Support Changes during Online Human Motion Imitation by a Humanoid Robot using Task Specification

Louise Penna Poubel[1], Sophie Sakka[2], Denis Ćehajić[3] and Denis Creusot[4]

Abstract— **This paper presents a method based on inverse kinematics with task specification for online human to humanoid motion imitation. We particularly focus on the problem of lifting and placing feet on the floor during the motion, allowing change of support during stepping or locomotion. The approach avoids the use of motion primitives that limit the robot motions to what had been learned. A direct transposition of movements is generated, allowing the robot to move freely in space as the human model does, at a velocity close to the reference one. The approach is validated on the humanoid robot NAO and shows very promising results for the use of online motion imitation.**

I. Introduction

Humanoid robots are made to the image of human beings. Mechanically, their bodies try to emulate the human body in several aspects: whole-body robots possess two arms, two legs and a head. If humanoid robots are to interact with human beings, it is imperative that their gestures are human-like since much of human communication is non-verbal. However, programming each aspect of the motion detail by detail in order to make it human-like is time-consuming and not fit to handle the immense variety and complexity of human behaviors. A natural alternative is to look for inspiration in human movements to generate motion for the humanoid.

Motion imitation does not come without its challenges. Even the most elaborate humanoid robots have less degrees of freedom (NAO: 25, Asimo: 34, HRP-4C: 42) than the human body. This considerably limits the redundancy of the robot in relation to the human body. There are also differences in the link lengths, joint ranges, velocities, accelerations and torques, which must be properly mapped to the considered robot morphology during imitation. Moreover, issues such as self-collision and singularities must be addressed.

Riley *et al.* [1] scaled the captured motion in joint space in order to fit the joint ranges of the robot. The scaling was performed globally, which did not preserve nuances of the movement. This was addressed by Pollard *et al.* [3], who locally scaled angles and velocities in order to preserve as much as possible local variations in the motion

imitated by the Sarcos robot. Safonova *et al.* [4] addressed the issue of the robot overall configuration by inserting a term in the optimization which maintains the relative positions between certain key points on the body. In the previously mentioned works, the robot did not have to stand its own weight or remain balanced. Moreover, these were validated offline, allowing forward-backward loops considering the whole motion performed, or optimization processes.

Real-time or online imitation is necessary for interactive applications or teleoperation. In such case, time becomes a rigid constraint to be met to maintain as much as possible the artificial motion performance at the same speed rate than the model motion. Riley *et al.* [5] divided the inverse kinematics computation into 6 hierarchical chains to speed up computation. They were able to perform whole body real-time imitation, but once again balance was not considered. Montecillo *et al.* [6] sped up the retargeting process by developing a "Humanoid normalized model" which can be retargeted online for marker-based motion capture systems.

Koenemann *et al.* [7] proposed an online imitation by interpolating trajectories between captured human poses using inverse kinematics (IK). This resulted in a fluid motion but not retaining the nuances of the human reference motion and introducing a visible delay. On the other end, inverse kinematics was proposed by Sakka *et al.* [2] showing an almost non existent delay between the human motion and the humanoid one, but with a less fluid motion.

Kinematics retargeting is enough to perform imitation in slow velocities (quasi-static). When moving with higher accelerations however, inertial forces come into play and the system dynamics must be considered. As the robot body masses and achievable accelerations are different from those of the human, some dynamics retargeting must be made.

A common approach to keep balance is using a different controller for upper and lower body [8]. Other works have used human motion in order to design realistic trajectories for the zero-moment-point (ZMP) during imitation [9]. Nakaoka *et al.* [10] pointed out that due to limitations such as the lack of toes and the impossibility of crossing the legs, many humanoid robots can't properly imitate human motion. Thus, they use motion primitives for the legs and inverse kinematics for the arms. This allows for easier computation of balance using ZMP since the legs can only assume three different patterns, but over-simplifies the motion being imitated.

