion during a contact, some force error will result as shown in Fig. 16.

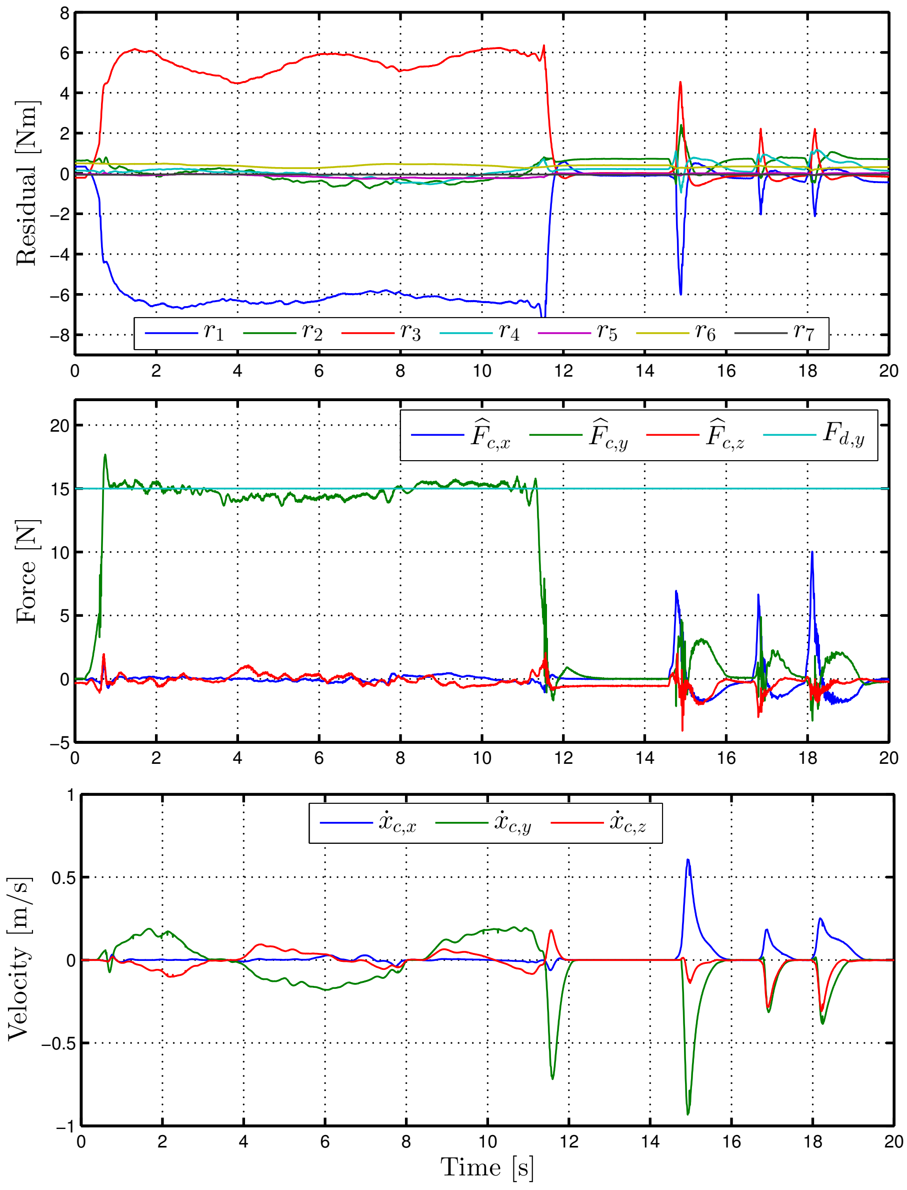


Figure 15: Contact force control with drift motion due to task incompatibility. [Top] Components of the residual vector. [Center] Modulus of the estimated contact force. [Bottom] Components of the Cartesian contact velocity $\dot{\boldsymbol{x}}_c$.

