A Multi-modal Architecture for Human Robot Communication

Arjun K. Arumbakkam, Taizo Yoshikawa, Behzad Dariush and Kikuo Fujimura

Abstract— **In this paper we present a human-friendly control framework and an associated system architecture for performing compliant trajectory tracking of multimodal human gesture information on a position controlled humanoid robot in realtime. The contribution of this paper includes a system architecture and control methodology that enables real-time compliant control of humanoid robots from demonstrated human motion and speech inputs. The human motion consists of the body and head pose. The human body motion, represented by a set of Cartesian space motion descriptors, is captured using a single depth camera marker-less vision processing module. The human head pose , represented by two degrees of freedom, is estimated and tracked using a single CCD camera. The architecture also enables fine motion control through human speech commands processed by a dedicated speech processing system. Motion description from the three input modes are synchronized and retargeted to the joint space coordinates of the humanoid robot in real-time. The retargeted motion adheres to the robot's kinematic constraints and represents the reference joint motion that is subsequently executed by a model based compliant control framework through a torque to position transformation system. The compliant and low gain tracking performed by this framework renders the system physically safe and therefore friendly to humans interacting with the robot. Experiments were performed on the Honda humanoid robot and the results are presented here.**

I. INTRODUCTION

Transfer of human gesture information to humanoid robots and physics based animated characters is a widely researched topic [1]–[5]. The applications foreseen for such research are varied with some being imitation learning from gesture information and tele-operation for applications such as search and rescue and therapy for autistic children. Many of these applications require the robot to operate in a workspace that is intrinsically unstructured, unpredictable and could possibly include humans thereby making safety a key concern. However, robotic manipulators have traditionally been stiff and high gain position controlled mechanisms making them unsafe for operation in such scenarios, with the reason being that they have primarily been associated with industrial applications requiring high precision under heavy loads. It is therefore highly relevant to develop controllers and a system architecture that is flexible enough to allow for precise tracking of high dimensional and multimodal human gesture information on position controlled robots in a safe manner without having to modify the hardware of the robot. Such a system architecture would typically require a trajectory generator that tracks the gestures made by the human demonstrator, respects joint limit constraints of the robot, performs self collision avoidance and avoids kinematic singularities in the robot. It would also require a controller that takes into account the nonlinear and highly coupled nature of the dynamics of the robot and some mechanism to convert the torque commands generated by such a controller to position commands that can be followed by the position controlled robot. In this paper we show how our system architecture and controller design addresses each of these requirements.

The motivation for the work presented here is derived from the need for a system architecture that has minimum system delay and provides smooth and safe joint and Cartesian space trajectory tracking on a position controlled robot. Such a system architecture augments and in many cases is essential for realizing the applications listed earlier. Tele-operation tasks frequently involve situations where the robot is interacting with humans, animals or just fragile objects. For example, there is anecdotal evidence that autistic children respond better to therapy when it is provided through a robotic interface by a therapist in the loop [6]–[10]. For such applications, it is essential for the robot to be compliant and backdrivable to help prevent injury to the subject. Search and rescue robots frequently have to operate in unpredictable and possibly fragile environments with human or animal subjects requiring care. In all these situations, we see the common need for fast, smooth, precise and safe transfer of human gestures to a robot.

Imitation learning is another popular topic that lies within the domain of the problems addressed by our system architecture. Gesture information and motor skills are learnt by the robot by observing joint or task space trajectories generated from a human demonstrator. Torque commands that allow compliant tracking of these trajectories satisfying kinematic and dynamic constraints for a particular robot platform would only augment the utility of the function being learned for a given goal or task [11]–[15].

Safety of human robot interaction has been widely studied in the past [16]–[18]. Some well known techniques to make robots safe for interaction with humans are back-drivability of joints, low inertia in the links and motors that are most likely to come in contact, low gain and compliant tracking of input trajectories, disturbance observers that detect contact, covering the links with soft and compressible material, etc. In many cases, the workspace and the speeds at which the links are allowed to move for the robot are highly restricted to reduce the risk of accidental contact and injury to humans. As is evident, some of these techniques restrict and hinder the complexity of the motions that can possibly be achieved by the robot. In our work, we address the safety issue by

All the authors are with the Honda Research Institute USA, Inc., Mountain View, CA 94043, U.S.A, email-ids: akarjun@gmail.com, dariush@hra.com, tyoshikawa@hra.com, kfujimura@honda-ri.com

using a low gain and compliant trajectory tracking controller to track input trajectories at the servo level of the robot. Tests of the architecture were performed on the Honda humanoid robot which is conventionally an independent joint high gain position controlled robot that does not exhibit compliant behavior.

