

# Multi-Contact Motion Retargeting from Human to Humanoid Robot

Alessandro Di Fava<sup>1</sup>, Karim Bouyarmane<sup>2</sup>, Kevin Chappellet<sup>3</sup>, Emanuele Ruffaldi<sup>1</sup>, Abderrahmane Kheddar<sup>3,4</sup>

**Abstract**—We propose a framework for real-time online and offline retargeting of a human actor motion to a humanoid robot motion involving multi-contact configuration changes between the human/humanoid and their environments. The framework is based on the specification within a multi-contact QP control formulation of tracking tasks for a selected set of body-segments/links, the ones either used for a manipulation task from a fixed multi-contact stance or susceptible to be used as contact supports or as locomotion supports on the environment in the course of the execution of the motion. The framework is applied in an online setting for simultaneous human-robot motion tracking (in the case of no contact configuration change) and in an offline setting for tracking the playback of the recorded human motion that is pre-processed to extract from it the sequence of contact change events as a necessary motion information exploited by the tracking algorithm. We present a real robot experiment with HRP-4 for the online setting and a dynamics simulation experiment for the offline setting to validate the proposed approach.

## I. INTRODUCTION

Human-to-humanoid motion retargeting and humanoid multi-contact motion planning are two problems that were extensively studied in recent years [1]–[11]. We propose here to combine both approaches in a common framework to solve new classes of problems that will unlock novel applications. Examples of these envisioned applications are the tasks and motions expected to be carried out by a humanoid robot in a confined or highly constrained environment that was designed for human operators, as was motivated in the recent DARPA Robotics Challenge (DRC) or in the growing use of humanoid robots in manufacturing spaces such as aircraft manufacturing scenarios ([www.comanoid.eu](http://www.comanoid.eu)).

In our previous works [12], [13] we proposed a fully autonomous approach to solve the problem of humanoid motion in such confined human operator environments. The two-stage approach starts with a multi-contact planning stage. The planning algorithm autonomously explores the contact space, from which it extracts a sequence of contact transitions to bring the humanoid from its initial configuration to a desired goal configuration. Each step of the sequence consists of a multi-contact configuration that differs from the precedent step by exactly one contact, either removing that

contact or adding it. Each multi-contact configuration is valid if and only if a posture-generator module finds a full-body robot configuration that matches the contact configuration while being collision-free in the confined space, in addition to satisfying to all other physical and mechanical constraints of the robot. The second stage of the approach is a full-body quadratic program (QP)-based motion control stage [14]. The QP controller steers the full-body configuration of the robot from one step of the sequence to the following step by calculating in real-time, at each control loop iteration, the joint configuration command to send to the robot to realize a given set of tasks that are automatically derived to reach the next step of the sequence.

The above two-stage approach proved successful in solving some the DRC-related problems such as the car egress, ladder climbing, and valve manipulation scenarios [15]–[17]. It however can be computationally expensive and might require up to several tens of minutes for solution extraction. The human operator might also want to manually change or adjust the autonomous plan for various reasons unaccounted for by the autonomous planner.

Hence we propose in this paper to investigate an alternative approach to tackle the problem of multi-contact motion in confined human operator environment. We do not want to strictly and exclusively rely on the geometrical information and physics-law constraints for fully autonomous planning. Instead, the idea is to get “guided” (in the sense precised in the paper) by the motion performed by a human actor in the same environment. The robot then replicates that motion in its environment, simultaneously and in parallel with the human demonstration, or with a delay after offline processing and analysis of the human motion to extract from it additional information required for the robot controller.

To this end, we use an Xsens MVN inertial-marker-based system ([www.xsens.com](http://www.xsens.com), Fig. 1) to capture the motion of the human demonstrator in the confined space, and the multi-contact QP-based controller to retarget the motion on the humanoid robot HRP-4 [18] (34 degrees of freedom).

