Experimental Evaluation of Task Space Position/Orientation Control Towards Compliant Control for Humanoid Robots

Jun Nakanishi*[†], Michael Mistry[‡], Jan Peters[‡], and Stefan Schaal[†‡]
* *ICORP, Japan Science and Technology Agency, Kyoto, Japan*
[†] *ATR Computational Neuroscience Laboratories, Kyoto, Japan*
[‡] *Computer Science and Neuroscience, University of Southern California, Los Angeles, USA*
jun@atr.jp, {mmistry, jrpeters, sschaal}@usc.edu

***Abstract*— Compliant control will be a prerequisite for humanoid robotics if these robots are supposed to work safely and robustly in human and/or dynamic environments. One view of compliant control is that a robot should control a minimal number of degrees-of-freedom (DOFs) directly, i.e., those relevant DOFs for the task, and keep the remaining DOFs maximally compliant, usually in the null space of the task. This view naturally leads to task space control. However, surprisingly few implementations of task space control can be found in actual humanoid robots. This paper makes a first step towards assessing the usefulness of task space controllers for humanoids by investigating which choices of controllers are available and what inherent control characteristics they have—this treatment will concern position and orientation control, where the latter is based on a quaternion formulation. Empirical evaluations on an anthropomorphic Sarcos master arm illustrate the robustness of the different controllers as well as the ease of implementing and tuning them. Our extensive empirical results demonstrate that simpler task space controllers, e.g., classical resolved motion rate control or resolved acceleration control can be quite advantageous in face of inevitable modeling errors in model-based control, and that well chosen formulations are easy to implement and quite robust, such that they are useful for humanoids.**

I. Introduction

One of the important goals in humanoid robotics is to achieve control with similar compliance and robustness towards unanticipated perturbations as humans have. For instance, when stepping on an unforeseen obstacle, the stiff PD control of many humanoids is likely to tip the robot out of balance—the high compliance in a human foot, however, would easily absorb this perturbation. Or, the collision with a suddenly appearing child would make the robot fall and/or hurt the child, partially due to the stiff control gains and lack of back-drivability of most humanoid robot controllers—in contrast, a human would absorb the impact much more softly due to the compliance in all the body's DOFs, and the risk of injury would be reduced.

One potential component to achieve higher levels of compliance is to avoid explicit position-based reference trajectories, and rather work with velocity, acceleration, or, ideally, force control. A second component is to control as few DOFs of the robot as possible with explicit criteria, but rather focus on only task relevant variables and leave other DOFs to compliant redundancy resolution in the null space of the task. This view naturally leads to task space control [1]–[9]. While impressive results have been generated with advanced task space controllers on simulated humanoid robots [10], [11], there seems to be a lack of implementation of these approaches on actual robots. Until a few years ago, one of the potential reasons for this lack of applications was simply the missing computational power to compute the necessary ingredients of these controllers in real-time. Another potential problem with task space controllers is their vulnerability towards geometric and algorithm singularities [12]–[14]. And a third issue with task space control is the increased mathematical complexity and need for automatic code generation tools to deal with this complexity. At last, it is also not easy to tune task space controllers since task space gains affect null space gains and performance in often complex and non-intuitive ways.

However, since task space control seems to be such a theoretically appealing and promising approach to compliant control in humanoids, it seems to be time to examine existing techniques in a more unified and empirical way such that pros and cons can be highlighted. For this reason, this paper investigates a large number of task space controllers in actual implementation of different test tasks on a seven DOF hydraulically actuated anthropomorphic robot arm (Sarcos Master Arm). Unlike many existing robots with stiff joints, the joints of Sarcos Master Arm are designed to be back-drivable and compliant. We focus on the most prominent velocity-based, acceleration-based, and force-based controllers in the literature. All the controllers are formulated in a unified notational framework, including quaternion-based orientation control [15], [16]. Efficient implementations are achieved using Featherstone's spatial notation [17] in a Mathematica-based automatic programming tool. The next sections first briefly introduce all the controllers used in this paper, before evaluating them in different task scenarios.

II. Controller Formulations

Consider the rigid body dynamics of a robot

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} \tag{1}$$

where $\mathbf{q} \in \Re^n$ is the joint angle vector, $\mathbf{M}(\mathbf{q})$ is the inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ is the Coriolis/centripetal vector, $\mathbf{g}(\mathbf{q})$ is the gravity vector, and $\boldsymbol{\tau}$ is the joint torque vector. We use the following notation: $\mathbf{x} \in \Re^m$ is the task coordinate vector, $\mathbf{J}$ is the Jacobian matrix describing the differential relationship

Preprint submitted to 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received April 9, 2007.

between $\mathbf{x}$ and $\mathbf{q}$ as

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \tag{2}$$

$\mathbf{J}^\dagger$ is the pseudoinverse defined by $\mathbf{J}^\dagger = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T)^{-1}$, and $\bar{\mathbf{J}}$ denotes the inertia-weighted pseudoinverse defined by $\bar{\mathbf{J}} = \mathbf{M}^{-1}\mathbf{J}^T(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1}$.

We consider the following eight controllers (for more details of each controller formulation, see [18] and the brief explanations in the next sub-sections):

- Velocity based Control
  1) Velocity based control with joint velocity integration
  2) Velocity based control without joint velocity integration
- Acceleration based Control
  3) Acceleration based control in [3]
  4) Simplified acceleration control variation 1 (with nullspace premultiplication of $\mathbf{M}$)
  5) Simplified acceleration control variation 2 (without nullspace premultiplication of $\mathbf{M}$)
- Force based Control
  6) Gauss control [7]
  7) Dynamical decoupling control variation 1 (without nullspace pre-multiplication of $\mathbf{M}$)
  8) Dynamical decoupling control variation 2 (with nullspace pre-multiplication of $\mathbf{M}$)

A. Velocity based Control

Velocity based control computes the desired joint velocities for a given endeffector velocity [1] in at least two different ways:

1) Velocity based Control with Joint Velocity Integration: In this classical base-line controller, given the reference task space velocity command $\dot{\mathbf{x}}_r$, the reference joint velocities $\dot{\mathbf{q}}_r$ are obtained from the Liegeois' null-space optimization [19]

$$\dot{\mathbf{x}}_r = \dot{\mathbf{x}}_d + \mathbf{K}_p(\mathbf{x}_d - \mathbf{x}) \tag{3}$$

$$\dot{\mathbf{q}}_r = \mathbf{J}^\dagger\dot{\mathbf{x}}_r - \alpha(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\nabla g \tag{4}$$

where $\dot{\mathbf{x}}_d$ and $\mathbf{x}_d$ are the desired task space velocities and positions, respectively, $\mathbf{K}_p$ is a positive definite gain matrix, α is a positive constant, and $g(\mathbf{q})$ is a null space cost function. In our work, $g(\mathbf{q})$ is chosen to be

$$g(\mathbf{q}) = \frac{1}{2}(\mathbf{q} - \mathbf{q}_{rest})^T\mathbf{K}_w(\mathbf{q} - \mathbf{q}_{rest}) \tag{5}$$

where $\mathbf{K}_w$ is a weighting positive definite diagonal matrix and $\mathbf{q}_{rest}$ is some rest (preferred) posture.

