The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Constrained Resolved Acceleration Control for Humanoids

Behzad Dariush, Ghassan Bin Hammam, and David Orin

Abstract— **Resolved acceleration control is a well-known strategy used in tracking control of robotic systems where the desired motion is specified in task-space. Typically, such controllers are developed for systems which exhibit redundancy with respect to execution of operational tasks. While redundancy fundamentally adds new capabilities (self-motion and subtask performance capability), the degree to which secondary objectives can be faithfully executed cannot be determined in advance unless the motion is planned and the environment is known. Therefore, execution of secondary objectives cannot be guaranteed. In fact, a robot which exhibits redundancy with respect to operational tasks may have insufficient degrees of freedom to fulfill more critical objectives such as enforcing constraints. In this paper, we present a generalized constrained resolved acceleration control framework to handle execution of operational tasks and constraints for redundant and non-redundant task (and constraint) specifications. The approach is particularly well suited for online control of complex robot structures such as humanoid robots. The current formulation considers joint limit and collision constraints. The efficacy of the proposed algorithm is demonstrated by simulated experiments of task level upper-body human motion replication on the Honda humanoid robot.**

I. Introduction

In the past four decades, a great deal of research has been devoted to understand and formulate robot movement coordination in task-space, whereby the control action can be represented at the velocity, acceleration [1], or force/torque levels [2]. In typical applications involving task-space control, the robot's number of degrees of freedom exceeds what is required to perform the operational tasks. In such circumstances, the robot is said to exhibit redundancy since there exist infinite joint motions that produce the specified task motion.

The occurrence of redundancy with respect to the specified tasks gives opportunity to achieve other objectives, such as avoiding obstacles [3], [4], avoiding structural limits (e.g. joint limits, and self collisions [5]–[7]), minimizing energy consumption, creating balanced motions, etc. In early research on redundancy resolution schemes, many of these objectives (including enforcing kinematic constraints) were considered as secondary tasks, performed in the null-space of the higher priority operational tasks [8], [9].

Formulating constraints as a secondary task cannot guarantee that constraints will be satisfied if there are insufficient number of degrees of freedom to satisfy both objectives. In many cases, satisfying constraints is critical and therefore must be given priority over execution of operational tasks. A suggested solution is to formulate constraints as the highest priority operation and project the operational tasks onto the constraint null-space [10]. In principle, this method is more appropriate since constraints which adhere to the mechanical limits of the robot or sustain balance are critical for safety. In practice, several unresolved issues remain, particularly for the case when the task and constraint spaces are not known in advance.

As an example, consider a scenario of real-time transfer of task level motion from human demonstrator to a humanoid robot [7], [11]–[13]. This scenario involves execution of an unplanned task motion subject to kinematic and balance constraints. The transferred motion may result in simultaneous self collisions occurring between multiple segment pairs, or violations of multiple joint limits. Two problems may arise under such circumstances. First, the null-space of the constraint Jacobian may not exist, making infeasible the execution of secondary objectives, including tracking of the operational tasks. Second, in case of self collision avoidance, the Cartesian positions corresponding to the minimum distances between two colliding body segments are generally discontinuous, resulting in numerical and algorithmic instabilities which require special care. Based on the above discussion, it is evident that regardless of the order of priority assigned to enforcing constraints, methods which are solely based on null-space projections will ultimately fail in executing secondary objectives if there is no redundancy.

This paper examines the problem of task-space control in the presence of joint limit and collision constraints and provides effective solutions to address issues associated with existing algorithms as described above. In particular, the proposed approach uses results from our previously reported constrained inverse kinematics procedure (see [14]), to arrive at a constrained resolved acceleration control formulation which is applicable for redundant as well as non-redundant robotic mechanisms.

The main contributions of the paper are as follows. We derive a novel kinematically constrained second-order closed loop inverse kinematics formulation applicable for redundant and non-redundant task specifications. The transformation from Cartesian-space to joint-space is handled using a weighted pseudo-inverse at the velocity and acceleration level. The weight matrix, also referred to as the constraint matrix, is constructed based on gradients of collision and joint limit potential functions which penalize and dampen motion approaching constraint surfaces. At the velocity level, the constraint matrix dampens joint velocities and effectively stops motion when the system approaches the constraint

B. Daruish is with the Honda Research Institute, Mountain View, CA 94041, U.S.A., email: dariush@hra.com

G. Bin Hammam and D. E. Orin are with the Department of Electrical & Computer Engineering, The Ohio State University, Columbus, OH 43210, U.S.A., email: bin-hammam.1@osu.edu, orin.1@osu.edu

manifold. At the acceleration level, the constraint matrix dampens joint accelerations, forcing the joint velocities to zero near the constraint manifold. The solutions obtained from kinematic inversion at the velocity and acceleration levels are unified using a derivative feedback control mechanism, producing a kinematically constrained resolved acceleration control input.

The efficacy of the proposed algorithm is demonstrated by simulated experiments of task level upper-body human motion replication on the Honda humanoid robot. The reference human motions are complex, fast, and exhibit frequent self collision and joint limit violations when retargeted onto the robot without provisions for handling the kinematic constraints. The constraint handling, energetic studies, and tracking of operational tasks are studied for redundant and non-redundant task specifications based on the proposed constrained resolved acceleration controller.

II. Background

The equations of motion of a robotic mechanism in joint-space can be written as:

$$\boldsymbol{\tau} = \boldsymbol{H}(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\,\dot{\boldsymbol{q}} + \boldsymbol{\tau}_g(\boldsymbol{q}) + \boldsymbol{J}^T\mathbf{f}_e, \quad (1)$$

where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$, and $\boldsymbol{\tau}$ denote *(n × 1)* generalized vectors of joint position, velocity, acceleration and force variables, respectively. $\boldsymbol{H}(\boldsymbol{q})$ is an $(n \times n)$ joint-space inertia matrix. $\boldsymbol{C}$ is an $(n \times n)$ matrix such that $\boldsymbol{C}\,\dot{\boldsymbol{q}}$ is the *(n × 1)* vector of Coriolis and centrifugal terms. $\boldsymbol{\tau}_g$ is the *(n × 1)* vector of gravity terms. $\boldsymbol{J}$ is a Jacobian matrix, and $\mathbf{f}_e$ is the external spatial force acting on the system.