Related work that accounted for changing contacts, in the dynamics of the retargeted motion presented the limitation of not considering multi-contact stances (e.g. not involving hand contacts) [19], [20]. In particular, among those, ZMP-based approaches are not appropriate for considering these stances. Other approaches similar to ours in the objectives [21] remain strictly kinematics-based and are therefore not adaptable to real robots. None demonstrated the results on real robots. To the best of our knowledge, we propose the first multi-contact motion retargeting framework and apply it on

<sup>1</sup>A. Di Fava and E. Ruffaldi are with Scuola Superiore Sant’Anna di Pisa, 56127 Pisa, Italy [a.difava@sssup.it](mailto:a.difava@sssup.it)

<sup>2</sup>K. Bouyarmane is with University of Lorraine - CNRS, Loria UMR 7503, Vandœuvre-lès-Nancy, F-54500, France, and with Inria, Villers-lès-Nancy, F-54600, France

<sup>3</sup>K. Chappellet and A. Kheddar are with CNRS University of Montpellier LIRMM, 34095 Montpellier Cedex 5, France

<sup>4</sup>A. Kheddar is also with CNRS-AIST JRL UMI3218/RL, 1-1-1 Umezono, Tsukuba 305-8568, Japan



Fig. 1: Front and rear view of the human subject wearing the Xsens system

a real robot, hence our contribution with respect to the state-of-the-art.

The rest of the paper is organized as follows. In Section II we recall the QP-based formalism that we use in our approach. Section III describes the full framework, with Section III-A focusing on the initialization phase to put the robot in the initial posture prior to motion tracking, Section III-B addressing the motion retargeting method in fixed contact configuration, and Section III-C describing the algorithm devised to account for changing contact configurations. Results in the forms of a real-robot experiment and a dynamics simulation are presented in Section IV. Finally Section V concludes the paper with expected follow-up work.

## II. MULTI-CONTACT QP CONTROL FORMALISM

At each control time-step  $\Delta t$  ( $\Delta t = 5$  ms for the HRP-4 robot), the QP controller [14], [17] solves the following QP program:

$$\min_{\dot{q}, u, f_C} \sum_{k=1}^m w_k \|\ddot{\tau}_k - \ddot{\tau}_k^d\|^2 \quad (1)$$

$$\text{subject to} \quad M\ddot{q} + N = Su + J_C^T \Lambda_C f_C \quad (2)$$

$$J_C \ddot{q} + \dot{J}_C \dot{q} = 0 \quad (3)$$

$$f_C \geq 0 \quad (4)$$

$$u_{\min} \leq u \leq u_{\max} \quad (5)$$

$$\frac{q_{\min} - q - \dot{q}\Delta t}{\frac{1}{2}\Delta t^2} \leq \ddot{q} \leq \frac{q_{\max} - q - \dot{q}\Delta t}{\frac{1}{2}\Delta t^2} \quad (6)$$

$$\frac{\dot{q}_{\min} - \dot{q}}{\Delta t} \leq \ddot{q} \leq \frac{\dot{q}_{\max} - \dot{q}}{\Delta t} \quad (7)$$

$$\ddot{d}(b_1, b_2) \geq \frac{1}{\Delta t} \left( -\xi \frac{d(b_1, b_2) - \delta_s}{\delta_i - \delta_s} - \dot{d}(b_1, b_2) \right) \quad (8)$$

where (1) is a weighted sum of  $m$  tasks  $\tau_k$  with respective weights  $w_k$  and desired task accelerations  $\ddot{\tau}_k^d$ , (2) is the dynamics equation of motion with mass matrix  $M$ , nonlinear

effects matrix  $N$ , actuation torques  $u$ , actuated-joint selection matrix  $S$ , contact links indexed in the set  $C$ , concatenated contact point Jacobians  $J_C$ , linearized friction cone generator matrix  $\Lambda_C$ , contact force coefficients along the generators  $f_C$ , (3) is the non-sliding contact constraint, (4) the friction cone constraint, (5) the torque limit constraint, (6) the joint limit constraint, (7) the joint velocity limit constraint, (8) the collision-avoidance constraint between body  $b_1$  of the robot and body  $b_2$  of the robot or the environment, with distance between the two bodies  $d(b_1, b_2)$ , damping coefficient  $\xi$ , security distance  $\delta_s$ , influence distance  $\delta_i$ . Note that there are as many constraints (8) as there are pairs of collision or self-collision bodies accounted for, and that although the distance function is not linear in the configuration of the robot, yet at any given state  $(q, \dot{q})$  equation (8) is linear in  $\ddot{q}$ . See [16], [17] for more details on the formulation.