The reference joint accelerations $\ddot{\mathbf{q}}_r$ and positions $\mathbf{q}_r$ are obtained by numerical differentiation and integration of the reference joint velocities (4), respectively. The final control signal is calculated using the computed torque control method with a PD controller as

$$\begin{aligned}\boldsymbol{\tau} = {} & \mathbf{M}(\mathbf{q}_r)\ddot{\mathbf{q}}_r + \mathbf{C}(\mathbf{q}_r, \dot{\mathbf{q}}_r) + \mathbf{g}(\mathbf{q}_r) \\ & + \mathbf{K}_{q,d}(\dot{\mathbf{q}}_r - \dot{\mathbf{q}}) + \mathbf{K}_{q,p}(\mathbf{q}_r - \mathbf{q})\end{aligned} \tag{6}$$

where $\mathbf{K}_{q,d}$ and $\mathbf{K}_{q,p}$ are positive definite gain matrices.

2) Velocity based Control without Joint Velocity Integration: As we wish to avoid explicit position reference trajectories in joint space such that higher compliance can be accomplished, an alternative controller formulations uses equations (3)–(5) as before, but defines the control law as a proper inverse dynamics controller with the addition of a joint velocity feedback term as

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_r + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \mathbf{K}_{q,d}(\dot{\mathbf{q}}_r - \dot{\mathbf{q}}) \tag{7}$$

where the reference joint accelerations $\ddot{\mathbf{q}}_r$ are calculated through numerical differentiation.

B. Acceleration based Control

Instead of velocities, the acceleration based control computes the desired joint accelerations for a given endeffector reference, this time specified as a reference acceleration [3]–[6]. Various versions of such controllers exist:

1) Acceleration based Controller in [3]: This approach was proposed in [3]. For a given reference task space acceleration

$$\ddot{\mathbf{x}}_r = \ddot{\mathbf{x}}_d + \mathbf{K}_d(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) + \mathbf{K}_p(\mathbf{x}_d - \mathbf{x}), \tag{8}$$

the control law is given by

$$\boldsymbol{\tau} = \mathbf{M}\left(\mathbf{J}^\dagger(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \boldsymbol{\phi}_N\right) + \mathbf{C} + \mathbf{g} \tag{9}$$

where

$$\boldsymbol{\phi}_N = (\mathbf{I}-\mathbf{J}^\dagger\mathbf{J})(\dot{\mathbf{h}}+\mathbf{K}_N\mathbf{e}_N)-(\mathbf{J}^\dagger\dot{\mathbf{J}}\mathbf{J}^\dagger+\dot{\mathbf{J}}^\dagger)\mathbf{J}(\mathbf{h}-\dot{\mathbf{q}}) \tag{10}$$

$$\mathbf{e}_N = (\mathbf{I}-\mathbf{J}^\dagger\mathbf{J})(\mathbf{h}-\dot{\mathbf{q}}) \tag{11}$$

$\mathbf{h}$ is a vector function $\mathbf{h} = -\alpha\nabla g$, and $\mathbf{K}_N$ is a positive definite matrix. Derivations of this more complex looking control law can be found in [3] and [18].

2) Simplified Acceleration based Control Variation 1 (with nullspace pre-multiplication of $\mathbf{M}$*):* Given the complex null space terms in the controller above [3], a simplified acceleration based controller can be formulated as [18]:

$$\boldsymbol{\tau} = \mathbf{M}\ddot{\mathbf{q}}_r + \mathbf{C} + \mathbf{g} \tag{12}$$

where

$$\ddot{\mathbf{q}}_r = \mathbf{J}^\dagger(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + (\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\boldsymbol{\xi}_2 \tag{13}$$

$$\boldsymbol{\xi}_2 = -\mathbf{K}_{q,d}\dot{\mathbf{q}} - \alpha\nabla g \tag{14}$$

and $\ddot{\mathbf{x}}_r = \ddot{\mathbf{x}}_d + \mathbf{K}_d\dot{\mathbf{e}} + \mathbf{K}_p\mathbf{e}$, where $\mathbf{e} = \mathbf{x}_d - \mathbf{x}$.

The final control law is

$$\boldsymbol{\tau} = \mathbf{M}\mathbf{J}^\dagger(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{C} + \mathbf{g} + \mathbf{M}(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\boldsymbol{\xi}_2. \tag{15}$$

Note that in this controller, the inertia matrix effectively premultiplies the null space term, as seen in the last summand in (15) above.

3) Simplified Acceleration based Control Variation 2 (without nullspace pre-multiplication of $\mathbf{M}$*):* In the previous controller, a null space optimization term was introduced in acceleration space as in (13). Pre-multiplying this null space term with the inertia matrix $\mathbf{M}$ to obtain joint torques, can, however, be problematic if the inertia matrix is not well modeled. Thus, as a variant, we introduce a control law with null space-projected PD control term, i.e., a torque (and not an acceleration) null space command:

$$\begin{aligned}\boldsymbol{\tau} &= \mathbf{M}\ddot{\mathbf{q}}_r + \mathbf{C} + \mathbf{g} + (\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\boldsymbol{\xi}_2 & (16)\\ &= \mathbf{M}\mathbf{J}^{\dagger}(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{C} + \mathbf{g} + (\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})(-\mathbf{K}_{q,d}\dot{\mathbf{q}} - \alpha\nabla g) & (17)\end{aligned}$$

where

$$\begin{aligned}\ddot{\mathbf{q}}_r &= \mathbf{J}^{\dagger}(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) & (18)\\ \boldsymbol{\xi}_2 &= -\mathbf{K}_{q,d}\dot{\mathbf{q}} - \alpha\nabla g & (19)\end{aligned}$$

and $\ddot{\mathbf{x}}_r = \ddot{\mathbf{x}}_d + \mathbf{K}_d\dot{\mathbf{e}} + \mathbf{K}_p\mathbf{e}$.