When changing support feet, there is a sudden change in the support area and it is necessary to ensure the projection of the CoM or the ZMP are within the new support polygon before effectively landing or taking off a foot. It is also

[1]Louise Penna Poubel is with IRCCyN and École Centrale de Nantes, France `Louise.Penna-Poubel@eleves.ec-nantes.fr`

[2]Sophie Sakka is with IRCCyN and University of Poitiers, France `Sophie.Sakka@irccyn.ec-nantes.fr`

[3]Denis Ćehajić is with the Institute for Information-oriented Control, Technische Universitt Mnchen, Munich, Germany `Denis.Cehajic@tum.de`

[4]Denis Creusot is with IRCCyN and École Centrale de Nantes, France `Denis.Creusot@irccyn.ec-nantes.fr`

important to make sure the foot interacts with the floor while flat. Montecillo *et al.* [6] anticipated a change of support of the feet to maintain balance during support transition by controlling the robot head and CoM displacement in one axis.

This paper will introduce a strategy which allows a change of support while maintaining the nuances of the human motion during online imitation. This is possible due to the solution of IK with task specification [2], which uses the robot redundancy to place its feet at different poses at each support change, as the human does. In the next section, the general method to scale the human motion to any humanoid robot dimensions will be introduced, and the method of task specification will be described. Next, the scaling and IK goals will be extended to different support phases and the transitions among them. Finally, the method will be validated with the NAO robot and several human actors, whose respective motions are tracked by a simple markerless motion capture system (Kinect).

II. METHOD

A. Scaling human motion

Imitation contains two main actions: reproduce the task and reproduce the manner the task is performed. The task in our case is to track the reference end effectors trajectories (the human hand and feet displacements). To do so by any humanoid robot, a geometric scaling must be performed beforehand so the human dimensions match the humanoid ones.

In this work, each human segment is scaled to the dimensions of the robot while keeping the segments respective directions. The scaled skeleton has the dimensions of the robot, with the same body pose as the human, as shown in Fig.1. This scaling can be used with any motion capture system which provides human joint positions/orientations in the Cartesian space. The 3 Cartesian coordinates of a joint ℓ are denoted $\mathbf{p}_\ell = [x_\ell, y_\ell, z_\ell]^T$. Points p_ℓ^h on the human skeleton are translated and become points on an equivalent robot skeleton p_ℓ^r. The iterative process starts from a point on the support foot, which is fixed to the ground, and moves upwards toward each limb extremity, joint by joint. The scaling is performed in 3 steps:

1) The direction of a human segment ℓ is taken (vector normalization, free vector);
2) The free vector is multiplied by the corresponding segment length on the robot l_ℓ^r;
3) The scaled segment is placed on the kinematic chain after its antecedent $\mathbf{p}_a^r$, where a stands for the antecedent of frame ℓ

In summary:

$$\mathbf{p}_\ell^r = \frac{\mathbf{p}_\ell^h - \mathbf{p}_a^h}{\|\mathbf{p}_\ell^h - \mathbf{p}_a^h\|} l_\ell^r + \mathbf{p}_a^r \qquad (1)$$

Due to the limited number of degrees of freedom (dof) in robots, there are points which can move in relation to each other on the human, but not on the robot. That is
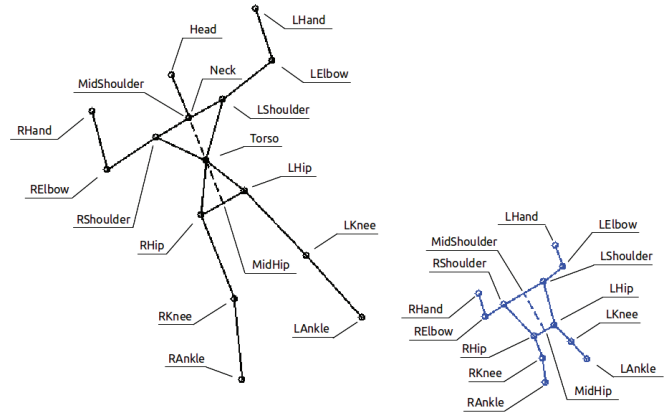

Fig. 1. Scaling human joint positions (black) to humanoid joint positions (blue). Example using the Kinect sensor and the NAO robot.

notable for the spine, for example. In case the robot torso is rigid, to maintain the distances connecting shoulders and hips should be constant after scaling, a segment which goes from the $MidHip$ to the $MidShoulder$ is scaled, preserving symmetry.

During online imitation, the scaling is performed at each time step. No previous knowledge of the human dimensions is needed as the joint positions are used directly. Therefore, the scaling works for any actor and is not affected by segment length variations in the motion capture system. The scaled motion does not respect the robot limits or ensures balance. It is used to find a reference motion fit to the robot dimensions.

