Task-Oriented Whole Body Motion for Humanoid Robots

Michael Gienger, Herbert Janßen, Christian Goerick

2005

Task-Oriented Whole Body Motion for Humanoid Robots

Michael Gienger, Herbert Janßen and Christian Goerick
Honda Research Institute Europe GmbH
63073 Offenbach / Main, Germany
Email: michael.gienger@honda-ri.de

***Abstract*— We present a whole body motion control algorithm for humanoid robots. It is based on the framework of *Liégeois* [10] and solves the redundant inverse kinematics problem on velocity level. We control the hand positions as well as the hand and head attitude. The attitude is described with a novel 2-dof description suited for symmetrical problems. Task-specific command elements can be assigned to the command vector at any time, such enabling the system to control one or multiple effectors and to seamlessly switch between such modes while generating a smooth, natural motion. Further, kinematic constraints can be assigned to individual degrees of freedom. The underlying kinematic model does not consider the leg joints explicitly. Nevertheless, the method can be used in combination with an independent balance or walking control system, such reducing the complexity of a complete system control. We show how to incorporate walking in this control scheme and present experimental results on *ASIMO*.**

***Index Terms*— Humanoid robot - Whole body motion - Redundant control**

I. Introduction

Whole body motion control experiences increasing popularity in the last few years. Originally a method coming from computer animation, the performance of today's state of the art computers allows a real-time implementation to control humanoid robots with many degrees of freedom.

For multi degrees-of-freedom robots, significant advantages arise due to the increased operation space and the higher robustness against singular configurations. Since such systems are often redundant with respect to the task, it is possible to separate the configuration space into a task space and an orthogonal null space. The null space can be exploited to satisfy additional criteria. Further, manipulation and balance control can be combined in a quite elegant way.

A comprehensive book on redundant control techniques has been published by *Nakamura* [13]. *Buss* [3] gives an overview on rigid body kinematics with tree-like structure. Particular emphasis is devoted to the inverse kinematics problem, addressing Jacobian transpose, pseudo-inverse and damped least squares methods. How to deal with ill-conditioned or singular mappings is shown by *Maciejewski* [11]. Pseudo-inverse method and extended Jacobian approach are compared and applied to a 30 dof robot by *Tevatia et al.* [15] . The balance problem is addressed in [2], [16], [18]. *Sian et al.* develop a teleoperation framework for their humanoid robot *HRP-2* that enables the remote operator to command different effectors. The contribution of individual joints to the motion is selected according to the distance of the target. A similar method has been developed by *Nishiwaki et al.* [14]. They propose a whole body motion generator that adaptively selects postures from a predefined set (kneeling, standing, etc.) in order to reach the target. An advanced foot step planning algorithm maintains balance while approaching the target.

Many works focus on exploiting the null space to satisfy further objectives. Joint limit and obstacle avoidance are addressed in [4]–[6], [12], [19]. *Baerlocher et al.* incorporate task prioritization for different criteria within the null space [1].

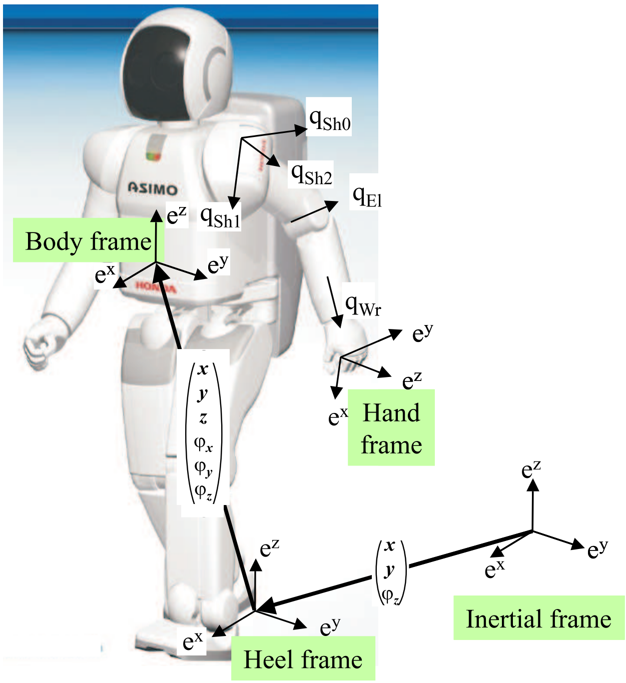


Figure 1: Kinematic model for whole body motion

While above mentioned works are based on resolved motion rate control, *Khatib* employs the *Operational Space Formulation* which describes the relationship between end effector accelerations and forces [9]. Similar to *Liégeois'* method, the problem is separated into a task- and null space. *Sentis et al.* extend this framework to prioritized objectives and include methods for joint limit avoidance [17].

We intend to use the proposed whole body motion control as a building block within a behavior system. It will correspond to the motor action part, while the perceptual information will be gained by visual, auditory, tactile and other sensors. The whole body control should process commands autonomously and in real-time, such that reactive control loops with multiple high level behaviors can be created. Therefore the commands should be composed of few parameters, such giving transparency to the high level tasks while leaving the complexity of the whole body interaction to the motion control.