C. Force based Control

Force based control directly computes the desired joint torques for a given endeffector reference command, given as a force [7], [8]. Again, multiple versions can be considered:

1) Gauss Controller (Operational Space Controller in [7]): A prominent framework for force based redundancy resolution was proposed by Khatib [7]. The desired task dynamics are specified as

$$\bar{\mathbf{M}}(\mathbf{x})\ddot{\mathbf{x}} + \bar{\mathbf{C}}(\mathbf{x}, \dot{\mathbf{x}}) + \bar{\mathbf{g}}(\mathbf{x}) = \mathbf{F} \qquad (20)$$

where

$$\begin{aligned}\bar{\mathbf{M}} &= (\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1} & (21)\\ \bar{\mathbf{C}} &= (\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1}(\mathbf{J}\mathbf{M}^{-1}\mathbf{C} - \dot{\mathbf{J}}\dot{\mathbf{q}}) & (22)\\ \bar{\mathbf{g}} &= (\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1}\mathbf{J}\mathbf{M}^{-1}\mathbf{g}. & (23)\end{aligned}$$

From these equations, a control law for the desired task space dynamics (20) is designed as

$$\mathbf{F} = \bar{\mathbf{M}}\ddot{\mathbf{x}}_r + \bar{\mathbf{C}} + \bar{\mathbf{g}} \qquad (24)$$

where

$$\ddot{\mathbf{x}}_r = \ddot{\mathbf{x}}_d + \mathbf{K}_p(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) + \mathbf{K}_p(\mathbf{x}_d - \mathbf{x}). \qquad (25)$$

The corresponding joint torque control law becomes

$$\begin{aligned}\boldsymbol{\tau} &= \mathbf{J}^T\mathbf{F} + (\mathbf{I} - \mathbf{J}^T\bar{\mathbf{J}}^T)\boldsymbol{\tau}_{null} & (26)\\ &= \mathbf{M}\bar{\mathbf{J}}\left(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}\mathbf{M}^{-1}(\mathbf{C}+\mathbf{g})\right) + (\mathbf{I} - \mathbf{J}^T\bar{\mathbf{J}}^T)\boldsymbol{\tau}_{null} & (27)\end{aligned}$$

where $\boldsymbol{\tau}_{null} = -\mathbf{K}_{q,d}\dot{\mathbf{q}} - \alpha\nabla g$. This control law has many interesting characteristics [7], [8], [20], including that it dynamically decouples task and null space dynamics, and that it interferes with the natural dynamics of the robot is a minimal way, according to the so-called Gauss principle [21].

2) Dynamical Decoupling Controller Variation 1 (without nullspace pre-multiplication of $\mathbf{M}$*, and compensation of* $\mathbf{C}$ *and* $\mathbf{g}$ *in joint space):* In the previous control law, Coriolis and gravitational terms are compensated only in the task space. Motivated by the discussions in [8], we suggested a variant of force based control by pre-compensating Coriolis and gravitational terms in joint space [18]:

$$\begin{aligned}\boldsymbol{\tau} &= \mathbf{C} + \mathbf{g} + \mathbf{J}^T\mathbf{F} + (\mathbf{I} - \mathbf{J}^T\bar{\mathbf{J}}^T)\boldsymbol{\tau}_{null}\\ &= \mathbf{M}\bar{\mathbf{J}}(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{C} + \mathbf{g} + (\mathbf{I} - \mathbf{J}^T\bar{\mathbf{J}}^T)\boldsymbol{\tau}_{null} & (28)\end{aligned}$$

where

$$\begin{aligned}\mathbf{F} &= (\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1}(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) & (29)\\ \boldsymbol{\tau}_{null} &= -\mathbf{K}_{q,d}\dot{\mathbf{q}} - \alpha\nabla g. & (30)\end{aligned}$$

3) Dynamical Decoupling Controller Variation 2 (with nullspace pre-multiplication of $\mathbf{M}$*, and compensation of* $\mathbf{C}$ *and* $\mathbf{g}$ *in joint space):* In (28) above, it is possible to choose the null space vector $\boldsymbol{\tau}_{null}$ as

$$\boldsymbol{\tau}_{null} = \mathbf{M}\ddot{\mathbf{q}}_0. \qquad (31)$$

With this choice of the null space vector, the control law becomes

$$\begin{aligned}\boldsymbol{\tau} &= \mathbf{C} + \mathbf{g} + \mathbf{J}^T\mathbf{F} + (\mathbf{I} - \mathbf{J}^T\bar{\mathbf{J}}^T)\mathbf{M}\ddot{\mathbf{q}}_0\\ &= \mathbf{C} + \mathbf{g} + \mathbf{J}^T\mathbf{F} + \mathbf{M}(\mathbf{I} - \bar{\mathbf{J}}\mathbf{J})\ddot{\mathbf{q}}_0\\ &= \mathbf{M}\left(\bar{\mathbf{J}}(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + (\mathbf{I} - \bar{\mathbf{J}}\mathbf{J})\ddot{\mathbf{q}}_0\right) + \mathbf{C} + \mathbf{g} & (32)\end{aligned}$$

where

$$\begin{aligned}\mathbf{F} &= (\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1}(\ddot{\mathbf{x}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) & (33)\\ \ddot{\mathbf{q}}_0 &= -\mathbf{K}_{q,d}\dot{\mathbf{q}} - \alpha\nabla g. & (34)\end{aligned}$$

As already suggested in acceleration based control, the omission of the pre-multiplication of the inertia matrix in the null space term can add useful robustness in face of modeling errors of $\mathbf{M}$.

III. Orientation Control with Quaternion Feedback

The enumeration of the controllers in the previous section did not make any attempts to be specific about the nature of the controlled task. In general, position and orientation control need to be considered in task space (besides force control, which we will not address explicitly here). While task space position control is rather straightforward, orientation control is more complex. An elegant and numerically robust solution to this problem can be formulated with the help of quaternions [15], instead of Euler angles or roll-pitch-yaw angles which are frequently used in the robotics literature. Quaternions have desirable properties, for example, a) quaternion derivatives can be integrated over time to obtain the resultant orientation representation, b) they do not have singularities as in Euler angles, and c) it is straightforward to convert between angular velocities and quaternion derivatives.