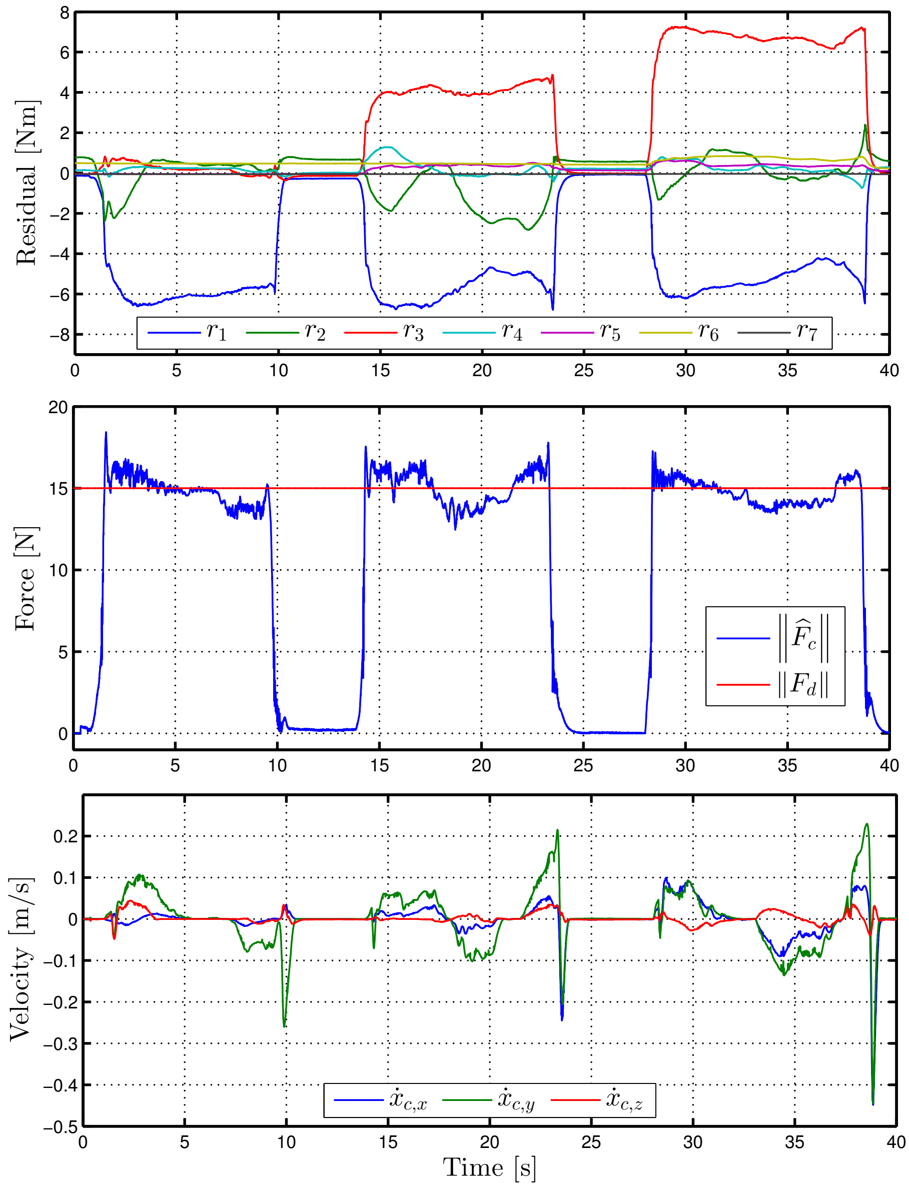


Figure 16: Contact force control designed for automatic task compatibility. [Top] Components of the residual vector. [Center] Modulus of the estimated contact force. [Bottom] Components of the Cartesian contact velocity $\dot{\boldsymbol{x}}_c$.

References

[1] A. Zube, “Cartesian Nonlinear Model Predictive Control of Redundant Manipulators Considering Obstacles,” Submitted to IEEE Int. Conf. on Industrial Technology (ICIT), 2015.

