Online and markerless motion retargetting with kinematic constraints

Behzad Dariush, Michael Gienger, Arjun Arumbakkam, Christian Goerick, Youding Zhu, Kikuo Fujimura

2008

Preprint:

This is an accepted article published in IEEE International Conference on Intelligent Robot and Systems. The final authenticated version is available online at: https://doi.org/[DOI not available]

Online and markerless motion retargeting with kinematic constraints

Behzad Dariush Michael Gienger Arjun Arumbakkam Christian Goerick Youding Zhu Kikuo Fujimura

***Abstract*— Transferring motion from a human demonstrator to a humanoid robot is an important step toward developing robots that are easily programmable and that can replicate or learn from observed human motion. The so called motion retargeting problem has been well studied and several off-line solutions exist based on optimization approaches that rely on pre-recorded human motion data collected from a marker-based motion capture system. From the perspective of human robot interaction, there is a growing interest in online and marker-less motion transfer. Such requirements have placed stringent demands on retargeting algorithms and limited the potential use of off-line and pre-recorded methods. To address these limitations, we present an online task space control theoretic retargeting formulation to generate robot joint motions that adhere to the robot's joint limit constraints, self-collision constraints, and balance constraints. The inputs to the proposed method include low dimensional normalized human motion descriptors, detected and tracked using a vision based feature detection and tracking algorithm. The proposed vision algorithm does not rely on markers placed on anatomical landmarks, nor does it require special instrumentation or calibration. The current implementation requires a depth image sequence, which is collected from a single time of flight imaging device. We present online experimental results of the entire pipeline on the Honda humanoid robot - ASIMO.**

I. INTRODUCTION

Learning from human demonstration, also referred to as imitation learning, has become an important topic of research in robotics with applications spanning across many disciplines such as robot motion control, human-robot interaction, and machine learning. Imitation learning promises to simplify the process of programming complex humanoid robot motions by replacing the time-consuming manual programming of a robot by an automatic programming process, solely driven by showing the robot the task by an expert teacher [1] [2]. Examples from a human demonstrator also provide a powerful mechanism for reducing the complexity of search spaces in learning algorithms by either starting the search from the observed locally optimal solutions, or by eliminating from the search space what are known as infeasible solutions.

A long standing trend in learning from demonstration methods has been to approach the problem from the standpoint of motion replication, although recent work inspired by this rational is not just about observing and replicating the motion, but rather about understanding the goals of a given action. Indeed, many aspects of imitation are goal-directed, that is, actions are meant to fulfill a specific purpose and convey the intention of the teacher [3]. Nevertheless, one can argue that an important pre-requisite step in learning from imitation involves mimicry, or the reproduction of the actions of the human demonstrator.

Mimicry can be viewed as a sophisticated transfer of the observed human motion to the robot, a procedure referred to as motion retargeting by the computer graphics community [4], [5], [6], [7], [8]. The representation of motion by descriptors which capture the essence of motion or encode meaningful information about the task is an important research problem. Typically, motion descriptors are simply described by either joint space or task (Cartesian) space coordinates. Task space methods offer an advantage in that the large number of mechanical degrees of freedom involved in the execution of movement tasks by humanoids can be represented by lower dimensional descriptors. These descriptors may be referred to as task descriptors because they are used to describe motion by higher level task variables which may be encoded to convey the intention of the human performer. A task oriented approach is also compatible with current views in motor learning that suggest that the central nervous system organizes or simplifies the control of large degrees of freedom during motion execution and motor learning phases. That is, the controlled operation of the neuromuscular system with an exorbitant number of degrees of freedom requires a reduction of mechanical redundancy, achieved by reducing the number of degrees of freedom [9].

Regardless of how the motion is represented, by task variables or joint variables, a suitable retargeting algorithm must deal with complex kinematic constraints due to differences in topology, anthropometry, joint limits, and degrees of freedom between the human demonstrator and the robot. Similar problems have been widely studied and partially addressed, particularly for transferring human motion to an animated character. The problem is often formulated and solved as a constrained non-linear optimization problem, where the objective is to minimize the error between the human motion and the target motion, subject to the kinematic constraints [10], [11]. Such approaches are often performed off-line, in static environments, using pre-recorded human motion obtained from marker based motion capture systems [12]. The resulting motion is then used as the reference trajectory to be executed by the robot's motion controller.