Let us denote a unit quaternion as

$$\boldsymbol{\alpha} = \begin{bmatrix} \eta \\ \boldsymbol{\epsilon} \end{bmatrix} = [\, \eta,\ \epsilon_1,\ \epsilon_2,\ \epsilon_3 \,]^T \tag{35}$$

where η is the scalar part and $\boldsymbol{\epsilon}$ is the vector part and $\eta^2 + \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2 = 1$. A spatial orientation can be described by a rotation of φ about an unit vector of the axis $\mathbf{r}$ ($||\mathbf{r}|| = 1$) and represented in terms of a quaternion as

$$\eta = \cos\left(\frac{\varphi}{2}\right) \text{ and } \epsilon = \mathbf{r}\sin\left(\frac{\varphi}{2}\right). \tag{36}$$

For orientation control, in [15], the orientation error is formulated using the unit quaternion as[1]

$$\mathbf{e}_o = \delta\boldsymbol{\epsilon} = \eta_d\boldsymbol{\epsilon} - \eta\boldsymbol{\epsilon}_d + [\boldsymbol{\epsilon}_d\times]\boldsymbol{\epsilon}. \tag{37}$$

where $\boldsymbol{\alpha}_d$ is the desired orientation and $\boldsymbol{\alpha}$ is the current orientation, both expressed as quaternions.

In velocity based control, given the desired task space translatory velocities $\dot{\mathbf{p}}_d$, positions $\mathbf{p}_d$, angular velocities $\boldsymbol{\omega}_d$ and orientation $\boldsymbol{\alpha}_d$ (i.e., as a quaternion), the task space translatory reference velocity $\dot{\mathbf{p}}_r$ and angular reference velocity $\boldsymbol{\omega}_r$ are defined, respectively, as

$$\dot{\mathbf{p}}_r = \dot{\mathbf{p}}_d + \mathbf{K}_p(\mathbf{p}_d - \mathbf{p}) \tag{38}$$

$$\begin{aligned}\boldsymbol{\omega}_r &= \boldsymbol{\omega}_d - \mathbf{K}_o\mathbf{e}_o \\ &= \boldsymbol{\omega}_d - \mathbf{K}_o(\eta_d\boldsymbol{\epsilon} - \eta\boldsymbol{\epsilon}_d + [\boldsymbol{\epsilon}_d\times]\boldsymbol{\epsilon})\end{aligned} \tag{39}$$

such that the augmented task space reference velocities are

$$\dot{\mathbf{x}}_r = \begin{bmatrix} \dot{\mathbf{p}}_r \\ \boldsymbol{\omega}_r \end{bmatrix}. \tag{40}$$

With this re-definition, all the controllers in Section II-A can be applied.

In analogy, in acceleration and force based control, the task space translatory reference acceleration $\ddot{\mathbf{p}}_r$ and angular reference acceleration $\dot{\boldsymbol{\omega}}_r$ are defined, respectively, as

$$\ddot{\mathbf{p}}_r = \ddot{\mathbf{p}}_d + \mathbf{K}_d(\dot{\mathbf{p}}_d - \dot{\mathbf{p}}) + \mathbf{K}_p(\mathbf{p}_d - \mathbf{p}) \tag{41}$$

$$\begin{aligned}\dot{\boldsymbol{\omega}}_r &= \dot{\boldsymbol{\omega}}_d + \mathbf{K}_\omega(\boldsymbol{\omega}_d - \boldsymbol{\omega}) - \mathbf{K}_o\mathbf{e}_o \\ &= \dot{\boldsymbol{\omega}}_d + \mathbf{K}_\omega(\boldsymbol{\omega}_d - \boldsymbol{\omega}) - \mathbf{K}_o(\eta_d\boldsymbol{\epsilon} - \eta\boldsymbol{\epsilon}_d + [\boldsymbol{\epsilon}_d\times]\boldsymbol{\epsilon})\end{aligned} \tag{42}$$

and the task space reference accelerations are augmented as

$$\ddot{\mathbf{x}}_r = \begin{bmatrix} \ddot{\mathbf{p}}_r \\ \dot{\boldsymbol{\omega}}_r \end{bmatrix}. \tag{43}$$

Then, all the controllers in Sections II-B and II-C can be applied.

IV. Estimation of Physically Consistent Rigid Body Parameters

For experimental evaluations, we used a Sarcos Master Arm, a seven DOF hydraulically actuated anthropomorphic robot arm (Figure 1). The joints of this robot are designed to be back-drivable and compliant. In order to implement the controllers introduced above, it is necessary to obtain the rigid body dynamics components in (1), which require knowledge of the inertial parameters of each robot link. Ideally, these parameters can be obtained from CAD data. However, due to the significant contribution of hydraulic actuation in our robot, CAD data turned out to be a poor fit to model the robot dynamics, such that we resorted to numerical parameter identification. While estimation techniques such as [22] generate good inverse dynamics models for simple joint space controllers like computed torque or classical inverse dynamics control, they run into problems with complex task space controllers, as investigated in this paper. The reason for these problems is due to the ordinary least-squares procedure employed for identifying the link parameters—this procedure has no mechanism to ensure physical correctness of the inertial parameters, i.e., positive mass parameters, positive-definite inertia matrices, and the constraints imposed by the parallel axis theorem in converting center-of-mass inertia to joint-axis coordinate systems. As a result, it was possible to obtain non-positive definite inertia matrix $\mathbf{M}(\mathbf{q})$ at certain configurations of the robot, which destabilized some of the suggested controllers.

In order to ensure physically consistent inertial parameters, we derived the following nonlinear parameter estimation method using a re-parametrization of the basic link parameters. There are 11 parameters (10 inertial and 1 friction parameters) to be estimated for each DOF (see Figure 2), mass m, three center of mass coefficients multiplied by the mass mc_x, mc_y, mc_z, six inertial parameters (in joint axis, not center-of-mass coordinates) $I_{xx}, I_{xy}, I_{xz}, I_{yy}, I_{yz}, I_{zz}$ (cf. [22]), and viscous friction d. These parameters are arranged

[1]Note that there is a sign change in the third term in this orientation error. This is due to the difference of the representation of the angular velocity in world and local coordinates.

Fig. 1. 7 degree-of-freedom hydraulic robot: Sarcos Master Arm

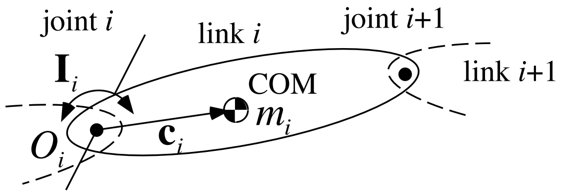


Fig. 2. Link inertial parameters and link coordinates for parameter identification. m is the link mass, $\mathbf{I}$ is the inertia matrix about the local link (joint-axis) coordinates, and $\mathbf{c}$ is the center of mass (COM) location in the joint-axis coordinates.