An  $n_k$ -dimensional task  $\tau_k$  is a mapping  $\mathbb{R}^n \rightarrow \mathbb{R}^{n_k}$  from the  $n$ -dimensional configuration space of the robot (6 dimensions of floating base plus number of degrees of freedom of the actuated joints) to the  $n_k$ -dimensional task space. For example,  $\tau_k$  can denote the position of a link ( $n_k = 3$ ) or its orientation ( $n_k = 3$ ), it can also denote the full-configuration of the robot ( $n_k = n$ ) or the actuated joint configuration (posture) ( $n_k = n - 6$ ), finally it can denote the center-of-mass (CoM) position of the robot ( $n_k = 3$ ) or its horizontal ground projection ( $n_k = 2$ ). The task velocity is  $\dot{\tau}_k = J_k \dot{q}$  and its acceleration is  $\ddot{\tau}_k = J_k \ddot{q} + \dot{J}_k \dot{q}$ , where  $J_k = \frac{\partial \tau_k}{\partial q}$  is the Jacobian of the task.

The desired task acceleration  $\ddot{\tau}_k^d$  in (1) can be derived in several ways, depending on whether we have an explicit time-parameterized reference trajectory to follow for the task, denoted  $t \mapsto \tau_k^{\text{ref}}(t)$ , or only a fixed target, denoted  $\tau_k^{\text{tgt}}$ . In the first case, the desired trajectory can be derived as

$$\ddot{\tau}_k^d(\tau_k^{\text{ref}}) = \ddot{\tau}_k^{\text{ref}} + 2\sqrt{K_k}(\dot{\tau}_k^{\text{ref}} - \dot{\tau}_k) + K_k(\tau_k^{\text{ref}} - \tau_k), \quad (9)$$

with  $K_k$  being the stiffness gain parameter of the task. This is a user-defined parameter. High-stiffness will force a close but stiff following of the reference trajectory while low stiffness will ensure more compliance with respect to external perturbations in the following of the task reference. The second case with fixed target is a particular case of the first one with  $\tau_k^{\text{ref}}(t) = \tau_k^{\text{tgt}}$  and  $\ddot{\tau}_k^{\text{ref}} = \dot{\tau}_k^{\text{ref}} = 0$ . Hence in that second case we derive the desired task acceleration as

$$\ddot{\tau}_k^d(\tau_k^{\text{tgt}}) = -2\sqrt{K_k}\dot{\tau}_k + K_k(\tau_k^{\text{tgt}} - \tau_k). \quad (10)$$

A third case occurs when we have a trajectory to follow without having its explicit time-parameterization. This typically happens for real-time online trajectory following of human motion. In that case we can either consider in a first approximation a fixed target tracking (10) updating the “moving fixed target”  $\tau_k^{\text{tgt}}$  at every time-step  $\Delta t$ , or in a second approximation use the trajectory following (9) by approximating  $\ddot{\tau}_k^{\text{ref}}$  and  $\dot{\tau}_k^{\text{ref}}$  from a measured  $\tau_k^{\text{ref}} = \tau_k^{\text{tgt}}$  after applying to it a smoothing and differentiation filter such as a Savitzky-Golay filter for example (note that since we use an inertial-sensor based system the tasks for link position

and orientation are themselves already filtered from their acceleration measurement [22]).

### III. MOTION RETARGETING USING THE QP

#### A. Initialization of the robot posture

We will first study retargeting of a human motion to the humanoid robot when the motion is performed on a fixed set of contacts, e.g. a manipulation motion with the two hands while both feet are fixed during the whole manipulation. In this subsection we describe the initialization procedure, that will also be common to the motion retargeting with changing contact configuration.

Let us denote  $\alpha$  the number of links (body-segments) of interest for the motion and  $E \subset \{1, \dots, l\}$  the set containing indexes of these links, among the total number  $l$  of links of the robot. By “link of interest” we mean any link of the robot that is susceptible to be used either for contact support to enable the motion or for a manipulation task during the course of the motion. In the above-mentioned example of dual-hand manipulation with both feet fixed on the ground, we have  $\alpha = 4$ , considering the two hands and two feet as the links of interest. Other motions however might involve knee or elbow contacts (crawling), buttocks parts that can be the legs or the waist (sitting down), head orientation (gaze). All of these links would thus be included in the list of links of interest.

The  $\alpha$  links in  $E$  can be partitioned into two subsets  $E_1$  and  $E_2$  of size  $\alpha_1$  and  $\alpha_2$  respectively, such that  $\alpha = \alpha_1 + \alpha_2$  and  $E = E_1 \cup E_2$ .  $E_1$  contains the indexes of the links that are in contact, and  $E_2$  the indexes of links that are used for the manipulation or that are susceptible to become future contact supports in the course of the motion. (i.e.  $E_2$  contains all the links that are tracked without being in contact in the current multi-contact stance).