II. System Architecture

At a high level, the system consists of a vision based feature point detection algorithm, a speech interface, a face tracking algorithm, a motion retargeting module, a computed torque based servo level controller, a torque-to-position transformer and an underlying socket and shared memory based software framework for relaying data to and from the robot.

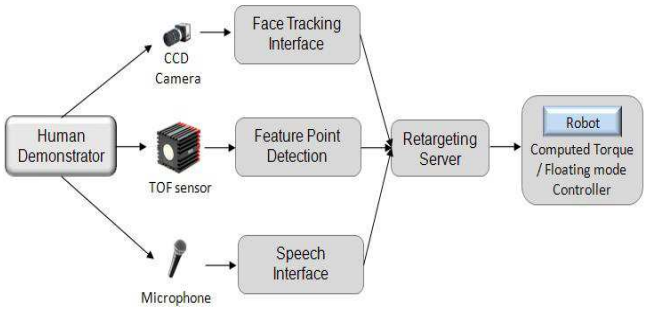

Fig. 1. High level system diagram

The system architecture that we use is distributed and complex but relatively flexible. The reason being that there are several data paths involved running at different servo rates providing multimodal task information that all need to be retargeted simultaneously. Further adding complexity, some of the software modules including the servo level computed torque controller, run on a realtime operating system and share computing time with other servo and monitor processes essential for the robots health.

The software architecture presented in figure Fig. 2 is designed to address the issue of enabling a robot to be an extended physical interaction medium between the human and the immediate environment of the robot. Robots in this application space are expected to be controlled using multi-modal task specific data. Safety issues also require the robot to be a compliant mechanism that will minimize possible physical injury to human subjects. As is evident, from a system architecture perspective, these requirements have disparate data paths and control methodologies. The system architecture needs to be capable of synchronized data sharing between the data paths. Some of the data paths have been highlighted in the system architecture as motion streams. These data paths begin at the input interface, which could be the feature point detector, the speech interface or the face tracker and end at the motion retargeting module, where they get retargeted into the joint angles of the robot, which forms a new distal data path. In short, the motion streams are task-space data paths and the distal data path is the joint-space data path. As is common in software design, these disparate data paths meet at a common shared memory interface, where they are acted upon by algorithms implementing the controllers.

A. Motion Streams

The motion streams are data path leading from the sensor interface modules to the motion retargeting module. They run at different rates depending on the data input or frame rate from the sensors and the processing rate of the algorithms that generate the task information from the raw input. For example, the motion stream from the Time-of-flight sensor and the associated feature point detection algorithm generates task level data at a rate of about 14 Hz, whereas the face tracking module generates the head joint angles at a rate of about 20 Hz. The speech interface however generates commands at the rate at which the human demonstrator feels necessary. Basically, the motion streams contain task level motion information for the robot.

The basic organization of the task level data is in the form of First In First Out (FIFO) data structures. The data stream comes over network socket interfaces and they are accumulated in thread safe FIFO data structures. This thread safe FIFO data structure is basically a shared memory area that is used for communication between the socket interface based server thread and the motion retargeting thread. Task information is processed on a FIFO basis and converted to joint-space commands consistent with pre-specified constraints. Details on the motion retargeting module are given in a following section.

B. Distal / Joint space data-path

The joint-space commands, thus generated, are transferred to another FIFO thread safe data structure, that serves as a communication interface between the motion retargeting thread and a thread that sends the joint space commands to the servo level controller at 200 Hz. The servo level controller is a gravity compensation based control algorithm that generates torque commands for the robot and runs on a real-time operating system. A network socket based server listens on the real time operating system for the joint-space commands from the motion retargeting module. It further collects these commands in a thread safe FIFO data structure and commands them to the servo controller at precisely 200 Hz through another shared memory interface.