Preprint submitted to 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received April 9, 2007.

in an 11-dimensional vector $\boldsymbol{\theta}$ as

$$\boldsymbol{\theta} = [m, mc_x, mc_y, mc_z, I_{xx}, I_{xy}, I_{xz}, I_{yy}, I_{yz}, I_{zz}, d]^T \tag{44}$$

To ensure physical consistency, the constraints in (45)–(50) need be satisfied for each DOF, which are derived from the parallel axis theorem, a Cholesky decomposition of the inertia matrix (positive definite inertia matrix around the center of mass),and the need for positive mass and friction parameters. Thus, one can conceive that the original parameter vector $\boldsymbol{\theta}$ was generated through a nonlinear transformation from an 11-dimensional virtual parameter vector $\tilde{\boldsymbol{\theta}} = [\tilde{\theta}_1, \cdots, \tilde{\theta}_{11}]^T$.

$$\theta_1 = \tilde{\theta}_1^2,\ \theta_2 = \tilde{\theta}_1^2\tilde{\theta}_2,\ \theta_3 = \tilde{\theta}_1^2\tilde{\theta}_3,\ \theta_4 = \tilde{\theta}_1^2\tilde{\theta}_4 \tag{45}$$

$$\theta_5 = \tilde{\theta}_5^2 + \tilde{\theta}_1^2\left(\tilde{\theta}_3^2 + \tilde{\theta}_4^2\right) \tag{46}$$

$$\theta_6 = \tilde{\theta}_5\tilde{\theta}_6 - \tilde{\theta}_1^2\tilde{\theta}_2\tilde{\theta}_3,\ \theta_7 = \tilde{\theta}_5\tilde{\theta}_7 - \tilde{\theta}_1^2\tilde{\theta}_2\tilde{\theta}_4 \tag{47}$$

$$\theta_8 = \tilde{\theta}_6^2 + \tilde{\theta}_8^2 + \tilde{\theta}_1^2\left(\tilde{\theta}_2^2 + \tilde{\theta}_4^2\right) \tag{48}$$

$$\theta_9 = \tilde{\theta}_6\tilde{\theta}_7 + \tilde{\theta}_8\tilde{\theta}_9 - \tilde{\theta}_1^2\tilde{\theta}_3\tilde{\theta}_4 \tag{49}$$

$$\theta_{10} = \tilde{\theta}_7^2 + \tilde{\theta}_9^2 + \tilde{\theta}_{10}^2 + \tilde{\theta}_1^2\left(\tilde{\theta}_2^2 + \tilde{\theta}_3^2\right),\ \theta_{11} = \tilde{\theta}_{11}^2 \tag{50}$$

Given the above formulation, any arbitrary set of virtual parameters gives rise to a physically consistent set of actual parameters for the rigid body dynamics model. For a robotic system with n DOFs, (45)–(50) are repeated for each DOF. For parameter estimation, we replaced the least-squares approach of [22] by a nonlinear parameter estimation technique with gradient descent, based on the above virtual parameters. A more advanced parameter identification procedure using a Bayesian approach is currently being developed [23].

V. Experimental Evaluations

A. Benchmark Tasks

We empirically evaluated the performance and properties of the various controllers introduced in the above sections. As benchmark tasks, the following eight experiments were considered:

- Tracking a “figure 8” pattern
  1) figure 8, slow speed, high task space gain
  2) figure 8, fast speed, high task space gain
- Drawing a “star-like” pattern
  3) star-like pattern, slow speed, high task space gain
  4) star-like pattern, fast speed, high task space gain
  5) star-like pattern, slow speed, low task space gain
  6) star-like pattern, fast speed, low task space gain
- Tracking a “figure 8” pattern with orientation control
  7) figure 8 with orientation, slow speed, high task space gain
  8) figure 8 with orientation, fast speed, high task space gain

In the experiments 1) and 2), the task is to track a planar “figure 8” pattern in the vertical plane (height and width: 0.225m) in task space at two different speeds (slow speed: 4 seconds per cycle, fast speed: 2 seconds per cycle). In the experiments 3)–6), the task is to draw a “star-like” pattern in the vertical plane by first pointing outwards from the center and then inwards back to the center in eight directions in a sequential manner (slow: 1 second and fast 0.5 seconds for each pointing movement). The desired pointing movement is specified by a minimum-jerk trajectory. This “star-like” pattern is often considered in biological human motor control behavior experiments and has parts of rather high acceleration. In the experiments 7) and 8), in addition to the task of tracking a the desired position trajectory of the “figure 8”, the endeffector’s orientation is also specified to keep it at a fixed desired posture.

In these experiments, for the high task space gain setting, we used $\mathbf{K}_p = 10\mathbf{I}$, $\mathbf{K}_o = 50\mathbf{I}$ for the velocity based controllers, and $\mathbf{K}_p = 50\mathbf{I}$, $\mathbf{K}_o = 1000\mathbf{I}$ and $\mathbf{K}_d = \sqrt{50}\mathbf{I}$, $\mathbf{K}_\omega = \sqrt{1000}\mathbf{I}$ for the acceleration and force based controllers. For the low task space gain setting, we used $\mathbf{K}_p = 5\mathbf{I}$, $\mathbf{K}_o = 25\mathbf{I}$ for the velocity based controllers, and $\mathbf{K}_p = 25\mathbf{I}$, $\mathbf{K}_o = 500\mathbf{I}$ and $\mathbf{K}_d = \sqrt{25}\mathbf{I}$, $\mathbf{K}_\omega = \sqrt{500}\mathbf{I}$ for the acceleration and force based controllers[2]. We collected the data of three runs for each experiment with the eight controllers described in Section II, respectively.

B. Experimental Results

Figures 3–5 show examples of tracking results of the experiments 1), 4) and 6) illustrating qualitative difference among the eight controllers. The gray line is the target trajectory and the solid line is the actual trajectory. Table I shows the root mean squared (RMS) errors between the actual and target task space trajectory (position), and Table II shows the RMS orientation error of the experiments 7) and 8). As an orientation error measure, the L_2 norm of the orientation error feedback term $\mathbf{e}_o$ in (37) is considered. In these tables, bold numbers indicate the best tracking performance and italic numbers indicate the second best tracking performance. A short video clip is attached to this submission to show examples of the movement of the robot with the simplified acceleration based control (controller 5) containing 1) demonstration of the level of compliance and robustness of the control by manually perturbing the robot during slow “figure 8” movement, 2) fast “figure 8” movement, 3) fast star-like movement, 4) “figure 8” with orientation control at slow speed demonstrating the effectiveness of orientation control by placing a cup containing water on the endeffector.

