2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Constrained Closed Loop Inverse Kinematics

Behzad Dariush Youding Zhu Arjun Arumbakkam Kikuo Fujimura

Abstract— **This paper introduces a kinematically constrained closed loop inverse kinematics algorithm for motion control of robots or other articulated rigid body systems. The proposed strategy utilizes gradients of collision and joint limit potential functions to arrive at an appropriate weighting matrix to penalize and dampen motion approaching constraint surfaces. The method is particularly suitable for self collision avoidance of highly articulated systems which may have multiple collision points among several segment pairs. In that respect, the proposed method has a distinct advantage over existing gradient projection based methods which rely on numerically unstable null-space projections when there are multiple intermittent constraints. We also show how this approach can be augmented with a previously reported method based on redirection of constraints along virtual surface manifolds. The hybrid strategy is effective, robust, and does not require parameter tuning. The efficacy of the proposed algorithm is demonstrated for a self collision avoidance problem where the reference motion is obtained from human observations. We show simulation and experimental results on the humanoid robot** ASIMO**.**

I. Introduction

Motion planning with kinematic constraints has been an important and widely studied problem since the inception of robotics technology. The majority of early research in this area was focused on obstacle avoidance, typically for applications involving mobile robot navigation and industrial manipulation [1], [2]. In these applications, the workspace was often predefined, static, or slowly varying. Moreover, application developers typically adopted the philosophy of segregating the workspace of robots and humans as a safety countermeasure to avoid collisions with humans.

Today, the field of robotics is moving towards development of high degree of freedom, human-like, and personal robots, which are often designed to share a common workspace and physically interact with humans. Such robots are often highly redundant which fundamentally adds new capabilities (self-motion and subtask performance capability). However, increased redundancy has also added new challenges for constraining the internal motion to avoid joint limits and self collisions. With these challenges, researchers have become increasingly aware of the need for robust joint limit and collision avoidance strategies to accommodate such applications.

In particular, self collision avoidance, which was largely overlooked or not required when obstacle avoidance strategies were first developed, has recently become an important topic of research [3], [4], [5]. Enforcing self collision constraints is challenging for humanoid robots performing human-like tasks, especially in a real-time or online setting. The strategy must not only accommodate multiple colliding segments simultaneously, but also tolerate smaller collision distance thresholds than those established for early obstacle avoidance algorithms. In addition, such constraints should not significantly alter the reference or originally planned motion. This is particulary important in applications involving reproduction of robot motion from observed human motion [5], [6], [7], [8].

This paper introduces an online, kinematically constrained motion generation algorithm for motion control of robots or other articulated rigid body systems in task space. The strategy is formulated in the framework of the closed loop inverse kinematics (CLIK) algorithm [9]. The presented CLIK formulation is based on a weighted and regularized pseudo-inverse solution which computes joint variables given a set of motion descriptors specified in Cartesian space.

The first contribution of this paper is the construction of an appropriate weighting matrix which results in collision free motion. The weighting matrix is based on the gradient of a potential function which penalizes and dampens joints whose motion directs the segments toward joint limit and collision constraints. The proposed method is particularly suitable for self collision avoidance of highly articulated systems which may have multiple and changing collision points among several segments. In this respect, the proposed method has a distinct advantage over existing gradient projection based methods which are susceptible to numerical instability when dealing with multiple and intermittently colliding segment pairs [1], [10].

The second contribution of this paper is to show that the proposed method may be augmented with our previously reported collision avoidance strategy to improve robustness [5]. In particular, the hybrid strategy is effective in guaranteeing collision free motion without the need to tune parameters for the construction of collision potential functions.

To demonstrate the effectiveness of the proposed algorithm, we illustrate simulated and experimental results on the Honda humanoid robot ASIMO, where the reference motion is obtained from captured human motion. The reference motions are complex, fast, and exhibit frequent self collisions under the traditional CLIK motion generation. With the proposed algorithm, the motions are shaped in real time to produce kinematically constrained motions. The algorithm is also implemented in our online motion retargeting framework and demonstrated on the ASIMO platform.

B.Dariush, A. Arumbakkam, and K. Fujimura are at the Honda Research Institute, USA, 800 California St. Suite 300, Mountain View CA 94041 dariush(aarumbakkam)(kfujimura)@honda-ri.com

Y. Zhu is at Department of Computer Science, The Ohio State University, zhu.81@osu.edu

II. Closed Loop Inverse Kinematics

Closed loop inverse kinematics (CLIK) is an effective method to perform trajectory conversion from task space to joint space [9]. A CLIK algorithm uses a set of task descriptors as input and estimates the robot joint commands that minimize the tracking error between the reference and predicted Cartesian motion. In this section, we present an overview of the CLIK formulation.