We propose a velocity-based whole body motion algorithm that is based on *Liégeois'* method. The underlying mathematical model and task description of our system will be presented. We then propose an adaptive way to assemble the system equations according to a time-variant task description. Further, a method to adaptively modify constraints for individual degrees of freedom will be explained. The null space is utilized for joint limit avoidance. The characteristics of the underlying objective functions are discussed. We show how the balance criteria can be incorporated in our framework, such having a computationally efficient solution by separating the control problem in two parts: Whole body and balance control. A brief example illustrates the applicability of our method with respect to planning and prediction. We finally present experimental results.

II. Kinematic Model

The kinematic model of the robot is depicted in figure 1. In initial configuration, the x-axis points forward, the z-axis points upward and the y-axis accordingly to the left. Pan, roll and tilt describe a rotation about the z-, x- and y-axis, respectively. The kinematic parameters and degrees of freedom of the links correspond to those of the humanoid robot *ASIMO* [7], [8]. The first link corresponds to the heel coordinate system comprising three degrees of freedom. It is centered between the feet and aligned with the heel edge. Its degrees of freedom are translations in the x- and y-direction as well as a rotation about the z-axis. The consecutive links correspond to the body segments of the robot. The upper body is undergoing three translations and Kardan rotations with respect to the heel frame. The head is connected to the upper body with pan and tilt joints. Further, the two arms comprise 5 dof each. Three joints drive the shoulder, one joint is located in the elbow, and the hand rotates about the forearm axis. An additional coordinate system with some offset to the hand origin defines a hand reference point. All together, the model comprises 21 dof.

Shifting and rotating the pelvis will result in a one-to-one mapping onto the leg joints. This mapping is implemented within the leg and balance control which is a separate, independent process. Therefore, the leg joints are not explicitly included in the model, they are accounted for by the upper body degrees of freedom. The state vector such comprises

$$\boldsymbol{q} = \begin{pmatrix} ({}_I x_{hl} \ {}_I y_{hl} \ {}_I \varphi_{z,hl})^T \\ {}_{hl}\boldsymbol{x}_{ub} \\ {}_{hl}\boldsymbol{\varphi}_{ub} \\ \boldsymbol{\varphi}_{arm,L} \\ \boldsymbol{\varphi}_{arm,R} \\ (\varphi_{pan} \ \varphi_{tilt})^T \end{pmatrix} \tag{1}$$

with indices I for the inertial frame, *hl* for heel and *ub* for upper body being the coordinate system in which the respective vector is represented.

III. Task description

The link rotation matrices and origin vectors can be computed according to the following recursive scheme, starting out with a unity matrix representing the inertial (world) coordinate frame. Hereby, the indices of a matrix $\boldsymbol{A}_{CP}$ denote a rotation from the "P"-frame in the "C"-frame.

$$\boldsymbol{A}_{CI} = \boldsymbol{A}_{CP} \ \boldsymbol{A}_{PI} \tag{2}$$

Index C stands for the current, P for the previous and I for the inertial coordinate system. The link origin vectors then compute as

$$_I\boldsymbol{r}_C = {}_I\boldsymbol{r}_P + \boldsymbol{A}_{PI}^T \ {}_P\boldsymbol{r}_{PC} \quad . \tag{3}$$

Vector $_P\boldsymbol{r}_{PC}$ describes the displacement of the current coordinate frame with respect to the previous link origin. In the coordinates of the previous link, it is constant.

The task vector is composed of three subsets: left hand, right hand and head motion. The hand tip position is described in cartesian coordinates x, y and z with respect to the world or heel coordinate frame. It corresponds to the above derived link origin vector of the respective hand reference point.

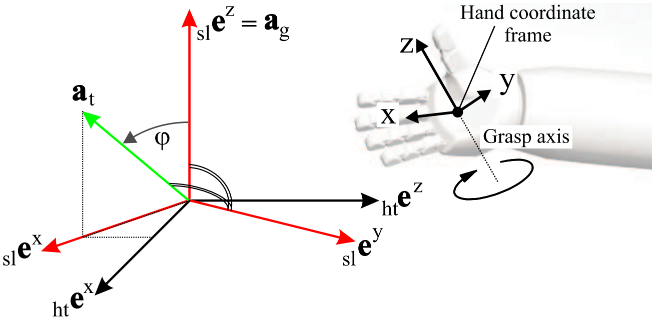


Figure 2: Two-dimensional hand attitude representation

Mostly, spacial orientations are described in 3 dimensions, e. g. by means of Kardan or Euler angles. Often, this is not required. This is particularly the case for grasping or manipulating cylindrical objects, or for the head motion, if a gaze direction is given. In such cases, the direction of the axis should be constrained, but the rotation of the end effector about this axis may be unconstrained. Such a description will decrease the dimensionality of the task, the gained degree of freedom can effectively be utilized for joint limit avoidance, singularity avoidance and further criteria. We therefore propose to control such an axis. In the following, it will be referred to as the grasp axis $\boldsymbol{a}_g$. Indices *sh*, *h* and *ht* denote shoulder, hand and hand reference point, respectively. With *l* and *r* specifying left and right, the task vector is

$$\boldsymbol{x}_{task} = \left(\boldsymbol{x}_{ht,l}^T \; \boldsymbol{a}_{g,ht,r}^T \; \boldsymbol{x}_{ht,r}^T \; \boldsymbol{a}_{g,ht,r}^T \; \boldsymbol{a}_{head}^T\right)^T \quad . \tag{4}$$

The trajectories are generated in these coordinates with a speed-limited higher order low pass filter between actual and target values.