The initialization of the posture of the robot in its environment will be performed by writing a *kinematic mode* of the QP in Section II, i.e. dropping in the QP the constraints (2) to (5) and the variables  $f_C$  and  $u$ , and by writing tasks for all  $\alpha$  links in  $E$ , plus an additional low-weight full-body posture task that is the default zero-posture ( $q = 0$ ) or the manufacturer-provided half-sitting posture of the robot. The kinematic posture initialization is performed while the robot is hanging in the air on a dedicated lifter, which is a security device commonly found and used in most laboratories hosting a full-size humanoid robot.

All tasks in the kinematic initialization are written as fixed target tasks. For every link  $k$  in  $E_1$  let us denote  $x_k$  its position relative to the base link of the robot (HRP-4 waist link) and  $x_k^{\text{mkr}}$  the position of the corresponding marker-containing body-segment on the human kinematic model relative to the marker-containing base-link (human pelvis in the Xsens human model). Both  $x_k$  and  $x_k^{\text{mkr}}$  are translated at an origin point that is located at the center of the contact surface of the link respectively for the robot and the human (e.g. center of sole if the link is the foot). We additionally add for the link  $k$  a *contact orientation* task, denoted  $\phi_k$ , to make the normal of the contact surface of the robot  $\nu_k$  match

the normal of the contact surface of the environment  $\nu_k^{\text{env}}$ . The task  $\phi_k$  is a one-dimensional task

$$\phi_k = \langle \nu_k | \nu_k^{\text{env}} \rangle, \quad (11)$$

where  $\langle \cdot | \cdot \rangle$  denotes the  $\mathbb{R}^3$  scalar product. The target value for that task is  $\phi_k^{\text{tgt}} = 1$ . The use of the contact orientation tasks allows us to bypass the issue of different frame orientation of the link between the human model and the robot one.

Finally, for every link  $k$  in  $E_2$  we add a position task  $x_k$  to follow the position of the corresponding marker-containing body-segment of the human model  $x_k^{\text{mkr}}$ .

All in all, the kinematic QP takes the form:

$$\begin{aligned} \min_{\dot{q}} \quad & \sum_{k \in E_1 \cup E_2} w_k \|\ddot{x}_k - \ddot{x}_k^d(x_k^{\text{mkr}})\|^2 \\ & + \sum_{k \in E_1} w'_k \|\ddot{\phi}_k - \ddot{\phi}_k^d(1)\|^2 + w_0 \|\ddot{q} - \ddot{q}^d(0)\|^2 \\ \text{subject to constraints} \quad & (6) \text{ to } (8). \end{aligned} \quad (12)$$

The human tries as much as possible to maintain his/her posture and to not move while waiting for the robot to reach the initial posture in the air using the kinematic QP (12). When the robot reaches the initial posture (link task errors below a threshold), it is put on the ground. The robot then notifies the human with a voice synthesis speech sentence (e.g. “intialization done”) and the human can start their motion.

#### B. Online motion tracking on a fixed contact configuration

After the robot is put on the ground we switch the QP controller from the kinematic mode to the full dynamic mode as described in Section II. In that dynamic mode the contact indexes set  $C$  is identified with the set of indexes  $E_1$  (i.e. we set  $C \leftarrow E_1$ ). We therefore remove from the tasks in (12) the position and orientation tasks for the links  $k$  in  $E_1$  since the constraint (3) takes care of fixing the position and orientation of the contact links.

When the robot is on the ground, balance becomes an issue. To address it we add a high-weight task on the ground projection of the CoM, that we denote  $c \in \mathbb{R}^2$ . The fixed target of the task  $c$  is computed as the center of the convex hull of the ground projection of the vertexes of the contact surfaces for the links in  $E_1$ . Let us call that convex-hull center  $h[E_1] \in \mathbb{R}^2$ . The fixed target for the task  $c$  is  $h[E_1]$ .

The dynamic QP that we use to track the motion of the human in real-time hence becomes:

$$\begin{aligned} \min_{\dot{q}, u, f_{E_1}} \quad & \sum_{k \in E_2} w_k \|\ddot{x}_k - \ddot{x}_k^d(x_k^{\text{mkr}})\|^2 \\ & + w_c \|\ddot{c} - \ddot{c}^d(h[E_1])\|^2 + w_0 \|\ddot{q} - \ddot{q}^d(0)\|^2 \\ \text{subject to constraints} \quad & (2) \text{ to } (8). \end{aligned} \quad (13)$$

The major difference between the  $x_k, \phi_k$  tasks in (12) and the  $x_k$  tasks in (13) (the link tracking tasks) is their stiffness  $K_k$ . For the initialization phase, since the robot starts in a configuration that is far away from the target configuration, we use low stiffness to avoid abrupt motion caused by an initially large task error (value used in the experiments  $K_k =$

5). For the motion-following phase, the task error is always small and we can keep close tracking of the motion with a high stiffness (value used in the experiments  $K_k = 40$ ).