Let n represent the number of robot degrees of freedom. Let the vector $q = [q_1, \cdots, q_n]^T$ describe degrees of freedom which fully characterize the configuration space of the robot. Suppose the task variables operate the full six dimensional task space, three for position and three for orientation and the number of task variables is n_t. Let k $(k = 1 \cdots n_t)$ be the index of the spatial velocity vector $\dot{x}_k$ corresponding to the k_{th} task descriptor. The associated Jacobian is given by $J_k = \frac{\partial x_k}{\partial q}$.The mapping between the joint space velocities and task space velocities is obtained by considering the differential kinematics relating the two spaces

$$\dot{x}_k = J_k(q)\,\dot{q}. \tag{1}$$

The spatial velocity vector is defined by

$$\dot{x}_k \;=\; \begin{bmatrix}\omega_k & \dot{p}_k\end{bmatrix}^T, \tag{2}$$

where ω_k and $\dot{p}_k$ are vectors corresponding to the angular velocity of the task frame and the linear velocity of the task position referenced to the base frame, respectively. We construct all task variables to form an augmented spatial velocity vector $\dot{x}$ and an augmented Jacobian matrix J as follows:

$$\dot{x} \;=\; \begin{bmatrix}\dot{x}_1^T & \cdots & \dot{x}_k^T & \cdots & \dot{x}_{n_t}^T\end{bmatrix}^T, \tag{3}$$

$$J \;=\; \begin{bmatrix}J_1^T & \cdots & J_k^T & \cdots & J_{n_t}^T\end{bmatrix}^T. \tag{4}$$

The Jacobian matrix may be decomposed to its rotational and translational components, denoted by J_o and J_p, respectively.

$$J = \begin{bmatrix}J_o \\ J_p\end{bmatrix}. \tag{5}$$

If for example, only a position descriptor p_k is observable, then the parameters in Equation 1 can be modified to $\dot{x}_k = \dot{p}_k$, and $J_k = J_{p_k}$.

Let the velocity of the reference task descriptors in the augmented space be described by

$$\dot{x}_r \;=\; \begin{bmatrix}\dot{x}_{r_1}^T & \cdots & \dot{x}_{r_k}^T & \cdots & \dot{x}_{r_{n_t}}^T\end{bmatrix}^T. \tag{6}$$

Given $\dot{x}_{r_k} = \dot{x}_k$, the standard inverse kinematics procedure involves calculation of $\dot{q}$ from Equation (1). A feedback error term is typically added to correct for numerical drift, resulting in the following closed loop inverse kinematics equation:

$$\dot{q} = J^*(\dot{x}_r + K\ e), \tag{7}$$

where J^* denotes the regularized right pseudo-inverse of J weighted by the positive definite matrix W, and defined by:

$$J^* = W^{-1}J^T\ (JW^{-1}J^T + \lambda^2 I)^{-1}. \tag{8}$$

The parameter $\lambda > 0$ is a damping term, and I is an identity matrix. The vector $\dot{x}_{r_k} = [\omega_{r_k} \quad \dot{p}_{r_k}]^T$ represents the spatial velocity. The rate of convergence of the error for the k_{th} descriptor is controlled by K_k, a diagonal 6×6 positive definite gain matrix. The vector e is the concatenation of the individual error terms $e_i = [e_{o_k}\ e_{p_k}]^T$, where($e_{p_k}$) and ($e_{o_k}$) are the position and orientation error vectors between the reference and computed task descriptors [11], [12], [9].

III. Constrained Closed Loop Inverse Kinematics

The CLIK algorithm described in the previous section does not explicitly enforce kinematic constraints. For redundant systems, a prioritized inverse kinematics strategy is often used whereby constraints can be projected onto the null space of the higher priority tasks [13]. Formulating constraints as a secondary task cannot guarantee that constraints will be satisfied. One solution is to enforce constraints as highest priority operation and project the operational tasks into the constraint null-space [10]. However, when there are multiple simultaneous constraints to satisfy (such as self collision avoidance in a humanoid robot), there may be insufficient degrees of freedom to satisfy both the constraints, and operational tasks. Task and algorithmic singularities arise which create numerical instabilities. The intermittent nature of the constraints could lead to further numerical instability especially with null-space projection methods.

In our earlier work, we described an algorithm for solving the constrained closed loop inverse kinematics problem which proved to be an effective and stable solution for self collision avoidance [5]. The method used a weighted least squares solution to constrain joints from violating their limits and a virtual surface control method to redirect the motion of segments away from collisions. While the virtual surface control method is effective in preventing collisions, the redirected task motion may be farther away from the reference motion than necessary.

In this section, we extend the weighted least squares solution used for joint limit avoidance to also handle self collision constraints. The objective is to generate the joint motion which minimizes the Cartesian tracking error subject to joint limit and collision constraints. The approach is based on time-local information and is targeted to real time control.