IV. Inverse Kinematics

To map the task space magnitudes into joint space, we employ the framework ”redundancy resolution” that has first been proposed by *Liégeois* [10] for redundant systems. With a given task Jacobian $\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}}$, the state motion rates can be computed as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\#}\,\dot{\boldsymbol{x}} - \alpha\,\boldsymbol{N}\boldsymbol{W}^{-1}\left(\frac{\partial H}{\partial \boldsymbol{q}}\right)^T \tag{5}$$

where $\boldsymbol{J}^{\#}$ is a weighted generalized pseudo-inverse of $\boldsymbol{J}$ with weighting matrix $\boldsymbol{W}$

$$\boldsymbol{J}^{\#} = \boldsymbol{W}^{-1}\boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^T)^{-1} \tag{6}$$

and $\boldsymbol{H}$ is some optimization criterion. Its gradient is mapped into the null space with projection matrix $\boldsymbol{N}$ and scalar α defining the step width. Matrix $\boldsymbol{E}$ denotes the identity matrix.

$$\boldsymbol{N} = \boldsymbol{E} - \boldsymbol{J}^{\#}\boldsymbol{J} \tag{7}$$

A. Manipulator Jacobian

The manipulator Jacobian relates the effector linear and angular velocities to the state velocities.

$$\dot{\boldsymbol{x}}_{effector} = (\dot{\boldsymbol{x}}_{ht,l}^T \; \boldsymbol{\omega}_{ht,l}^T \; \dot{\boldsymbol{x}}_{ht,r}^T \; \boldsymbol{\omega}_{ht,r}^T \; \boldsymbol{\omega}_{head}^T)^T = \boldsymbol{J}_{MP}\;\dot{\boldsymbol{q}} \tag{8}$$

The rotation Jacobian of the upper body maps the joint rates on the upper body angular velocities. Index T denotes the linear, R the rotational velocity mapping.

$$\begin{pmatrix} {}_I\dot{\boldsymbol{x}}_{ub} \\ {}_I\boldsymbol{\omega}_{ub} \end{pmatrix} = \begin{pmatrix} {}_I\boldsymbol{J}_{T,ub} \\ {}_I\boldsymbol{J}_{R,ub} \end{pmatrix} \dot{\boldsymbol{q}} \tag{9}$$

The rotational Jacobian can be constructed from the individual link rotation matrices. The first two columns and column 4-6 are zero as they correspond to the translational velocities of the heel frame and upper body frame. Column 3 is ${}_I\boldsymbol{e}_z$ as this degree of freedom describes a rotation about the vertical axis. Matrix $\boldsymbol{A}_{ub-I}$ yields the Kardan rotation axes of the upper body.

$${}_I\boldsymbol{J}_R^{ub} = \left(\begin{array}{ccc|c|c} 0 & 0 & 0 & & \\ 0 & 0 & 0 & \boldsymbol{0}_{3\times3} & \boldsymbol{A}_{ub-I}^T \\ 0 & 0 & 1 & & \end{array}\right) \tag{10}$$

The Jacobian of the upper body linear velocity computes as

$${}_I\boldsymbol{J}_T^{ub} = \left(\begin{array}{cc|c|c|c} 1 & 0 & & & \\ 0 & 1 & {}_I\tilde{\boldsymbol{r}}_{hl-ub}^T\;\boldsymbol{J}_{R,col3} & \boldsymbol{A}_{hl-I}^T & \boldsymbol{0}_{3\times3} \\ 0 & 0 & & & \end{array}\right) \quad . \tag{11}$$

The tilde symbol denotes the matrix multiplication with a skew symmetric matrix $\tilde{\boldsymbol{r}}$ such that $\tilde{\boldsymbol{r}}\boldsymbol{x} = \boldsymbol{r} \times \boldsymbol{x}$. Columns one and two are x- and y- unity vectors as they yield the heel translational velocity elements. Column 3 accounts for the upper body velocity due to the rotation of the heel frame about the vertical axis. Columns 4-6 are the direction vectors of the upper body translation axis, and the remaining columns are zero as the rotation is described with respect to the upper body reference point.

The velocity of a hand reference point with respect to the shoulder can be expressed in state velocities. The superscripts *row dof* in equation (12) denote the row of the rotation matrix that corresponds to the respective link rotation axis.

$${}_I\boldsymbol{J}_R^{arm} = \begin{pmatrix} \boldsymbol{A}_{sh0-I}^{row\,dof} \\ \boldsymbol{A}_{sh1-I}^{row\,dof} \\ \vdots \\ \boldsymbol{A}_{ht-I}^{row\,dof} \end{pmatrix}^T \tag{12}$$

The translational speed of the reference points is the outer product of the axis and the distance vector between axis and end effector point.

$${}_I\boldsymbol{J}_T^{arm} = \begin{pmatrix} ({}_I\tilde{\boldsymbol{r}}_{sh0-ht}^T\;{}_I\boldsymbol{J}_{R,sh-ht}^{col\,0})^T \\ ({}_I\tilde{\boldsymbol{r}}_{sh1-ht}^T\;{}_I\boldsymbol{J}_{R,sh-ht}^{col\,1})^T \\ \vdots \\ ({}_I\tilde{\boldsymbol{r}}_{h-ht}^T\;{}_I\boldsymbol{J}_{R,sh-ht}^{col\,n})^T \end{pmatrix}^T \tag{13}$$