In the absence of an external force acting on the system, controlling the system described by Eq. 1 can be handled using a nonlinear model-based compensation with the following structure,

$$\boldsymbol{\tau} = \hat{\boldsymbol{H}}(\boldsymbol{q})\;\boldsymbol{\alpha} + \hat{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\,\dot{\boldsymbol{q}} + \hat{\boldsymbol{\tau}}_g(\boldsymbol{q}) \quad (2)$$

The vector $\boldsymbol{\alpha}$ represents a *resolved acceleration* in terms of joint variables. The notation $\hat{\cdot}$ denotes estimates of the components of the dynamic model. Provided the model parameters in Eq. 2 match those of Eq. 1, $\boldsymbol{\alpha}$ can be conveniently designed as a reference joint acceleration vector $\ddot{\boldsymbol{q}}_r$ which decouples and linearizes the closed loop system:

$$\boldsymbol{\alpha} = \ddot{\boldsymbol{q}}_r. \quad (3)$$

In practice, additional stabilizing terms may be added to the control law in Equation 2 [15]. The literature on task level tracking control of robotic systems offers a variety of techniques for computing $\boldsymbol{\alpha}$. The underlying mechanism for such control techniques is referred to as resolved acceleration control and illustrated in Figure 1. The aim is to design a control law to ensure tracking of a desired position and/or desired orientation of a set of task variables specified in Cartesian-space.

In general, task variables can operate in the full six dimensional task-space, three for position and three for orientation. Suppose there are N task variables, each task variable indexed by k. The spatial velocity vector of the k_{th} task variable is given by,

$$\mathbf{v}_k = \begin{bmatrix}\omega_k & \dot{\boldsymbol{p}}_k\end{bmatrix}^T, \quad (4)$$

where ω_k and $\dot{\boldsymbol{p}}_k$ are vectors corresponding to the angular velocity of the task frame and the linear velocity of the task position, respectively. The mapping between joint variables and task variables is obtained by considering the differential kinematics relating the two spaces,

$$\mathbf{v} = \boldsymbol{J}\,\dot{\boldsymbol{q}}, \quad (5)$$

$$\mathbf{a} = \boldsymbol{J}\,\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}\,\dot{\boldsymbol{q}}, \quad (6)$$

where $\mathbf{a} = \dot{\mathbf{v}}$, and $\mathbf{v}$ and $\boldsymbol{J}$ correspond to the augmented spatial velocity vector and the augmented task Jacobian matrix formed by concatenation of the individual tasks:

$$\mathbf{v} = \begin{bmatrix}\mathbf{v}_1^T & \cdots & \mathbf{v}_k^T & \cdots & \mathbf{v}_N^T\end{bmatrix}^T, \quad (7)$$

$$\boldsymbol{J} = \begin{bmatrix}\boldsymbol{J}_1^T & \cdots & \boldsymbol{J}_k^T & \cdots & \boldsymbol{J}_N^T\end{bmatrix}^T. \quad (8)$$

The augmented desired spatial velocity and acceleration vectors, denoted by $(\mathbf{v}_d, \mathbf{a}_d)$, can be constructed in the same fashion.

A. Velocity-based control

One way to compute $\boldsymbol{\alpha}$ from Cartesian inputs is by a first-order trajectory conversion to obtain $\dot{\boldsymbol{q}}_r$ followed by numerical differentiation:

$$\dot{\boldsymbol{q}}_r = \boldsymbol{J}^{+}(\mathbf{v}_d + \mathbf{K}\,\boldsymbol{e}), \quad (9)$$

$$\boldsymbol{\alpha} = \frac{d}{dt}(\dot{\boldsymbol{q}}_r), \quad (10)$$

where $\mathbf{K}$ is a positive definite gain matrix and the vector $\boldsymbol{e}$ describes the orientation and position error between the desired and computed task descriptors [1], [16]:

$$\boldsymbol{e} = \begin{bmatrix}\frac{1}{2}(\boldsymbol{n}\times\boldsymbol{n}_d + \boldsymbol{s}\times\boldsymbol{s}_d + \boldsymbol{c}\times\boldsymbol{c}_d)\\ \boldsymbol{p}_d - \boldsymbol{p}\end{bmatrix}, \quad (11)$$

where $\boldsymbol{R}_d = [\boldsymbol{n}_d \quad \boldsymbol{s}_d \quad \boldsymbol{c}_d]$ and $\boldsymbol{R} = [\boldsymbol{n} \quad \boldsymbol{s} \quad \boldsymbol{c}]$ correspond to the desired and computed unit vector triple representation of the task frame orientation, respectively.

Equations 9 and 10 may be used in the inverse dynamics control law of Eq. 2. A joint velocity feedback term is sometimes added in the velocity based control to provide stability [15],

$$\boldsymbol{\tau} = \hat{\boldsymbol{H}}(\boldsymbol{q})\;\boldsymbol{\alpha} + \hat{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\,\dot{\boldsymbol{q}} + \hat{\boldsymbol{\tau}}_g(\boldsymbol{q}) + \mathbf{K}_{q,v}(\dot{\boldsymbol{q}}_r - \dot{\boldsymbol{q}}), \quad (12)$$

where $\mathbf{K}_{q,v}$ is a positive definite joint-space gain matrix. While control at the velocity level is simple to implement and exhibits a large amount of practical robustness due to error stabilization terms both in task-space as well as joint-space, the method has three important disadvantages. First, the required numerical differentiation tends to amplify noise and can lead to undesirably large accelerations. Further, the behavior of the task dynamics cannot be easily specified as compared to that of acceleration-based control. Finally, this method ignores information about the desired accelerations,