Off-line methods using prerecorded motions do not ac-

B.Dariush and 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

M. Gienger and C. Goerick are at Honda Research Institute, EU, Carl-Legien-Strae 30 63073 Offenbach/Main Germany, michael.gienger(christian.goerick)@honda-ri.de

Youding Zhu is with the Department of Computer and Information Science, The Ohio State University, Columbus OH zhu.81@osu.edu>

count for a dynamically changing environment and have no provision for sensory feedback from the robot's current state to the motion retargetter. Therefore, there is a lack of robustness to uncertainties in the environment. Assuming the environment is static, the edited motion is likely to be admissible by the robot's structure and can to a certain degree be executed by the robot's control scheme during runtime. Balance constraints [13], and obstacle avoidance [14] are sometimes considered. To our knowledge, self collisions constraints have not been explicitly considered as part of a motion retargeting procedure, although recent work demonstrated a control formulation for a real time self collision avoidance on a humanoid robot [15].

In many human-robot interaction applications, the requirements of interactivity in dynamically changing environments as well as simplicity in sensing and instrumentation have placed stringent demands on the retargeting procedure. These requirements include capturing the demonstrator's motions unobtrusively, without instrumenting them with markers. Also, human motion capture with multiple cameras in a special environment may neither be feasible nor practical. The imaging modality must be simple and the underlying retargeting mechanism must cope with this.

In this paper, we present an online, Cartesian space control theoretic retargeting formulation to generate robot joint motions that adhere to the robot's joint limit constraints, self-collision constraints, and balance constraints. The inputs to the proposed method include low dimensional normalized human task descriptors, detected and tracked using a vision based feature detection and tracking algorithm. The proposed vision algorithm does not rely on markers placed on anatomical landmarks, nor does it require special instrumentation or calibration. The current implementation requires a depth image sequence, which is collected from a single time of flight imaging device. We present online experimental results of the entire pipeline on the Honda humanoid robot - ASIMO.

II. Overview of the Entire Pipeline

Figure 1 illustrates an overview of the proposed online motion retargeting framework. The first step in this pipeline involves visual detection and tracking of a set of anatomical landmarks (or features) in the upper-body from image observations obtained using a time of flight depth imaging device [16]. The detected features, registered to a human model, correspond to 3D position vectors at the waist joint, two shoulder joints, two elbow joints, two wrist joints, and the head center (See right image in Figure 2). The output of the feature detection module is represented by the vector p_d, where the subscript d denotes detected features.

These features are subsequently low pass filtered and normalized (limb lengths re-scaled) to our humanoid robot model, ASIMO, which has different dimensions, physical parameters, geometry, and degrees of freedom than the human model. Furthermore, the filtered and scaled features are up-sampled to a higher rate (100 HZ) to achieve numerical stability and good tracking within the retargeting module. The resulting vector, denoted by the p_r, represents the reference motion of positional task descriptors [1] which are used as input in our retargeting module. Taking the reference task descriptors as input, the retargeting module outputs kinematically constrained robot joint variables which are commanded to the robot.

Fig. 1. System diagram of the entire pipeline.

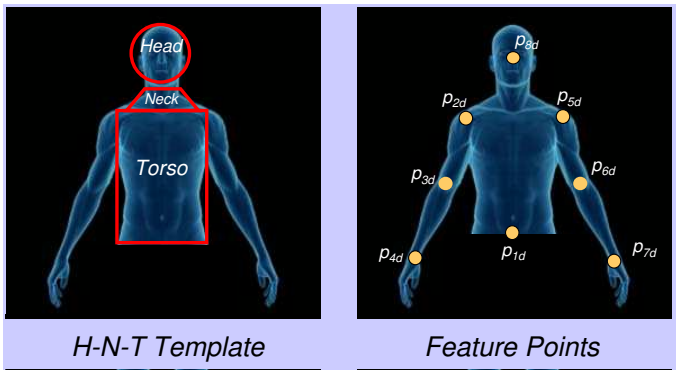

Fig. 2. Left) Head-Neck-Torso template. Right) Position descriptors used in our experiments

III. Feature Detection

We use depth image streams to extract anatomical landmarks (or features) in the upper body. Such data, obtainable by using active or passive stereo, or time-of-flight sensors, provides a blob corresponding to a human body. Our experimental results are based on a single 3D time of flight depth camera sensor [16] which captures depth and intensity images at approximately 15 frames per second. The details of the feature detection algorithm are described in [18]. In this section, we will describe the overview of the approach.

An important first step in our feature detector is the monitoring and tracking of the head and torso. We design a head-neck-torso (H-N-T) deformable template depicted by a circle, trapezoid, and rectangle, respectively (Left image in Figure 2). Let $L = \{H, N, T\}$ denote a configuration of the H-N-T template, that localizes the head circle, neck trapezoid, and torso rectangle. Let θ be a set of distribution parameters used to define the H-N-T template which are learned by collecting training examples from image processing operations and distribution functions. Let $P(I|L, \theta)$ be the likelihood function measured from the image observations, and let $P(L|\theta)$ be the prior probability of the H-N-T

[1]Note that a task descriptor may in general describe the position of landmark and/or the orientation of a coordinate frame attached to a rigid body. A retargeting framework which includes both position and orientation descriptors was described in our earlier work [17].