The rotation Jacobian for the head rotation computes accordingly. Skipping the left subscripts I for brevity, we can assemble the manipulator Jacobian:

$$\boldsymbol{J}_{MP} = \begin{pmatrix} \boldsymbol{J}_T^{ub} + {}_I\tilde{\boldsymbol{r}}_{ub-Ht,r}^T\;\boldsymbol{J}_R^{ub} | & \boldsymbol{J}_T^{arm,l} & | \; \boldsymbol{0}_{3\times7} \\ \boldsymbol{J}_R^{ub} & | \; \boldsymbol{J}_R^{arm,l} & | \; \boldsymbol{0}_{3\times7} \\ \boldsymbol{J}_T^{ub} + {}_I\tilde{\boldsymbol{r}}_{ub-Ht,r}^T\;\boldsymbol{J}_R^{ub} | & \boldsymbol{0}_{3\times5} \; | \; \boldsymbol{J}_T^{arm,r} & |\boldsymbol{0}_{3\times2} \\ \boldsymbol{J}_R^{ub} \; | & \boldsymbol{0}_{3\times5} \; | \; \boldsymbol{J}_R^{arm,r} & | \; \boldsymbol{0}_{3\times2} \\ \boldsymbol{J}_R^{ub} \; | & \boldsymbol{0}_{3\times10} & | \; \boldsymbol{J}_R^{head} \end{pmatrix} \tag{14}$$

B. Task-specific Jacobian

The linear hand velocities now can directly be computed with the respective rows of equation (14). The rotational Jacobian for a 3-dof attitude can be obtained by projecting the effector angular velocities on a suitable description, e. g. on Kardan or Euler angles. It is straight forward to rotate the respective rows of eq. (14) on the Kardan or Euler rates with the well-known projection matrices.

To compute the Jacobian of the grasp axes, some minor modifications have to be done. According to figure 2, vector $\boldsymbol{a}_g$ corresponds to the z-axis of the hand-fixed coordinate system. As the rows of the rotation matrix $\boldsymbol{A}_{h-I}$ contain the unit vectors of the hand-frame, represented in the I-frame, the current grasp axis $\boldsymbol{a}_g$ can be extracted from the third row of this projection: ${}_I\boldsymbol{a}_g = (\boldsymbol{A}_{h-I}^{row\,3})^T$. Vector $\boldsymbol{a}_g$ and the target grasp axis $\boldsymbol{a}_t$ span a plane and enclose angle φ.

$$\varphi = \arccos(\frac{\boldsymbol{a}_g \circ \boldsymbol{a}_t}{|\boldsymbol{a}_g||\boldsymbol{a}_t|}) \tag{15}$$

The current grasp axis shall move in this plane, the end point of the normalized grasp axis describes the shortest path on a unit sphere. We now construct an intermediate coordinate system $\boldsymbol{A}_{sl-I}$ in which the z-axis corresponds to the current grasp axis. The y-axis shall be orthogonal to the motion plane and such defines the rotation axis of angle φ.

$${}_{sl}\boldsymbol{e}^y = \frac{{}_I\boldsymbol{a}_g \times {}_I\boldsymbol{a}_t}{|\boldsymbol{a}_g \times \boldsymbol{a}_t|} \tag{16}$$

Note that the plane will degenerate if $\boldsymbol{a}_g$ and $\boldsymbol{a}_t$ coincide. Vector ${}_{sl}\boldsymbol{e}^x$ is perpendicular to y- and z-axis following the right-hand-rule. The angular velocity $\dot{\varphi}$ between current and target grasp axis can be projected onto ${}_I\boldsymbol{\omega}$ employing the above derived *sl*-frame. Specifiers *l* and *r* are skipped for brevity.

$${}_I\boldsymbol{\omega} = \ \boldsymbol{A}_{sl-I}^T \begin{pmatrix} 0 \\ \dot{\varphi} \\ 0 \end{pmatrix} = {}_I\boldsymbol{J}_{R,MP}^{ht}\ \dot{\boldsymbol{q}} \tag{17}$$

The hand rotation about the z-axis of the *sl*-frame does not change the angle φ. Therefore, the rotation Jacobian of the respective effector is rotated into the *sl*-frame, and the third row is skipped.

$$\begin{pmatrix} 0 \\ \dot{\varphi} \end{pmatrix} = \underbrace{\left\{ \boldsymbol{A}_{sl-I}\ {}_I\boldsymbol{J}_{R,MP}^{ht} \right\}_{row\,1,2}}_{\boldsymbol{J}_{grasp}} \dot{\boldsymbol{q}} \quad . \tag{18}$$

The same scheme is applied to the description of the head attitude. We map the head angular velocity on the view axis velocity of the camera system.

C. Setting up the Equation System

Often not all end effectors are to be controlled at a time. We therefore propose to adaptively assemble the Jacobian with respect to the desired command. According to the task vector, we command a binary vector $\boldsymbol{b}_J$ that denotes if a task vector element is active ("1") or passive ("0"). Employing this vector, the Jacobian now will be assembled row by row such that only rows of active task vector elements are incorporated. If all elements equal one, the Jacobian describes the mapping of the largest possible task command, comprising both hands position and grasp axes as well as the head view axis. Having e. g. zero elements for the grasp axes will lead to a mapping that only controls the hand positions, but not their attitude.

