Controlled human pose estimation from depth image streams

Youding Zhu
The Ohio State Univ.
Dreese Laborartory 395
2015 Neil Ave.
Columbus OH 43210
zhu.81.osu.edu

Behzad Dariush
Honda Research Institute USA
800 California St. Suite 300
Mountain View CA 94041
dariush@honda-ri.com

Kikuo Fujimura
Honda Research Institute USA
800 California St. Suite 300
Mountain View CA 94041
kfujimura@honda-ri.com

Abstract

This paper presents a model-based, Cartesian control theoretic approach for estimating human pose from features detected using depth images obtained from a time of flight imaging device. The features represent positions of anatomical landmarks, detected and tracked over time based on a probabilistic inferencing algorithm. The detected features are subsequently used as input to a constrained, closed loop tracking control algorithm which not only estimates the pose of the articulated human model, but also provides feedback to the feature detector in order to resolve ambiguities or to provide estimates of undetected features. Based on a simple kinematic model, constraints such as joint limit avoidance, and self penetration avoidance are enforced within the tracking control framework. We demonstrate the effectiveness of the algorithm with experimental results of upper body pose reconstruction from a small set of features. On average, the entire pipeline runs at approximately 10 frames per second on a standard 3 GHz PC using a 17 degree of freedom upper body human model.

1. Introduction

Recovering human pose from visual observations is one of the most challenging problems in Computer Vision because of the complexity of the models which relate observation with pose. An effective solution to this problem has many applications in areas such as video coding, visual surveillance, human gesture recognition, biomechanics, video indexing and retrieval, character animation, and man-machine interaction [6, 18, 8].

One of the major difficulties in estimating pose from visual input involves the recovery of the large number of degrees of freedom in movements which are often subject to kinematic constraints such as joint limit avoidance, and self penetration avoidance between two body segments. Such difficulties are compounded with insufficient temporal or spatial resolution, ambiguities in the projection of human motion onto the image plane, and when a certain configuration creates self occlusions. Other challenges include the effects of varying illumination and therefore appearance, variations of appearance due to the subject’s attire, required camera configuration, and real time performance for certain applications.

There are two main approaches in solving the pose estimation problem, categorized as model based approaches and learning based approaches. Model-based approaches rely on an explicitly known parametric human model, and recover pose either by inverting the kinematics from known image feature points on each body segment [3, 16], or by searching high dimensional configuration spaces which is typically formulated deterministically as a nonlinear optimization problem [11], or probabilistically as a maximum likelihood problem [15]. Methods based on optimization typically suffer from local minima and require good initialization. When an image sequence is available, temporal information is often used to track the human pose from a known initialization and an approximate dynamical model.

In contrast, learning based approaches directly estimate body pose from observable image quantities and do not require initialization and an accurate 3D model [2, 10]. In example based learning, inferring pose is typically formulated as a k-nearest neighbors search problem where the input is matched to a database of training examples whose 3D pose is known. Computational complexity of performing similarity search in high dimensional spaces and on very large data sets has limited the applicability of these approaches. Although faster approximate similarity search algorithms have been developed based on Locally-Sensitive Hashing [13], computation speed remains a challenge with learning based approaches.

The pose estimation formulation presented in this paper is a combination of a probabilistic method for detecting key feature points, and a model based approach for recovering
the pose from the detected features. Our framework consists of a unique prediction mechanism that provides feedback to resolve ambiguities when multiple candidate features are detected by the probabilistic inferencing algorithm. The feedback from the predicted pose is also used to estimate intermittently occluded or missing features.

The proposed framework allows the representation of the large number of human degrees of freedom involved in the execution of movement tasks to be expressed by a small number of features. These features describe motion by higher level Cartesian variables corresponding to position of landmarks on the human body. They may be keenly chosen to simplify their detection and tracking. We show that reasonable estimates of human pose can be constructed from a small set of features, provided we have an appropriate human kinematic model. Our pose estimator is very effective in tracking the detected features while satisfying joint limit and self penetration constraints.

2. Overview of the entire pipeline

Figure 1 illustrates the different modules in the pipeline. Our algorithm reconstructs human pose from a possible \( k \) features, corresponding to 3D positions of prominent anatomical landmarks on the body. Without loss of generality, we consider eight (\( k = 8 \)) such upper body features as illustrated in Figure 2(a).