### C. Offline motion tracking with changes in the contact configuration

In the subsection above the two sets  $E_1$  and  $E_2$  in (13) never change and keep the same elements throughout the motion. Hence the QP instance (13) stays the same. When considering now motions that involve a changes of the contact configuration, some elements are interchanged between the sets  $E_1$  and  $E_2$  during the contact change event (overall though the set  $E = E_1 \cup E_2$  stays the same). For example, starting with our previous example of dual-hand manipulation on fixed foot locations, if one of the hands, e.g. the right hand, comes in contact with a table of the environment, then the index of the right hand is removed from  $E_2$  and added to  $E_1$ . If and when the right hand contact is later released, its index is removed from  $E_1$  and added back to  $E_2$ . At each contact change event the instance of the QP (13) is changed.

To properly follow the motion, we need a way to detect the contact change events. The difficulty in extracting the contact change information from inertial marker kinematic data (position, velocity, acceleration) online is that there is no easy way to differentiate between the actual contact event and the hand stopping in the air at the end of its contact removing motion for example, without using additional sensors such as force sensors or proximity sensors in the hands. Since we only rely on geometrical information from inertial measurement units that constitute the Xsens markers, we chose to proceed in a different – offline – approach, based on recording the human motion data, processing it to extract the contact change event information from it, and replay the motion in simulation while simultaneously tracking it online with the robot that is now aware of the upcoming contact change events. After recording the human motion data, we export it from the Xsens provided software MVN Studio to a format compatible with the Unity simulation engine software ([www.unity3d.com](http://www.unity3d.com)).

The MVN Studio human model and the environment model are both imported inside Unity. We replay the human motion and use the Unity API to detect collision events between all the links in  $E = E_1 \cup E_2$  and the surfaces of the environment. A contact change event is defined as a 4-tuple

$$\eta = (e, s, t, \sigma) \in E \times S \times \mathbb{R} \times \{0, 1\}, \quad (14)$$

where  $e$  is the link index,  $s$  is the environment surface,  $t$  is the timing of the event, and  $\sigma$  a binary variable encoding the type of event ( $\sigma = 0$  for leaving the contact and  $\sigma = 1$  for entering the contact). As an output of this Unity motion processing phase we are provided with a chronological sequence of contact change events in the form of a list

$$L = (\eta_1, \dots, \eta_j, \dots, \eta_\mu), \quad (15)$$

where  $\mu$  is the total number of events and each event of the sequence is denoted  $\eta_j = (e_j, s_j, t_j, \sigma_j)$ .

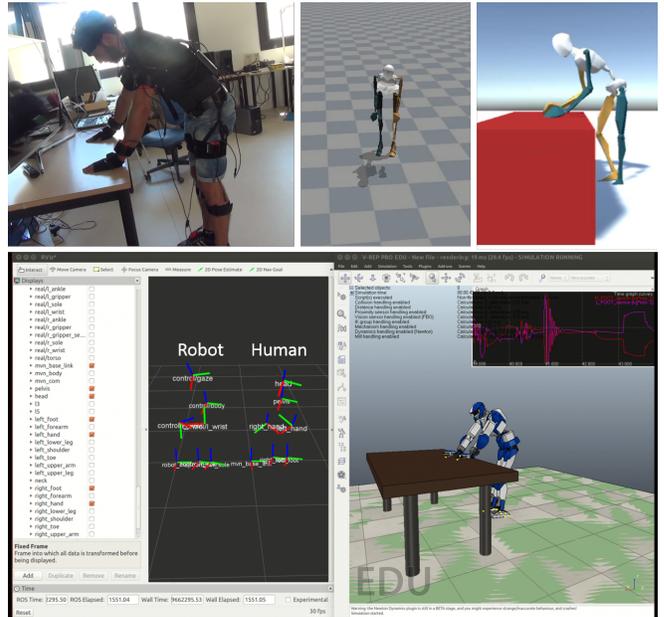


Fig. 2: Different layers of the control architecture. From left to right. Top: Real human, human model in Xsens MVN studio, human model exported to Unity. Bottom: QP controller in ROS, Simulation in VRep



Fig. 3: Real robot tracking

In the subsequent offline motion tracking phase we use the motion tracking formulation (13) and update  $E_1$ ,  $E_2$ , and (13) whenever the time  $t$  reaches the next event timing  $t_j$  along the sequence.