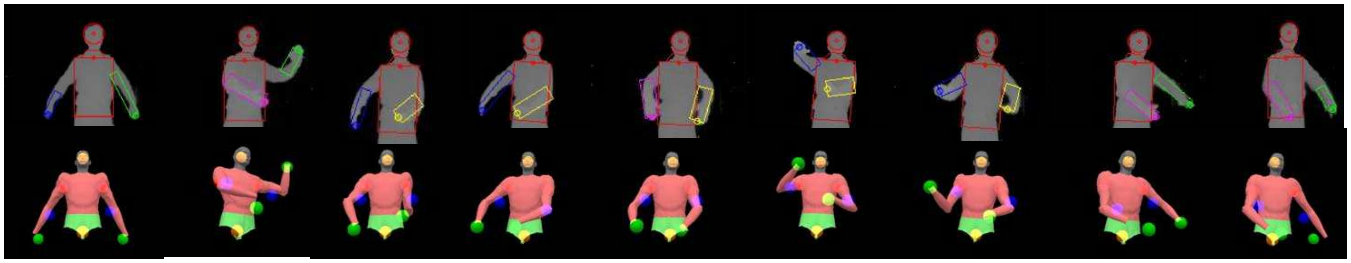

Fig. 3. Taiji sequence. Top row: depth image sequence with the detected arm and H-N-T templates. Bottom row: reconstructed pose.

configuration. From Bayes' rule, we can define the posterior distribution, $P(L|I,\theta)$, as,

$$P(L|I,\theta) \propto P(I|L,\theta)\ P(L|\theta) \tag{1}$$

The posterior probability given in Equation 1 leads to standard log-likelihood hypothesis test for determining the likelihood that the H-N-T template is detected [18]. If the H-N-T template is detected, we first perform a morphological operation referred to as skeletonization to detect an arm blob. If one or two arm blobs are detected, we further examine the arm blobs in order to determine the hand points corresponding to each detected arm blob. The hand blobs are located at the end-points of the distance transformed skeleton which have a sufficiently large distance values. If a hand point is detected, an arm template is formed by tracing back along the skeleton until we reach the torso template. If needed, i.e. one or fewer arm blobs are detected, we perform a second image processing operation that we refer to as *depth slicing* in order to form the arm template. This operation is typically necessary when the arms occlude the body. In this operation, we extract the connected blobs by decreasing the cut-off thresholds until the area of blob is too large to be an arm.

Once the arm templates are formed, they must accordingly be labeled as right arm or left arm. If the arm is detected by skeleton analysis, it can be labeled as right or left based on the location of the entry point (right or left) at the torso template. If the arm template is detected by depth-slicing, the arm label is assigned based on temporal continuity, i.e. the smaller distance to the left or right arm rectangles obtained from the previous frame. The top row in Figure 3 illustrates the results of the H-N-T template (outlined in red) as well as the arm detection using the skeletonization approach (outlined in blue for right arm and green for the left arm) and the depth-slicing approach (outlined in pink for the right arm and yellow for the left arm).

With the detected body parts including the head, torso, left and right arms, we localize the 3D features shown in the right image of Figure 2 for further processing. The head center feature is simply the 3D center of the head circle template. The right and left shoulder features correspond to the upper right and left corner point of the torso template, respectively. The waist joint feature is obtained by projecting a vector from the mid-point of the top edge of the torso template toward the midpoint of the bottom edge of the torso template.

Localizing the arm features, including left and right elbows and wrists is more challenging. If the arm is detected by skeleton analysis, the wrist joint is located near the end-point of the skeleton. The elbow joint feature is located at the intersection of the upper arm and forearm in the skeleton. If the arm is detected based on the depth slicing operation, we assume that the feature points are located approximately at either ends of the arm rectangle. If the H-N-T template is undetected, or if the features are occluded, we use temporal prediction to estimate the missing features [18].

IV. Retargeting

As previously described, Figure 2 illustrates a possible set of features corresponding to prominent landmarks on the body. The position of each detected feature is represented by the vector p_d and described in the base frame corresponding to the waist joint coordinate system. Subsequently, the detected features are normalized to the robot dimensions, low pass filtered, and interpolated. The resulting position vector represents the reference positions on the robot model which are analogous to the detected features on the human model. These reference positions, denoted by the vector p_r will be referred to as task descriptors because they correspond to Cartesian space (or task space) positions in our proposed task space control retargeting framework.

The proposed retargeting algorithm can be described as a local constrained optimization problem. The objective is to estimate the robot joint commands that minimize the tracking error between the reference and predicted task descriptors subject to kinematic and balance constraints. We previously presented a task space control framework to solve this problem without considering self collision constraints [17]. In this section, we highlight key features of the algorithm and introduce a novel self collision avoidance algorithm. As in our previous work [17], we will use the existing balance controller on the humanoid robot, ASIMO, and a whole body motion control system to ensure that the humanoid robot is balanced [19].