The input to the proposed algorithm includes depth image streams, captured at approximately 15 frames per second using a time of flight depth imaging device [1]. The depth images are used as input to a visual processing module which detects \( m \) (\( m = 0 \cdots k \)) upper body features, denoted by \( p_{\text{det}} \), at approximately 6-12 frames per second. Note that the number of detected features at each frame may be fewer than eight (i.e. \( m < k = 8 \)) due to occlusions or unreliable observations. For numerical stability in subsequent modules, the detected features are re-sampled to a higher rate (usually 100 HZ) and represented by the vector \( \bar{p}_{\text{det}} \).

Among the eight upper body features, those which are undetected may be estimated using feedback from the prediction mechanism in a pose estimation module (feedback path 1 in Figure 2). If \( m < k \), the detected features are augmented with \( (k - m) \) predicted features (\( \bar{p} \)) obtained from forward kinematics computations of the reconstructed pose. The augmented feature vector, denoted by \( \bar{p}_{\text{det}} \), represents the \( k = 8 \) desired features used as input to a constrained pose estimation and tracking module. The recovered pose, parameterized by the vector \( q \), describes the motion of the \( n = 17 \) degree of freedom upper body model.

The predicted features are also fed-back to resolve ambiguities in case multiple candidate for a given feature are detected (false positives detected) or if a given feature is missing or intermittently occluded. This scenario corresponds to feedback path 2 in Figure 2.

![Figure 1. System diagram of the entire pipeline.](image)

3. The Algorithm

The predicted position of each feature in Figure 2(a) is described by the vector \( p_i \) and referenced to a base frame corresponding to the waist joint coordinate system. As will be described in Section 3.3, it is possible to prioritize features according to their importance or the level of confidence in the observations. For example, since elbow positions are difficult to detect, we may designate them as secondary features while assigning others as primary features.

These features are expressed in Cartesian space. They do not necessarily define the degrees of freedom required to fully describe the motion of the human model. For an \( n \) degree of freedom human model, the configuration space, or joint space, described here by the vector \( q = [q_1, \cdots, q_n]^T \), fully characterizes the motion of the human model. The mapping between configuration space velocities and Cartesian space velocities is obtained by considering the differential kinematics relating the two spaces,

\[
\dot{p}_i = J_i(q) \dot{q}
\]

where \( J_i \in \mathbb{R}^{3 \times n} \) is the Jacobian of the \( i_{th} \) feature [5] and \( \dot{p}_i \) is the velocity of \( p_i \).

![Figure 2. (a) Features used in experiments. (b) H-N-T template.](image)

3.1. Feature Detection

We use depth image streams to detect the upper body parts and extract the anatomical features. Our experimental results are based on a single 3D time of flight depth...
camera sensor [1] which captures images at approximately 15 frames per second. Unlike the existing body part detection methods that depend on the trained classifiers such as svm classifiers [9, 12] or boosted weak classifiers [7], our method is based on low-level depth image analysis that can take the depth image property into consideration, thus achieve better detection performance.

3.1.1 Head-Torso Initialization and Tracking

An important first step in the proposed feature detection 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, as shown in Figure 2(b). To initialize the tracking, the subject is asked to first assume an open arm configuration as illustrated in Figure 2. The initialization process involves the registration of a H-N-T template to the depth pixel blob. The torso is represented as rectangular box with parameters $T = \{x_0, y_0, w_T, h_T, \alpha\}$, where $w_T$ and $h_T$ represent the width and height of the torso box, respectively, $\alpha$ describes the inclination angle of the torso in the image plane relative to the upright posture, and $(x_0, y_0)$ are the frontal (image) plane coordinates at the midpoint of the top edge in the torso box. The torso box is initialized around the foreground gravity center after a few iterations of expanding and shrinking operations. After initializing the torso box, we predict the head circle based on the learned H-N-T template as described in the next section. The head circle template is parameterized by $H = \{x_{H0}, y_{H0}, r_0\}$, where $r_0$ represents the radius of the head circle template and $(x_{H0}, y_{H0})$ are the head center coordinates.

The neck trapezoid is parameterized by $N = \{x_0, y_0, w_{N1}, w_{N2}, h_N, \alpha\}$, where $w_{N1}$ and $w_{N2}$ correspond to the width of the upper and lower trapezoid edges, respectively. The relative edge lengths of the H-N-T template are obtained based on anthropometric studies reported in the biomechanics literature [19], which report body segment measurements as a fraction of the total body height.