The pseudo-inverse $\boldsymbol{J}^\#$ and the null space projection $\boldsymbol{N}$ then can be computed by means of equations (6) and (7). We chose a diagonal weighting matrix $\boldsymbol{W}$. Its elements correspond to the work range of the individual degrees of freedom. Such, each joint will contribute to the overall motion according to its available motion range.

D. Constraining Individual Degrees of Freedom

In many cases, it is useful to constrain individual degrees of freedom to remain fixed. One example is to "freeze" the position and orientation of the heel, when the robot should stand on a fixed position. This can be done by constraining the heel dofs. Unconstraining the heel dofs will allow the heel coordinate frame to "slide" on the ground. This can effectively be used for real-time planning of how to optimally approach an object. For this reason, we command a binary vector $\boldsymbol{b}_C$ with the dimension of the state vector. Each element of this vector denotes if the respective state vector element is fixed ("0") or free to move ("1"). If all elements equal one, the motion is generated by utilizing all dofs. A zero-element will eliminate the respective dof, it will not be moved. To realize this, the respective column of the Jacobian has to be set to zero. This modified Jacobian will lead to a null space projection matrix in which the respective column and row is zero except for the diagonal elements which will equal 1. These diagonal elements also have to be set to zero in order to comply with the imposed constraints.

E. Null Space Criteria

We implemented and qualitatively compared two cost functions to exploit the null space: The "squared sum of the joint center deviations" (see e. g. [19]) and a “cubic sum of joint center deviations” which is zero in some region about the joint center. This zero region has also been proposed in [12] as "activation threshold". The gradients for a single element are depicted in figure 3.

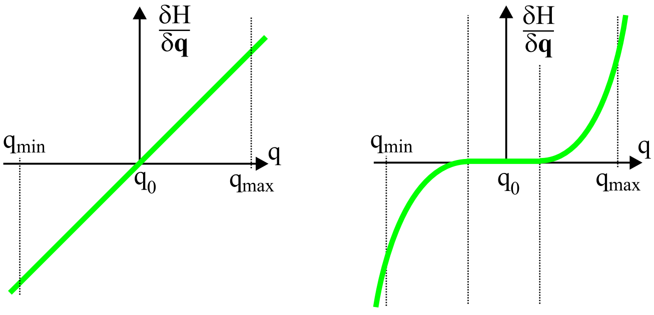


Figure 3:
Left: Gradient of "squared sum of the joint center deviations"
Right: Gradient of "cubic penalty with activation threshold"

Both cost functions show a good performance in avoiding joint limits. The latter one assigns higher motion rates to the boundary-close joints since they are weighted more heavily compared to the center-closed joints. Further, the “zero penalty region” leads to some hysteresis in the motion, which adds a human-like characteristic to the motion. We therefore favor the cubic penalty approach.

Having excessively large elements in the gradient vector might lead to oscillations and instability. We deal with this problem by scaling down the null space velocities such that the permissable joint speeds are not violated.

It has to be stated that these strategies do help, but do not guarantee that the limits are not violated. In such cases, the joint angles are clamped to their limit value.

When the constraints or task vector elements are modified during the motion, the structure of the Jacobian changes and instantaneous velocity jumps on joint level can arise. This is particularly the case if constrained dofs or many task elements are released, as large gradients will be mapped on the null space velocities. To deal with this, we apply a jerk limit for the null space motion rates. If this limit is exceeded, the null space velocity vector is scaled down accordingly.

Figure 4 shows the smooth progression of the arm joint angles in such switching situations. In the upper diagram, no task is specified in the white region, while the arm z position is moved to a high vertical target in the grey region. The lower diagram shows switching from the task ”hand z position” (white) to controlling the hand position and attitude with all 5 task elements. It can be seen that there is a smooth transition in the joint angles.

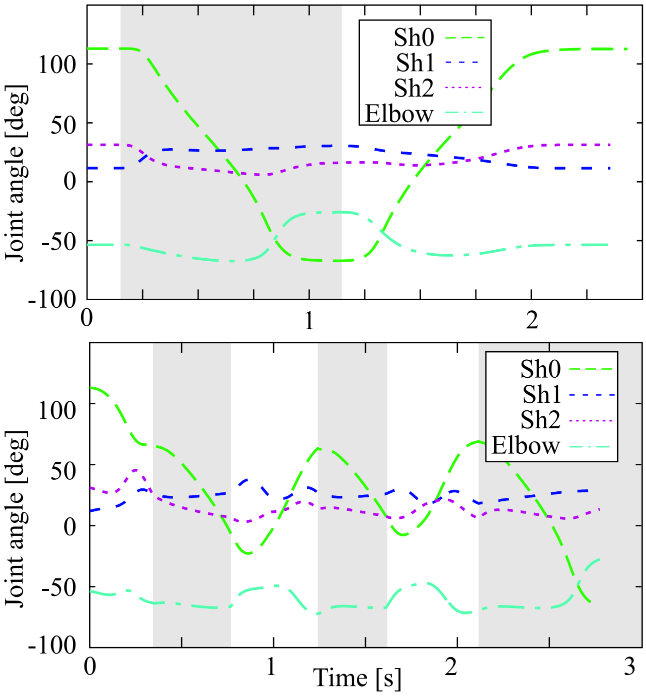


Figure 4: Joint angles under different task switching situations

V. Incorporating Balance Control

The controller as presented did not yet consider the constraints that are required to maintain balance during standing and walking. As mentioned in the introduction, these aspects are not handled within this framework, but rather by a separate walking and balancing controller in *ASIMO* [7], [8].