However, when entering a contact  $\sigma_k = 1$  and before updating  $E_1$ ,  $E_2$ , and (13) we temporarily pause the human motion tracking at a timing  $t = t_j - t^\epsilon$  where  $t^\epsilon$  is a small threshold (e.g.  $t^\epsilon = 1s$ ) to allow for *contact stabilization*, that is, fine positioning and orientating of the link  $e_j$  of the event  $\eta_j$  to reach the contact surface area  $s_j$ . During that

phase we instantiate the following QP:

$$\begin{aligned}
\min_{\dot{q}, u, f_{E_1}} \quad & \sum_{k \in E_2 \setminus \{e_j\}} w_k \|\dot{x}_k - \ddot{x}_k^d(x_k^{\text{mkr}})\|^2 \\
& + w_{e_j} \|\ddot{x}_{e_j} - \ddot{x}_{e_j}^d(x_{s_j}^{\text{surf}})\|^2 + w'_{e_j} \|\ddot{\phi}_{e_j} - \ddot{\phi}_{e_j}^d(1)\|^2 \\
& + w_c \|\ddot{c} - \ddot{c}^d(h[E_1])\|^2 + w_0 \|\ddot{q} - \ddot{q}^d(0)\|^2
\end{aligned}
\tag{16}$$

subject to constraints (2) to (8),

where  $x_{s_j}^{\text{surf}}$  is the projection of the center of the contact surface of the link  $e_j$  on the environment surface  $s_j$  along the direction of the surface normal  $\nu_{s_j}$ , and  $\phi(e_j)$  is the orientation task

$$\phi_{e_j} = \langle \nu_{e_j} | \nu_{s_j} \rangle. \tag{17}$$

All in all the algorithm for the motion tracking with offline contact change event processing is shown in Algorithm 1.

---

**Algorithm 1:** Motion tracking with contact change events

---

**Input:** Sequence of contact change events  
 $L = (\eta_1, \dots, \eta_\mu)$

**begin**

Kinematic posture initialization with QP (12);  
 Initialization of  $E_1$  and  $E_2$  with initial contact configuration;  
 Start tracking time  $t$ ;

**for**  $j = 1$  **to**  $\mu$  **do**

**if**  $\sigma_j = 0$  **then**

**while**  $t \leq t_j$  **do**

└ Motion tracking with QP (13);

//  $t = t_j$

$E_1 \leftarrow E_1 \setminus \{e_j\}$ ;

$E_2 \leftarrow E_2 \cup \{e_j\}$ ;

└ Update QP (13);

**else**  $\sigma_j = 1$

**while**  $t \leq t_j - t^\epsilon$  **do**

└ Motion tracking with QP (13);

//  $t = t_j - t^\epsilon$

**repeat**

└ Contact stabilization with QP (16);

**if**  $t \geq t_j$  **then**

└ Pause tracking time at  $t = t_j$

**until** *Contact positioned* **and**  $t = t_j$ ;

Resume tracking time at  $t = t_j$ ;

$E_1 \leftarrow E_1 \cup \{e_j\}$ ;

$E_2 \leftarrow E_2 \setminus \{e_j\}$ ;

└ Update QP (13);

// at that point  $t = t_\mu$  there is no more contact change event

**while**  $t \leq t^{\text{end of motion}}$  **do**

└ Motion Tracking with QP (13)

---

## IV. EXPERIMENTS

We set up two experiments to validate our approach, the first one with the real robot HRP-4 and the second one in dynamics simulation in VRep ([www.coppeliarobotics.com](http://www.coppeliarobotics.com)). See the Video Attachment<sup>1</sup>.

The first experiment (Fig. 3) is without contact change. The subject is standing while moving the hands without touching any object. The robot replicates the motion in real-time, while maintaining balance and avoiding self-collisions.

The second experiment (Fig. 2) involves 6 contact change events, starting from an initial position with only the two feet in contact (2 contacts). The sequence is as follows:

- 1) put the right hand on the table (3 contacts)
- 2) remove the right hand from the table (2 contacts)
- 3) put the right hand back on the table (3 contacts)
- 4) put the left hand on the table (4 contacts)
- 5) remove the left hand from the table (3 contacts)
- 6) remove the right hand from the table (2 contacts)

The robot in VRep dynamic simulation followed the motion in real-time on the playback after the motion was initially processed in Unity to extract the sequence above. The robot in VRep dynamic simulation was again able to follow the human motion without falling down, avoiding self-collisions, and collisions with the table other than the desired contact events<sup>2</sup>. Fig. 4 shows the tracking performances for the right and left hand links when making and breaking contacts with the table.