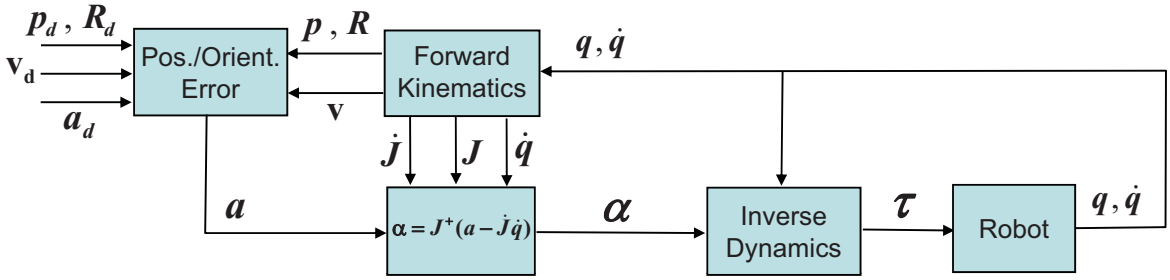

Fig. 1. Block diagram of the well-known resolved acceleration control system used for tracking control in task-space. The desired position and orientation of task variables are denoted by the vector $\boldsymbol{p}_d$ and the rotation matrix $\boldsymbol{R}_d$, respectively. The desired spatial velocities and accelerations of the task variables are denoted by $\mathbf{v}_d$ and $\mathbf{a}_d$, respectively. The vector $\boldsymbol{\alpha}$ represents the joint-space resolved acceleration vector that is commanded to the inverse dynamics control law.

which may lead to lower tracking performance or the need for high task-space gains and servo rates.

Nevertheless, the first-order kinematic inversion in Eq. 9 has important characteristics which can be exploited in enforcing kinematic constraints [7], [14]. As will be described in the next section, enforcing constraints at the velocity level can be used in constrained control at the acceleration level. This property is an important part of the proposed resolved acceleration control.

B. Acceleration-based control

An effective method to decouple and linearize the closed loop system is to form the resolved acceleration control input $\boldsymbol{\alpha}$ in Eq. 3 using the following relationship,

$$\boldsymbol{\alpha} = \boldsymbol{J}^{+}(\mathbf{a}_r - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}), \tag{13}$$

where the matrix $\boldsymbol{J}^{+}$ denotes the pseudo-inverse of $\boldsymbol{J}$ and

$$\mathbf{a}_r = \mathbf{a}_d + \mathbf{K}_v\,(\mathbf{v}_d - \mathbf{v}) + \mathbf{K}_p\,\boldsymbol{e}\,, \tag{14}$$

where $\mathbf{K}_p$ and $\mathbf{K}_v$ are positive definite diagonal position and velocity feedback gain matrices, respectively.

III. Handling Kinematic Constraints

The first and second-order closed loop inverse kinematics procedures described by Eq. 9 and Eq. 13, respectively, are effective methods to perform trajectory conversion from task-space to joint-space [16]. To a certain degree, the null space terms can be designed to perform secondary objectives such as keeping the robot away from kinematic constraints. In these methods, the gradients of an objective function are projected into the null-space of the Jacobian and constraints are handled as secondary tasks.

In our earlier work, we described an algorithm for solving the first-order constrained closed loop inverse kinematics (CCLIK) problem which proved to be an effective and stable solution for self collision avoidance [7], [14]. The method used a weighted least squares solution to constrain joints from violating their limits and to avoid collisions. In this section, we present an overview of the CCLIK method which will be used in the derivation of our constrained resolved acceleration control formulation described in Section III-D.

As described in [7], [14], a singularity robust constrained closed loop inverse kinematics formulation which produces constrained joint velocities, $\dot{\boldsymbol{q}}_c$, is given by

$$\dot{\boldsymbol{q}}_c = \boldsymbol{J}^{+}(\mathbf{v}_d + \mathbf{K}_p\ \boldsymbol{e}), \tag{15}$$

where $\boldsymbol{J}^{+}$ denotes the right pseudo-inverse of $\boldsymbol{J}$ weighted by the positive definite matrix $\boldsymbol{W}$,

$$\boldsymbol{J}^{+} = \boldsymbol{W}^{-1}\boldsymbol{J}^{T}\ (\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^{T})^{-1}. \tag{16}$$

Furthermore, if $\boldsymbol{J}$ is a square non-singular matrix, and $\boldsymbol{W}$ is the identity matrix, we can simply replace $\boldsymbol{J}^{+}$ by the standard matrix inverse $\boldsymbol{J}^{-1}$. In Eq. 16, kinematic constraints can be enforced by construction of an appropriate weight matrix, $\boldsymbol{W}$, to penalize and dampen joints whose motion directs the segments toward the constraint manifold. We construct $\boldsymbol{W}$ as a diagonal matrix whose elements are derived by considering the gradients of the joint limit and collision potential functions. The weight matrix $\boldsymbol{W}$ is influenced by the $n \times n$ joint limit weight matrix $\boldsymbol{W}_{JL}$ and the $n \times n$ collision avoidance weight matrix $\boldsymbol{W}_{COL}$. In the following section, we describe the formulation of the joint limit and collision avoidance matrices.

A. Joint limit constraints

Joint limit avoidance may be achieved by the proper selection of the diagonal matrix $\boldsymbol{W}_{JL}$ [7], [17]. To construct $\boldsymbol{W}_{JL}$, we consider a candidate joint limit potential function, denoted by $h(\boldsymbol{q})$, that has higher values when joints near their limit and tends to infinity at the joint limits. The gradient of h, denoted as ∇h, represents the joint limit gradient function, an $n \times 1$ vector whose entries point in the direction of the fastest rate of increase of h. The gradient associated with the i_{th} $(i = 1 \cdots n)$ degree of freedom is denoted by,

$$\nabla h_i = \frac{\partial h(\boldsymbol{q})}{\partial q_i}, \tag{17}$$

and described as follows [17],

$$\nabla h_i = \frac{(q_{i,max} - q_{i,min})^2\ (2q_i - q_{i,max} - q_{i,min})}{4(q_{i,max} - q_i)^2\ (q_i - q_{i,min})^2},$$

where q_i represents the generalized coordinates of the i_{th} degree of freedom, and $q_{i,min}$ and $q_{i,max}$ are the lower and

upper joint limits, respectively. The gradient ∇h_i is equal to zero if the joint is at the middle of its range and goes to infinity at either limit. As described in [17], we construct the joint limit weight matrix $\boldsymbol{W}_{JL}$ by an $n \times n$ diagonal matrix with diagonal elements w_{JL_i}. The scalars w_{JL_i} are defined by

$$\mathrm{w}_{JL_i} = \begin{cases} 1 + |\nabla h_i| & \text{if} \quad \Delta|\nabla h_i| \geq 0, \\ 1 & \text{if} \quad \Delta|\nabla h_i| < 0. \end{cases} \tag{18}$$