The experimental results can be summarized as follows:

- **Overall performance comparison:** As already observed in a preliminary earlier study [18], the simplified acceleration based control (controller 5) is the most promising approach. The more comprehensive experimental evaluation in this paper clearly demonstrates the effectiveness of this approach in face of inevitable modeling errors. In particular, the results of experiment 6) (fast star-like pattern with low gain), which

[2]In the acceleration based controller 3) (Section II-B.1), we had to reduce orientation gains because of instability, and $\mathbf{K}_o = 400\mathbf{I}$, $\mathbf{K}_\omega = \sqrt{400}\mathbf{I}$ are used in the experiments reported in this paper.

Preprint submitted to 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received April 9, 2007.

is the most demanding task of all, showed that the simplified acceleration based control (controller 5) still achieves remarkably good tracking performance while other acceleration/force based controllers' performance was significantly degraded.

- **Velocity based approach:** Velocity based controllers (controllers 1 and 2) achieved overall good performance. However, as we have mentioned in [18], there is a practical limitation in the choice of the task space position gain because the task space position gain effectively appears as a task space damping gain due to numerical differentiation of the joint space reference velocity (4). This effect implies that a) we cannot increase position gains too much to improve the tracking performance, as it may lead to instability due to too large damping gain, and b) the impedance behavior of the task space dynamics cannot be easily specified because of this coupling of task space position and damping gains. From a subjective point of view, all velocity based controllers "felt" highly overdamped and not very compliant when manually perturbing the robot.
- **Acceleration based controller in [3] (controller 3):** The experimental results demonstrated that we could not achieve very good performance and we found that the controller was not very robust and really hard to tune. For orientation control, the choice of the orientation gain of $\mathbf{K}_o = 1000\mathbf{I}$ resulted in instability while the same choice of the orientation gain worked well in other acceleration and force based controllers. Also, this controller was highly sensitive to noise form numerical differentiation, which was generated when obtaining time derivatives of Jacobin and pseudo-inverse matrices. Subjectively, it felt that this controller is overly complex, without that this controller would achieve particularly useful control properties.
- **Force based control with inertia-weighted pseudoinverse:** Force based control approaches using the inertia-weighted pseudoinverse have the desirable property of dynamical decoupling between the task and null space. However, in practice, performance of these force based controllers (controllers 6–8) was not as good as the simplified acceleration based control (controller 5). This effect is in particular due to inaccuracies of the estimated inertia matrix, as this matrix and its inverse are used in many different places of the control law. Computationally, the calculation of the inertia-weighted pseudoinverse requires explicit extraction of the inertia matrix from the rigid body dynamics, which is computationally expensive.
- **Explicit use of the inertia matrix:** The performance of the algorithms which explicitly use the inertia matrix (force based algorithms with inertia-weighted pseudoinverse (controllers 6-8) or controllers including nullspace term premultplied by the inertia matrix (controllers 3, 4 and 8)) significantly degrade especially in the tasks with fast movements. This implies that these algorithms require highly accurate inertia matrix estimation to be successful.
- **Orientation control with quaternion feedback:** The quaternion based orientation control was successfully implemented for all controllers. We achieved good overall performance in the regulation of the desired posture of the endeffector while tracking the figure 8 pattern except for the controller 3 (the one which we could not achieve good performance in position control tasks mentioned above). We could place a cup containing water on the endeffector without spilling water in the slow figure 8 task by keeping the endeffector's posture fixed while the endeffector was tracking the figure 8 pattern (see Figure 6). However, we observed that the task space position tracking performance was not as good as that of controllers without orientation constraints.

VI. CONCLUSION

In this paper, we presented extensive empirical evaluations of various task space control algorithms with redundancy resolution, with the goal to provide a critical examination of these controllers for their use in humanoid robotics. We formulated all the controllers in a unified notational framework, including quaternion-based orientation control. We also introduced a parameter estimation algorithm for rigid body dynamics that ensures physically consistent parameters identification. Our experimental results demonstrate that the simplified acceleration-based controller (controller 5) had the overall best performance in terms of tracking results and general robustness. It even worked surprisingly well for the demanding task of the fast star-like movement with low task space gain settings, in which the performance of many other controllers significantly degraded. This simplified acceleration based controller is easy to implement in the framework of efficient Newton-Euler rigid body dynamics formulations. For that tasks that we investigated, more complex operational space controllers like force based controllers, did not perform very well. To achieve high quality results with these theoretically very appealing methods, significant emphasis must be placed on accurate model identification of the robot, which can be rather hard in complex humanoids.

Future work will address applications of the task space control laws for balancing and locomotion control for biped robots, including control situations with switching constraints and the integration of task space control with joint-space based pattern generators.

ACKNOWLEDGMENTS

We would like to thank Prof. Roy Featherstone for discussions and clarifications of the theoretical issues on redundancy resolution. This research was supported in part by National Science Foundation grants ECS-0325383, IIS-0312802, IIS-0082995, ECS-0326095, ANI-0224419, the DARPA program on Learning Locomotion, a NASA grant AC#98-516, an AFOSR grant on Intelligent Control, the ERATO Kawato Dynamic Brain Project funded by the Japan Science and Technology Agency, and the ATR Computational Neuroscience Laboratories.

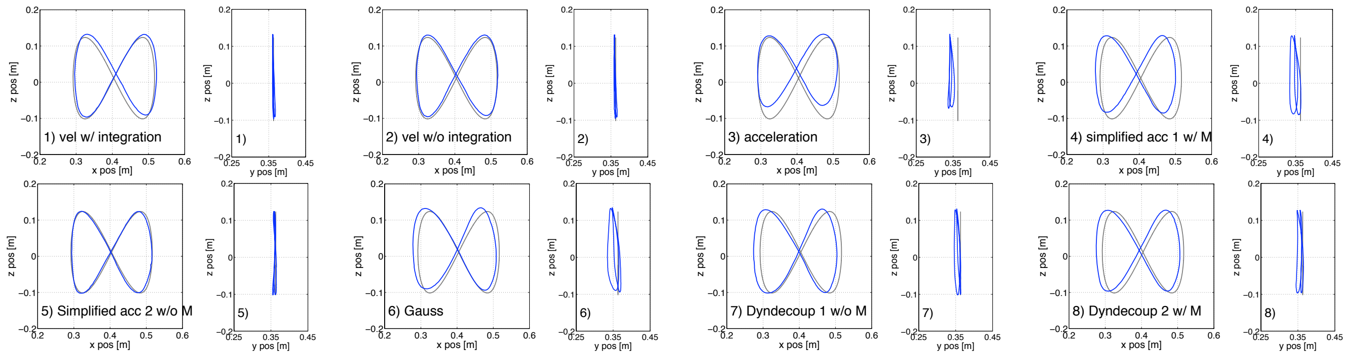


Fig. 3. Tracking results of slow figure 8 movement (4 seconds per period)