B. Task specification

The robot redundancy will be used to specify several constraints in its motion. *Task specification* (or *Task classification*, or *Task prioritization*) allows adding terms to the IK forcing the robot configurations into desired ones or minimizing additional terms [11]. Let us describe a strict task j by an equality constraint equation:

$$\dot{\mathbf{q}} = \mathbf{J}_j^+ \dot{\mathbf{X}}_j, \qquad (2)$$

and a minimization task k by an optimization constraint equation:

$$\dot{\mathbf{q}} = \kappa_k \nabla_q f_k(\mathbf{q}). \qquad (3)$$

where $\mathbf{J}_j$ denotes the Jacobian matrix related to task $\mathbf{X}_j$; κ_k is a weight tuned according to the task importance and $\nabla_q f_k(\mathbf{q})$ is the gradient of function f_k with respect to the joint angle vector $\mathbf{q}$. The iterative process considering the N equality constraints associated to their respective priority is the following, for $j = 0..N$.

$$\begin{aligned} \dot{\mathbf{q}}_0 &= 0 \\ \dot{\mathbf{q}}_{j+1} &= \dot{\mathbf{q}}_j + (\mathbf{J}_{j+1}\mathbf{P}_j^a)^+(\dot{\mathbf{X}}_{j+1} - \mathbf{J}_{j+1}\dot{\mathbf{q}}_j) \end{aligned} \qquad (4)$$

where $\dot{\mathbf{q}}_j$ is the joint velocities vector realizing strict tasks 0 to j. $\mathbf{P}_j$ and $\mathbf{P}_j^a$ are the projectors on the kernels of the task Jacobian matrices $\mathbf{J}_j$ and $\mathbf{J}_j^a$ respectively:

$$\begin{aligned} \mathbf{P}_j &= \mathbf{I} - \mathbf{J}_j^+ \mathbf{J}_j \\ \mathbf{P}_j^a &= \mathbf{I} - (\mathbf{J}_j^a)^+ \mathbf{J}_j^a \end{aligned} \qquad (5)$$

I being the identity matrix. $\mathbf{J}_j^a$ denotes the augmented Jacobian matrix $\mathbf{J}_j^a$ defined as the concatenation of matrices $\mathbf{J}_1$ to $\mathbf{J}_j$. Introducing M optimization constraints, we obtain the following equation.

$$\dot{\mathbf{q}}_M = \dot{\mathbf{q}}_N + \mathbf{P}_N^a \sum_{k=1}^{M} \kappa_k \nabla_q f_k(\mathbf{q}) \tag{6}$$

As many tasks as wished can be added as equality or optimization constraints, as long as the robot's redundancy is sufficient. Some tasks related to humanoid imitation of human motion are further described.

1) Cartesian trajectory tracking: The task vector $\mathbf{X}_t$ consists of the scaled poses for the robot end effectors. The vector contains at most 6 coordinates (3 translations, 3 rotations) for each tracked effector. Typical effectors are hands, feet, head and waist, but in fact any other frames in the human body can also be tracked. Denoting the length of $\mathbf{q}$ (the robot dof) as n^r, for m_t tracked coordinates, the dimension of Jacobian $\mathbf{J}_t$ is $n^r \times m_t$.

2) Keeping balance: The task vector $\mathbf{X}_c' = (x_c \ \ y_c)^t$ contains the absolute position of the center of mass (CoM) projected on the horizontal plane. $\mathbf{J}_c$ denotes the $n^r \times 2$ Jacobian matrix transforming the CoM velocity vector into the joint velocity vector. In this approach, we have constrained the CoM projection to remain superposed to a fixed reference point on the robot sole.

3) Avoiding robot joints limits: Avoiding robot joints limits is very important to perform efficient imitation, because if the solution surpasses physical limits, the balance may not be met. This criterion is defined by minimization:

$$f_\ell = \sum_{i=1}^{n^r} \left(\frac{q^r(i) - \bar{q}^r(i)}{q_{\max}^r(i) - q_{\min}^r(i)} \right)^2 \tag{7}$$

where $q^r(i)$ is the i-th component of vector $\mathbf{q}^r$, $\bar{q}^r = \frac{1}{2}(q_{\max}^r + q_{\min}^r)$, and $q_{\max}^r$ and $q_{\min}^r$ denote respectively the maximum and minimum joint limits.