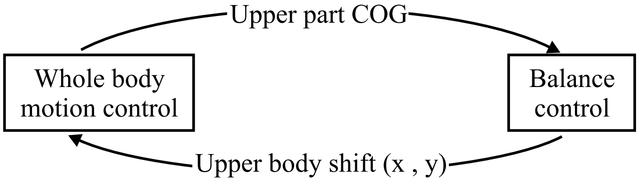


Figure 5: Separation of balance- and whole body posture control

The motion generated by the whole body motion controller will lead to a displacement of the overall center of gravity. This displacement is computed and interpreted by the balance controller as a disturbance. We will now assume that the balance control is computing some upper body shift in x- and y-direction to control the balance. This is the case for our system, and it also seems to be the standard way according to literature. Therefore, kinematic constraints are imposed on the upper body x- and y-dofs according to the above description, such being excluded from the whole body motion generation.

Given that the task vector contains elements that depend on the upper body shift, the task velocity vector is split up into a whole body motion part (Index wbm) and a balance part (Index bal).

$$ {}_I\dot{\boldsymbol{x}}_{task} = \underbrace{\begin{pmatrix} {}_I\dot{x}_{wbm,x} \\ {}_I\dot{x}_{wbm,y} \\ {}_I\dot{x}_{wbm,z} \\ \dot{\boldsymbol{\varphi}}_{task} \\ \vdots \end{pmatrix}}_{\dot{\boldsymbol{x}}_{wbm}} + \underbrace{\begin{pmatrix} {}_I\dot{x}_{bal,x} \\ {}_I\dot{x}_{bal,y} \\ 0 \\ 0 \\ \vdots \end{pmatrix}}_{\dot{\boldsymbol{x}}_{bal}} \tag{19} $$

Now we account for the upper body shift by incorporating its velocity $\dot{\boldsymbol{x}}_{bal}$

$$ \dot{\boldsymbol{x}}_{task} = \boldsymbol{J}\,\dot{\boldsymbol{q}} + \dot{\boldsymbol{x}}_{bal} \tag{20} $$

and solve for the joint motion rates according to eq. (5).

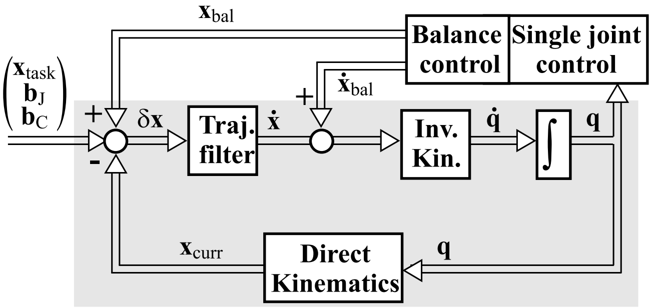
$$ \dot{\boldsymbol{q}} = \boldsymbol{J}^{\#}\left(\dot{\boldsymbol{x}}_{task} - \dot{\boldsymbol{x}}_{bal}\right) - \alpha \boldsymbol{N}\boldsymbol{W}^{-1}\left(\frac{\partial H}{\partial \boldsymbol{q}}\right)^{T} \tag{21} $$

Since the upper body shift is considered in the inverse kinematics computation, the commanded magnitudes will be tracked very accurately. Stability problems are unlikely to be encountered, since a large motion of an end effector will only lead to a small displacement of the overall center of gravity. This displacement is effectively compensated by the balance control. Figure 6 shows the control scheme in a block diagram.

VI. Incorporating Walking

In the following, we will show how the method can be utilized for real-time planning. This example illustrates how to solve the problem of grasping an object that is out of range.

First, some criterion that is used to trigger walking has to be defined. This could e. g. be the violation of a joint limit, reaching a threshold in the null space cost function, or some virtual force acting on the system.



$\mathbf{x}_{\text{bal}}$
Balance control
Single joint control
$\begin{pmatrix}\mathbf{x}_{\text{task}}\\ \mathbf{b}_J\\ \mathbf{b}_C\end{pmatrix}$
+
$\dot{\mathbf{x}}_{\text{bal}}$
+
$\delta\mathbf{x}$
Traj. filter
$\dot{\mathbf{x}}$
Inv. Kin.
$\dot{\mathbf{q}}$
$\int$
$\mathbf{q}$
-
$\mathbf{x}_{\text{curr}}$
Direct Kinematics
$\mathbf{q}$

Figure 6: Block diagram of the control structure

When a command is issued, the controller will initially start to move the effector towards the target while the robot is standing.

If the chosen success criterion is violated, walking will be started. Hereby, the kinematic constraints on the heel coordinate frame will be released. In our model, this leads to a ”floating” heel basis that will converge to a position and orientation that is optimal with respect to the null space criterion. At the same time, the robot is commanded to walk to the ”floating” heel frame. Due to the released heel constraints, the hands will automatically move to a more “relaxed” posture according to the null space criterion. It is also possible to control the hands with another strategy while walking, e. g. with an arm swinging motion.

