Real-time Simultaneous Pose and Shape Estimation for Articulated Objects Using a Single Depth Camera

Mao Ye mao.ye@uky.edu
Ruigang Yang ryang@cs.uky.edu
University of Kentucky
Lexington, Kentucky, USA, 40506

Abstract

In this paper we present a novel real-time algorithm for simultaneous pose and shape estimation for articulated objects, such as human beings and animals. The key of our pose estimation component is to embed the articulated deformation model with exponential-maps-based parametrization into a Gaussian Mixture Model. Benefiting from the probabilistic measurement model, our algorithm requires no explicit point correspondences as opposed to most existing methods. Consequently, our approach is less sensitive to local minimum and well handles fast and complex motions. Extensive evaluations on publicly available datasets demonstrate that our method outperforms most state-of-art pose estimation algorithms with large margin, especially in the case of challenging motions. Moreover, our novel shape adaptation algorithm based on the same probabilistic model automatically captures the shape of the subjects during the dynamic pose estimation process. Experiments show that our shape estimation method achieves comparable accuracy with state of the arts, yet requires neither parametric model nor extra calibration procedure.

1. Introduction

The topic of pose estimation for articulated objects, in particular human pose estimation [17, 22], has been actively studied by the computer vision community for decades. In recent years, due to the increasing popularity of depth sensors, studies have been conducted to capture the pose of articulated objects using one or more such depth sensors (detailed in Sec. 2). Despite of the substantial progress that have been achieved, there are still various limitations.

Discriminative approaches [23, 25, 21] in general are capable of handling large body shape variations. Yet it has been shown that most existing discriminative and hybrid approaches cannot achieve high accuracy with complex motions [13]. The majority of generative approaches require building point correspondences, mostly with variants of ICP methods. Thus they are prone to be trapped in local minimum, especially in the case of fast and complex motions. When a template model is used, as in generative or hybrid approaches, the consistency of body shape (limb lengths and girths) between the model and the subject is critical for accurate pose estimation. Most existing approaches either require given shapes [11, 29], small variations from the template [12], or specific initialization [27, 13]. Apparently, these requirements limit the applicability of these methods in home environments.

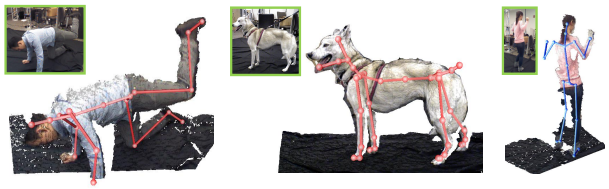

Figure 1. Our novel algorithm effectively estimates the pose of articulated objects using one single depth camera, such as human and dogs, even with challenging cases.

To overcome the limitations mentioned above, we propose a novel (generative) articulated pose estimation algorithm that **does not require explicit point correspondences and captures the subject's shape automatically during the pose estimation process.** Our algorithm relates the observed data with our template using Gaussian Mixture Model (GMM), without explicitly building point correspondences. The pose is then estimated through probability density estimation under articulated deformation model parameterized with exponential maps [2]. Consequently, the algorithm is **less sensitive to local minimum and well accommodates fast and complex motions**. In addition, we develop a novel shape estimation algorithm within the same probabilistic framework that is seamlessly combined with our pose estimation component. Last but not the least, our pipeline is implemented on GPU with CUDA, and achieves real-time performance (> 30 frames per second).

We conduct extensive evaluations on publicly available datasets. The experiments show that our algorithm outperforms most existing methods [11, 23, 30, 1, 12, 13] with large margin, especially in the case of complex motions.

Our method is flexible enough to handle animals as well, as shown in Figure 1 and Figure 8.

2. Related Work

Pose Estimation from Depth Sensors The approaches for depth-based pose estimation can be classified into discriminative, generative and hybrid approaches. Existing discriminative approaches either performed body part detection by identifying salient points of the human body [20], or relied on classifiers or regression machines trained offline to infer the joint locations [23, 25]. Shotton et al. [23] trained a Random Forest classifier for body part segmentation from a single depth image, and then estimated joint locations using mean shift algorithm. Also based on Random Forest, Taylor et al. [25] directly inferred mode-to-depth correspondences for off-line pose optimization. As will be shown in Sec. 4, our algorithm achieves significantly higher accuracy in the case of complex motions compared to [23] and the KinectSDK [16].

The generative approaches fit a template, either parametric or non-parametric, to the observed data, mostly with variants of ICP. Ganapathi et al. [12] used Dynamic Bayesian Network (DBN) to model the motion states and demonstrated good performance with an extend ICP measurement model and free space constraint. Yet their oversimplified cylindrical template cannot capture the subject's shape. The work by Gall et al. [9], built upon [10], used both depth and edge information to guide the tracker also within the ICP framework. Compared to this line of works, our method does not require explicit point correspondences and is more robust in dealing with fast complex motions.

The complementary characteristics of these two groups of approaches have been combined to achieve higher accuracy. Ye et al. [30] and Baak et al. [1] use database lookup to locate a similar pose, with PCA of normalized depth images and salient point detection, respectively. Helten et al. [13] extended [1] to obtain personalized tracker that can handle larger body shape variations and achieves real-time performance. Ganapathi et al. [11] use body part detector [20] to guide their tracker within the DBN framework. This DBN model is combined with the Random Forest classifier [23] by Wei et al. [27] to achieve high accuracy with real-time performance. The generative component of these approaches [13, 11, 27] are also ICP-based.