To ensure that the values will be fit into the allowed range, a clamping loop [11] is added. After the IK including all tasks has been solved, the loop checks whether all joints are within their limits. For those which are not, their values are fixed to their limits and their respective columns in the Jacobians are zeroed. The IK is then solved again, with less dof. This is repeated until all joints fit their limits.

4) Tracking human joint positions: The robot tracks human joint values which correspond to its dof as an optimization constraint. Here, $\mathbf{q}^{h'}$ is the human generalized position vector matching the size and the joints of $\mathbf{q}^r$.

$$f_h = \sum_{i=1}^{n^r} (q^r(i) - q^{h'}(i))^2 \tag{8}$$

III. CHANGE OF SUPPORT

A. Scaling different support phases

To deal with support changes, the scaling process is adapted to each support phase (right RS, left LS or double support DS). A foot is considered to be in support if its

vertical distance from the ground is lower than a given threshold. For the single support phases (right or left), the data is scaled as described in section II.A, beginning by the support foot and moving toward each end-effector as a tree structure.

During double support (DS), it is important to ensure that the tracked trajectories for both feet are at ground level. To that end, both feet heights are fixed beforehand even if the data from the motion capture system shows a difference between the reference feet heights. The closed loop formed by the legs is scaled from the point between the ankles to that between the hips without going through the knees. From the hips onwards the scaling is the same as for single support. An example for the scaling of each support is seen in Fig. 2.

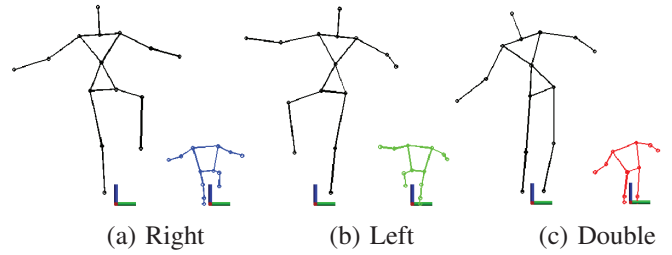(a) Right   (b) Left   (c) Double

Fig. 2.   Scaling human joint positions to humanoid joint positions for different support phases.

When the human lifts a foot beyond the support threshold, the scaling changes from DS scaling to a single support scaling. Conversely, when the human places a foot on the floor, the scaling fixes the foot on ground level before scaling other joints. The horizontal coordinates of the robot foot are chosen proportionally to the position of the human foot, fitting into an area on the floor where the robot is able to place the foot flat. This area is determined beforehand experimentally.

B. Tracked coordinates for various supports

To reduce the number of tracked coordinates, the robot is modeled with one foot fixed to the world frame (implicit constraint). For single support, the support foot frame is taken as the origin. The scaled data is transformed to this frame before the IK is solved. The reference position for the CoM projection is under the ankle of the support foot, to minimize the torque needed on the ankle.

For double support, the robot is also modeled with one foot fixed to the world frame. Let us choose the right foot to be the origin. In this case, it is necessary to ensure that the left foot is flat on the same plane as the right foot. Therefore, although during single support it would be fine to track only the left foot position, for example, during double support 6 dof must be tracked to ensure both feet are flat on the ground. As for the projection of the CoM, it is set to the point between the two feet during double support.

The transition between single and double supports happens in 2 steps. When a foot is to be taken off, first the projection of the CoM is moved to the other foot, then the foot is

lifted parallel to the ground. When a foot is to be landed, the steps go in an opposite order: first the swing foot is lowered parallel to the ground, then the projection of the CoM is brought back to between the feet.

To increase stability while a foot is being lifted or lowered, the CoM goal is set not to the projection of the ankle, but to a point closer to the center of the foot. To facilitate the transition, the number of constraints on the movement can be reduced by setting all effectors free except for the foot.

IV. EXPERIMENTATIONS

A. Setting

The humanoid robot NAO (Aldebaran Robotics) was used for the experimental validation. This small robot only has 23 dof for its body, when not considering the open and close of the hands. This considerably limits the number of tasks possible in the stack as the redundancy order remains low.

The 3D human data were captured using a Microsoft Kinect sensor. This sensor tracks 15 points in the human body, previously shown in Fig. 1. Although the data are quite noisy, no filtering is being performed, not to waste imitation time. To avoid the legs shaking during DS, both feet positions are kept the same as the previous time step. A foot is considered to be in support if the data received is below the vertical threshold of 200 mm. This is a large number in order to avoid false positives due to jittering.