The term $\Delta|\nabla h_i|$ represents the change in the magnitude of the joint limit gradient function. A positive value indicates the joint is moving toward its limit while a negative value indicates the joint is moving away from its limit.

B. Collision constraints

Constructing the appropriate collision weight matrix $\boldsymbol{W}_{COL}$ is more complex. Consider collision between two unconnected segments (or segments which do not share a joint). Let d ($d \geq 0$) correspond to the minimum distance between two segment pairs. Let $f(\boldsymbol{q}, d)$ represent a candidate collision function that has a maximum value at $d = 0$ and decays exponentially toward zero as d increases.

We define the gradient of f, denoted as ∇f, as the collision gradient function, an $n \times 1$ vector whose entries point in the direction of the fastest rate of increase of f. The collision gradient function may be described as,

$$\nabla f = \frac{\partial f}{\partial \boldsymbol{q}} = \frac{\partial f}{\partial d} \frac{\partial d}{\partial \boldsymbol{q}}. \tag{19}$$

In case of self collisions, the second term in Eq. 19 may be computed as follows,

$$\frac{\partial d}{\partial \boldsymbol{q}} = \frac{1}{d} \left[\boldsymbol{J}_a^T (\boldsymbol{p}_a - \boldsymbol{p}_b) + \boldsymbol{J}_b^T (\boldsymbol{p}_b - \boldsymbol{p}_a) \right]^T, \tag{20}$$

where $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ represent position vectors, referred to the base, of the two collision points, and $\boldsymbol{J}_a$ and $\boldsymbol{J}_b$ are the associated Jacobian matrices. The coordinates $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ can be obtained using a standard collision detection software package [18]. In case of collision with the environment, the Jacobian associated with the environment collision point is zero. Similar to the joint limit weight function, $\boldsymbol{W}_{COL}$ may be constructed by an $n \times n$ diagonal matrix with diagonal elements w_{COL_i} ($i = 1 \cdots n$) defined by

$$\mathrm{w}_{COL_i} = \begin{cases} 1 + |\nabla f_i| & \text{if} \quad \Delta|\nabla f_i| \geq 0, \\ 1 & \text{if} \quad \Delta|\nabla f_i| < 0. \end{cases} \tag{21}$$

The elements of ∇f represent the degree to which each degree of freedom influences the distance to collision. It is appropriate to select a function f such that its gradient is zero when d is large and infinity when d approaches zero. One such candidate function is,

$$f = \rho e^{-\alpha d} d^{-\beta}, \tag{22}$$

where α and β are parameters to control the rate of decay and ρ controls the amplitude. The partial derivative of f with respect to d is

$$\frac{\partial f(\boldsymbol{q})}{\partial d} = -\rho e^{-\alpha d} d^{-\beta} (\beta \; d^{-1} + \alpha). \tag{23}$$

It follows that ∇f may be computed from Eqs. 19, 20, and 23.

The term $\Delta|\nabla f|$ in Eq. 21 represents the change in the magnitude of the collision gradient function. A positive value indicates the joint motion is causing the collision point to move toward collision while a negative value indicates the joint motion is causing the collision point to move away from collision. When a collision point is moving toward collision, the associated weight factor, described by the first condition in Eq. 21, becomes very large causing the joints affecting the motion of the collision point to slow down. When two segments are about to collide, the weight factor is near infinity and the joints contributing to collision virtually stop. If two segments are moving away from collision, there is no need to restrict or penalize the motions. In this scenario, the second condition in Eq. 21 allows the joint to move freely.

Suppose a total of N_c segment pairs are checked for self collision. Let j ($j = 1 \cdots N_c$) be the index of the j_{th} collision pair, and d_j the minimum distance between the two colliding segments. Let $\boldsymbol{p}_{a_j}$ and $\boldsymbol{p}_{b_j}$ represent the coordinates, referred to the base, of the two colliding point pairs for the j_{th} collision pair. The candidate potential function for each collision pair is given by,

$$f_j = \rho_j e^{-\alpha_j d_j} d_j^{-\beta_j}. \tag{24}$$

Its gradient can be computed as before,

$$\nabla f_j = \frac{\partial f_j}{\partial \boldsymbol{q}} = \frac{\partial f_j}{\partial d_j} \frac{\partial d_j}{\partial \boldsymbol{q}}. \tag{25}$$

It follows that the collision weight matrix for each collision pair, denoted by $\boldsymbol{W}_{COL_j}$ can be computed as outlined above. The collision weight matrix is comprised of the contribution of each collision pair as given by,

$$\boldsymbol{W}_{COL} = \frac{1}{N_c} \sum_{j=1}^{N_c} \boldsymbol{W}_{COL_j}. \tag{26}$$