Shape Estimation The topic of shape estimation has also been extensively studied, however mainly for rigid bodies [3, 15]. Some methods can deal with small degree of motions for articulated objects [28, 5, 31]. Only a few methods allow the subject to perform free movements. Using multi-view setup, detailed geometry can be recovered [7, 26, 14, 24]. High quality monocular scans of strictly articulated objects under different poses can also be accurately merged to obtain a complete shape [4]. However, the noise from current depth sensors could downgrade the method's accuracy by a large degree [5]. With a single depth sensor, usually only the overall shapes are recovered [28, 13]. Compared to these methods, our approach does not require parametric models. Besides, our algorithm adapts the template shape to the subject's automatically during the pose estimation process, without requiring the subject to perform extra specific motions. Experiments show that our algorithm can deal with large shape variations.

3. GMM-based Pose and Shape Estimation

Our pose tracker uses a skinning mesh model as template. Specifically, the template consists of four components, the surface vertices $\mathcal{V} = \{\boldsymbol{v}_m^0 | m = 1, \cdots, M\}$, the surface mesh connectivity $\mathcal{F}$, the skeleton $\mathcal{J}$ and the skinning weights $\mathcal{A}$. In general, the hierarchy of the joints in the skeleton $\mathcal{J}$ forms a tree structure (kinematic tree). The goal of pose estimation is to identify the pose Θ, under which the deformed surface vertices, denoted as $\mathcal{V}(\Theta) = \{\boldsymbol{v}_m^\Theta | m = 0, \cdots, M\}$, best explain the observed point cloud $\mathcal{X} = \{\boldsymbol{x}_n | n = 1, \cdots, N\}$.

3.1. The General Probabilistic Model

Our algorithm assumes that the observed point cloud $\mathcal{X}$ follows a Gaussian Mixture Model (GMM) whose centroids are the deformed template vertices $\mathcal{V}(\Theta)$. Similar to [19], an extra uniform distribution is included to account for outliers. Therefore, the probability of each observed point $\boldsymbol{x}_n \in \mathcal{X}$ can be expressed as

$$p(\boldsymbol{x}_n) = (1-u) \sum_{m=1}^{M} p(\boldsymbol{v}_m^\Theta) p(\boldsymbol{x}_n | \boldsymbol{v}_m^\Theta) + u \frac{1}{N} \quad (1)$$

$$p(\boldsymbol{x}_n | \boldsymbol{v}_m^\Theta) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(\frac{-\|\boldsymbol{x}_n - \boldsymbol{v}_m^\Theta\|^2}{2\sigma^2}\right) \quad (2)$$

where d is the dimensionality of the point set ($d = 3$ in our case) and u is the weight of the uniform distribution that roughly represents the percentage of the outliers in $\mathcal{X}$. Here a single variance parameter σ^2 is used for all Gaussians for simplicity. Similar to [19, 6], we assume uniform distribution for the prior, that is $p(\boldsymbol{v}_m^\Theta) = 1/M$.

Under this probabilistic mode, pose estimation is cast as a probability density estimation problem that minimizes the following negative log-likelihood:

$$E(\Theta, \sigma^2) = -\sum_{n=1}^{N} \log\left(\sum_{m=1}^{M} \frac{1-u}{M} p(\boldsymbol{x_n} | \boldsymbol{v}_m^\Theta) + \frac{u}{N}\right) \quad (3)$$

which is normally solved iteratively using the Expectation-Maximization (EM) algorithm [8]. During the E-step, the posteriors $p^{\text{old}}(\boldsymbol{v}_m^\Theta | \boldsymbol{x}_n)$ are calculated using the parameters estimated from the previous iteration based on Bayes rule:

$$p_{mn} \equiv p^{\text{old}}(\boldsymbol{v}_m^{\Theta}|\boldsymbol{x}_n) = \frac{\exp\left(\frac{-\|\boldsymbol{x}_n - \boldsymbol{v}_m^{\Theta}\|^2}{2\sigma^2}\right)}{\sum_{m=1}^{M}\exp\left(\frac{-\|\boldsymbol{x}_n - \boldsymbol{v}_m^{\Theta}\|^2}{2\sigma^2}\right) + u_c} \quad (4)$$

where $u_c = \frac{(2\pi\sigma^2)^{d/2}uM}{(1-u)N}$. During the M-step, the parameters are updated via minimizing the following complete negative log-likelihood (upper bound of Eq. 3):

$$Q(\Theta, \sigma^2) = -\sum_{n,m} p_{mn}\Big(\log\big(\frac{1-u}{M}p(\boldsymbol{x_n}|\boldsymbol{v}_m^{\Theta})\big) + \log\frac{u}{N}\Big)$$

$$\propto \frac{1}{2\sigma^2}\sum_{n,m} p_{mn}\|\boldsymbol{x}_n - \boldsymbol{v}_m^{\Theta}\|^2 + \frac{d}{2}P\log\sigma^2 \quad (5)$$

$$\text{where } P = \sum_{n,m} p_{mn}; \quad \sum_{n,m} \equiv \sum_{n=1}^{N}\sum_{m=1}^{M} \quad (6)$$

So far, the probabilistic model is independent of the form of deformation model in $\mathcal{V}(\Theta)$. Cui et al. [6] and Myronenko et al. [19] used this model for rigid and non-rigid point set registration. Different from their work, we derive the pose estimation formulation under the articulated deformation model, which is more suitable for a large variety of articulated-like objects (e.g., human and many animals).