We employ a step generator that tracks the target heel position and orientation. Once this has been reached, the walking motion will be stopped, the kinematic heel constraints will be activated and the robot continues to grasp the object.

This type of planning is very general as it can be applied to arbitrarily composed task vectors. The resulting heel frame position and orientation will always be a local optimum with respect to the null space criterion. Employing such a strategy allows to command the target in a very general way. Only the object position and attitude has to be commanded, while the whole body motion control will autonomously decide if it has to walk or not.

VII. Experiments

The control system has been tested extensively on ASIMO. For the presented experiments, the task has been commanded in a teleoperation-like manner by specifying the targets interactively.

In the upper picture series of figure 7, a target position for the right hand is specified. While the head and left arm joints remain at their center positions, the right arm joints as well as the upper body vertical position and attitude contribute to the task motion.

In the middle snapshot series, both hand positions and attitudes as well as the head attitude are controlled. While the hands are instructed not to move, the head view axis is moved from the left to the right side. In this case, the task vector yields 12 elements. This means that a 12×12-matrix needs to be inverted, which is the worst case concerning computational effort. The computation time is less than 1 ms on *ASIMO’s* CPU or a standard PC board. In the picture series, it can be very nicely seen how the upper body rotates about the vertical axis while the hands remain at fixed positions. Comparing the first and last picture also shows that the hands are slightly rotating about the grasp axis.

The bottom picture row shows the motion when just the left hand vertical position is commanded to reach for a high target. The hand motion in x- and y-direction now results from the ”joint limit avoidance” criterion that is optimized in the null space. It can be seen that the upper body starts to rotate as soon as the arm reaches an almost straight configuration. This somewhat looks similar to a human motion. Another apparent aspect is the higher robustness against singularities. Whereas the pure arm Jacobian would be very close to singularity due to the straight arm configuration, the whole body Jacobian is far more robust as the upper body rotation and z-position also contribute to the task motion.

VIII. Conclusion

The paper describes a flexible whole-body motion method for humanoid robots. We represent the end effector orientation with a 2-dimensional description suitable for problems with symmetry, e. g. grasping cylindrical objects or looking along a given view vector. The proposed scheme allows to assemble the kinematic equations according to the task description at run-time. Further, constraints can be applied to individual degrees of freedom. We showed how balance control can be incorporated, making the system applicable during walking. Further, it is shown that real-time planning capabilities can be easily incorporated: Walking can be seamlessly integrated in the control scheme to achieve a given manipulation task.

The method works in real-time and has successfully been tested on *ASIMO*. The algorithms can be used for real-time control of the robot as well as for motion simulation.

Future work will go in the direction of force interaction, planning and real-time simulation, such providing efficient tools for decision making processes and learning.

Acknowledgment

The authors would like to thank the members of Honda’s *ASIMO* and robot research teams in Japan for their support.

References

[1] P. Baerlocher and R. Boulic. Task-priority Formulations for the Kinematic Control of Highly Redundant Articulated Structures. In *IEEE International Conference on Intelligent Robot and Systems (IROS)*, pp. 323 – 329, November 1998.

[2] R. Boulic, R. Mas and D. Thalmann. Interactive Identification of the Center of Mass Reachable Space for an Articulated Manipulator. In *IEEE International Conference of Advanced Robotics (ICAR)*, pp. 589 – 594, Monterey, July 1997.

[3] S. R. Buss. Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares Methods. Internet page. http://www.math.ucsd.edu/ sbuss/ResearchWeb/ikmethods/.