A. Differential Kinematics

Let n represent the number of upper body joint variables on the robot model and the vector $q = [q_1, \cdots, q_n]^T$ describe the degrees of freedom which fully characterize the configuration space, or joint space, of the upper-body humanoid robot. Let i $(i = 1 \cdots k)$ be the index of the

i_{th} task descriptor velocity $\dot{p}_i$ and the associated Jacobian $J_i = \frac{\partial p_i}{\partial q}$. We form an augmented spatial velocity vector $\dot{p}$ and an augmented Jacobian matrix J as follows,

$$\dot{p} = \begin{bmatrix} \dot{p}_1^T & \cdots & \dot{p}_i^T & \cdots & \dot{p}_k^T \end{bmatrix}^T \quad (2)$$

$$J = \begin{bmatrix} J_1^T & \cdots & J_i^T & \cdots & J_k^T \end{bmatrix}^T \quad (3)$$

The mapping between the joint space velocities and task space velocities is obtained by considering the differential kinematics relating the two spaces,

$$\dot{p} = J(q)\,\dot{q} \quad (4)$$

B. Cartesian tracking control

We wish to create a control policy that produces the robot joint commands (q) such that the Cartesian error between the predicted robot task descriptors and the reference task descriptors (normalized and processed from the detected human features) are minimized. The tracking performance is very much subject to the robot's kinematic constraints, as well as the execution of multiple and often conflicting task descriptor requirements. Our formulation of such a constrained optimization is based on a Cartesian space kinematic control method known as closed loop inverse kinematics (CLIK). The CLIK equation is given by,

$$\dot{q} = J^*(\dot{p}_r + K\ e) \quad (5)$$

where J^* denotes the regularized right pseudo-inverse of J weighted by the positive definite matrix W_1,

$$J^* = W_1^{-1} J^T\ (J W_1^{-1} J^T + \lambda^2 I)^{-1} \quad (6)$$

The parameter $\lambda > 0$ is a damping term, and I is an identity matrix. The vector $\dot{p}_r$ corresponds to the concatenated velocity of the individual reference task descriptors. The vector e expresses the position error given by $e = p_r - p$. The rate of convergence of the error is controlled by K, a diagonal positive definite gain matrix.

Managing multiple task descriptors as described above is referred to as task *augmentation*, i.e. the concatenation of individual spatial velocities and the associated Jacobian matrices and feedback gain matrices. The tracking error rate for each element of a task descriptor can be controlled by the augmented feedback gain matrix K. The trajectory tracking error convergence rate depends on the eigenvalues of the feedback gain matrix: the larger the eigenvalues, the faster the convergence. A particular task descriptor or its individual components can be more tightly tracked by increasing the eigenvalues of K associated with that direction. By modulating the elements of K, we can effectively encode the importance or the relative level of confidence we have in our observations. Measurements which are more important or have higher confidence values will be assigned higher feedback gain values.

Alternatively, if the system exhibits redundancy, task management can be more tightly enforced by *prioritization*. That is, we can execute task descriptors according to the order of priority. Recursive methods which handle an arbitrary number of prioritized task descriptors have been described elsewhere [17], [20].

C. Joint limit avoidance constraints

Chan and Dubey [21] proposed joint limit avoidance based on a Weighted Least-Norm (WLN) solution. The WLN 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. One such candidate function is given by

$$H(q) = \frac{1}{4} \sum_{i=1}^{n} \frac{(q_{i,max} - q_{i,min})^2}{(q_{i,max} - q_i)(q_i - q_{i,min})}$$

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 upper and lower joint limits represent the more conservative limits between the physical joint limits and the virtual joint limits used to avoid self collision as will be described in the Section IV-D. Note that $H(q)$ is normalized to account for variations in the range of motion. 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.

$$\nabla H = \frac{\partial H}{\partial q} = \begin{bmatrix} \frac{\partial H}{\partial q_1} & , \cdots , & \frac{\partial H}{\partial q_n} \end{bmatrix} \quad (7)$$

The element associated with joint i is given by

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

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 [21], we define the joint limit gradient weighting matrix, denoted by W_{JL}, by an $n \times n$ diagonal matrix with diagonal elements w_{JLi} $(i = 1 \cdots n)$. The scalars w_{JLi} are defined by

$$\mathrm{w}_{JLi} = \begin{cases} 1 + |\frac{\partial H}{\partial q_i}| & if \quad \Delta|\partial H/\partial q_i| \geq 0 \\ 1 & if \quad \Delta|\partial H/\partial q_i| < 0 \end{cases} \quad (8)$$

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 8, becomes very large causing the motion to slow down. When the joint nearly reaches its limit, the weighting factor is near 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 (8) allows the joint to move freely.