[2] H. Sadeghian, L. Villani, M. Keshmiri, B. Siciliano, “Task-Space Control of Robot Manipulators With Null-Space Compliance”, IEEE Transactions on Robotics, 2014, pp. 493–506.

[3] A. De Luca, R. Mattone, “Sensorless robot collision detection and hybrid force/motion control,” *Proc. IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 999–1004.

[4] O. Khatib, “Inertial properties in robotic manipulation: an object-level framework,” International Journal of Robotics Research, vol. 14, no. 1, 1995, pp. 19–36.

[5] A. De Luca, G. Oriolo, “Nonholonomic behavior in redundant robots under kinematic control,” IEEE Trans. Robot. Autom., vol. 13, no. 5, 1997, pp. 776–782.

[6] A. Cirillo, P. Cirillo, G. De Maria, C. Natale and S. Pirozzi, “An Artificial Skin Based on Optoelectronic Technology,” Sensors & Actuators: A. Physical, vol. 212, 2014, pp. 110–122.

[7] G. De Maria, C. Natale, S. Pirozzi, “Force/Tactile Sensor for Robotic Applications,” Sensors & Actuators: A. Physical., vol. 175, 2012, pp. 60–72.

[8] H. Sadeghian, L. Villani, M. Keshmiri, B. Siciliano, “Dynamic Multi-Priority Control in Redundant Robotic Systems,” Robotica, 2013, pp. 1155–1167.

[9] A. Cirillo, F. Ficuciello, C. Natale, S. Pirozzi, L. Villani, “A sensitive skin for safe and controlled human-robot physical interaction,” *submitted to* IEEE Int. Conf. on Robotics and Automation, 2015.

[10] A. De Luca and R. Mattone, “Sensorless robot collision detection and hybrid force/motion control,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 1011–1016.

[11] E. Magrini, F. Flacco, and A. De Luca, “Estimation of contact forces using a virtual force sensor,” in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2014, pp. 2126–2133.

[12] A. De Luca, A. Albu-Schäffer, S. Haddadin, and G. Hirzinger, “Collision detection and safe reaction with the DLR-III lightweight robot arm,” in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2006, pp. 1623–1630.

[13] N. Hogan, “Impedance control: An approach to manipulation: Part I - Theory, Part II - Implementation, Part III - Applications,” *ASME J. of Dynamic Systems, Measurement, and Control*, vol. 107, 1985, pp. 1–24.

[14] B. Siciliano and L. Villani, “Robot Force Control,” Kluwer, 1999.

[15] F. Flacco and A. De Luca, “A reverse priority approach to multi-task control of redundant robots,” in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2014, pp. 2421–2427.

[16] F. Flacco and A. De Luca, “Discrete-time velocity control of redundant robots with acceleration/torque optimization properties,” in *IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 5139–5144.

[17] T. L. Boullion and P. L. Odell, “Generalized Inverse Matrices,” Wiley-Interscience, 1971.

[18] A. Liégeois, “Automatic supervisory control of the configuration and behavior of multibody mechanisms,” *IEEE Trans. on Systems, Man and Cybernetics*, vol. 7, no. 12, 1977, pp. 868–871.

[19] A. De Luca and R. Mattone, “Sensorless robot collision detection and hybrid force/motion control,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 1011–1016.

[20] A. De Luca, A. Albu-Schäffer, S. Haddadin, and G. Hirzinger, “Collision detection and safe reaction with the DLR-III lightweight robot arm,” in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2006, pp. 1623–1630.

[21] F. Flacco and A. De Luca, “Unilateral constraints in the Reverse Priority redundancy resolution method,” submitted to *IEEE Int. Conf. on Robotics and Automation*, 2015.

[22] E. Magrini, F. Flacco, and A. D. Luca, “Control of generalized contact motion and force in physical human-robot interaction,” submitted to *IEEE Int. Conf. on Robotics and Automation*, 2015.