## V. CONCLUSION

We presented a complete framework for the retargeting of human motion to a full-size humanoid robot. The framework was based on the formulation inside a QP of motion tracking tasks for the links of interest in the motion, effortlessly overcoming all the difficulties raised by the differences between the kinematic tree models of the robot and the human. The additional tasks on the CoM and all the constraints of the QP allowed the robot to autonomously avoid self-collisions and maintain balance – even with the completely different geometric and dynamic models the human and the robot – which are two common problems in human-to-humanoid motion retargeting.

We plan to improve this framework in the future by devising a way to anticipate and detect online and in real-time the contact change events without resorting to offline processing of the motion. We also plan to enhance the motion capture system by exploiting information gathered from the Xsens shoe sensors that we can equip the human actor with. Finally we are currently implementing the work presented in this paper for a real-life problem using HRP-4 in an aircraft manufacturing environment.

<sup>1</sup>Higher quality version of the Video Attachment is available at [www.loria.fr/~kbouyarmane/humanoids2016.mp4](http://www.loria.fr/~kbouyarmane/humanoids2016.mp4)

<sup>2</sup>that second motion was not performed on the real robot since the robot was sent from the LIRMM lab back to Kawada Industries in Japan for maintenance and was not available at the time when the simulation was completed.

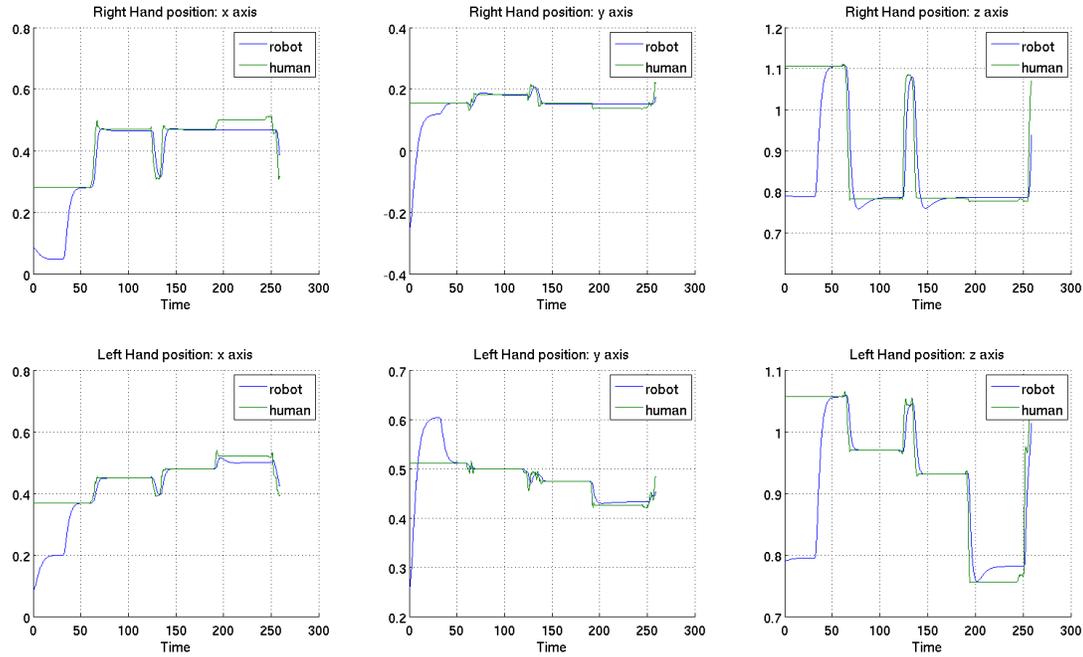


Fig. 4: Motion tracking performances of right hand and left hand links  $(x, y, z)$  positions in the contact change event experiment (blue: robot trajectories, green: human trajectories)