Two programs were developed in C++ and run at the same time. One program acquires the motion capture data from the Kinect at 60 Hz, detects the type of support and scales the motion accordingly. The other receives the scaled data, calculates the IK and sends the results to the robot via *wi-fi*.

In total, four tasks are performed. The two equality tasks are: keeping balance with the highest priority, followed by Cartesian tracking. The two optimization tasks, avoiding limits and joint space tracking, are projected into the kernel of the previous tasks and thus have a lower priority. A weight five times larger was given to the limits task than to the joints task ($\kappa_l = -0.10, \kappa_h = -0.02$). A clamping loop was also added. A time step consists of computation time plus motion time. The robot speed was set to 10% of its maximum speed in order to avoid high accelerations, which is the main time constraint of the system.

B. Tracked coordinates

Two robot models were implemented using modified DH parameters [12], one based on the right foot (the *RFoot* model, used for RS and DS phases), and one based on the left foot (*LFoot* model, for LS). All models are described with the z axis pointing upwards in the vertical and the x axis pointing to the robot front. The tracked coordinates $\mathbf{X}_t$ and $\mathbf{X}'_c$ vary according to the support phase and the step during transition. A summary of all tracked coordinates is shown in Tab. I. The referred frames are detailed in Fig. 3.

Since one foot is always attached to the absolute frame, at most 3 effectors are being tracked at the same time: right hand, left hand and the free ankle. Due to the limited number of dof in the NAO robot, the effectors orientations
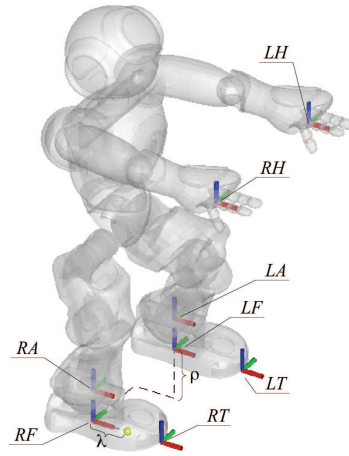

Fig. 3.   Frames and offsets tracked during online imitation

are not tracked unless it is necessary to place a foot flat on the ground. Thus, only the 3 Cartesian coordinates for each effector are tracked during single support phases.

During DS, the left foot must be kept flat on the ground. Experimentally, it was noted that the IK cannot come to a solution respecting balance and feet yaw (orientation about the vertical axis) constraints at the same time, especially because NAO legs have a total of 11 dof together, and only one of these dof (the pelvis joint) affects the yaw of the feet. This same dof also influences the orientation of the torso, greatly affecting the position of the CoM. Due to this kinematic limitation, the left foot yaw is not being tracked and only 5 dof are being constrained: 3 coordinates of the left ankle ($\mathbf{p}_{LA}$) and the vertical height of two points on the sole (z_{LF}, z_{LT}).

During transitions, the hands are left free and only the 5 dof for the swing foot are being tracked. During CoM placing tasks, the feet are maintained flat on the ground with the vertical coordinates $z_\ell = 0$ and the CoM is placed with an offset $\lambda = 15mm$. The transitions into single support finish when the swing foot is lifted to a height $\rho = 30mm$.

C. Poses

The system was tested by several actors performing a wide range of slow motions. The robot was able to track the human motion in time and space while keeping balance during all support phases. Support transitions happened smoothly, taking a minimum of 3 s and a maximum of 12 s to be finalized. The average time step took 420 ms (2.4 Hz), 10% of this time spent solving the IK and 90% spent moving.

D. Cartesian tracking

Fig. 5 shows the imitation online for a DS-RS-DS motion. In other words, a step forward. The tracking of end-effectors with respect to the absolute frame is shown in time in Fig. 6 and in space in Fig. 7. For the *DS* to *RS* transition, the robot takes a while to finally lift the foot from the floor, since it has to carefully move the *CoM* and lift the foot parallel to