D. Avoiding self collision

Self collision avoidance may be categorized as one of two types: 1) collision between two connected segments, and 2) 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.

1) Collision avoidance between two connected bodies: If two segments are connected at a common rotational joint, i.e. connected segments, self collision may be handled by limiting the joint range as described in Section IV-C. 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 does not occur. 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 IV-C.

2) Collision avoidance between unconnected bodies: Consider two unconnected rigid bodies, i.e. bodies which do not share a joint, as shown in Figure 4. In general, Body A and body B may both be in motion. However, for simplicity of presentation and without loss of generality, suppose body A is moving toward a stationary body B. Let p_a and p_b represent the coordinates of the shortest distance $d(d \geq 0)$ between the two bodies, described in the base reference frame. Hereafter, we refer to p_a and p_b as collision points. The coordinates p_a and p_b can be obtained using a standard collision detection software. In this work, we use the SWIFT++ library [22].

Let $\hat{n}_a = \frac{p_b - p_a}{|p_b - p_a|}$ be the unit normal vector and $\vec{d} = d\ \hat{n_a}$ the vector from p_a to p_b. Consider a 3D virtual surface surrounding body A, shown by a dashed line in Figure 4. For every point on body A, its associated virtual surface point is located by the vector $\vec{d_c} = d_c\ \hat{n}$, where d_c is the critical distance, and $\hat{n}$ is the unit normal vector at the surface point. Let p_{vs_a} be the coordinates of a point on the virtual surface of A defined by

$$p_{vs_a} = p_a + d_c \hat{n}_a \tag{9}$$

We define the region between the actual surface of body A and its virtual surface as the critical zone. If body B is stationary, we can redirect the motion at p_a to prevent collision in the critical zone. This redirection is invoked when $(d < d_c)$.

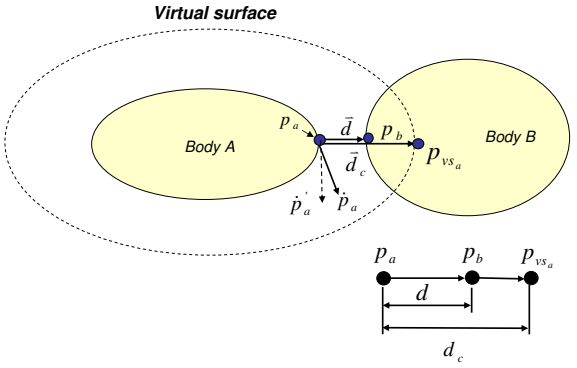

Fig. 4. Body A moving towards a fixed body B

In our CLIK control framework, one way to control (or redirect) the motion of p_a is by modifying the trajectory of the desired task descriptor p_r. Let us specify a redirected motion of p_a by p'_a and its associated velocity by $\dot{p}'_a$. To find the mapping between $\dot{p}'_a$ and $\dot{p}_r$, consider first the equivalent redirected joint velocity vector, given by

$$\dot{q}' = J_a^*\ \dot{p}'_a + SJ^*(\dot{p}_r + K\ e) \tag{10}$$

where $J_a = \partial p_a / \partial q$ is the Jacobian at the collision point(s) and J_a^* is its weighted Damped Least Squares inverse. The matrix $S = diag(s_1 \cdots s_n)$ is a diagonal selection matrix where $s_i = 1$ when the i_{th} column of J_a has all zero entries and $s_i = 0$ elsewhere. The term $J^*(\dot{p}_r + K\ e)$ is simply the joint velocity solution obtained from Equation 5. The physical interpretation of Equation 10 is as follows. The first term determines the joint velocities needed to redirect the collision point velocities along $\dot{p}'_a$. Any zero column of J_a (all zero entries) implies that the associated degree of freedom does not contribute to the motion of the collision point. The second term in Equation 10 is the orthogonal complement of the first term which computes the entries for those joint velocities which do not affect the motion of the collision point(s).

Intuitively, it would seem more appropriate to formulate Equation 10 using a two priority inverse kinematics strategy similar to the control of redundant manipulators [23]. In such a strategy, the first priority term corresponds to satisfying self collision avoidance by redirection (as in first term in Equation 10). Utilizing redundancy, the second priority term can be constructed to satisfy the requirements for tracking the task descriptors. However, our first implementation of such an approach resulted in jerky motions under certain situations.

A redesigned position task descriptor trajectory may be computed as follows

$$\dot{p}'_r = J\ \dot{q}' \tag{11}$$

The closed loop inverse kinematics equation with the modified parameters is given by

$$\dot{q} = J^*(\dot{p}'_r + K'\ e') \tag{12}$$

where $e' = p'_r - p'$ and K' is an adaptively changing diagonal feedback gain matrix whose values decrease as the distance d decreases. Note that p'_r at the current time t may be computed by a first order numerical integration.