III. Feature Point Detection

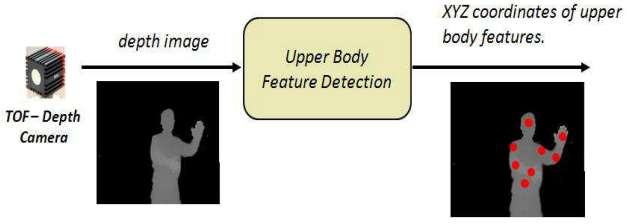

Fig. 3. Keypoint detection system

A time of flight sensor is used to capture depth images of the human demonstrator to be processed by a feature

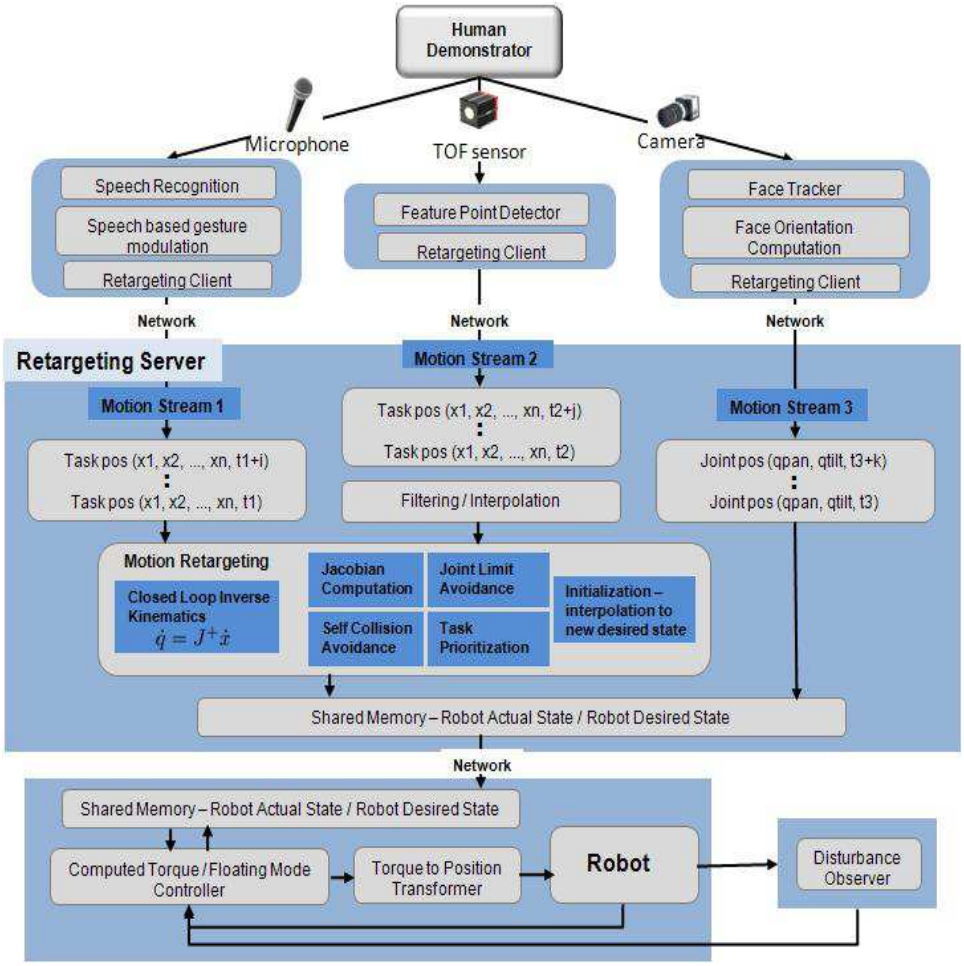

Fig. 2. System Architecture

detection algorithm in real time. The details of the feature detection algorithm are described in [19]. The feature detection algorithm generates a stream of Cartesian space task points associated with pre-specified features on the human demonstrator, such as the neck, shoulders, elbows and waist. This information is generated at a rate of about 14 Hz. There are two types of features, (i) extremum points such as hands and (ii) non-extremum key points such as shoulder or belly. For extremum key points, we use endpoints of the skeletal structure of the human body silhouette and points that are singularly closest to the camera in a depth image as candidates. For non-extremum keypoints, we make use of shape-fitting approach to track blobs corresponding to the torso and head. Once the torso (face) blob is located, keypoints associated with it (e.g., shoulder points, belly point, head point) are consequently identified.