The proposed method involves construction of an appropriate weighting matrix, W, in Equation 8 to penalize and dampen joints whose motion directs the segments toward the constraint manifold. We construct W as a diagonal matrix whose elements are derived by considering the gradients of the joint limit and collision potential functions. The matrix W is influenced by the $n \times n$ joint limit weighting matrix W_{JL} and the $n \times n$ collision avoidance weighting matrix W_{COL}.

A. Joint limit constraints

In this section, we review the weighted least squares solution to handle joint limit constraints as reported in [5], [14]. Joint limit avoidance is achieved by the proper selection of the weighting matrix W in Equation 8. The solution

considers a candidate joint limit function, denoted by $H(q)$, 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.

$$\nabla H = \frac{\partial H}{\partial q} = \left[\frac{\partial H}{\partial q_1} \quad , \cdots , \quad \frac{\partial H}{\partial q_n} \right]. \tag{9}$$

The gradient of a candidate H associated with the i_{th} ($i = 1 \cdots n$) degree of freedom is given by [14],

$$\frac{\partial H(q)}{\partial q_i} = \frac{(q_{i,max} - q_{i,min})^2 \ (2q_i - q_{i,max} - q_{i,min})}{4(q_{i,max} - q_i)^2 \ (q_i - q_{i,min})^2},$$

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 $\frac{\partial H(q)}{\partial q_i}$ is equal to zero if the joint is at the middle of its range and goes to infinity at either limit. As described in [14], we construct the joint limit weighting matrix W_{JL} by an $n \times n$ diagonal matrix with diagonal elements w_{JL_i}. The scalars w_{JLi} are defined by

$$\mathrm{w}_{JL_i} = \begin{cases} 1 + |\frac{\partial H}{\partial q_i}| & if \quad \Delta|\partial H/\partial q_i| \geq 0, \\ 1 & if \quad \Delta|\partial H/\partial q_i| < 0. \end{cases} \tag{10}$$

The term $\Delta|\partial H/\partial q_i|$ 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. When a joint moves toward its limit, the associated weighting factor, described by the first condition in Equation 10, becomes very large causing the motion to slow down. When the joint nearly reaches its limit, the weighting factor approaches infinity and the corresponding joint virtually stops. If the joint is moving away from the limit, there is no need to restrict or penalize the motions. In this scenario, the second condition in Equation (10) allows the joint to move freely.

B. Collision constraints

Collision avoidance may be categorized as self-collision avoidance or obstacle avoidance. Self collision avoidance involves two segments coming into contact; whereas obstacle avoidance involves contact between an object in the environment (See Figure 1). Self collision avoidance may be classified as one of two types: 1) avoiding collision between two connected segments, and 2) avoiding collision between two unconnected segment pairs. By connected segment pairs, we imply that the two segments are connected at a common joint and assume that the joint is rotational.

If two segments are connected, self collision may be handled by limiting the joint range. Joint limits for self collision avoidance need not correspond to the physical joint limits; rather, they may be more conservative virtual joint limits whose values are obtained by manually verifying the bounds at which collision occurs. Therefore, for two segments connected by a rotation joint, joint limit avoidance and self collision avoidance may be performed by using the same formulation presented in Section III-A.

For those segment pairs that do not share the same joint, the collision avoidance strategy must consider the minimum Euclidian distance between the two colliding segments as an input. Let d ($d \geq 0$) correspond to the minimum distance between two segment pairs. Let $P(q, d)$ 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 P, denoted as ∇P, as the collision gradient function, an $n \times 1$ vector whose entries point in the direction of the fastest rate of increase of P.

$$\nabla P = \frac{\partial P}{\partial q} = \left[\frac{\partial P}{\partial q_1} \quad , \cdots , \quad \frac{\partial P}{\partial q_n} \right]. \tag{11}$$

The collision gradient can be computed as follows

$$\frac{\partial P}{\partial q} = \frac{\partial P}{\partial d} \ \frac{\partial d}{\partial q}. \tag{12}$$

Consider first the case of self collision, or collision between two bodies of the articulated chain. In this scenario, it can be shown that the second term in Equation 12 is

$$\frac{\partial d}{\partial q} = \frac{1}{d} \left[J_a^T (p_a - p_b) + J_b^T (p_b - p_a) \right]^T, \tag{13}$$

where p_a and p_b represent position vectors, referred to the base, of the two collision points, and J_a and J_b are the associated Jacobian matrices. The coordinates p_a and p_b can be obtained using a standard collision detection software. In this work, we use the SWIFT++ library [15]. Figure 1 illustrates an example of a set of collision points between the forearm and trunk.