3.1.2 Head-Torso Detection

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 $\theta$ be a set of distribution parameters used to define the H-N-T template,

$$\theta = \{\lambda_1, \cdots, \lambda_5, (\mu_1, \sigma_1), \cdots, (\mu_4, \sigma_4)\}$$

These parameters are learned by collecting training examples from image processing operations and distribution functions given below. 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 configuration. From Bayes’ rule, we can define the posterior distribution, $P(L|I, \theta)$, as,

$$P(L|I, \theta) \propto P(I|L, \theta) P(L|\theta)$$

Assuming the image likelihood functions for the H-N-T parameters are independent, we obtain

$$P(I|L, \theta) = P(I|H) P(I|N) P(I|T)$$

The prior distribution over the H-N-T includes:

$$P(L|\theta) = P(r_0|\theta) P(w_T|\theta) P(h_T|\theta) P(c|\theta)$$

where $c$ is the distance from the head center to top edge midpoint of the neck trapezoid. Then the H-N-T template is either detected or rejected based on the following criterion imposed on the likelihood function,

$$L(H, N, T) = \begin{cases} \text{yes if } \log(P(L|I, \theta)) > \text{thr} & \text{otherwise} \end{cases}$$

where the threshold (thr) is determined empirically during training by computing the likelihood function $L$ for several hundred frames and observing the H-N-T detection results.

For the head likelihood function, we use the distribution function $P(I|H) = e^{-\lambda_1 \lambda_2 \lambda_3 \lambda_4 \lambda_5}$, where $\lambda_{10}$ and $\lambda_{01}$ represent the number of false negative and false positive pixels, respectively. More precisely, $\lambda_{10}$ is the number of background pixels in the head circle, and $\lambda_{01}$ is the number of foreground pixels in the buffered head boundary (green striped region above the head in Figure 2(b)). Similarly, for neck likelihood function, we use a distribution function $P(I|N) = e^{-\lambda_2 \lambda_1 \lambda_3 \lambda_4 \lambda_5}$, where $\lambda_{10}$ is the number of background pixels in the neck trapezoid, and $\lambda_{01}$ is the number of foreground pixels in the buffered neck boundary (yellow striped region on the right and left side of the neck template in Figure 2(b)). For the torso likelihood function, we use the distribution function $P(I|T) = e^{-\lambda_4 \lambda_3 \lambda_2 \lambda_1 \lambda_5}$, where $\lambda_{10}$ is the number of background pixels in the torso box.

Note that the false positive pixels are not considered since the arm frequently occludes the torso box. Finally, the prior distribution over the H-N-T includes:

$$P(r_0|\theta) = \eta(\mu_1, \sigma_1)$$

$$P(w_T|\theta) = \eta(\mu_2, \sigma_2)$$

$$P(h_T|\theta) = \eta(\mu_3, \sigma_3)$$

$$P(c|\theta) = \eta(\mu_4, \sigma_4)$$

Figure 3 illustrates detected (top row) and rejected (bottom row) H-N-T template results, showing the effectiveness of the proposed algorithm in removing unreliable observations and occasional drifts in the tracking.
3.1.3 Arm Blob Detection

If the H-N-T template is detected, we first perform an image processing operation referred to here as skeleton analysis 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 blob 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. A few examples from this type of operation are shown as green rectangles in Figure 4.

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. A few examples from this operation are shown as blue rectangles in Figure 4.

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.

3.1.4 Upper Body Joint Localization

With the detected body parts including the head, torso, left and right arms, we localize the 3D features shown in Figure 2(a) for further processing. The head center feature is simply the 3D center of the head circle template. Note that the depth values from the depth images are used to determine the z coordinates associated with the 2D image coordinates. 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. The length of this vector is obtained from the relative anthropometric parameters obtained from the literature [19]. If the H-N-T template is undetected, or if the features are occluded, we use temporal prediction to estimate the missing features.

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 endpoint 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. They are extracted based on the following two effective assumptions:

1. If arm is in front of the body, the left arm should point to right, and the right arm should point to left.
2. If arm is beside the body, the elbow and wrist are to be labeled based on the closeness to the predicted model elbow and wrist positions.

As shown in Fig. 5, we illustrate the application of first assumption for right arm at the bottom left image, and the application of first assumption for left arm at the bottom middle image, where wrist points are shown as solid red circles and elbow points are showed as solid yellow circles. The application of second assumption is illustrated at the bottom right image, where the predicted model elbow and wrist positions from 3D model posture are displayed as unfilled circles. In the above discussion, we have taken advantage of feedback information (See Figure 1) from the predicted model pose to improve the feature detection by resolving ambiguities and estimating features which may be intermittently occluded.