IV. Gesture Control Interface and Motion Retargeting

In our earlier work, we described an algorithm for retargeting detected human motion (described by a set of motion descriptors) to a humanoid robot under kinematic constraints. The method is based on a first-order constrained closed loop inverse kinematics (CCLIK) problem which proved to be an effective and stable solution for obtaining robot joint commands in the presence of kinematic constraints [20], [21].

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

$$\mathbf{v}_k = \begin{bmatrix} \omega_k & \dot{\boldsymbol{p}}_k \end{bmatrix}^T, \tag{1}$$

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

$$\mathbf{v} = \boldsymbol{J}\ \dot{\boldsymbol{q}} \tag{2}$$

where $\mathbf{v}$ and $\boldsymbol{J}$ correspond to the augmented spatial velocity vector and the augmented task Jacobian matrix formed by concatenation of the individual motion descriptors:

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_1^T & \cdots & \mathbf{v}_k^T & \cdots & \mathbf{v}_N^T \end{bmatrix}^T, \tag{3}$$

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_1^T & \cdots & \boldsymbol{J}_k^T & \cdots & \boldsymbol{J}_N^T \end{bmatrix}^T. \tag{4}$$

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

As described in [20], [21], a singularity robust constrained closed loop inverse kinematics formulation which produces constrained joint velocities, $\dot{\boldsymbol{q}}_c$, is given by

$$\dot{\boldsymbol{q}}_c = \boldsymbol{J}^*(\mathbf{v}_d + \mathbf{K}_p\ \boldsymbol{e}), \tag{5}$$

where $\boldsymbol{J}^*$ denotes the singularity robust right pseudo-inverse of $\boldsymbol{J}$ regularized by the damping factor λ.

$$\boldsymbol{J}^* = \boldsymbol{W}^{-1}\boldsymbol{J}^T\ (\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^T + \lambda^2\ \boldsymbol{I})^{-1}, \tag{6}$$

In Eq. 6, kinematic constraints can be enforced by construction of an appropriate weight matrix, $\boldsymbol{W}$, to penalize and dampen motion at joints that direct motion away from 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 $\boldsymbol{W}_{JL}$ [20], [22]. 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,

$$\nabla h_i = \frac{\partial h(\boldsymbol{q})}{\partial q_i}, \tag{7}$$

and described as follows [22],

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

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 [22], 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

$$\mathrm{w}_{JL_i} = \begin{cases} 1 + |\nabla h_i| & \text{if} \quad \Delta|\nabla h_i| \geq 0, \\ 1 & \text{if} \quad \Delta|\nabla h_i| < 0. \end{cases} \tag{8}$$

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,

$$\nabla f = \frac{\partial f}{\partial \boldsymbol{q}} = \frac{\partial f}{\partial d}\ \frac{\partial d}{\partial \boldsymbol{q}}. \tag{9}$$

In case of self collisions, the second term in Eq. 9 may be computed as follows,

$$\frac{\partial d}{\partial \boldsymbol{q}} = \frac{1}{d}\left[\boldsymbol{J}_a^T(\boldsymbol{p}_a - \boldsymbol{p}_b) + \boldsymbol{J}_b^T(\boldsymbol{p}_b - \boldsymbol{p}_a)\right]^T, \tag{10}$$

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 [23]. 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

$$\mathrm{w}_{COL_i} = \begin{cases} 1 + |\nabla f_i| & \text{if} \quad \Delta|\nabla f_i| \geq 0, \\ 1 & \text{if} \quad \Delta|\nabla f_i| < 0. \end{cases} \tag{11}$$

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,

$$f = \rho e^{-\alpha d} d^{-\beta}, \tag{12}$$

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

$$\frac{\partial f(\boldsymbol{q})}{\partial d} = -\rho e^{-\alpha d} d^{-\beta}(\beta\ d^{-1} + \alpha). \tag{13}$$

It follows that ∇f may be computed from Eqs. 9, 10, and 13.