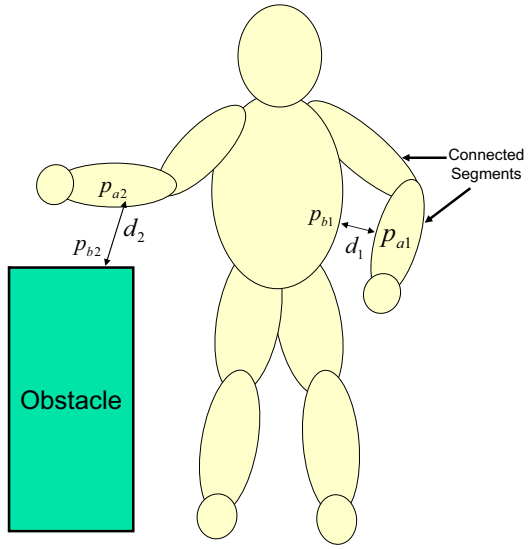

Fig. 1. Illustration of self collision and collision with an obstacle.

If one of the collision point pairs, for example point b, represents a point on an environment obstacle that is not attached to the articulated structure (See Figure 1), then 13 simplifies to

$$\frac{\partial d}{\partial q} = \frac{1}{d} \left[J_a^T (p_a - p_b) \right]^T. \tag{14}$$

The collision gradient function in Equation 11 represents the degree to which each degree of freedom influences the distance to collision. We wish to select a function $P(q)$, such that the gradient $\frac{\partial P(q)}{\partial q_i}$ is zero when d is large and infinity when d approaches zero. We define the collision gradient weighting matrix, denoted by W_{COL}, by an $n \times n$ diagonal matrix with diagonal elements w_{col_i} $(i = 1 \cdots n)$ defined by

$$W_{COL} = \begin{bmatrix} \mathrm{w}_{col_1} & 0 & 0 & 0 \\ 0 & \mathrm{w}_{col_2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \mathrm{w}_{col_n} \end{bmatrix}. \quad (15)$$

The scalars w_{col_i} are defined by

$$\mathrm{w}_{col_i} = \begin{cases} 1 + |\frac{\partial P}{\partial q_i}| & if \quad \Delta|\partial P/\partial q_i| \geq 0, \\ 1 & if \quad \Delta|\partial P/\partial q_i| < 0. \end{cases} \quad (16)$$

The collision function P can have different forms. For example, we consider the following candidate function.

$$P = \rho e^{-\alpha d} d^{-\beta}. \quad (17)$$

The function is at infinity when $d = 0$ and decays exponentially as d increases. The rate of decay is controlled by adjusting the parameters α and β. By increasing α, we can control the exponential rate of decay so that the function approaches zero more quickly. The parameter ρ controls the amplitude. The partial derivative of P with respect to d is

$$\frac{\partial P(q)}{\partial d} = -\rho e^{-\alpha d} d^{-\beta} (\beta \ d^{-1} + \alpha). \quad (18)$$

The quantity $\frac{\partial P}{\partial q}$ in Equation 12 can be analytically computed from Equation 13 (or Equation 14) and Equation 18.

The term $\Delta|\partial P/\partial q_i|$ in Equation 16 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 weighting factor, described by the first condition in Equation 16, becomes very large causing the joints affecting the motion of the collision point to slow down. When two segments are about to collide, the weighting 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 Equation (16) allows the joint to move freely.

The next step is to construct a weighting matrix W. This matrix is comprised of the joint limit weighting matrix W_{JL} and the collision weighting matrix W_{COL}. Suppose a total of N_c segment pairs are checked for self collision. Let j $(i = 1 \cdots N_c)$ be the index of the j_{th} collision pair, and d_j the minimum distance between the two colliding segments. Let p_{a_j} and p_{b_j} represent the coordinates, referred to the base, of the two colliding point pairs for the j_{th} collision pairs.

The candidate potential function for each collision pair is given by,

$$P_j = \rho e^{-\alpha_j d_j} d_j^{-\beta_j} \quad (19)$$

Its gradient can be computed as before,

$$\frac{\partial P_j}{\partial q} = \frac{\partial P_j}{\partial d_j} \frac{\partial d_j}{\partial q} \quad (20)$$

It follows that the collision weighting matrix for each collision pair, denoted by W_{COL_j} can be computed as outlined above. The collision weighting matrix is comprised of the contribution of each collision pair as given by,

$$W_{COL} = \frac{1}{N_c} \sum_{j=1}^{N_c} W_{COL_j} \quad (21)$$

C. The composite constraint matrix

The next step is to construct a composite constraint weighting matrix W comprised of the joint limit weighting matrix W_{JL} and the collision weighting matrix 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 weighting matrix is given by,

$$W = a \ W_{JL} + (1 - a) \ W_{COL}, \quad (22)$$

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

$$a = \frac{1}{(N_c + 1)}. \quad (23)$$

IV. Results

We consider the human to robot motion retargeting application to test the efficacy of the proposed algorithm [5]. The humanoid robot ASIMO was the platform used in the simulation and experimental results. We first associate a set of desired upper body human motion descriptors with the human model. The human motion descriptors consist of up to eight upper body Cartesian positions corresponding to the waist joint, two shoulder joints, two elbow joints, two wrist joints, and the neck joint (see Figure 2). The human motion data was obtained from the Carnegie Mellon University (CMU) human motion data base [16] as well as from a marker-less system we have previously developed [17]. The human motion data was low pass filtered, interpolated, and scaled to the humanoid robot ASIMO dimensions. The resulting motion corresponds to the reference robot motion descriptors used as input in our proposed kinematically constrained CLIK formulation.