3.2. Twist-based Articulated Deformation Model

We use twist and exponential maps to represent the 3D transformations of each joint in the skeleton, similar to [2, 10, 13]. With this parametrization, the transformation of a joint in the kinematic tree can be represented as:

$$T_i = e^{\hat{\xi}_g}\prod_{k=1}^{K} e^{\delta_{ki}\hat{\xi}_k\theta_k} \quad (7)$$

$$\text{where } \delta_{ki} = \begin{cases} 1 & \text{joint } k \text{ is an ancestor of joint } i \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

Here $\hat{\xi}_g$ and $\hat{\xi}_k\theta_k$ represent the global transformation and local rotation of joint k respectively (see [18] for details). Without loss of generality, we assume the index of a joint is smaller than its children throughout this paper.

The skinning model deforms a vertex $v_m \in \mathcal{V}$ with the weighted sum of the transformations of its controlling joints as follows:

$$\boldsymbol{v}_m^{\Theta} = \sum_{k=1}^{K}\alpha_{mk}T_i^{\Theta}\boldsymbol{v}_m^0 \quad (9)$$

where $\{\alpha_{mk} \in \mathcal{A}\}$ are the skinning weights from our template. Notice that throughout the texts, homogeneous and inhomogeneous coordinates are used interchangeably without explicit differentiation for notational simplicity.

As shown in Section 3.1, the pose will be updated in an iterative fashion. At iteration t, with the assumption of small changes of both joints angles $\{\Delta\theta_k^t\}$ and global transformation $\boldsymbol{\Delta\xi}_g^t$, the deformation model in Eq. 9 can be linearized (see supplemental materials for derivation):

$$\boldsymbol{v}_m^{t+1} \approx \boldsymbol{v}_m^t + I_m^t\boldsymbol{\Delta\xi}_g^t + \sum_{k=1}^{K}\beta_{mk}\hat{\xi}_k'^t\boldsymbol{v}_m^t\Delta\theta_k^t \quad (10)$$

$$\text{where} \quad I_m^t = \begin{bmatrix} I_3 & [\boldsymbol{v}_m^t]_\times \end{bmatrix}; \quad (11)$$

$$\beta_{mk} = \sum_{i=1}^{K}\delta_{ki}\alpha_{mi}; \quad \hat{\xi}_k'^t = T_k^t\hat{\xi}_k(T_k^t)^{-1} \quad (12)$$

Here I_3 is a 3×3 identity matrix, and the operator $[\cdot]_\times$ converts a vector to a skew-symmetric matrix. The weight β_{mk} accumulates the corresponding skinning weights along all joints and represents the global influence of joint k over vertex $\boldsymbol{v}_m$. The term $\hat{\xi}_k'^t$ is a coordinate transformed twist of $\hat{\xi}_k$ via the transformation of joint k, namely T_k^t.

3.3. The Tracking Algorithm

The core of our tracking algorithm is the combination of the articulated deformation model in Sec. 3.2 with the probabilistic framework in Sec. 3.1. Plugging Eq. 10 into Eq. 5, we get the following objective function (superscript ignored for notational simplicity):

$$Q(\boldsymbol{\Delta\Theta}, \sigma^2) = \frac{1}{2\sigma^2}\sum_{n,m}\Big(p_{mn}\|\boldsymbol{x}_n - \boldsymbol{v}_m - I_m\boldsymbol{\Delta\xi}_g - \sum_{k=1}^{K}\beta_{mk}\hat{\xi}_k'\boldsymbol{v}_m\Delta\theta_k\|^2\Big) + \frac{d}{2}P\log\sigma^2$$

$$= \sum_{n,m}\frac{p_{mn}}{2\sigma^2}\|\boldsymbol{x}_n - \boldsymbol{v}_m - A_m\boldsymbol{\Delta\Theta}\|^2 + \frac{d}{2}P\log\sigma^2 \quad (13)$$

$$\text{where } A_m = \begin{bmatrix} I_m & \beta_{m1}\hat{\xi}_1'\boldsymbol{v}_m & \cdots & \beta_{mK}\hat{\xi}_K'\boldsymbol{v}_m \end{bmatrix} \quad (14)$$

$$\boldsymbol{\Delta\Theta} = \begin{bmatrix} \boldsymbol{\Delta\xi}_g^T & \Delta\theta_1 & \cdots & \Delta\theta_K \end{bmatrix}^T \quad (15)$$

In order to solve for the parameters $\{\boldsymbol{\Delta\Theta}, \sigma^2\}$, the partial derivatives of $Q(\boldsymbol{\Delta\Theta}, \sigma^2)$ over the parameters are set to zero to obtain the following equations:

$$\sum_{n,m}\frac{p_{mn}}{\sigma^2}A_m^TA_m\boldsymbol{\Delta\Theta} = \sum_{n,m}\frac{p_{mn}}{\sigma^2}A_m^T(\boldsymbol{x}_n - \boldsymbol{v}_m) \quad (16)$$

$$\sigma^2 = \frac{1}{dP}\sum_{n,m}p_{mn}\|\boldsymbol{x}_n - \boldsymbol{v}_m\|^2 \quad (17)$$