We have compared the results of the feature detection algorithm versus ground truth marker data obtained from a motion capture system. The results are tabulated in Figure 6 for a Taiji motion sequence.

3.2. Cartesian tracking control

Cartesian tracking control refers to a control policy that produces the joint variables (q) such that the Cartesian error between the estimated features and the desired (from
error between the observed and computed features. The position error is simply defined as $e = p_d - p$, where $p_d$ and $p$ correspond to the observed and computed feature positions, respectively.

### 3.3. Managing multiple features

The formulation above considers estimation of human pose from a single feature. Multiple features can be handled in two ways, namely by augmentation or prioritization. These methods are described in detail in robot motion control literature [14]. In this paper, we consider feature augmentation which refers to the concatenation of the individual spatial velocities and the associated Jacobian matrix and feedback gain matrix.

Let $i$ ($i = 1 \cdots k$) be the index of the $i_{th}$ feature $\dot{p}_i$ and the associated Jacobian $J_i$. We form a $3k \times 1$ augmented spatial velocity vector $\dot{p}$ and a $3k \times n$ augmented Jacobian matrix $J$ as follows,

$$\dot{p} = \begin{bmatrix} \dot{p}_1^T & \cdots & \dot{p}_i^T & \cdots & \dot{p}_k^T \end{bmatrix}^T \quad (9)$$

$$J = \begin{bmatrix} J_1^T & \cdots & J_i^T & \cdots & J_k^T \end{bmatrix}^T \quad (10)$$

Likewise, $\dot{p}_d$ in the augmented space is the concatenation of the individual feature velocity vectors. The solution of tracking control algorithm in the augmented system follows exactly the same way as that previously described by Equation (7). The tracking error rate for each element of a feature can be controlled by the augmented feedback gain matrix $K$, which represents a $3k \times 3k$ diagonal matrix in the augmented space. The trajectory tracking error convergence rate depends on the eigenvalues of the feedback gain matrix in Equation (7): the larger the eigenvalues, the faster the convergence. In practice, such systems are implemented as discrete time approximation of the continuous time system; therefore, it is reasonable to predict that an upper bound exists on the eigenvalues, depending on the sampling time. A particular feature or its individual components can be more tightly tracked by increasing the eigenvalue of $K$ associated with that direction. By modulating the elements of $K$, we can effectively encode the relative level of confidence we have in our observations. Measurements with higher confidence will be assigned higher feedback gain values.

### 3.4. Joint limit avoidance constraints

Chan and Dubey [4] developed a joint limit avoidance algorithm 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 the joints near their limits and tends to infinity at the joint limits. One such candidate function is given by

$$H(q) = \frac{1}{4} \sum_{i=1}^{n} \frac{(q_{i,\text{max}} - q_{i,\text{min}})^2}{(q_{i,\text{max}} - q_i)(q_i - q_{i,\text{min}})}$$
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 conservatiss limits between the anatomical joint limits and the virtual joint limits used to avoid self penetration as will be described in the next Section. The gradient of $H$, denoted as $\nabla H$, represents the joint limit gradient function, an $n \times 1$ vector whose entries point in the direction of the fastest rate of increase of $H$.

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

(11)

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 [4], 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

$$w_{JLi} = \begin{cases} \frac{1}{1 + |\frac{\partial H}{\partial q_i}|} & \text{if } \Delta|\frac{\partial H}{\partial q_i}| \geq 0 \\ 1 & \text{if } \Delta|\frac{\partial H}{\partial q_i}| < 0 \end{cases}$$

(12)

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 12, 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 12 allows the joint to move freely.

3.5. Avoiding self penetration

Self penetration avoidance may be categorized as one of two types: 1) penetration between two connected segments, and 2) penetration 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 at a common rotational joint, i.e. connected segments, self collision may be handled by limiting the joint range as described in Section 3.4. Joint limits for self penetration avoidance need not correspond to the anatomical 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 rotational joint, joint limit avoidance and self penetration avoidance may be performed by using the same formulation presented in Section 3.4.

Considering the case of self penetration between two unconnected bodies, i.e. bodies which do not share a joint, we refer to the two bodies shown in Figure 7. 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 [17].