The total upper-body degrees of freedom utilized in our ASIMO model was 14 (6 at torso, 3 at each shoulder, 1 at each elbow). The dimension of Cartesian motion when using all eight position task descriptors is 24, resulting in an over-constrained task specification. If four task descriptors at the waist, two wrists, and the neck are used, the task dimension

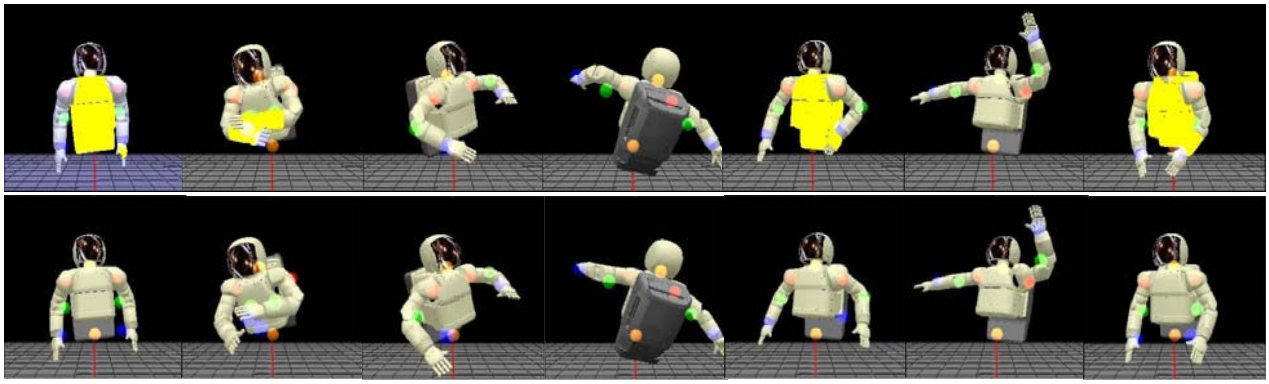

Fig. 3. Snapshots of simulated dancing motion with and without collision avoidance.

is 12. This task specification is under-constrained and creates 2 degrees of redundancy. The experiments conducted in this section use 4 task descriptors. In section V, we utilize both 4 as well as 8 task descriptors.

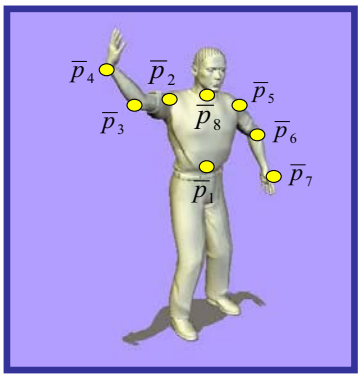

Fig. 2. Eight upper body human motion descriptors, represented by $\bar{p}_i$ are used to control the humanoid robot. These motion descriptors are normalized to the dimensions of the humanoid robot ASIMO.

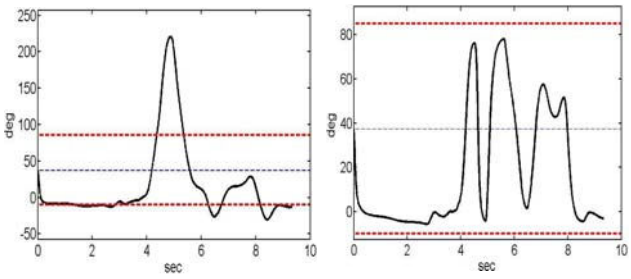

Fig. 4. Euler rotation corresponding to the Adduction/Abduction motion of the left shoulder. Left: without joint limits. Right: with joint limits

Figure 3 illustrates snapshots of simulated retargeting results of a fast dancing motion (obtained from the CMU database) with a full body 360 degree twisting. These simulated results are generated using the humanoid robot ASIMO's model and geometry. Since this motion is high speed and involves complex twisting, the motion cannot be performed on the physical robot. Nevertheless, we can test the collision avoidance algorithm in simulation. We have restricted the motion to the upper body. The top row in Figure 3 illustrates snapshots of the motion without invoking the collision avoidance algorithm. The colliding segments, detected using the SWIFT++ collision detection software, are highlighted in yellow. The bottom row illustrates the results of the same motion when the collision avoidance was used.