These two equations comprise the core of our novel tracking algorithm. To better regularize the optimization, we add the following two terms to the objective function:

$$E_r(\boldsymbol{\Delta\Theta}) = \|\boldsymbol{\Delta\Theta}\|^2 \quad (18)$$

$$E_p(\boldsymbol{\Delta\Theta}) = \sum_{k=1}^{K}\big(\theta_k^{\text{prev}} + \sum_{\tau=1}^{t}\Delta\theta_k^\tau - \theta_k^{\text{pred}}\big)^2 \quad (19)$$

The term E_r ensures that the solution complies with the small pose change assumption during linearization in Eq. 10. In E_p, $\{\theta_k^{\text{prev}}\}$ are the joint angles from previous frame, and $\{\theta_k^{\text{pred}}\}$ are the predicted joints angles using linear third order autoregression similar to [10]. The second term penalizes a solution's large deviation from the prediction, assuming relatively continuous motions in tracking scenario. This term is helpful in dealing with occlusions, in which case the joints corresponding to invisible parts can be relatively well constrained.

Initialize the template with previous pose
Sample a subset of points from each point set
while *Pose not converged* **do**
  E-step: Compute posteriors via Eq. 4.
  M-Step:
  - Minimize Eq. 20 for $(\mathbf{\Delta\Theta}, \sigma^2)$.
  - Update template vertices via Eq. 9.

end

Algorithm 1: The pose estimation procedure.

Our complete objective function is the weighted sum of these three terms:

$$E = Q(\mathbf{\Delta\Theta}, \sigma^2) + \lambda_r E_r(\mathbf{\Delta\Theta}) + \lambda_p E_p(\mathbf{\Delta\Theta}) \quad (20)$$

The partial derivative of E_r and E_p over $\mathbf{\Delta\Theta}$ are added to Eq. 16 and the entire linear system is solved at each iteration for the pose update $\mathbf{\Delta\Theta}$. Eq. 17 is solved for the Gaussian variance σ^2. After each pose update, the surface vertices are updated via the skinning deformation in Eq. 9. In the E-step, the posteriors are calculated according to Eq. 4. The procedure iterates until the maximum surface vertex movements is below a certain threshold ($1mm$ in our experiments). **Note that the small pose update is only enforced between two iterations, while large pose change between two frames is allowed (see our video).**

Since the computational complexity is in the order of MN, we use only a subset from each point cloud during the optimization. For the template mesh, random sampling strategy is used. The observed point set is uniformly subsampled based on the regular image grid. The pose estimation process for each new frame is summarized in Alg. 1.

3.4. Monocular Scenario

In monocular setup, the missing data introduces additional difficulties for the algorithm above. Specifically the algorithm attempts to use all given template vertices to fit the observed data. However, since the template surface is complete while the observed surface is partial, the observed partial surface will typically end up inside the complete template surface (between frontal and back surfaces). An intuitive strategy is to use only the visible part of the template from previous frame, as being adopted by Helten et al. [13]. However, such strategy can not well handle rotation of body parts, as the visible parts will change. Towards this end, we propose a two-step coarse-to-fine strategy to handle this situation. In the first step, the entire set of template vertices are used for sampling, and the pose are updated until convergence. In most cases, the body parts will be very close to their correct places. Then the visibility of the template surface is determined and only the visible part are used to refine the pose.

Another type of missing data is the occlusion of an entire body segment, either due to self occlusion or camera field of view limitation. By using only visible points in the second refining step, these two issues could be partially resolve, as the joint angles of the corresponding invisible parts will not be updated. However, they might still be affected during the first step. Therefore, we limit our sampling candidates to the set of template vertices inside the camera view. Besides, we rely on the autoregression prediction in Eq. 19 to constraint the occluded parts that are inside the camera view. With these strategies, our algorithm can effectively and reliably estimate the pose using only one single depth camera. The entire pose estimation procedure is summarized in Alg. 2.

begin Step One
  Exclude template points outside the camera view;
  Perform Alg. 1;

end
begin Step Two
  Perform visibility test and exclude invisible points from the template;
  Perform Alg. 1;

end

Algorithm 2: Pose estimation for monocular data.

3.5. Template Shape Adaptation and Initialization

The consistency of body shape (limb lengths and body part girths) between the template and the subject plays a critical role in pose estimation. In this section, we describe our novel algorithm for automatic body shape estimation, followed by the initialization procedure of our tracker.

3.5.1 Limb Lengths Estimation

In order to adjust the limb lengths of template to fit the subject, existing methods either assume presence of a personalized template model [10] or estimate the body size before tracking [27, 13]. We follow the later strategy because apparently it is more general. However, our method requires neither parametric model as in [13], nor body part detectors as in [27]. Instead, we parameterize the template vertices on limb lengths and utilize the probabilistic model in Sec. 3.1 to estimate the optimal body size.