| Support	Step	Model	$\mathbf{X}_t$	$\mathbf{X}'_c$
Continuous support				
RS	RFoot	$\mathbf{X}_t^{(9\times1)} = [\mathbf{p}_{LH}^{(3\times1)}, \mathbf{p}_{RH}^{(3\times1)}, \mathbf{p}_{LA}^{(3\times1)}]^T$	$\mathbf{X}_c'^{(2\times1)} = [0,0]^T$
LS	LFoot	$\mathbf{X}_t^{(9\times1)} = [\mathbf{p}_{LH}^{(3\times1)}, \mathbf{p}_{RH}^{(3\times1)}, \mathbf{p}_{RA}^{(3\times1)}]^T$	$\mathbf{X}_c'^{(2\times1)} = [0,0]^T$
DS	RFoot	$\mathbf{X}_t^{(11\times1)} = [\mathbf{p}_{LH}^{(3\times1)}, \mathbf{p}_{RH}^{(3\times1)}, \mathbf{p}_{LA}^{(3\times1)}, z_{LF}=0, z_{LT}=0]^T$	$\mathbf{X}_c'^{(2\times1)} = [\mathbf{x}_{LA}/2, \mathbf{y}_{LA}/2]^T$
Support transition				
DS to RS	CoM to *RFoot*	RFoot	$\mathbf{X}_t^{(5\times1)} = [\mathbf{p}_{LA}^{(3\times1)}, z_{LF}=0, z_{LT}=0]^T$	$\mathbf{X}_c'^{(2\times1)} = [\lambda, 0]$
DS to RS	Lift *LFoot*	RFoot	$\mathbf{X}_t^{(5\times1)} = [\mathbf{p}_{LA}^{(3\times1)}, z_{LF}=\rho, z_{LT}=\rho]^T$	$\mathbf{X}_c'^{(2\times1)} = [\lambda, 0]$
DS to LS	CoM to *LFoot*	RFoot	$\mathbf{X}_t^{(5\times1)} = [\mathbf{p}_{LA}^{(3\times1)}, z_{LF}=0, z_{LT}=0]^T$	$\mathbf{X}_c'^{(2\times1)} = [\mathbf{x}_{LA}, \mathbf{y}_{LA}]^T$
DS to LS	Lift *RFoot*	LFoot	$\mathbf{X}_t^{(5\times1)} = [\mathbf{p}_{LA}^{(3\times1)}, z_{RF}=\rho, z_{RT}=\rho]^T$	$\mathbf{X}_c'^{(2\times1)} = [\lambda, 0]$
RS to DS	Lower *LFoot*	RFoot	$\mathbf{X}_t^{(5\times1)} = [\mathbf{p}_{LA}^{(3\times1)}, z_{LF}=0, z_{LT}=0]^T$	$\mathbf{X}_c'^{(2\times1)} = [\lambda, 0]$
RS to DS	CoM to middle	RFoot	$\mathbf{X}_t^{(5\times1)} = [\mathbf{p}_{LA}^{(3\times1)}, z_{LF}=0, z_{LT}=0]^T$	$\mathbf{X}_c'^{(2\times1)} = [\mathbf{x}_{LA}/2, \mathbf{y}_{LA}/2]^T$
LS to DS	Lower *RFoot*	LFoot	$\mathbf{X}_t^{(5\times1)} = [\mathbf{p}_{LA}^{(3\times1)}, z_{RF}=0, z_{RT}=0]^T$	$\mathbf{X}_c'^{(2\times1)} = [\lambda, 0]$
LS to DS	CoM to middle	RFoot	$\mathbf{X}_t^{(5\times1)} = [\mathbf{p}_{LA}^{(3\times1)}, z_{LF}=0, zz_{LT}=0]^T$	$\mathbf{X}_c'^{(2\times1)} = [\mathbf{x}_{LA}/2, \mathbf{y}_{LA}/2]^T$
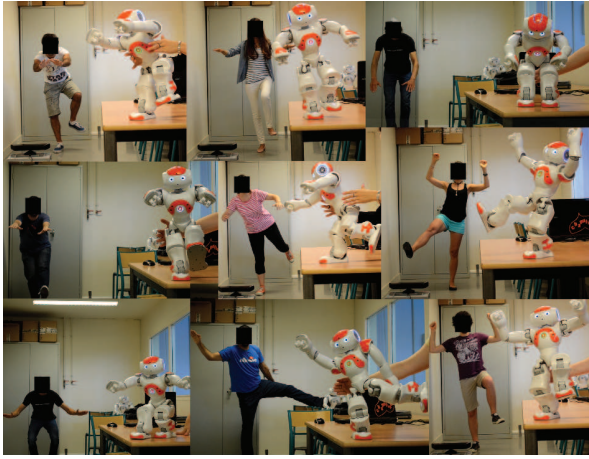
Fig. 4. Example of poses achievable during online imitation including support changes.