For the dancing sequence, Figure 5 and Figure 6 show the minimum distance between collision points on the left hand and torso segment pairs. The minimum distances are plotted with and without using the collision avoidance algorithm. Without collision avoidance, the collision points attached to the left hand and torso segment penetrate the collision zone and eventually collide between frames 470 and 505 as shown in more detail in Figure 6. Note that a negative distance implies collision and penetration of the two bodies. Penetration distance is clamped when penetration of the two bodies is beyond $-2.5\ cm$. When collision avoidance is turned on, contact between the two segments does not occur. Figure 7 shows more dramatic contact between the left hand segment and the torso segment. The collision avoidance algorithm can successfully avoid penetration. In this simulation and all other simulations, the collision function parameters were set at $\rho = 1$, $\alpha = 50$, $\beta = 2$. For high speed motions such as those from the CMU motion capture database, the collision avoidance strategy was effective in more than 98 $\%$ of the frames analyzed. However, the parameter α was tuned to avoid collision in all frames.

For the dancing motion, we also plotted the results of the joint limit avoidance strategy which is also used to prevent self collisions of connected body segments (see Figure 4). In the left plot, the 2nd Euler rotation (Adduction/Abduction) of the left shoulder joint is shown when joint limit avoidance is not used. The right plot shows the results after enforcing joint limits. The upper and lower joint limits are shown by the dashed lines. If joint limit avoidance is not used, the shoulder joint rotates rapidly well beyond its limit, resulting in collision between the arm and the torso. With joint limit avoidance, the joint angle dampens gradually when it approaches its limits, creating a very smooth and natural

looking motion.

We have also performed online motion retargeting based on human motion data obtained from a real time, marker-less human pose tracking system developed at Honda Research Institute, USA, Inc. [17], [18]. The collision avoidance algorithm had very little difficulty with these motions because they are generally less dynamic than those obtained from the CMU motion database. Snapshots from the collision free online motion retargeting experiments are shown in Figure 8. We have shown some explicit postures in which the human performer is self colliding or comes near collision, yet ASIMO'S motion is collision free.

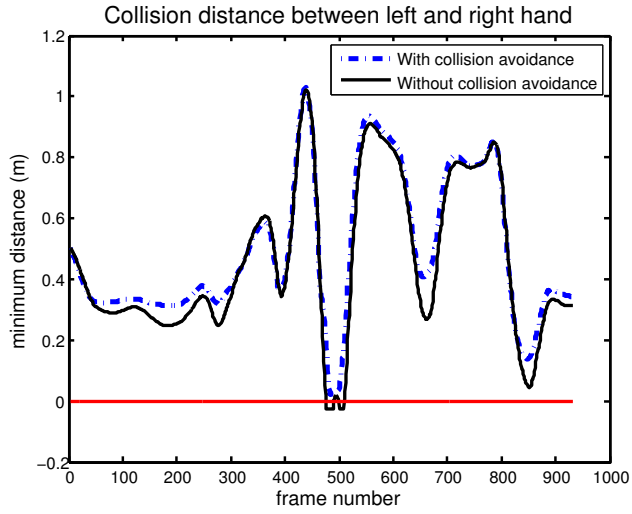

Fig. 5. Minimum distance between left and right hand collision points for a dancing sequence [16] using the weighted least squares solution (weighting method).

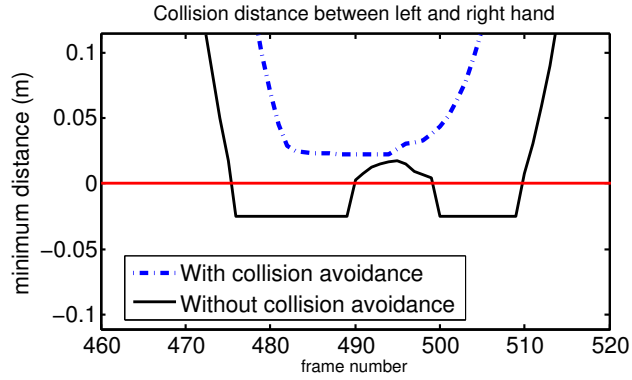

Fig. 6. Left and right hand minimum distance: Zoomed in Figure 5 to observe frames 460 - 520.