C. Composite constraint matrix

The next step is to construct a composite constraint weight matrix $\boldsymbol{W}$ comprised of the joint limit weight matrix $\boldsymbol{W}_{JL}$ and the collision weight matrix $\boldsymbol{W}_{COL}$. While a rigorous formulation of this integration is warranted and is currently being examined, we present a simple and effective solution based on our empirical results. The proposed composite weight matrix is given by,

$$\boldsymbol{W} = a \; \boldsymbol{W}_{JL} + (1 - a) \; \boldsymbol{W}_{COL}, \tag{27}$$

where a is a scalar index which can be used to modulate the contribution of the joint limit weight matrix and the collision weight matrix. We have found that the following index is effective for the various motions considered,

$$a = \frac{1}{(N_c + 1)}. \tag{28}$$

D. Constrained resolved accelerations

Let $\boldsymbol{\alpha}_c$ be a vector corresponding to the constrained resolved accelerations, which may be folded into the dynamic equations of motion as in the control law of Eq. 2,

$$\boldsymbol{\tau} = \hat{\boldsymbol{H}}(\boldsymbol{q})\ \boldsymbol{\alpha}_c + \hat{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\ \dot{\boldsymbol{q}} + \hat{\boldsymbol{\tau}}_g(\boldsymbol{q})\ . \tag{29}$$

We construct $\boldsymbol{\alpha}_c$ as follows:

$$\boldsymbol{\alpha}_c = \boldsymbol{\alpha} + \boldsymbol{G}_v(\dot{\boldsymbol{q}}_c - \dot{\boldsymbol{q}}), \tag{30}$$

where $\dot{\boldsymbol{q}}_c$ is computed from the first-order constrained kinematics inversion of Eqs. 15, and $\boldsymbol{G}_v$ is a diagonal positive definite joint velocity gain matrix. For systems which exhibit redundancy, it is advantageous to exploit the redundancy and include a null-space projection term when computing $\boldsymbol{\alpha}$, such that

$$\boldsymbol{\alpha} = \boldsymbol{J}^{+}(\mathbf{a}_r - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}) + \boldsymbol{N}\boldsymbol{\zeta}, \tag{31}$$

where $\boldsymbol{\zeta}$ is an arbitrary vector and $\boldsymbol{N} = \boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J}$ projects $\boldsymbol{\zeta}$ into the null space of $\boldsymbol{J}$. The closed loop task error dynamics of the control law given by Eq. 30 is described by,

$$\ddot{\boldsymbol{e}} + \mathbf{K}_v\dot{\boldsymbol{e}} + \mathbf{K}_p\boldsymbol{e} = -\boldsymbol{J}\boldsymbol{G}_v(\dot{\boldsymbol{q}}_c - \dot{\boldsymbol{q}}). \tag{32}$$

The left side of Eq. 32 describes the task error dynamics. The right side characterizes the constraint error dynamics, i.e. the error in tracking the reference constraint velocities $\dot{\boldsymbol{q}}_c$. Equation 32 reveals that as long as $\dot{\boldsymbol{q}}_c - \dot{\boldsymbol{q}}$ is non-zero, the terms on the right side will interfere with the task-space tracking error dynamics. Provided there are sufficient number of degrees of freedom to execute the tasks and satisfy the constraints, the gain $\boldsymbol{G}_v$ can be modulated to force the task-space tracking error to zero. The larger the gain, the faster the convergence. However, an excessively large $\boldsymbol{G}_v$ may introduce instability. The vector $\boldsymbol{\zeta}$ may be keenly designed to further stabilize the self-motion. A natural choice is given by,

$$\boldsymbol{\zeta} = \boldsymbol{G}_{v_N}(\dot{\boldsymbol{q}}_c - \dot{\boldsymbol{q}}), \tag{33}$$

where $\boldsymbol{G}_{v_N}$ is a diagonal positive definite matrix. We select $\boldsymbol{G}_{v_N} = \boldsymbol{G}_v$. Compliance and damping characteristics of the self-motion may be further improved by designing an adaptive $\boldsymbol{G}_v$ to have small values when the system is away from the constraints and large values near the constraints. This issue will be explored in more detail in the future.

In practice, considering the occurrence of singularities in the matrix $\boldsymbol{J}$, the pseudo-inverse is replaced with

$$\boldsymbol{J}^{*} = \boldsymbol{W}^{-1}\boldsymbol{J}^{T}\ (\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^{T} + \lambda^2\ \boldsymbol{I})^{-1}, \tag{34}$$

where $\boldsymbol{J}^{*}$ denotes the singularity robust right pseudo-inverse of $\boldsymbol{J}$ regularized by the damping factor λ. Also in Eq. 34, $\boldsymbol{W}$ represents the constraint weight matrix as given in Eq. 27.

Let us examine the effect of $\boldsymbol{W}$ in the solution of $\boldsymbol{\alpha}$ in Eq. 31. Recall that in the first-order inverse kinematics solution described by Eq. 15, the weight matrix $\boldsymbol{W}$ forced any joint velocity contributing to violation of a kinematic constraint to zero. In contrast, the use of the constraint matrix $\boldsymbol{W}$ in Eq. 31 forces the acceleration $\boldsymbol{\alpha}$ contributing to violation of a kinematic constraint to zero. Therefore, the use of $\boldsymbol{W}$ in Eq. 31 does not guarantee that a constraint will be maintained. Nevertheless, this approach effectively dampens motion as the robot configuration approaches the constraint manifold, providing additional stability and robustness to the controller.

Based on the above discussion, $\boldsymbol{\alpha}$ is computed as follows, depending on availability of redundant degrees of freedom to perform the original tasks:

$$\boldsymbol{\alpha} = \begin{cases} \boldsymbol{J}^{+}(\mathbf{a}_r - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}) + \boldsymbol{N}\boldsymbol{\zeta} & \text{if} \quad m < n, \\ \boldsymbol{J}^{+}(\mathbf{a}_r - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}) & \text{if} \quad m > n. \end{cases} \tag{35}$$

where m is the dimension of the primary (operational) tasks and n is the total number of available degrees of freedom.

Equations 30 and 35 constitute the generalized constrained resolved acceleration vectors. The solution is feasible even when there are insufficient degrees of freedom to execute both the constraints and the operational tasks. In contrast, approaches that use null space gradient projections to satisfy secondary objectives break down when the Jacobian of the primary objective becomes rank deficient due to task singularities.

IV. Simulation Results

The efficacy of the proposed control system was examined through simulated experiments involving reproduction of upper-body human motion on the humanoid robot model. The total upper-body degrees of freedom utilized in the robot model is $n = 11$ (three at the torso for orientation, three rotations at each shoulder, and one rotation at each elbow). The waist translation is considered as a prescribed motion and therefore treated as a known degree of freedom in the inverse kinematics procedure.