To achieve linear parametrization, we adopt the differential bone coordinates introduced by Straka et al. [24], which is defined in a way similar to the Laplacian coordinates:

$$\boldsymbol{\eta}_m = \sum_{k=1}^{K} \alpha_{mk} \boldsymbol{\eta}_{mk}; \ \boldsymbol{\eta}_{mk} = \boldsymbol{v}_m - (\boldsymbol{g}_k - (1-\gamma_{mk})\boldsymbol{d}_k) \quad (21)$$

As explained in Fig. 2, the local geometric shape information is encoded inside these coordinates.

The adjustment of the limb lengths are achieved by introducing a scale for each bone, denoted as $\mathcal{S} = \{s_k\}$. The scaled bone vectors become $\{s_k \boldsymbol{d}_k\}$, assuming unchanged pose. Consequently, the scaled joint positions can be computed as:

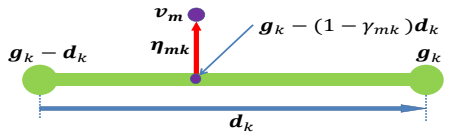

Figure 2. The differential bone coordinates [24]. $\boldsymbol{g}_k$ is the location of joint k and $\boldsymbol{d}_k$ is the vector from parent of joint k to itself. Each vector $\boldsymbol{\eta}_{m,k}$ encodes the relative position of the vertex $\boldsymbol{v}_m$ with respect to bone k. The differential bone coordinate defined in Eq. 21 accumulates such relative positions along all controlling bones of this vertex and encodes the information of surface geometry.

$$\boldsymbol{g}_k(\mathcal{S}) = \boldsymbol{g}_r + \sum_{j=1}^{K} \delta_{jk} s_j \boldsymbol{d}_j \tag{22}$$

where $\boldsymbol{g}_r$ is the position of the root that only changes with pose. Combining Eq. 22 and Eq. 21, we can reconstruct the vertex positions for the scaled template with the following equation:

$$\boldsymbol{v}_m(\mathcal{S}) = \boldsymbol{\eta}_m + \boldsymbol{g}_r + \sum_{k=1}^{K} \rho_{mk} s_k \boldsymbol{d}_k \tag{23}$$

$$\text{where} \quad \rho_{mk} = \beta_{mk} - \alpha_{mk}(1-\gamma_{mk}) \tag{24}$$

The idea of our limb lengths adaptation is to estimate the scales, such that the vertices of the scaled template defined in Eq. 23 maximizes the objective function defined in Eq. 13. Therefore, plugging Eq. 23 into Eq. 13 and changing the unknown from pose to the scales, we obtain the following objective function for limb length scales estimation:

$$Q(\mathcal{S}) = \sum_{nm} \frac{p_{mn}}{2\sigma^2} \Big(\sum_{k=1}^{K} \rho_{mk} s_k \boldsymbol{d}_k + \boldsymbol{\eta}_m + \boldsymbol{g}_r - \boldsymbol{x}_n \Big) \tag{25}$$

Again, setting the partial derivatives of the above objective function over the scales to zero yields:

$$\sum_{j=1}^{K} \Big(\frac{1}{\sigma^2} \boldsymbol{d}_k^T \boldsymbol{d}_j \sum_{n,m} p_{mn} \rho_{mj} \rho_{mk} \Big) s_j = \frac{\boldsymbol{d}_k^T}{\sigma^2} \cdot$$

$$\sum_{n,m} p_{mn} \rho_{mk} (\boldsymbol{x}_n - \boldsymbol{\eta}_m - \boldsymbol{g}_r); \quad k = 1, \cdots, K \tag{26}$$

In order to assure a reasonable overall shape of the scaled template, we further enforce consistency of the estimated scales for a set of joints pairs:

$$\lambda_s \omega_{i,j} s_i = \lambda_s \omega_{i,j} s_j; \quad < i, j > \in \mathcal{B} \tag{27}$$

Here the global weight λ_s leverages the relative importance of this regularization term with respect to the data term in Eq. 26, and $\omega_{i,j}$ represents the relative strength of the similarity for the pair (s_i, s_j) among others. For human and animals, we set $\omega_{i,j} = 1$ for symmetric parts (e.g left/right arm) and 0.5 for connected bones. Combining Eq. 26 and Eq. 27, we can solve for the scales so that the template is best fit to the observed points under our probabilistic model.

Notice that the parametrization in Eq. 23 requires unchanged pose. However, before the template is appropriately scaled, the pose might not be well estimated. Therefore, we iterate between pose estimation and scale estimation until the estimated scales converge. The procedure is summarized in Alg. 3. The effectiveness of our template limb length adaptation is illustrated in Fig. 9 and our supplemental video.

while *Scales not converged* **do**
  Estimate pose Θ using Alg. 2;
  Calculate the joints $\{\boldsymbol{g}_k, \boldsymbol{g}_r\}$ and $\{\boldsymbol{d}_k\}$ from Θ;
  Compute $\{\boldsymbol{\eta}_m\}$ according to Eq. 21;
  Estimate scales $\mathcal{S}$ via Eq. 26 and Eq. 27;
  Update the template with the scales $\mathcal{S}$
end

Algorithm 3: Limb lengths estimation procedure.