Let $\hat{n}_a = \frac{p_b - p_a}{||p_b - p_a||}$ be the unit normal vector and $\hat{d} = d \hat{n}_a$ the vector from $p_a$ to $p_b$. Consider a 3D virtual surface surrounding body $A$, shown by a dashed line in Figure 7. For every point on body $A$, its associated virtual surface point is located by the vector $\hat{d}_c = d_c \hat{n}$, 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

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

(13)

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 penetration in the critical zone. This redirection is invoked when ($d < d_c$).

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 feature $p_d$. 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 $p'_a$ and $p_d$, consider first the equivalent redirected joint velocity vector, given by

$$\dot{q}' = J'_a \dot{p}'_a$$

(14)
where $J_a = \partial p_a / \partial q$ is a $3 \times n$ Jacobian matrix and $J_a^*$ is its Damped Least Squares inverse. A redesigned task feature trajectory may be computed by

$$\dot{p}_a^d = J \dot{q}'$$  \hspace{1cm} (15)$$

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

$$\dot{q} = J^*(\dot{p}_a^d + K^* e')$$  \hspace{1cm} (16)$$

where $e' = p_a^d - p'$ and $K^*$ is a diagonal feedback gain matrix whose values are modulated as a function of the distance $d$. Note that $p_a^d$ at the current time $t$ may be computed by a first order numerical integration. The instantaneous redirection $\hat{p}_a \rightarrow \dot{p}_a^d$, as described above, produces a discontinuous first derivative of $p_a$ at the boundary $d = d_c$. The discontinuity at $p_a$ results in a discontinuity in $p_a^d$, as given by the solution in Equation 15. To preserve first order continuity, we may blend the solutions of $\dot{p}_a^d$ before and after redirection occurs. A blended solution to Equation 15 is given by

$$\dot{p}_a^d = (1 - b) \dot{p}_a + b J_p \dot{q}'$$  \hspace{1cm} (17)$$

where $b$ is a suitable blending function. A remaining question is how to specify the magnitude and direction of $\dot{p}_a^d$. An effective strategy would be to redirect the collision point so that it slides along a direction which is tangent to the surface at the penetration point, as shown in Figure 7.

$$p_a^d = \hat{p}_a - < \hat{p}_a, \hat{n}_a > \hat{n}_a$$  \hspace{1cm} (18)$$

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.

### 4. Pose Reconstruction Results

Experiments were performed using a single time-of-flight range image sensor [1]. The human performer was asked to perform a Taiji dance motion and a tennis serve motion. Figures 8 and 9 show snapshots of depth images acquired from the range sensor, the desired features (all eight upper body features), as well as the 3D reconstructed pose. The detected elbow positions are assigned a lower tracking priority as compared to the other six features. The detected features are shown by the colored circles overlaid on the depth image while the predicted features are shown by the colored spheres in the 3D reconstructed pose. Note that in some frames, some features may be undetected by the feature detection algorithm. The feature detection algorithm relies on feedback from the pose estimation module to resolve ambiguities in the detection as well as estimate those features which are missing or intermittently occluded.

We have confirmed that the joint limits and self-penetration constraints are not violated in these sequences as well as several other sequences we obtained using the time-of-flight sensor.

### 5. Summary

We have presented a computationally fast, model based control theoretic approach to estimate human pose from a small number of features detected using a probabilistic inferencing algorithm. We illustrated the results for several difficult motion sequences which many previous algorithms would have difficulty. We showed that human pose can be reliably recovered from a small set of features provided we have an adequate kinematic model and a good formulation of tracking control subject to anatomical constraints such as joint limit and self-penetration avoidance. We speculate that as the accuracy of the human model increases and by using a dynamic formulation of tracking control which incorporates physical characteristics of motion such as inertial, coriolis, and gravitational effects, we can further improve the results.

Our feature detection and pose recovery modules are tightly integrated with a prediction mechanism that allows us to resolve ambiguities when the feature detection algorithm reports multiple possible candidate features. The predicted descriptors can also be used to estimate intermittently missing or occluded features.

We have effectively used the output of our human pose estimation algorithm to transfer human motion to a humanoid robot in real time. The resulting robot motions are smooth, continuous, and very natural. In the future, we wish to extend these results to whole body pose estimation.

### References


Figure 8. Taiji sequence. Top row: depth image sequence with the detected features. Bottom row: reconstructed pose.

Figure 9. Tennis serve sequence. Top row: depth image sequence with the detected features. Bottom row: reconstructed pose.