The instantaneous redirection $\dot{p}_a \rightarrow \dot{p}'_a$, as described above, produces a discontinuous first derivative of p_a at the boundary $d = d_c$. The discontinuity at $\dot{p}_a$ results in a discontinuity in $\dot{p}_r$, as given by the solution in Equation 11. To preserve first order continuity, we may blend the solutions of $\dot{p}'_r$ before and after redirection occurs. A blended solution to Equation 11 is given by

$$\dot{p}'_r = (1 - b)\ \dot{p}_r + b\ J\ \dot{q}' \tag{13}$$

where b is a suitable blending function such as,

$$b(d) = \frac{e^{-\alpha(d/d_c - \delta)}}{1 + e^{-\alpha(d/d_c - \delta)}} \tag{14}$$

where α and δ are scalar parameters used to modulate the blending rate and shift of the blending function, respectively.

Figure 5 shows the plot of the b in relation to the ratio d/d_c for $\alpha = 15$. The blending function is plotted for $\delta = .5$ and $\delta = 1.0$. The parameter δ may be used to shift the distance d where blending is initiated and terminated. In the case $\delta = .5$, when $d > d_c$ the function $b(d) \approx 0$, implies that the second term in Equation 13 is effectively zero so that there is no redirection of the original task descriptor velocity (i.e. $\dot{p}'_r = \dot{p}_r$). At the other extreme, when $d = 0$, the function $b(d) = 1$, implies that the first term in Equation 13 is zero and the reference trajectory is altered in order to redirect the collision points along the tangent surface. To be more conservative, we may chose $\delta = 1.0$ in the blending function. This way, blending initiates even before the collision points reach their critical distance.

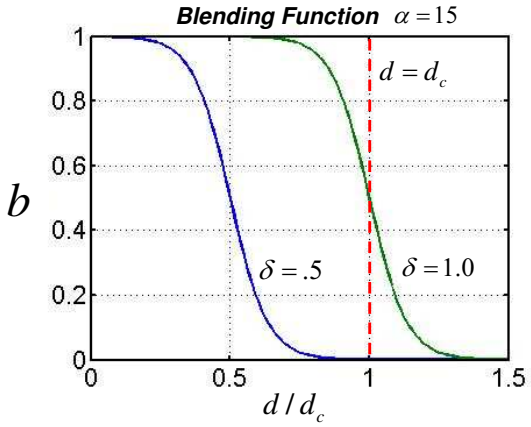

Fig. 5. Blending function at two values of δ

Thus far, we have not described how to specify the magnitude and direction of $\dot{p}'_a$. The most straightforward, and perhaps conservative solution is to redirect the collision point in a direction opposite to the unit normal vector $\hat{n}_a$. A more effective strategy is to redirect the collision point so that it slides along a direction which is tangent to the surface at the collision point, as shown in Figure 4.

$$\dot{p}'_a = \dot{p}_a - < \dot{p}_a, \hat{n}_a > \hat{n}_a \qquad (15)$$

In theory, the above redirection vector will guide the collision point motion along the virtual surface boundary, producing a more natural motion toward the target. The case when body A is stationary and body B is in motion is the dual of the problem considered above. When both body A and body B are in motion, we can specify the redirection vectors at the collision points p_a and p_b and use task augmentation to control both critical points.

Figure 6(a) and (b) illustrate two snapshots of a Taiji motion sequence and the corresponding collision detection and avoidance results. These simulated results are generated using the humanoid robot ASIMO's model and geometry. In each snapshot, the depth image and the reconstructed human pose are shown in the upper left and upper right image, respectively. The bottom left image illustrates collision between the hand and the torso segment when collision avoidance is turned off. The colliding body segments, detected using the SWIFT++ collision detection software, are highlighted in yellow. The lower right image illustrates that no collision is detected when collision avoidance is invoked. Figure 7 shows the minimum distance between two collision

Fig. 6. Snapshots of simulated Taiji motion with and without collision avoidance

points attached to the hand and head segments as a human performer moves his hand toward his head in a drinking motion sequence. The blending parameter was set at $\delta = .5$ such that blending is initiated at the critical distance of $d_c = 0.05$ meters; therefore, collision points are not fully redirected at the virtual surface. Redirection is gradual, and penetration into the critical zone occurs. However, the two bodies do not collide.