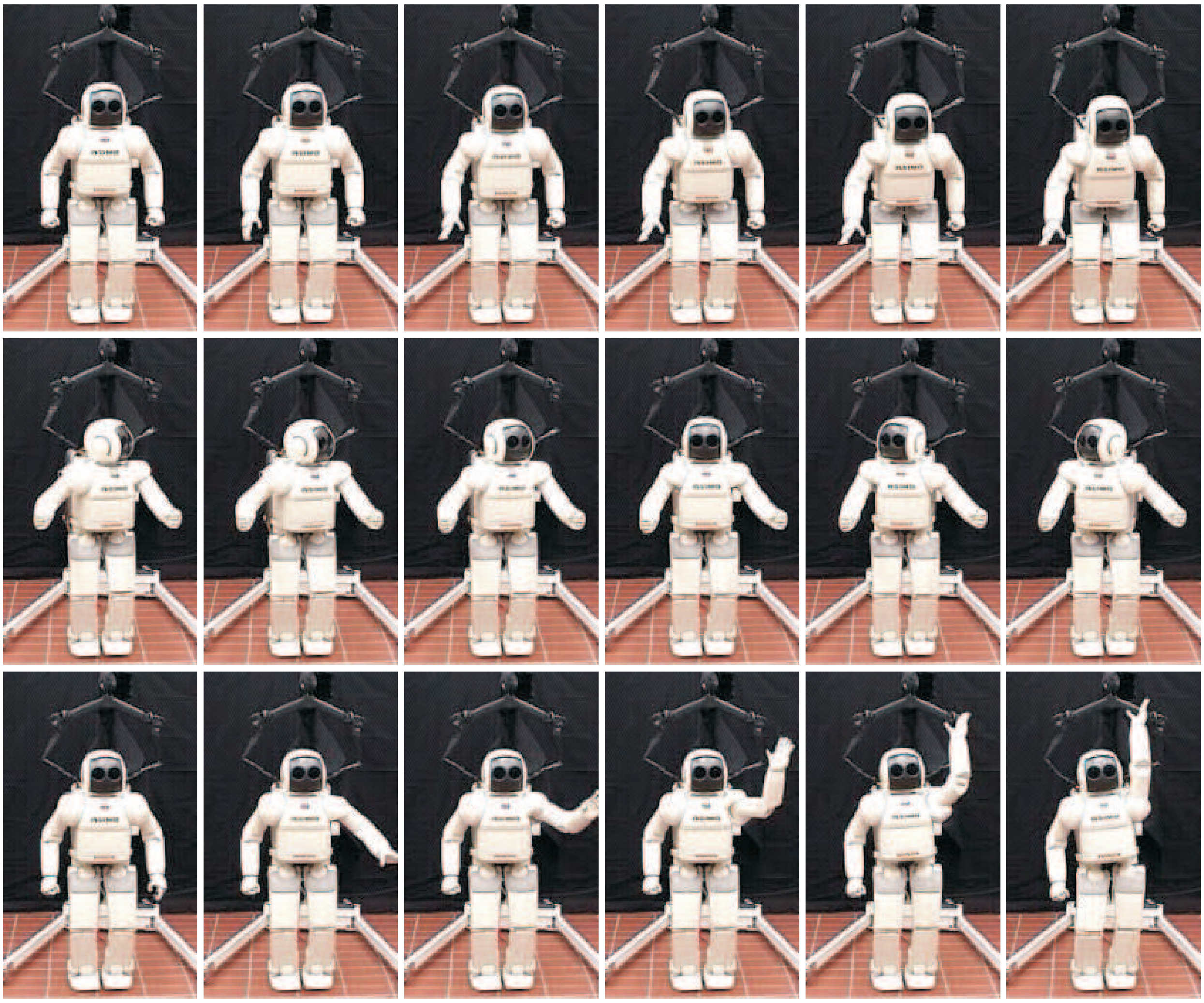


Figure 7: Upper row: Controlling the right hand x-, y- and z-position, Middle row: Changing gaze direction while maintaining hand positions and attitudes, Lower row: Controlling left hand z-position only

[4] F. Chaumette and E. Marchand. A Redundancy-Based Iterative Approach for Avoiding Joint Limits: Application to Visual Servoing. *IEEE Transactions on Robotics ans Automation*, Vol. 17, No. 5, October 2001.

[5] S. I. Choi and B. K. Kim. Obstace Avoidance Control for Redundant Manipulators Using Collidability Measure. *Robotica*, Vol. 18, pp. 143 – 151, 2000.

[6] J. D. English and A. A. Maciejewski. On the Implementation of Velocity Control for Kinematically Redundant Manipulators. In *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 30 of *Part A: Systems and Humans*, pp. 233 – 237, May 2000.

[7] M. Hirose, Y. Haikawa, T. Takenaka and K. Hirai. Development of Humanoid Robot ASIMO. In *IEEE/RSJ International Conference on Intelligent Robots and Systems – Workshop 2*, Hawaii, USA, 2001.

[8] Honda Motor. Internet page. http://www.world.honda.com/ASIMO.

[9] O. Khatib. A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation. *IEEE International Journal of Robotics and Automation*, RA-3(1):43 – 53, 2 1987.

[10] A. Liégeois. Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-7, No. 12, December 1977.

[11] A. A. Maciejewski. Dealing with the Ill-conditioned Equations of Motion for Articulated Figures. *IEEE Comput. Graphics Appl.*, Vol. 10, No. 3, pp. 63 – 71

[12] E. Marchand, F. Chaumette and A. Rizzo. Using the Task Function Approach to Avoid Robot Joint Limits and Kinematic Singularities in Visual Servoing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 3, pp. 1083-1090, Osaka, Japan, November 1996.

[13] Y. Nakamura. *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley Publishing, 1991.

[14] K. Nishiwaki, M. Kuga, S. Kagami, N. Inaba and H. Inoue. Whole-Body Cooperative Balanced Motion Generation for Reaching. In *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, Los Angeles, USA, November 2004.

[15] G. Tevatia and S. Schaal. Inverse Kinematics for Humanoid Robots. In *IEEE Conference on Robotics and Automation (ICRA)*, April 2000.

[16] N. E. Sian, K. Yokoi, S. Kajita and K. Tanie. A Framework for Remote Execution of Whole Body Motions for Humanoid Robots. In *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, Los Angeles, USA, November 2004.

[17] L. Sentis and O. Khatib. Prioritized Multi-objective Dynamics and Control of Robots in Human Environments. In *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, Los Angeles, USA, November 2004.

[18] T. Sugihara and Y. Nakamura. Whole-body Cooperative Balancing of Humanoid Robot using COG Jacobian. In *Proceedings of the IEEE-RSJ International Conference on Intelligent Robots and Systems*, 2002.

[19] R. L. Williams. Local Performance Optimization for a Class of Redundant Eight-Degree-of-Freedom Manipulators. NASA Technical Paper 3417, NASA, March 1994.