The term $\Delta|\nabla f|$ in Eq. 11 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. 11, 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. 11 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,

$$f_j = \rho_j e^{-\alpha_j d_j} d_j^{-\beta_j}. \tag{14}$$

Its gradient can be computed as before,

$$\nabla f_j = \frac{\partial f_j}{\partial \boldsymbol{q}} = \frac{\partial f_j}{\partial d_j} \frac{\partial d_j}{\partial \boldsymbol{q}}. \tag{15}$$

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,

$$\boldsymbol{W}_{COL} = \frac{1}{N_c} \sum_{j=1}^{N_c} \boldsymbol{W}_{COL_j}. \tag{16}$$

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,

$$\boldsymbol{W} = a\ \boldsymbol{W}_{JL} + (1 - a)\ \boldsymbol{W}_{COL}, \tag{17}$$

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,

$$a = \frac{1}{(N_c + 1)}. \tag{18}$$

V. Speech Command Retargeting

The speech commands were used for fine manipulation of the robots end effectors and were generated using a Natural Language Processing module that converted speech commands from the human demonstrator to a context sensitive grammar specified in advance. The grammar mapped Natural Language Segments such as "Move Right Hand Forward" or "Open Left Hand" to accepted gesture modulation commands. Commands that involved opening or closing the hands on the humanoid robot were routed to the low level servo controllers hand interface and bypassed the task control module. However, for commands that involved fine manipulation such as "Move Right Hand Forward", a displacement vector of Task coordinates was computed and incorporated into the retargeting framework. The reference task vector from the feature detection algorithm was considered as the base point and a displacement task vector that had translation coordinates for the particular task that was being commanded through the speech interface was added to it. Further, since the retargeting framework is a coupled system with the task variables augmented together into a single task command vector, a diagonal selection matrix s was used to set the priorities for tracking of task variables other than the task variable commanded through the speech interface to zero. The weighting matrix used to enfore kinematic constraints that was explained in the previous section was then premultiplied with this selection matrix to enforce a higher priority for the task variable being commanded through the speech interface.

$$s = \begin{bmatrix} 0 & & \\ & s_i & \\ & & 0 \end{bmatrix}_{(N \times N)}$$

$$p_d = p_r + \begin{bmatrix} 0 & \dots & \Delta p_i^T & \dots & 0 \end{bmatrix}^T$$

VI. Head Pose Retargeting

For the head pose, we use an incremental approach to track the pan and tilt angles. We assume that the human demonstrator is fixed to a spot while performing the gestures and therefore the demonstrators head does not translate in space. However, the Face tracking algorithm we use translates the Face bounding box whenever the demontrators head rotates along the pan and tilt axes. We convert these incremental translations in the X and Y directions of the Face bounding box, u and v, to the head pan and tilt angles, θ_p and θ_t, as follows,

$$\theta_p = atan(u)$$

$$\theta_t = atan(v)$$

VII. Dynamic Control

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

$$\boldsymbol{\tau} = \boldsymbol{H}(\boldsymbol{q})\ \ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\ \dot{\boldsymbol{q}} + \boldsymbol{\tau}_g(\boldsymbol{q}) + \boldsymbol{J}^T \mathbf{f}_e, \tag{19}$$

where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$, and $\boldsymbol{\tau}$ denote n-dimensional 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 vector of Coriolis and centrifugal terms. $\boldsymbol{\tau}_g$ is the 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, a classical strategy to control this system is to use a nonlinear model-based compensation to dynamically decouple and linearize the system via feedback using the following inverse dynamics control law,

$$\boldsymbol{\tau} = \hat{\boldsymbol{H}}(\boldsymbol{q})\ \boldsymbol{\alpha} + \hat{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\ \dot{\boldsymbol{q}} + \hat{\boldsymbol{\tau}}_g(\boldsymbol{q}), \tag{20}$$

where the notation $\hat{\cdot}$ denotes estimates of the components of the dynamic model. The vector $\boldsymbol{\alpha}$ represents a *resolved acceleration* in terms of joint variables. Provided the model parameters in Eq. 20 match those of Eq. 19, $\boldsymbol{\alpha}$ can be conveniently designed as a reference joint acceleration vector $\ddot{\boldsymbol{q}}_r$ which decouples and linearizes the closed loop system:

$$\boldsymbol{\alpha} = \ddot{\boldsymbol{q}}_r. \tag{21}$$

Typically error and rate of change in error terms are included in the resolved acceleration term as follows.