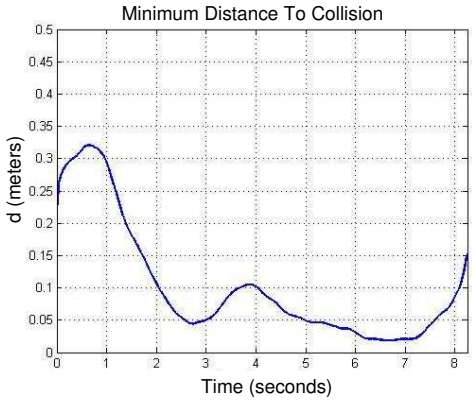

Fig. 7. Minimum distance between two collision points attached to the hand and head segments as a human performer moves his hand toward his head in a drinking motion

E. Balance Control

The presented control scheme does not yet consider the constraints that are required to maintain balance during standing and walking. These aspects are not handled within the retargeting framework, but rather by a separate walking and balancing controller that is described in [24], [25].

In detail, the retargeted motion is commanded to the whole body motion controller. The motion generated by the whole body controller will cause some momentum and moment of momentum from a desired reference. This deviation is compensated by the ZMP based balance controller by shifting the

upper body in forward and/or lateral direction. The whole body control and the ZMP control operate cooperatively.

To account for the body shift, the upper body translational degrees of freedom are incorporated in the kinematic model of the robot. However, they are not actively driven, but rather considered as external input into the controller equations of the whole body control. Whole body control and ZMP control are coupled through momentum and state feedback, which turns out to be an efficient way to separate these controllers.

V. Motion Interface

We have developed a motion interface to provide a communication link and command interface between off-board computations to generate retargeted joint commands and the on-board real time control. The motion interface provides a comprehensive way to give motion commands to the robot, without having the user to care about issues such as synchronization, delays, or on-board control for maintaining balance. Such an interface is desirable since the real-time implementation requires synchronization between critical control processes that may not be satisfied dependably with a network connection.

Issues like balance control and other critical aspects are handled within the real-time controller. There is also a second (fail safe) collision avoidance system onboard the robot within the real-time controller. The onboard collision avoidance is designed to be faster, but more conservative than the off-board collision avoidance algorithm described in Section IV-D. Details of the onboard collision avoidance algorithm are given in [19], [15]. The motion interface has been successfully used in various other applications, as for instance in [26].

VI. Experimental Results

Experiments were performed on the Honda humanoid robot, ASIMO, using a single time-of-flight range image sensor [16], to obtain depth images at approximately 15 frames per second. The visual processing algorithm generates eight upper-body task descriptors, which represent Cartesian coordinates corresponding to the waist, shoulders, elbows, wrists, and head center. This module operates at approximately 10 frames per second. The retargeting module then generates collision free joint commands for the Humanoid Robot ASIMO at 100 Hz. A socket program sends the joint commands to the motion interface, described in Section V, over a wireless network. The whole body motion interface then communicates these joint space coordinates to low level controllers on the robot, including the balance controller, that run on a dedicated real time processing computer onboard the robot, through UDP sockets. The Experimental setup of the entire pipeline is illustrated Figure 8.

Figure 9 illustrates snapshots of the online motion retargeting for a Taiji sequence. A slight delay was observed in the tracking of the human motion by ASIMO during experiments, although the visual processing and retargeting performed with little software latency. This delay could be attributed to network latency or other software overhead. Efforts to identify and resolve these latency issues are in progress.

VII. Summary and Future Work

We have presented an online (real-time) Cartesian control theoretic approach to retarget human motion to humanoid robots based on low dimensional human task descriptors obtained from marker-less visual observations. Although several systems have previously demonstrated human to humanoid motion retargeting, our approach is among a few to use an online and marker-less algorithm based on a single camera. To our knowledge, the system presented in this paper may be the first to explicitly enforce self collision constraints within the retargeting formulation and to demonstrate it on a humanoid robot.

Although we have reported results of eight upper body task descriptors, the proposed formulation can handle arbitrary number of task descriptors. The algorithm is suitable when there is redundant degrees of freedom as well as when the system is over-constrained. In fact, for many of the motions tested, we observed that utilizing as few as four task descriptors (waist, two hands, head) could reproduce realistic and natural looking robot motions. This attribute enables flexibility in sensing and instrumentation required to acquire human motion, as well as flexibility in controlling the robot from a reasonable, yet arbitrary number of task descriptors.

Our upper body retargeting algorithm relies on ASIMO's existing balance controller in order to make balance adjustments by modulating the pelvis position. In future work, we plan to extend the current framework to whole body motion retargeting which may require the robot to take steps to regain stability if necessary.

References

[1] S. Schaal. Learning from demonstration. In M.C. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, chapter 9, pages 1040–1046. MIT Press, 1997.

[2] 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.

[3] H. Bekkering, A. Wohlschlaeger, and M. Gattis. Imitation of gestures in children is goal directed. *Quarterly Journal of Experimental Psychology*, 53A(1):153–164, 2000.

[4] M. Gleicher. Retargeting motion to new characters. In *Proceedings of SIGGRAPH98*, pages 33–42, ACM, New York, 1998.