V. Hybrid Approach

In our earlier work, we introduced a collision avoidance strategy based on redirection of collision points along a virtual surface manifold surrounding each link segment [5]. The

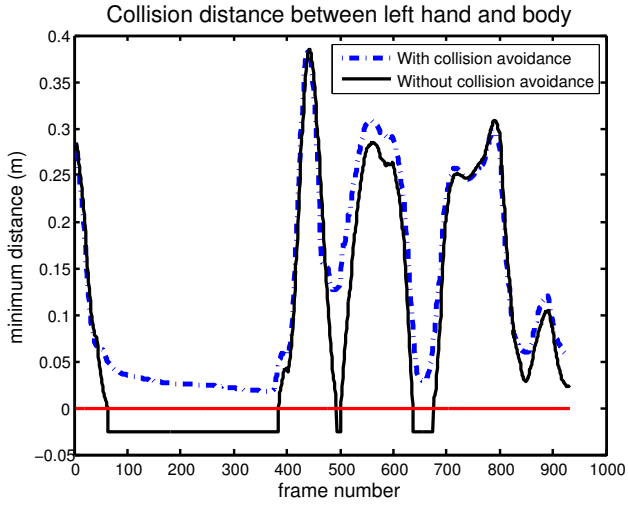

Fig. 7. Minimum distance between left hand and torso collision points for a dancing motion using the weighted least squares solution (weighting method).

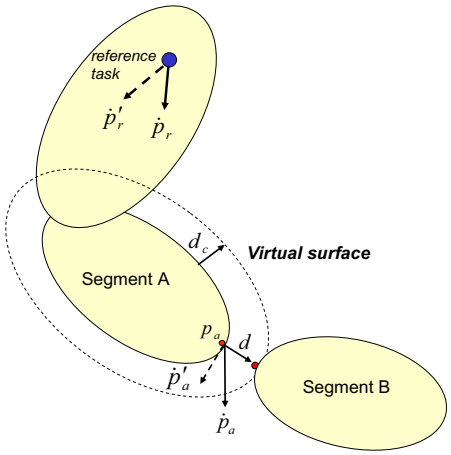

Fig. 9. Segment A approaches a stationary segment B. Upon penetration into a critical zone, the collision point p_a is redirected along the virtual surface tangent line by altering the trajectory of the reference task descriptor, p_r.

virtual surface method is conceptually illustrated in Figure 9. Suppose segment A approaches a stationary segment B. Consider a 3D virtual surface surrounding segment A. The goal is to redirect the collision point p_a along the virtual surface tangent when the minimum distance between the two segments is less than a critical distance (i.e. $d < d_c$). To avoid a discontinuous redirection, a blending approach is used. In order to redirect the collision point, the reference task descriptor trajectory is modified using a series of transformations which relate the redirected collision point velocities $\dot{p}'_a$ to the redirected reference velocities $\dot{p}'_r$. The closed loop inverse kinematics equation with the modified parameters is given by

$$\dot{q} = J^*(\dot{p}'_r + K' \; e'), \tag{24}$$

Fig. 8. Snapshots of ASIMO replicating the motion of a human demonstrator in real time.

where $e' = p'_r - p$ and K' is an adaptively changing diagonal feedback gain matrix whose values decrease as the distance d decreases.

The virtual surface method is effective in directly controlling the collision distance to be within a specified tolerance (determined by the critical distance d_c) for any type of motion without having to tune any parameter. The method is especially suitable in enforcing collision constraints while executing tasks which do not exhibit redundancy. However, since the reference motion is redirected, the performance of the virtual surface method in Cartesian tracking control of under-constrained task trajectories is not optimal.

The weighted least squares solution proposed in Section III-B is effective in handling redundant(under-constrained) as well as over-constrained task specifications. However, the limitation in the approach (as with gradient projection methods) is that the collision distance cannot be directly controlled. The collision avoidance function parameters may need to be tuned to produce a desirable effect or to guarantee that the constraints are enforced. A fixed set of parameters may not work in general, especially for high speed motions.

To address this issue, we consider a hybrid approach, combining the virtual surface and weighting matrix methods to exploit the benefits of each. The integration of the two methods is straightforward. We compute W and J^* as described in Section III-B and use the result in Equation 24. In the hybrid approach, we give the weighting method a higher priority and rely on the virtual surface method as the second layer of protection which does not interfere with the weighting method until $d < d_c$. Selecting a relatively small critical distance d_c will produce a desirable outcome to minimize interference with the weighting method unless necessary.

We investigated the hybrid approach using a simulated experiment. The critical distance d_c used in the virtual surface method and the hybrid method was set at $5cm$ and $2.5cm$, respectively. A motion captured boxing sequence obtained from the CMU motion capture database [16] was used as the reference motion. The original motion is characterized by considerable amount of self-collisions between the two hands and the torso. First, we examine the over-constrained case where eight upper-body task descriptors are used. The root mean square (RMS) deviation of the modified task trajectories from the reference trajectory when collision avoidance is invoked is illustrated in Figure 10 for the three approaches: 1) virtual surface, 2) weighting matrix, and 3) hybrid (virtual surface combined with weighting). In this over-constrained case, the three approaches yielded similar results.