$$\boldsymbol{\alpha} = \ddot{\boldsymbol{q}}_r + k_v \dot{\mathbf{e}} + k_p \mathbf{e} \tag{22}$$

In our controller, as is typical, we set $k_v = 2\sqrt{k_p}$. For the controller used on the upper body of our humanoid system, we use a hybrid setup. The controller switches between a computed torque controller that tracks the input joint space trajectory and a non-tracking controller that only compensates gravity torques acting on the robot. The gravity compensation controller is activated when a kinematic disturbance observer senses that one of the robots upper limbs might have come in physical contact with its environment. This controller renders the robot compliant and safe for interaction with humans.

VIII. Servo Level Robot Control

A torque transformer [24]–[26] was applied to command the robot by torque commands generated by the Computed Torque and Gravity compensation based control formulations. The torque transformer was developed to convert desired joint torque command into instantaneous increments of joint position command or joint velocity command, because the motor controller in our robot was designed as a position control unit and is hard to modify.

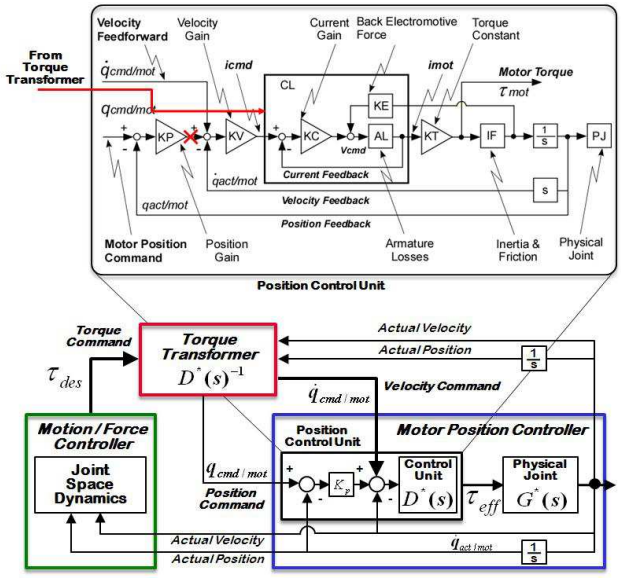

Fig. 4. Torque to Position Transformer

Fig. 4 shows the framework of the torque control system. The left block shows the Motion/Force Controller. A Computed Torque formulation was applied to account for the dynamics of the robot and command the robot by the torque command. The right block shows the Motor Position Controller. In the motor position controller, (i) the position control unit is designed with position feedback loop, velocity feedback loop and control unit and (ii) the resulting physical joint are defined. The inverse model of the ideal position control unit is applied as Torque Transformer which transforms a torque command into an instantaneous velocity command. In this framework, position command is ignored by commanding a joint position actual or by commanding position gain as zero. Through frequency response analysis or identification of the individual motor controller, the transformer has to be identified previously. Once this inverse model is generated, the torque command is directly sent to the motor current command by cancelling the effects of the inner feedback loops.

A. Whole Body Control

The controller consists of two independent components, an upper body force controller and a lower body balance controller. The balance controller uses an inverse pendulum model, and it treats the upper body simply as a mass. The dynamic model of the arm is not actively included in the balance controller, but these two systems have to be connected to achieve coordinated whole body control. The communication between the upper body and the lower body is shown in Fig 5.