The desired task motion to be executed by the robot is derived from a set of upper-body motions obtained from the Carnegie Mellon University (CMU) human motion capture data base [19]. We consider four highly dynamic and complex motions that produce multiple, simultaneous violations of joint limits and self collisions when the motion, in its raw form, is transferred to the robot model. The four motions include a dancing sequence, exercise sequence, reaching sequence, and boxing sequence.

A set of eight $3D$ marker positions were extracted from each of the four aforementioned motions. These marker positions correspond to the waist joint, two shoulder joints, two elbow joints, two wrist joints, and the neck joint. Motion description involving orientation variables (such as hand orientation) was not included and not considered in this study. The marker positions were low pass filtered, interpolated, and scaled to the robot dimensions. The resulting motion corresponds to a set of possible desired task descriptors which are to be tracked using the proposed constrained resolved acceleration control framework. We performed simulated experiments on two possible task description sets, as described below.

1) **Experiment 1: Five degrees of redundancy**
   The specified task involves two position descriptors for

the left and right wrist positions. The task dimension is $m = 6$, providing five degrees of redundancy.

2) **Experiment 2: Overconstrained tasks**
The specified task involves seven position descriptors for the two shoulders, two elbows, two wrists, and neck. The task dimension is $m = 21$ [1]. Since $m > n$, this scenario represents an over-constrained task specification.

In all experiments, the values for the feedback gain parameters are: $\mathbf{K} = 30\boldsymbol{I}$, $\mathbf{K}_p = 300\boldsymbol{I}$, $\mathbf{K}_v = 2\sqrt{\mathbf{K}_p}$, $\boldsymbol{G}_v = \boldsymbol{G}_{v_N} = 30\boldsymbol{I}$ ($\boldsymbol{I}$ is the identity matrix). For all collision pairs, the parameters used for the collision avoidance function in Eq. 24 are: $\rho_j = 1$, $\beta_j = 2$, and $\alpha_j = 50$. The selected damping factor is $\lambda = 0.1$.

Consider a human to robot motion retargeting scenario of a highly dynamic dancing sequence involving 360 degree twisting. In Fig. 2, we show a histogram of the number of constraint violations at different motion periods if the kinematic constraints are not enforced. Each constraint violation translates to a reduction of one degree of freedom. In Experiment 1, eleven degrees of freedom are used to execute two operational tasks at the wrists (requiring six degrees of freedom). Therefore, redundancy can only be established if the constraint violations are less than five. As shown in the figure, there are periods in the latter half of the motion where the number of constraint violations exceeds five. During these periods, the robot does not have sufficient degrees of freedom to execute both the constraints and the operational tasks. This simulation illustrates that even when the operational tasks exhibit significant amount of redundancy, it is not always possible to determine actual redundancy (including the constraints) if the motion is not planned in advance. This is particularly true for complex robot structures such as humanoids.

To illustrate the effectiveness of the proposed approach in satisfying constraints in the absence of redundancy, consider the simulation results in Figure 3. In particular, the plots show self-collision distance results for the scenario described in Experiment 2 for the dancing sequence. The figure shows the minimum distances between thirty segments pairs in the upper-body after the motion has been dynamically filtered to avoid self-collisions. The simulated robot snapshots are approximately aligned with the time-line (frame-number). The yellow highlighted region in the snapshots indicates occurrence of self-collisions before filtering. The minimum distance of collision points between the left hand and right hand segment is labeled for reference. The minimum distance profiles indicate a pattern having roughly three peaks and three valleys. As observed in the snapshots, the peaks correspond to large separations between segments when the arms are extended away from the body and there are no self-collisions. The valleys correspond to periods when the arms are close to the body, triggering multiple self-collisions. Note that the minimum distance plots are always positive,

[1]In all experiments, the waist position is specified as a floating base, and not considered in the calculation of the task dimension.

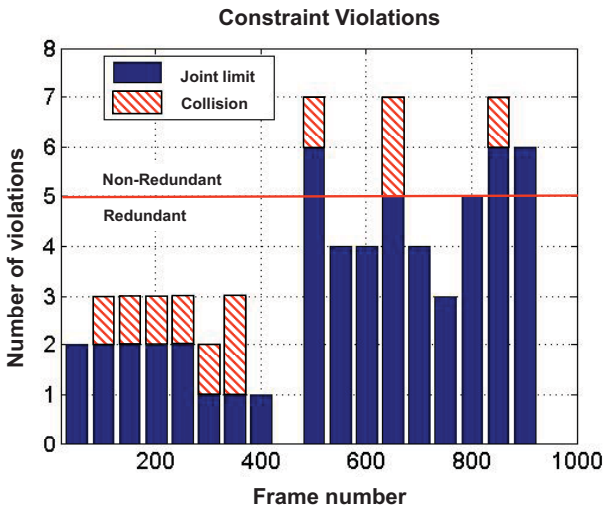

Fig. 2. Histogram of constraint violations for the dancing sequence. For executing two operational tasks (Experiment 1), redundancy is established only if the number of constraint violations are less than five.

indicating no penetration of the segment pairs.

We also simulated the task tracking error and the total average power consumption of the proposed resolved acceleration control for the four motion sequences. In particular, we compared the results using constrained resolved acceleration control at the velocity level, versus the results at the acceleration level.

Figures 4 and 5 illustrate results for the scenario involving execution of two operational tasks at the left and right wrists (Experiment 1). Although there are five degrees of redundancy with respect to the operational tasks, actual redundancy does not exist during parts of each motion when constraints are enforced. If there is no redundancy, the tasks cannot be perfectly tracked, introducing task errors. Additionally, the damping term in the damped least squares inverse introduces task error.

Although we expect improved task error performance for trajectory conversion at the acceleration level, our simulation of the task error does not reveal a significant difference between the two methods. One explanation is that we have used the same damping factor in both cases. Since we can generally tolerate lower damping factors when performing trajectory conversion at the acceleration level, we can observe improved performance by reducing the damping factor. Also, we believe implementation on a physical robot will further validate the task tracking benefits of trajectory conversion at the acceleration level.