The task tracking benefit of the weighting and hybrid methods becomes evident when the number of tasks is reduced to 4 (under-constrained) as shown in Figure 11. As compared to the over-constrained case, the (RMS) deviation of the modified collision free trajectory from the reference trajectory increased significantly in the virtual surface method. The weighting matrix method, however, had a similar RMS deviation. As in the over-constrained case (Figure 10), the hybrid approach produced similar results as the weighting matrix approach. The benefit of the hybrid approach is that that it is as effective as the weighted least squares solution in both over-constrained and under-constrained situations. At the same time, it offers an extra layer of protection to prevent collision without parameter tuning.

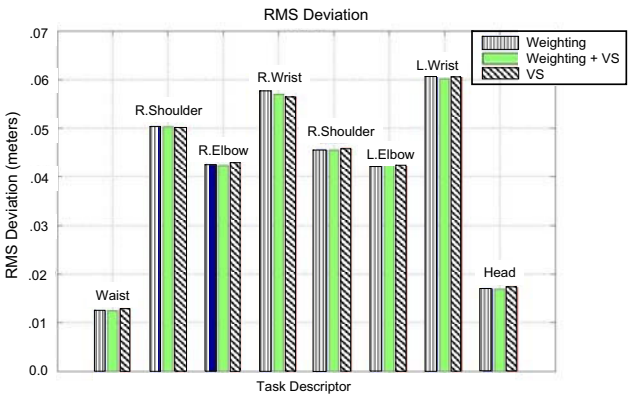

Fig. 10. RMS Deviation of modified task trajectories from the reference trajectory when collision avoidance is invoked using all eight task variables. In this over-constrained case, the three methods have similar performance.

VI. SUMMARY

Enforcing kinematic constraints due to self collisions is particularly challenging in highly articulated robots with tree structures because of the possibility that a large number of segments pairs can simultaneously approach and move away from collision. We described a weighted least squares solution which is effective in handling over-constrained as well as under-constrained task specifications. One shortcoming of this approach for high speed motions is that it requires tuning the collision function parameters to obtain a desirable

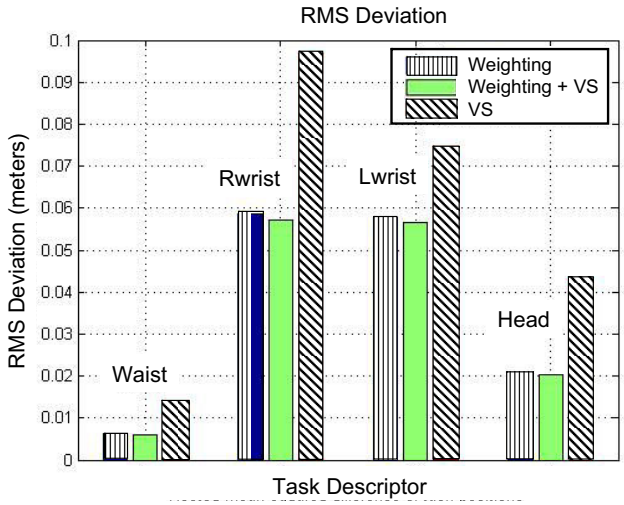

Fig. 11. RMS Deviation of modified task trajectories from the reference trajectory when collision avoidance is invoked using four task variables. In this under-constrained case, the weighting method and the hybrid method have similar performance and outperform the virtual surface method. The hybrid method is advantageous since it can guarantee collision free motion without parameter tuning.

effect. To address this issue, we presented a hybrid approach which combined the weighted least squares method with our previously reported virtual surface control method. The efficacy of the proposed algorithm was demonstrated based on simulations and experiments performed on the Honda humanoid robot ASIMO. Currently, we're working towards formulating this approach for higher order task space as well as joint space control methodologies.

REFERENCES

[1] 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.

[2] O. Khatib. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *The International Journal of Robotics Research (IJRR)*, 5(1):90–98, 1986.

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

[4] 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 *Proceedings of ICRA*, pages 3200–3205, Pasadena CA, 2008.

[5] 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.

[6] A. Nakazawa, S. Nakaoka, K. Ikeuchi, and K. Yokoi. Imitating human dance motions through motion structure analysis. In *Intl. Conference on Intelligent Robots and Systems (IROS)*, pages 2539–2544, Lausanne, Switzerland, 2002.

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

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

[9] 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.

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

[11] B. Dariush, M. Gienger, B. Jian, C. Goerick, and K. Fujimura. Whole body humanoid control from human motion descriptors. In *Int. Conf. Robotics and Automation (ICRA)*, Pasedena, CA, 2008.

[12] 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.

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

[14] 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.

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

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

[17] Y. Zhu, B. Dariush, and K. Fujimura. Controlled human pose estimation from depth image streams. In *CVPR Workshop on Time of Flight Computer Vision*, Anchorage, Alaska, 2008.

[18] Y. Zhu and K. Fujimura. Constrained optimization for human pose estimation from depth sequences. In *Proceedings of Asian Conference on Computer Vision*, Tokyo, Japan, 2007.