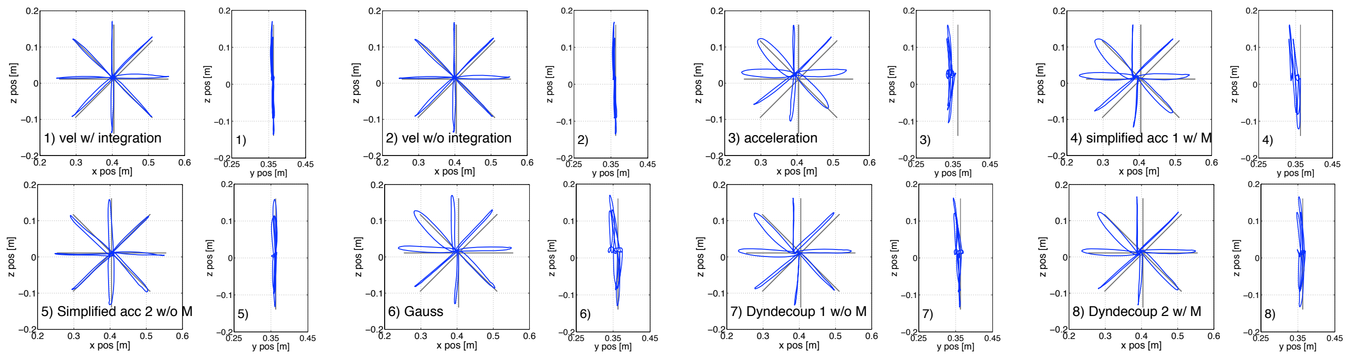


Fig. 4. Tracking results of fast star movement (0.5 seconds for each pointing movement) with high gains

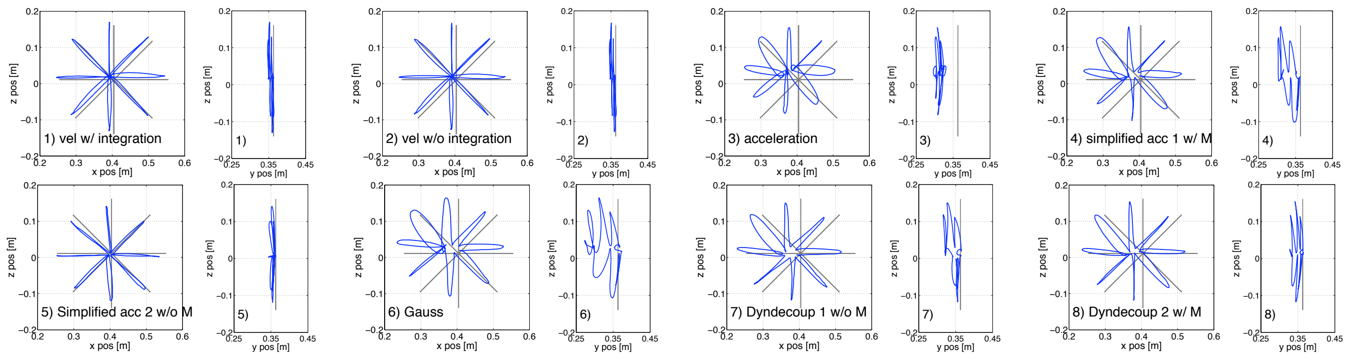


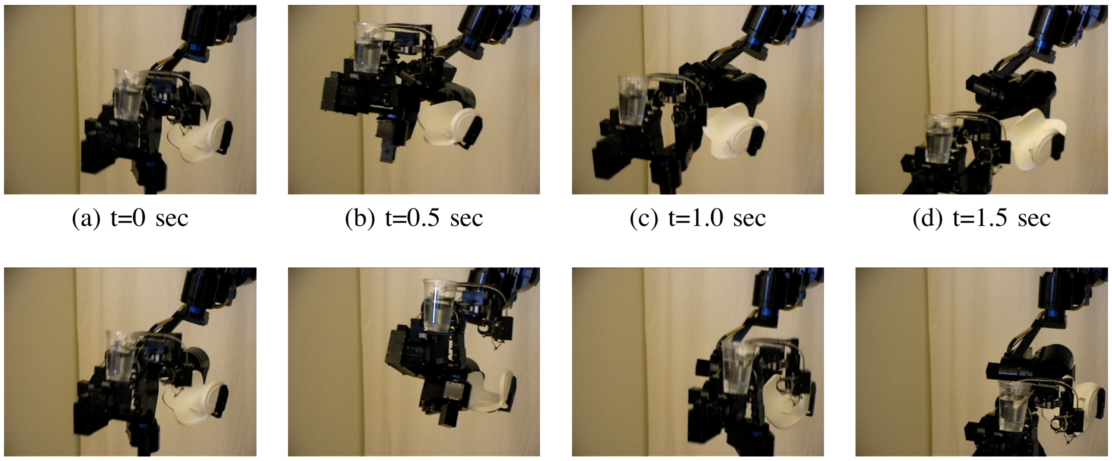
Fig. 5. Tracking results of fast star movement (0.5 seconds for each pointing movement) with low gains

TABLE I

ROOT MEAN SQUARED (RMS) TRACKING ERROR [m] OF THE BENCHMARK TASKS USING DIFFERENT CONTROL LAWS AVERAGED OVER THREE EXPERIMENTAL RUNS. THE BOLD NUMBERS INDICATE THE SMALLEST TRACKING ERROR AND THE ITALIC NUMBERS INDICATE THE SECOND SMALLEST TRACKING ERROR FOR EACH TASK, RESPECTIVELY.

	1) vel. w/ integration	2) vel. w/o integration	3) acc.	4) acc. 1 w/ $\mathbf{M}$	5) acc. 2 w/o $\mathbf{M}$	6) Gauss	7) dyn. dec. 1 w/o $\mathbf{M}$	8) dyn. dec. 2 w/ $\mathbf{M}$
figure 8 (slow)	0.0100	*0.0090*	0.0312	0.0256	**0.0080**	0.0218	0.0191	0.0187
figure 8 (fast)	**0.0144**	**0.0144**	0.0322	0.0317	*0.0258*	0.0293	0.0325	0.0291
star (slow, high gain)	**0.0072**	0.0090	0.0323	0.0265	*0.0074*	0.0180	0.0178	0.0133
star (fast, high gain)	*0.0117*	0.0125	0.0351	0.0309	**0.0144**	0.0240	0.0246	0.0221
star (slow, low gain)	**0.0153**	0.0188	0.0662	0.0451	*0.0155*	0.0570	0.0405	0.0343
star (fast, low gain)	*0.0209*	0.0223	0.0685	0.0518	**0.0197**	0.0600	0.0439	0.0396
figure 8 w/ orient (slow)	**0.0087**	**0.0087**	0.0216	0.0177	*0.0166*	0.0194	0.0188	0.0251
figure 8 w/ orient (fast)	*0.0160*	**0.0155**	0.0314	0.0390	0.0380	0.0396	0.0405	0.0387

Preprint submitted to 2007 IEEE/RSJ International Conference
on Intelligent Robots and Systems. Received April 9, 2007.