3.5.2 Geometric Shape Adaptation

Besides limb length adaptation, we further develop an automatic method to capture the overall geometry of the subject directly inside the pose estimation process, which does not require the subject to perform any additional specific motions as in [13]. The key insight is that upon convergence of pose estimation, the maximum of posteriors $p(\boldsymbol{v}_m|\boldsymbol{x}_n)$ over all $\{x_n\}$ naturally provide information about the point correspondences. Moreover, the corresponding posteriors can serve as a measure of uncertainty. With a sequence of data, we can treat each such correspondence as an observed samples of our target adapted surface vertex $\hat{\boldsymbol{v}}_m$. Consequently, the weighted average over all the samples can be used to represent our adapted template:

$$\bar{\boldsymbol{v}}_m = \sum\nolimits_f \omega(\boldsymbol{x}_{(m)}^f) \boldsymbol{x}_{(m)}^f \Big/ \sum\nolimits_f \omega(\boldsymbol{x}_{(m)}^f) \tag{28}$$

where $\boldsymbol{x}_{(m)}^f$ is the correspondence identified via maximum of posteriors at frame f. The weight $\omega(\boldsymbol{x}_{(m)}^f)$ is designed to take into account both the uncertainty based on the posterior and the sensor noise based on depth, and is defined as follows:

$$\omega(\boldsymbol{x}_{(m)}^f) = p(\boldsymbol{v}_m|\boldsymbol{x}_{(m)}^f) / [\boldsymbol{x}_{(m)}^f]_z^2 \tag{29}$$

where the quantity $[\cdot]_z$ denotes the z (depth) component.

In order to ensure smoothness of final adapted surface and to further handle noises, we allow the movement of a surface vertex only along its original normal. Consequently, a displacement d_m is estimated for each vertex $\boldsymbol{v}_m$, by optimizing the following objective function:

$$E_d = \sum\nolimits_{m=1}^{M} \Big(\omega_m \| d_m \boldsymbol{n}_m - (\bar{\boldsymbol{v}}_m - \boldsymbol{v}_m^0) \|^2 + \lambda_d \| d_m \|^2 \Big) + \lambda_e \sum_{<m,l> \in \mathcal{F}} \| d_m - d_l \|^2 \tag{30}$$

where the weight $\omega_m = 1$ if the vertex has correspondence up to the current frame and 0 otherwise. The first term moves the vertex to the projection of the weighted sum defined in Eq. 28 on the normal $\boldsymbol{n}_m$. The second term penalizes large movements and the third one enforces smoothness

	SMMC-10	EVAL	PDT
Subjects	One male	Two males, one female	Three males, one female
Data size	28 Sequences, 100 or 400 frames each (~50% each case)	7 sequences per subject, around 500 frames per sequence	4 sequences per subject, 1000~2000 frames per sequence
Motion complexity	Relatively simple	Moderate to complex (cart-wheels, hand standing, sitting on floor, etc.)	Moderate to complex (jumping, sitting on floor, dancing, etc.)
Ground truth data	Markers	Joints	Joints + Transformations

Figure 3. The three datasets we use for quantitative evaluations.

between connected vertex pairs. In our implementation, we prune correspondences based on euclidean distance in each frame to remove noises, and perform this adaptation only every L ($L = 5$ in our experiments) frames because nearby frames in general provide little new information.

3.5.3 Template Initialization

As opposed to most existing methods that require prior knowledge of initial pose, our method can handle large pose variations. In general, our tracker only requires knowledge of the rough global orientation of the subject, for example, whether the subject is facing towards the camera or to the left, etc. The local configuration of each limb can be effectively derived using our tracking algorithm in most cases. Please see our supplemental materials for examples. For pose tracking applications, the limb length estimation process described in Sec. 3.5.1 is performed using the first F frames ($F = 5$ in our experiments), as it requires repeated pose estimation and is relatively more time-consuming. Notice that the algorithm favors all segments of the articulated object being visible, as the scales of the invisible parts can only be estimated via the regularization term in Eq. 27 and might not be accurate. In addition, poses that resolves more joint ambiguities are preferred in this process. For example, an arm bending pose better defines the length of both the upper arm and forearm than a T-pose.

4. Experiments

In order to take advantage of the parallelism of the computations in our algorithm, especially the calculation of posteriors, we implement our pipeline on GPU with CUDA. With our current un-optimized implementation, each iteration of our pose estimation together with geometric shape adaptation takes about 1.5ms on average, with sub-sampling of around 1000 points for each point set. The running time is measured on a machine with Intel Core i7 3.4GHz CPU and Geforce GTX 560 Ti graphics card. Since our algorithm normally requires < 15 iterations in total to converge during tracking, the entire pipeline runs at real time. In our experiments, we assume segmentation of the target subject is relatively simple using depth information. Therefore, the computational time for segmentation is not considered here.

Parameters: The parameters in our algorithm are empirically set to fixed values throughout our experiments and are listed in Table 1. The only exception is the u in Eq. 1, which denotes the degree of noise and is data dependent. It is set to 0.01 unless significant noises are present. Although in other related methods[19, 6], σ^2 is initialized from the input data directly, we found that such strategy generally largely overestimates the value of σ^2 and will try to collapse the template to fit the input point cloud. Different from shape registration applications, for articulated shapes tracking, it would destroy the temporal information and introduce extra ambiguities. Therefore we use a fixed value instead. Due to the multiplier $\frac{1}{\sigma^2}$ in Eq. 13, which is normally $\geq 10^4$, the regularization terms generally require large global weights.