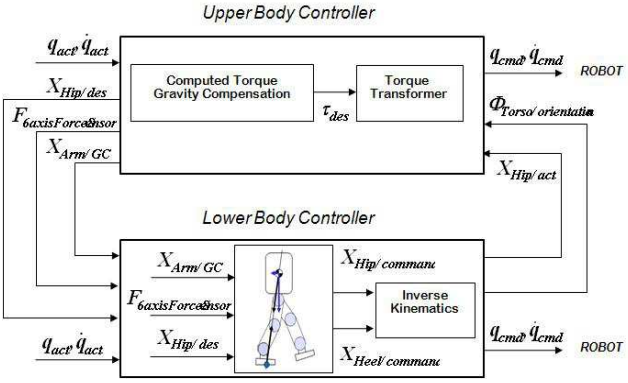

Fig. 5. Whole Body Control Framework

The upper body controller sends the desired position and orientation command of the hip to the balance controller. Then, the balance controller tries to move to the commanded position and orientation as long as stability is maintained. To calculate the upper body dynamics, the information of the actual hip position and orientation is necessary to compensate the gravity torque. The controller has to compensate for the hip position and orientation because the base of the arm moves according to the motion of the lower part. In this robot, a gyro sensor is mounted on the torso and is filtered and modified to estimate the absolute orientation of the torso. This filtered information is sent to the upper body dynamic controller.

IX. Experimental Results

The experiments were performed on the Honda Humanoid Robot. A human demonstrator performs some gestures and the robot imitates the motion in realtime in a compliant fashion respecting joint limit and self collision avoidance constraints. We present the results of two experiments that will help highlight the human friendly and multi-modal attributes of our system. The figures below are frames captured from video recordings of the experiments are in a clockwise sequence beginning from the top left of the image.

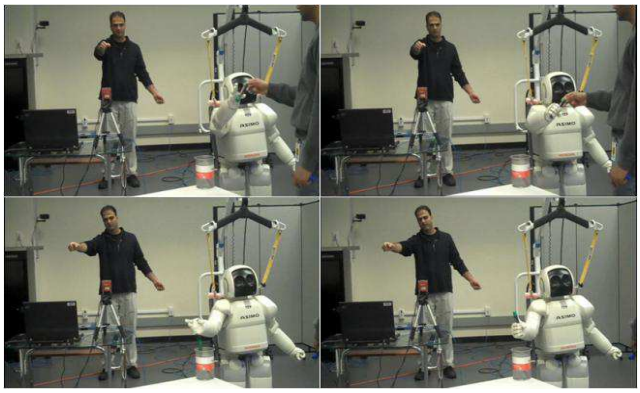

Fig. 6. Multimodal robot tele-operation

In the first experiment, illustrated by Fig. 6, a human demonstrator directs the motion of the end-effectors/hands of the humanoid robot to grasp objects presented to it by a human subject and place them in a bin. The human demonstrator uses speech commands to help with fine manipulation tasks as he deems appropriate. The speech commands are retargeted to task displacements in the motion retargeting module and are also used to open and close the robot's hands to graps and release the objects presented to it.

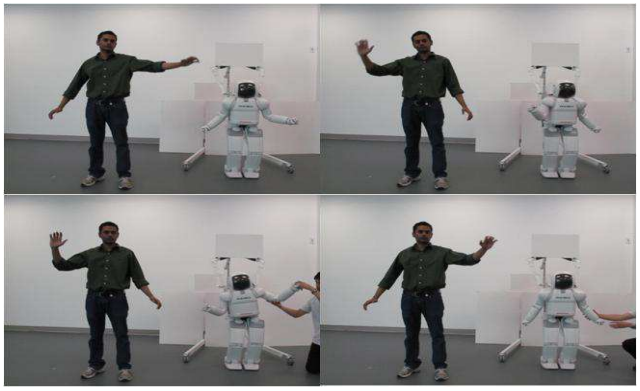

Fig. 7. Compliant Hybrid Control

In the second experiment, illustrated by Fig. 7, we outline the human friendly aspects of the system design. A human demonstrator performs gestures that are retargeted online to the humanoid robot. A compliant hybrid controller is used where the robot tracks an input trajectory using a computed torque controller and at other times, when the robot comes in physical contact with the environment, enters a *floating mode* where it does gravity compensation but does not track any input trajectory. A kinematic disturbance observer is used to identify any physical contact made by the robot with its environment. In the *floating mode*, where it compensates for gravity torques but does not track the input trajectory, the robot can be pushed away in a compliant fashion, thereby rendering it safe for interaction with humans. The authors will make available the video recordings of the experiments by email when contacted by conference organizers possibly on behalf of reviewers. Currently the authors are unable to post the recordings on a publicly accessible internet site due to the publicity protocols in place at the Honda Research Institute.