(a) t=0 sec (b) t=0.5 sec (c) t=1.0 sec (d) t=1.5 sec

(e) t=2.0 sec (f) t=2.5 sec (g) t=3.0 sec (h) t=3.5 sec

Fig. 6. Snapshots of slow figure 8 movement with orientation control. The task is to track the figure 8 pattern by keeping the endeffector's posture fixed. In order to demonstrate orientation control, a cup filled with water was placed on the endeffector. See also an accompanied video to this submission.

TABLE II

ROOT MEAN SQUARED (RMS) ORIENTATION ERROR OF THE FIGURE 8 TASK WITH ORIENTATION CONTROL USING DIFFERENT CONTROL LAWS AVERAGED OVER THREE EXPERIMENTAL RUNS. AS AN ORIENTATION ERROR MEASURE, THE L_2 NORM OF THE ORIENTATION ERROR FEEDBACK TERM $\mathbf{e}_o = \eta_d \boldsymbol{\epsilon} - \eta \boldsymbol{\epsilon}_d + [\boldsymbol{\epsilon}_d \times] \boldsymbol{\epsilon}$ IN (37) IS CONSIDERED. THE BOLD NUMBERS INDICATE THE SMALLEST TRACKING ERROR AND THE ITALIC NUMBERS INDICATE THE SECOND SMALLEST TRACKING ERROR FOR EACH TASK, RESPECTIVELY.

	1) vel. w/ integration	2) vel. w/o integration	3) acc.	4) acc. 1 w/ $\mathbf{M}$	5) acc. 2 w/o $\mathbf{M}$	6) Gauss	7) dyn. dec. 1 w/o $\mathbf{M}$	8) dyn. dec. 2 w/ $\mathbf{M}$
figure 8 w/ orient (slow)	**0.0219**	*0.0240*	0.0751	0.0296	0.0260	0.0311	0.0307	0.0274
figure 8 w/ orient (fast)	**0.0259**	*0.0299*	0.0814	0.0428	0.0429	0.0455	0.0443	0.0393

REFERENCES

[1] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *International Journal of Robotics Research*, vol. 6, no. 2, pp. 3–15, 1987.

[2] L. Sciavicco and B. Siciliano, "A solution algorithm of the inverse kinematic problem for redundant manipulators," *IEEE International Journal of Robotics and Automation*, vol. 4, no. 4, pp. 403–410, 1988.

[3] P. Hsu, J. Hauser, and S. Sastry, "Dynamic control of redundant manipulators," *Journal of Robotic Systems*, vol. 6, no. 2, pp. 133–148, 1989.

[4] A. D. Luca and G. Oriolo, "Issues in acceleration resolution of robot redundancy," in *3rd IFAC Symposium on Robot Control*, 1991, pp. 93–98.

[5] K. Senda, "Quasioptimal control of space redundant manipulators," in *AIAA Guidance, Navigation, and Control Conference*, 1999, pp. 1877–1885.

[6] J. M. Hollerbach and K. C. Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 4, pp. 308–316, 1987.

[7] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 1, pp. 43–53, 1987.

[8] R. Featherstone and O. Khatib, "Load independence of the dynamically consistent inverse of the Jacobian matrix," *International Journal of Robotics Research*, vol. 16, no. 2, pp. 168–170, 1997.

[9] S. Arimoto, M. Sekimoto, H. Hashiguchi, and R. Ozawa, "Natural resolution of ill-posedness of inverse kinematics for redundant robots:a challenge to Bernstein's degrees-of-freedom problem," *Advanced Robotics*, vol. 19, no. 4, pp. 401–434, 2005.

[10] O. Khatib, O. Brock, K. Chang, D. Ruspini, L. Sentis, and S. Viji, "Human-centered robotics and interactive haptic simulation," *International Journal of Robotics Research*, vol. 23, no. 2, pp. 167–478, 2004.

[11] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 505–518, 2005.

[12] C. Klein, C. Chu-Jenq, and S. Ahmed, "A new formulation of the extended Jacobian method and its use in mapping algorithmic singularities for kinematically redundant manipulators," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 1, pp. 50–55, 1995.

[13] J. Baillieul and D. P. Martin, "Resolution of kinematic redundancy," in *Proceedings of Symposia in Applied Mathematics*, vol. 41, 1990, pp. 49–89.

[14] C. L. Luck and S. Lee, "Self-motion topology for redundant manipulators with joint limits," in *Proceedings of IEEE International Conference on Robotics and Automation*, 1993, pp. 626–631.

[15] J. S.-C. Yuan, "Closed-loop manipulator control using quaternion feedback," *IEEE Journal of Robotics and Automation*, vol. 4, no. 4, pp. 434–440, 1988.

[16] F. Caccavale, C. Natale, B. Siciliano, and L. Villani, "Resolved-acceleration control of robot manipulators:," *A Critical Review with Experiments*, vol. 16, no. 5, pp. 565–573, 1998.

[17] R. Featherstone, *Robot Dynamics Algorithms*. Kluwer Academic Publishers, 1987.

[18] J. Nakanishi, R. Cory, M. Mistry, and S. Schaal, "Comparative experiments on task space control with redundancy resolution," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 1575–1582.

[19] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, no. 12, pp. 868–871, 1977.

[20] H. Bruyninckx and O. Khatib, "Gauss' principle and the dynamics of redundant and constrained manipulators," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2000, pp. 2563–2568.

[21] F. E. Udwadia and R. E. Kalaba, *Analytical Dynamics: A New Approach*. Cambridge University Press, 1996.

[22] C. H. An, C. G. Atkeson, and J. M. Hollerbach, *Model-Based Control of a Robot Manipulator*. MIT Press, 1988.

[23] J.-A. Ting, M. Mistry, J. Peters, S. Schaal, and J. Nakanishi, "A Bayesian approach to nonlinear parameter identification for rigid body dynamics," in *Robotics: Science and Systems Conference*, 2006.

Preprint submitted to 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received April 9, 2007.