	Eq. 20		Eq. 27	Eq. 30	
Initial σ^2	λ_r	λ_p	λ_s	λ_d	λ_e
$0.02^2(m^2)$	1000	500	1000	1	0.1

Table 1. The parameter settings for our experiments.

4.1. Tracking Accuracy Evaluations

The accuracy of our algorithm is evaluated on three publicly available datasets, namely SMMC-10 [11], EVAL [12] and PDT [13], that contain ground truth joint (marker) locations. A summary of these datasets is provided in Fig. 3. Due to the discrepancy of ground truth data format provided and joint definitions across trackers, different methods for accuracy measurement are needed. For the SMMC-10 and EVAL datasets, we use the same strategy as in [30]. Specifically, we align our template to one single frame and mount the corresponding markers to our template. The markers are then transformed with our estimated pose and directly compared with the ground truth. For the PDT dataset, we follow the strategy described in their paper [13]. The estimated joints are transformed via the provided transformations to

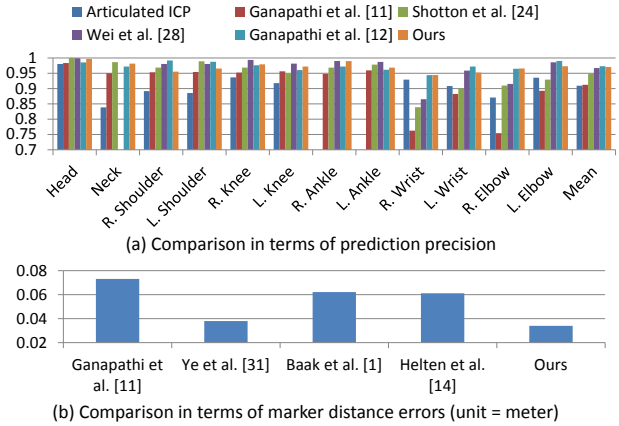

Figure 4. Quantitative evaluation of our tracker on the SMMC-10 dataset [11] with two error metrics. Notice that in (b), although the method by Ye et al. [30] achieves comparative accuracy, their reported computation time is much higher.

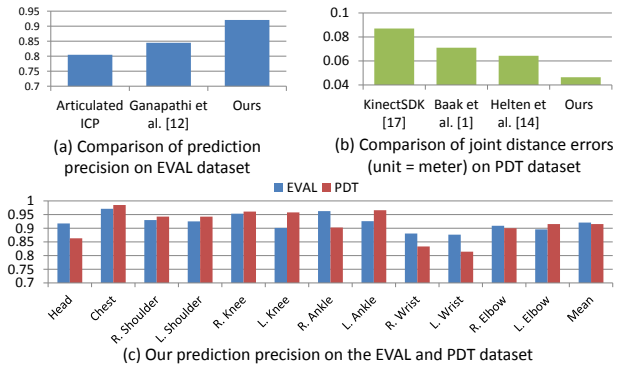

Figure 5. Quantitative evaluations of our tracker's accuracy with comparisons with the state of the arts.

local coordinate system of each corresponding joint, and the mean of displacement vectors for each joint is subtracted to account for joint definition difference.

Previous methods reported their accuracy according to two different error metrics, directly measurement of the euclidean distance between estimated and ground truth joints, or the percentage of correctly estimated joints (euclidean distance error within $0.1m$). Therefore, we show our results in both ways for comparison purpose. The accuracy of our tracker on the SMMC-10 dataset is shown in Fig. 4. In this relatively simple dataset, our method outperforms most existing methods, and is comparable with two recent work [12, 27]. The comparisons on the other two more challenging datasets are shown in Fig. 5. The results clearly show that our method outperforms all the compared methods by a large margin. That means our approaches can handle complex motions more accurately and robustly, while still achieves real-time performance. (The numbers for the compared methods are reproduced from their own papers, except for the Articulated ICP and the KinectSDK of which the numbers are obtained from [12] and [13], respectively.)

As mentioned before, our tracker well handles complex motions. The tracking results of various complex poses from our own data and the publicly available datasets are shown in Fig. 1 and in Fig. 6 respectively. Furthermore, visual comparisons between our results and the skeletons estimated by KinectSDK are shown in Fig. 7. (**More results are provided in our supplemental materials.**)

Figure 7. Visual comparison of our results (second row) with KinectSDK [16] (first row) on some relatively complex poses.

Figure 8. Example results of our algorithm applied to dog data.

Non-human Subjects Our algorithm can also be applied to articulated objects other than human beings. To demonstrate its applicability, we test on data captured from a dog. A major challenge in this data is the severe self occlusions. Some of our results are show in Fig. 1 and Fig. 8. Note that for discriminative and hybrid approaches, a database of this animal will be needed; while we only need a skinning template. Besides, none of existing generative methods have reported results on this type of data, except with multi-view setup that resolves the severe occlusion [10].

4.2. Shape Adaptation Accuracy

To quantitatively evaluate the performance of our body shape adaptation algorithm, we compare our results on the PDT dataset [13] with their ground truth shape data. Note that in our pipeline, the body shapes are **automatically** deduced during the tracking process, while a specific calibration procedure is required in [13]. To measure fitting errors, we estimate the pose of the ground truth shape using our algorithm and deform our template accordingly. On average, we achieve comparative accuracy as Heltel et al. [13]: $0.012m$ v.s $0.010m$. To compare, the mean fitting error reported by Weiss et al. [28] on their own test data is around $0.010m$. It should be emphasized that [13] requires an extra calibration procedure and [28] assumes small motion between their input, while our method directly operates on the dynamic input. It should also be pointed out that the ground truth shapes from [13] were obtained by fitting SCAPE models to data from scanner. Thus they do not well capture the effects of subject's clothing, which is captured by our algorithm through fitting with the input data. In Fig. 9, the effectiveness of our adaptation procedure is visualized.