[5] J. Lee and S. Y. Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of SIGGRAPH99*, pages 39–48, ACM, New York, 1999.

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

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

[8] K. J. Choi and H. S. Ko. On-line motion retargeting. *Journal of Visualization and Computer Animation*, 11(5):223–235, 2000.

[9] N. Bernstein. *The coordination and regulation of movements*. Pergamon, London, 1967.

[10] A. Ude, C.G. Atkeson, and M. Riley. Programming full-body movements for humanoid robots by observation. *Robotics and Autonomous Systems*, 47:93–108, 2004.

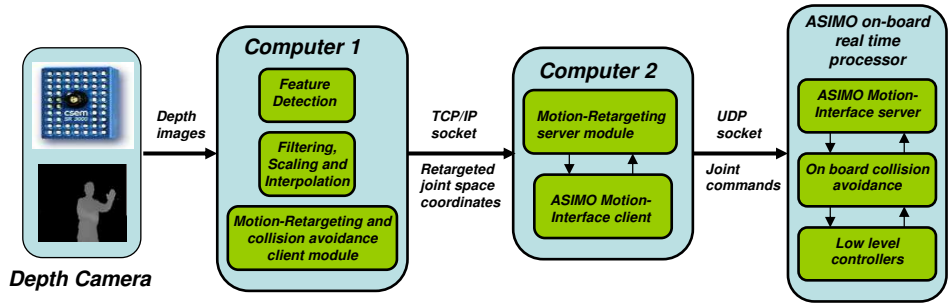

Fig. 8. Set-up for experiments

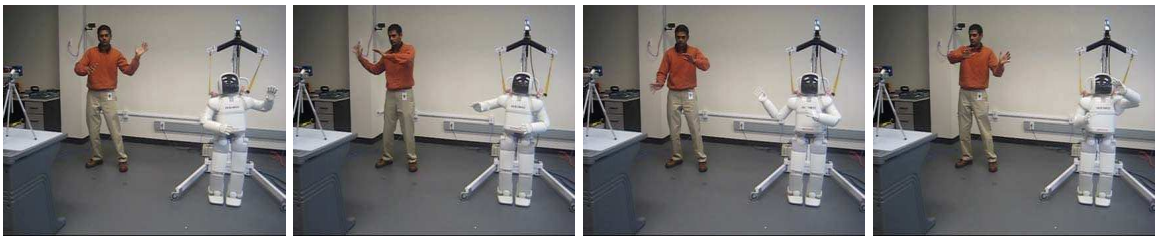

Fig. 9. Snapshots from Taiji online motion retargeting to ASIMO

[11] S. Nakaoka, A. Nakazawa, F. Kanehiro, K. Kaneko, M. Morisawa, H. Hirukawa, and Katsushi Ikeuchi. Learning from observation paradigm: Leg task models for enabling a biped humanoid robot to imitate human dances. *Int. J. Robotics Research*, pages 829–844, 2007.

[12] Nancy Pollard, Jessica K Hodgins, M.J. Riley, and Chris Atkeson. Adapting human motion for the control of a humanoid robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '02)*, May 2002.

[13] Y. Tamiya M. Inaba S. Kagami, F. Kanehiro and H. Inoue. Autobalancer: An online dynamic balance compensation scheme for humanoid robots. In *Int. Workshop Alg. Found. Robot.(WAFR)*, Lausanne, Switzerland, 2000.

[14] 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 (ICRA06)*, Orlando, FL, 2006.

[15] 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.

[16] *Swisss Ranger SR-3000 3D time of flight camera*. http://http://www.swissranger.ch/.

[17] 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)*, Pasadena, CA, 2008.

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

[19] M. Gienger, H. Janssen, and C. Goerick. Task-oriented whole body motion for humanoid robots. In *Proceedings of the 2005 5th IEEE-RAS International Conference on Humanoid Robots*, pages 238–244, Los Angeles, USA, 2005.

[20] B. Siciliano and J. Slotine. A general framework for managing multiple tasks in highly redundantrobotic systems. In *International conference on Advanced Robotics*, volume 2, pages 1211–1216, Pisa, Italy, 1991.

[21] 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.

[22] The University of North Carolina at Chapell Hill. Speedy walking via improved feature testing for non-convex objects. Internet page. http://www.cs.unc.edu/ geom/SWIFT++/.

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

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

[25] Honda Motor Corp. The Honda humanoid robot ASIMO. Internet page. http://www.world.honda.com/ASIMO.

[26] B. Bolder, M. Dunn, M. Gienger, H. Janssen, H. Sugiura, and C. Goerick. Visually guided whole body interaction. In *Proceedings of the Int. Conf. on Robotics and Automation*, Rome, Italy, 2007.