Fig. 6. Tracked Cartesian trajectories in time referent to the motion in Fig. 5

the ground. After that, the robot catches up with the actor's movement during *RS*.

During the transition from *RS* to *DS*, the foot was not initially placed on its final goal. Once the foot was already on the ground, during the following time steps in *DS*, the foot was correctly slid to the position it should be on the ground. This detour was allowed to improve convergence rates.

E. Balance

For the same DS-RS-DS movement, the projection of the *CoM* on the floor and the position of the support feet (when in contact with the floor) are plotted in Fig. 8. The movement of the CoM from in-between the feet to the single support foot and then back to the point between the feet is clearly seen. It is also possible to observe that the left foot was placed in a position 137 mm ahead of the initial one, performing a step forward.
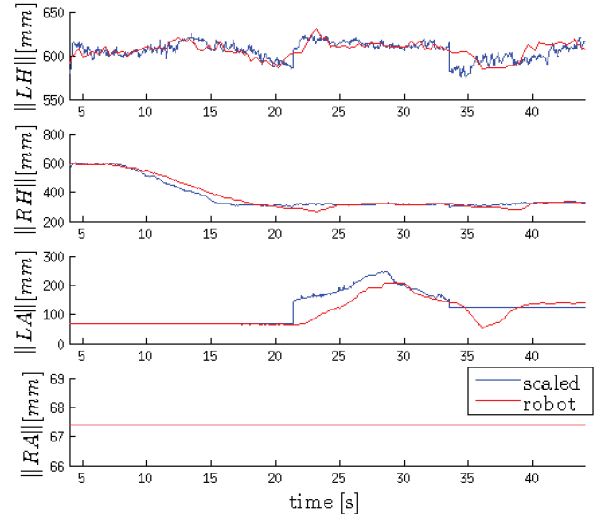
V. CONCLUSIONS

In this paper, a method to convert online the human change of support into humanoid robot motion was introduced. Taking a kinematic approach based on inverse kinematics with task specification, four tasks were performed: Cartesian tracking, keeping balance, avoiding joint limits and joints tracking. This method allows changing the support leg by altering the coordinates tracked for the effectors and for the center of mass at different support phases. To reinforce the limits avoidance, a clamping loop was added.

It was shown that a segment by segment scaling of the human motion to robot proportions is enough to define trajectories in the Cartesian space which maintain the overall posture throughout imitation and allows for precise support changes.

The method was validated using NAO robot and a Kinect motion capture system. Experiments with several performers
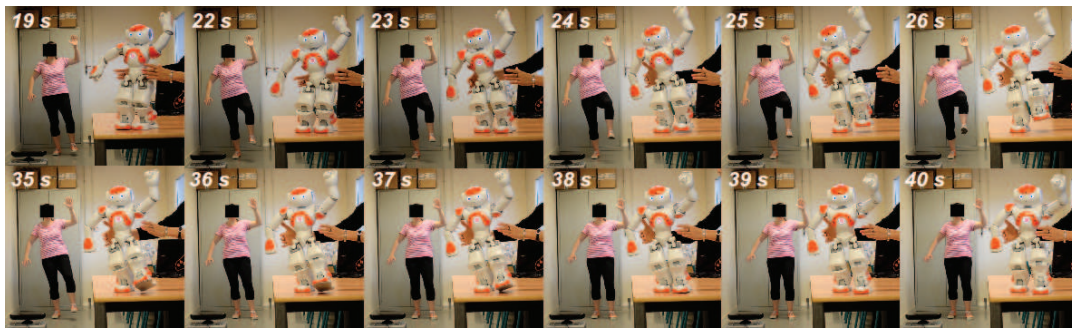
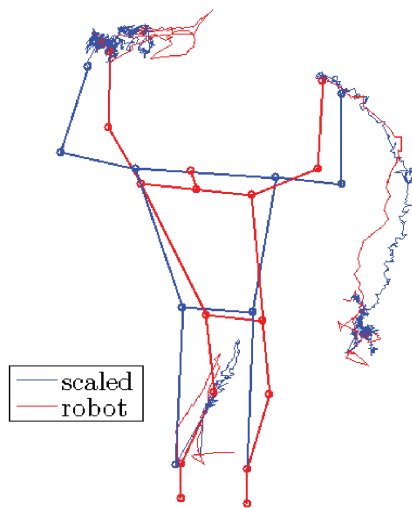Fig. 5. Online imitation changing support: double support to right support to double support.