X. Summary and Future work

A system architecture with associated modules was presented to perform real time human motion imitation on a humanoid robot in a compliant fashion. The parameters that were set for the system were met satisfactorily as far as the safety of the motion imitation and multimodal nature of the task input were concerned. Through the work presented here, we were able to better understand the benefits as well as the limitations of our approach. The dynamics based controller, while offering more compliant and low gain tracking of input trajectories was also observed to have less precise tracking when compared with the joint position controlled system. The benefits, as was highlighted throughout the paper, were manifold. Other than rendering the robot safe for interaction with humans, the computed torque controller decouples the nonlinear system dynamics allowing for stable tracking of fast and dynamic motions by the robot. We anticipate that this will allow the use of our humanoid robot platform in experiments that will involve closer physical interaction with human subjects in the future. In the future, we plan to implement a more sophisticated disturbance observer that can detect collision between the robot and a human subject and incorporate it into the servo level controller in a modular and extensible fashion.

References

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

[2] 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.

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

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

[5] Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popovic. Style-based inverse kinematics. *Proceedings of SIGGRAPH 2004*, 2004.

[6] Mark B. Colton, Daniel J. Ricks, Michael A. Goodrich, Behzad Dariush, Kikuo Fujimura, and Martin Fujiki. Toward therapist-in-the-loop assistive robotics for children with autism and specific language impairment. *New Frontiers in Human Robot Interaction*, April, 2009.

[7] B. Robins, P. Dickerson, P. Stribling, and K. Dautenhahn. Robot-mediated joint attention in children with autism: A case study in robot-human interaction. *Interaction Studies*, 2004.

[8] D.J. Feil-Seifer and M. Mataric. Robot-assisted therapy for children with autism spectrum disorders. *Procs. Conf. on Interaction Design for Children: Children with Special Needs*, 2008.

[9] C.M. Stanton, P.H. Kahn Jr., R.L. Severson, J.H. Ruckert, and B.T. Gill. Robotic animals might aid in the social development of children with autism. *Procs. ACM/IEEE Int. Conf. on Human-Robot Interaction*, 2008.

[10] F. Tanaka, A. Cicourel, and J.R. Movellan. Socialization between toddlers and robots at an early child education center. *Procs. of the National Academy of Sciences of the United States of America*, 2007.

[11] 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.

[12] 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.

[13] 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.

[14] Manuel Muhlig, Michael Gienger, Sven Hellbach, Jochen Steil, and Christian Goerick. Task-level imitation learning using variance-based movement optimization. *IEEE International Conference on Robotics and Automation*, pages 1177–1184, 2009.

[15] M. Kawato, K. Furukawa, and R. Suzuki. A hierarchical neural-network model for control and learning of voluntary movement. *Biological Cybernetics*, 57:169–185, 1987.

[16] J. Heinzmann and A. Zelinsky. Quantitative safety guarantees for physical human-robot interaction. 22(7–8):479–504, 2003.

[17] M. Zinn, O. Khatib, B. Roth, and J.K. Salisbury. Playing it safe - human friendly robots. 11:12–21, 2002.

[18] Sami Haddadin, Alin Albu-Schaffer, Alessandro De Luca, and Gerd Hirzinger. Collision detection and reaction: A contribution to safe physical human-robot interaction. 2008.

[19] 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.

[20] 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.

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

[22] 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.

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

[24] Taizo Yoshikawa and Oussama Khatib. Compliant motion control for a humanoid robot in contact with the environment and humans. 2008.

[25] T. Yoshikawa, J. Park, and O. Khatib. The torque to position transformer: A framework for compliant motion control of industrial position-controlled robots. *39th International Symposium on Robotics, Seoul, Korea*, 2008.

[26] Taizo Yoshikawa and Oussama Khatib. Compliant humanoid robot control by the torque transformer. 2009.