Figure 5 shows a comparison of the total average power for the four motions. The results show a reduction of the calculated power when trajectory conversion is performed at the acceleration level. The efficiency is attributed to two factors. First, the acceleration based approach does not require numerical differentiation of the constrained joint rates, which may introduce torque spikes. Second, additional damping

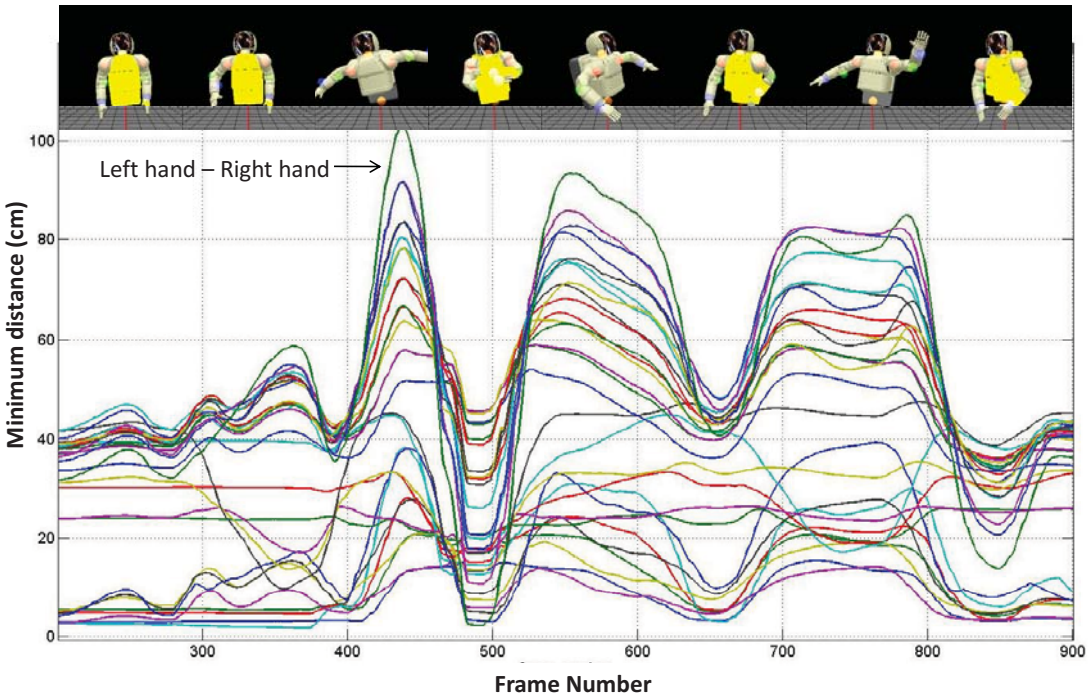

Fig. 3. CMU dancing sequence: the minimum distances between thirty segment pairs after the motion is dynamically filtered to avoid collisions. The snapshots are approximately aligned with the time-line (frame-number). The yellow highlighted region indicates occurrence of collisions before filtering. The minimum distance between the left hand and right hand is labeled.

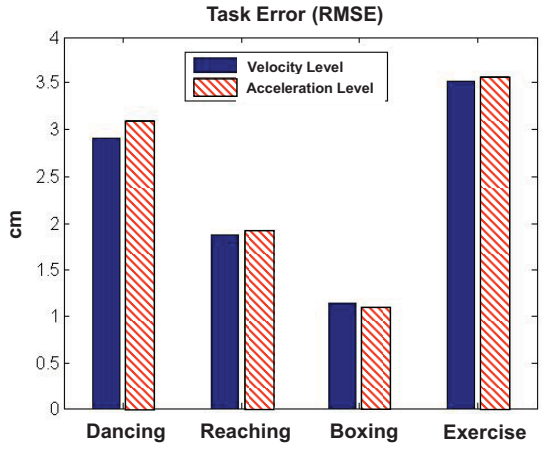

Fig. 4. Root mean squared task error for Experiment 1.

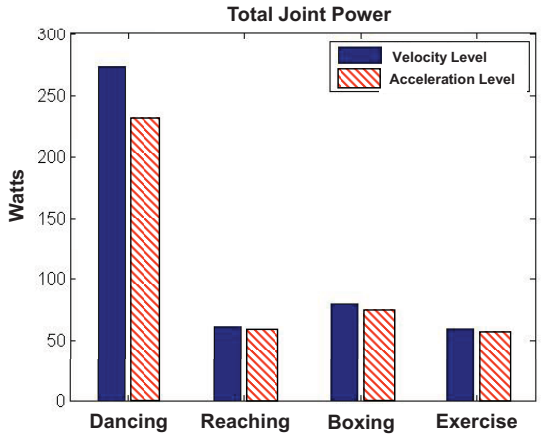

Fig. 5. The calculated absolute average power (over all degrees of freedom) for Experiment 1.

is introduced by using the constraint matrix $\boldsymbol{W}$ in the weighted pseudo-inverse solution at the acceleration level. The resulting accelerations are damped when the motion reaches a constraint manifold, producing a smoother dynamic response.

The root mean squared task error and the average power results for the over-constrained task specification scenario (Experiment 2) are shown in Figs. 6 and 7, respectively. In this scenario, the task error involves the tracking error of the seven upper-body operational tasks. Since the task motion is over-specified, there is an increase in the task error as compared to the results of Experiment 1. A comparison of the task error and total power between velocity level control and acceleration level control does not show a significant difference. This is attributed to the absence of null-space motion which prevents widely varying joint space solutions to emerge, regardless of whether the control action is performed at the velocity or the acceleration level. The

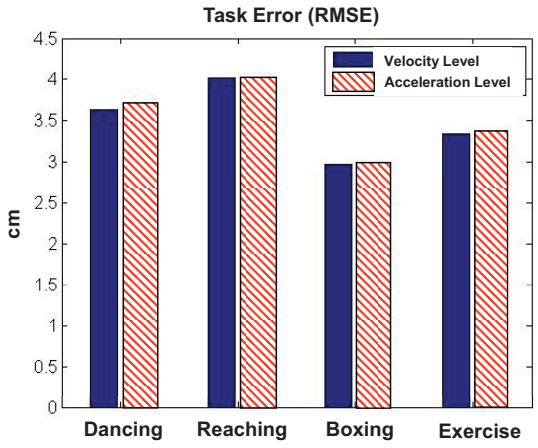

Fig. 6. Root mean squared task error for Experiment 2.