Fig. 7. Tracked Cartesian trajectories in space referent to the motion in Fig. 5

Fig. 8. Tracked trajectories for the CoM projection referent to the motion in Fig. 5

showed satisfactory results for the imitation of a wide variety of support changes. The end-effectors are successfully tracked both in time and space. The same can be said for the projection of the CoM on the ground. Compared to other methods in the literature, the present approach better preserves the nuances of human motion during whole-body online teleoperation while working with an uncluttered motion capture system, thanks to the absence of motion primitives.

Future work will focus on tracking the ZMP to increase movements velocity and validating the method using other robotic platforms for generalization purpose.

REFERENCES

[1] M. Riley, A. Ude, and C. G. Atkeson, Methods for motion generation and interaction with a humanoid robot: Case studies of dancing and catching, in Proc. 2000 Workshop on Interactive Robotics and Entertainment, Robotics Inst., Carnegie Mellon Univ, pp. 35–42, 2000.
[2] S. Sakka, L. Penna Poubel, D. Cehajic and D. Creusot, Tasks prioritization for whole-body IK-based on-line imitation of human motion by humanoid robots, International Journal of Humanoid Robotics, *to appear*, 2014.
[3] N. Pollard, J. Hodgins, M. Riley, and C. Atkeson, Adapting human motion for the control of a humanoid robot, in Proceedings 2002 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1390–1397, IEEE, 2002.
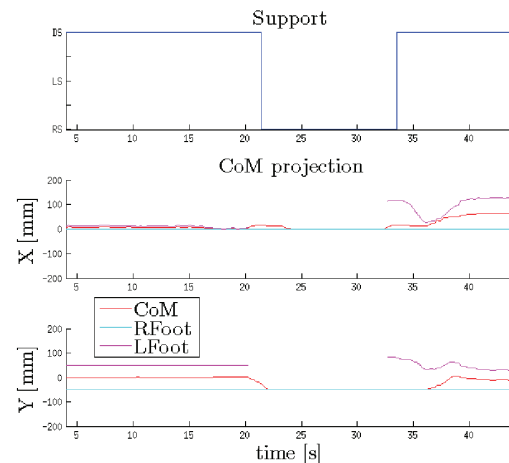[4] A. Safonova, N. Pollard, and J. Hodgins, Optimizing human motion for the control of a humanoid robot, 2nd International Symposium on Adaptive Motion of Animals and Machines (AMAM2003), 2003.
[5] M. Riley, A. Ude, K. Wade, and C. Atkeson, Enabling real-time full-body imitation: A natural way of transferring human movement to humanoids, Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on, vol. 2, pp. 2368–2374, 2003.
[6] F. Montecillo P., M. Sreenivasa, and J.-p. Laumond, On real-time whole-body human to humanoid motion transfer, International Conference on Informatics in Control, Automation and Robotics (ICINCO 2010), pp. 22–31, 2010.
[7] J. Koenemann and M. Bennewitz, Whole-body imitation of human motions with a nao humanoid, in Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction, HRI '12, (New York, NY, USA), pp. 425–426, ACM, 2012.
[8] B. Dariush, M. Gienger, B. Jian, C. Goerick, and K. Fujimura, Whole body humanoid control from human motion descriptors, IEEE International Conference on Robotics and Automation, ICRA 2008. pp. 2677–2684.
[9] A. Dasgupta and Y. Nakamura, Making feasible walking motion of humanoid robots from human motion capture data, in *ICRA*, pp. 1044–1049, 1999.
[10] S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, and K. Ikeuchi, Generating whole body motions for a biped humanoid robot from captured human dances, Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on, vol. 3, pp. 3905–3910, 2003.
[11] P. Baerlocher and R. Boulic, An inverse kinematic architecture enforcing an arbitrary number of strict priority levels, The Visual Computer, vol. 20, pp. 402–417, 2004.
[12] W. Khalil and E. Dombre, Modeling, Identification and Control of Robots. Bristol, PA, USA: Taylor & Francis, Inc., 3rd ed., 2002.