#### REFERENCES

- [1] M. Do, P. Azad, T. Asfour, and R. Dillmann, "Imitation of human motion on a humanoid robot using non-linear optimization," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2008, pp. 545–552.
- [2] C. Mandery, J. Borrás, M. Jochner, and T. Asfour, "Analyzing whole-body pose transitions in multi-contact motions," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2015, pp. 1020–1027.
- [3] C. Kim, D. Kim, and Y. Oh, "Adaptation of human motion capture data to humanoid robots for motion imitation using optimization," *Integrated computer-aided engineering*, vol. 13, no. 4, pp. 377–389, 2006.
- [4] D. Matsui, T. Minato, K. F. MacDorman, and H. Ishiguro, "Generating natural motion in an android by mapping human motion," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005, pp. 3301–3308.
- [5] A. Ude, C. G. Atkeson, and M. Riley, "Programming full-body movements for humanoid robots by observation," *Robotics and autonomous systems*, vol. 47, no. 2, pp. 93–108, 2004.
- [6] N. S. Pollard, J. K. Hodgins, M. J. Riley, and C. G. Atkeson, "Adapting human motion for the control of a humanoid robot," in *IEEE Int. Conf. on Robotics and Automation*, vol. 2, 2002, pp. 1390–1397.
- [7] M. Riley, A. Ude, K. Wade, and C. G. Atkeson, "Enabling real-time full-body imitation: a natural way of transferring human movement to humanoids," in *IEEE Int. conf. on Robotics and Automation*, vol. 2, 2003, pp. 2368–2374.
- [8] P. Azad, T. Asfour, and R. Dillmann, "Toward an unified representation for imitation of human motion on humanoids," in *IEEE Int. Conf. on Robotics and Automation*, 2007, pp. 2558–2563.
- [9] Ö. Terlemez, S. Ulbrich, C. Mandery, M. Do, N. Vahrenkamp, and T. Asfour, "Master motor map (mmm) framework and toolkit for capturing, representing, and reproducing human motion on humanoid robots," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2014, pp. 894–901.
- [10] J.-S. Monzani, P. Baerlocher, R. Boulic, and D. Thalmann, "Using an intermediate skeleton and inverse kinematics for motion retargeting," in *Computer Graphics Forum*, vol. 19, no. 3. Wiley Online Library, 2000, pp. 11–19.
- [11] K. Hausser, T. Bretl, and J.-C. Latombe, "Non-gaited humanoid locomotion planning," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2005, pp. 7–12.
- [12] K. Bouyarmane and A. Kheddar, "Humanoid robot locomotion and manipulation step planning," *Advanced Robotics*, vol. 26, no. 10, pp. 1099–1126, 2012.
- [13] A. Escande, A. Kheddar, and S. Miossec, "Planning contact points for humanoid robots," *Robotics and Autonomous Systems*, vol. 61, no. 5, pp. 428–442, 2013.
- [14] K. Bouyarmane and A. Kheddar, "Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2011, pp. 4414–4419.
- [15] K. Bouyarmane, J. Vaillant, F. Keith, and A. Kheddar, "Exploring humanoid robots locomotion capabilities in virtual disaster response scenarios," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2012, pp. 337–342.
- [16] J. Vaillant, A. Kheddar, H. Audren, F. Keith, S. Brossette, A. Escande, K. Bouyarmane, K. Kaneko, M. Morisawa, P. Gergondet, et al., "Multi-contact vertical ladder climbing with an hrp-2 humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 561–580, 2016.
- [17] J. Vaillant, K. Bouyarmane, and A. Kheddar, "Multi-character physical and behavioral interactions controller," *IEEE Transactions on Visualization and Computer Graphics*, vol. PP, no. 99, pp. 1–1, 2016.
- [18] K. Kaneko, F. Kanehiro, M. Morisawa, K. Akachi, G. Miyamori, A. Hayashi, and N. Kanehira, "Humanoid robot hrp-4-humanoid robotics platform with lightweight and slim body," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE, 2011, pp. 4400–4407.
- [19] K. Yamane and J. Hodgins, "Control-aware mapping of human motion data with stepping for humanoid robots," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE, 2010, pp. 726–733.
- [20] K. Yamane and Y. Nakamura, "Dynamics filter-concept and implementation of online motion generator for human figures," *IEEE transactions on robotics and automation*, vol. 19, no. 3, pp. 421–432, 2003.
- [21] H. J. Shin, J. Lee, S. Y. Shin, and M. Gleicher, "Computer puppetry: An importance-based approach," *ACM Transactions on Graphics (TOG)*, vol. 20, no. 2, pp. 67–94, 2001.
- [22] D. Roetenberg, H. Luinge, and P. Slycke, "Xsens mvn: full 6dof human motion tracking using miniature inertial sensors," *Xsens Motion Technologies BV, Tech. Rep*, 2009.