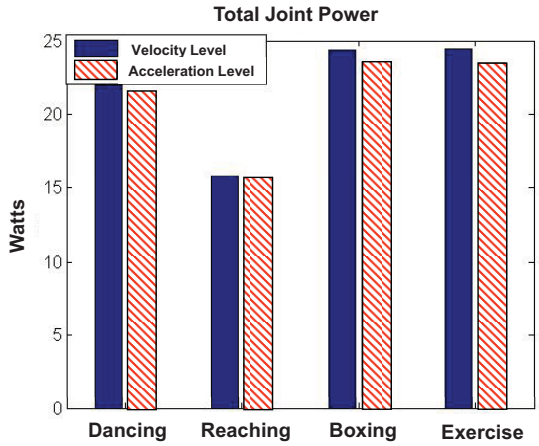

Fig. 7. The calculated absolute average power (over all degrees of freedom) for Experiment 2.

slight improvement in the power curves of the acceleration based control is likely to be attributed to improved numerical stability.

Interestingly, in comparison with the results of Experiment 1, the total joint power is significantly reduced when the task motion is over-specified (Experiment 2). Since the task motion is obtained from a human demonstrator, the energetics of the resulting retargetted motion onto a human-like humanoid robot is characterized by a highly efficient human motion dynamics. In contrast, when the task motion is under-specified (Experiment 1), the solution of the self-motion based on kinematic inversion does not minimize the kinetic energy. To improve the energetics in Experiment 1, the use of a dynamically consistent weight matrix is warranted.

V. Summary

A generalized constrained resolved acceleration control algorithm has been proposed. The approach is suitable for execution of operational tasks in the presence of kinematic constraints even when a robotic systems does not exhibit redundancy. Unlike gradient projection methods, satisfaction of the constraints is guaranteed and is numerically stable. This problem is particularly suitable for online control of humanoid robot motion, where the possibility of a large number of kinematic constraint violations exists and is not known if the motion is not planned in advance. Notable extensions of the current algorithm are to incorporate dynamic constraints, such as torque limits and balance constraints (in the case of bipedal humanoids).

References

[1] J.Y.S. Luh, M.W. Walker, and R.P.C. Paul. Resolved-acceleration control of mechanical manipulators. *IEEE Transactions on Automatic Control*, 25:468–474, 1980.

[2] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal of Robotics and Automation*, RA-3(1):43–53, 1987.

[3] A. A. Maciejewski and C. A. Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *International Journal of Robotics Research*, 4:109–117, 1985.

[4] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1):90–98, 1986.

[5] H. Sugiura, M. Gienger, H. Janssen, and C. Goerick. Real-time collision avoidance with whole body motion control for humanoid robots. In *IEEE/RJS Int. Conf. on Intelligent Robots and Systems*, 2007.

[6] O. Stasse, A. Escande, N. Mansard, S. Miossec, P. Evrard, and A. Kheddar. Real-time (self)-collision avoidance task on a HRP-2 humanoid robot. In *IEEE Int. Conf. Robotics and Automation*, pages 3200–3205, Pasadena CA, 2008.

[7] B. Dariush, M. Gienger, A. Arumbakkam, Y. Zhu, B. Jian, K. Fujimura, and C. Goerick. Online transfer of human motion to humanoids. *International Journal of Humanoid Robotics*, 6:265–289, 2009.

[8] Y. Nakamura. *Advanced Robotics, Redundancy and Optimization*. Addisson-Wesley, 1991.

[9] P. Hsu, J. Hauser, and S. Sastry. Dynamic control of redundant manipulators. *J. Robotic Systems*, 6(2):133–148, 1989.

[10] L. Sentis and O. Khatib. A whole-body control framework for humanoids operating in human environments. In *IEEE Int. Conf. Robotics and Automation*, Orlando, FL, 2006.

[11] A. Nakazawa, S. Nakaoka, K. Ikeuchi, and K. Yokoi. Imitating human dance motions through motion structure analysis. In *IEEE/RJS Int. Conf. on Intelligent Robots and Systems*, pages 2539–2544, Lausanne, Switzerland, 2002.

[12] S. Tak, O. Song, and H. Ko. Motion balance filtering. *Comput. Graph. Forum. (Eurograhics)*, 19(3):437–446, 2000.

[13] S. Tak and H. Ko. A physically-based motion retargeting filter. *ACM Trans. on Graphics*, 24(1):98–117, 2005.

[14] B. Dariush, Y. Zhu, A. Arumbakkam, and K. Fujimura. Constrained closed loop inverse kinematics. *Int. Conf. Robotics and Automation*, 2010.

[15] J. Nakanishi, R. Cory, M. Mistry, and J. Peters. Operational space control: A theoretical and empirical comparison. *International Journal of Robotics Research*, 27(6):737–757, 2008.

[16] F. Chiaverini, B. Siciliano, and O. Egeland. Review of damped least-squares inverse kinematics with experiments on an industrial robot manipulator. *IEEE Trans. Control Systems Tech.*, 2(2):123–134, 1994.

[17] T. F. Chan and R. V. Dubey. A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators. *IEEE Transactions on Robotics and Automation*, 11(2), 1995.

[18] UNC Chapell Hill: Swift++ Library. Speedy walking via improved feature testing for non-convex objects. Internet page. http://www.cs.unc.edu/ geom/SWIFT++/.

[19] Carnegie Mellon University. CMU graphics lab motion capture database. Internet page. http://mocap.cs.cmu.edu/.