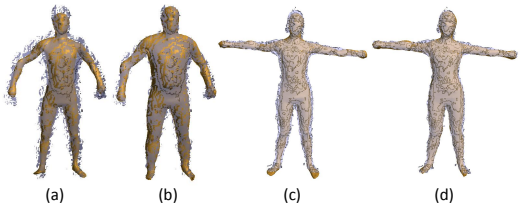

Figure 9. Visual results of our shape adaptation algorithm. (a) and (c) are results of only pose estimation (male and female respectively), while (b) and (d) are results with shape adaptation during tracking. Input meshes are overlaid on each result. Notice the adjustment of limb lengths, in particular the arms and feet of the male subject.

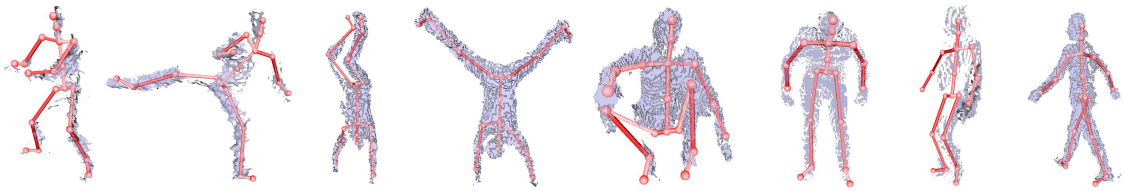

Figure 6. Visualization of example tracking results on the publicly available datasets.

5. Conclusion and Future Work

In this paper, we proposed a novel algorithm for simultaneous pose and shape estimation for articulated objects using one single depth camera. Our pipeline is fully automatic and runs in real time. Through extensive experiments, we have demonstrated the effectiveness of our algorithm, especially in handling complex motions. In particular, we have shown results of tracking an animal, which have not been demonstrated in previous methods with monocular setup.

Our pose tracker could be further improved, for example by taking into account free space constraints as in [12]. Besides, the computational complexity can be greatly reduced through fast Gauss transform during computation of the posteriors [19]. Looking into the future, we would like to explore schemes for adaptive limb lengths estimation along with pose estimation, instead of simply using the first few frames, which usually does not provide complete desired information. Besides, with dynamic scenes, similar to most existing techniques, our method only captures the subject's overall shape. The challenging problem of reconstructing detailed geometry is also interesting.

Acknowledgments This work is supported in part by US National Science Foundation grants IIS-1231545 and IIS-1208420 , and Natural Science Foundation of China (No.61173105,61373085, 61332017).

References

[1] A. Baak, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *ICCV*, pages 1092–1099. IEEE, Nov. 2011.

[2] C. Bregler, J. Malik, and K. Pullen. Twist based acquisition and tracking of animal and human kinematics. *IJCV*, 2004.

[3] B. J. Brown and S. Rusinkiewicz. Global non-rigid alignment of 3-d scans. In *SIGGRAPH)*. ACM, 2007.

[4] W. Chang and M. Zwicker. Global registration of dynamic range scans for articulated model reconstruction. *ACM TOG*, 2011.

[5] Y. Cui, W. Chang, T. Nöll, and D. Stricker. Kinectavatar: Fully automatic body capture using a single kinect. In *ACCV 2012 Workshop on Color Depth Fusion in Computer Vision*, Nov 2012.

[6] Y. Cui, S. Schuon, S. Thrun, D. Stricker, and C. Theobalt. Algorithms for 3d shape scanning with a depth camera. *TPAMI*, PP(99):1, 2012.

[7] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun. Performance capture from sparse multi-view video. *ACM TOG*, 2008.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.

[9] J. Gall, A. Fossati, and L. Van Gool. Functional categorization of objects using real-time markerless motion capture. In *CVPR*, 2011.

[10] J. Gall, C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel. Motion capture using joint skeleton tracking and surface estimation. In *CVPR*. IEEE, 2009.

[11] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. Real time motion capture using a single time-of-flight camera. In *CVPR*, 2010.

[12] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. Real-time human pose tracking from range data. In *ECCV*, 2012.

[13] T. Helten, A. Baak, G. Bharaj, M. Müller, H.-P. Seidel, and C. Theobalt. Personalization and evaluation of a real-time depth-based full body tracker. In *3DV*, 2013.

[14] H. Li, L. Luo, D. Vlasic, P. Peers, J. Popović, M. Pauly, and S. Rusinkiewicz. Temporally coherent completion of dynamic shapes. *ACM TOG*, 31(1):2:1–2:11, Feb. 2012.

[15] H. Li, E. Vouga, A. Gudym, L. Luo, J. T. Barron, and G. Gusev. 3d self-portraits. *SIGGRAPH ASIA*, 32(6), November 2013.

[16] Microsoft. Kinectsdk. http://www.microsoft.com/en-us/kinectforwindows/, 2013.

[17] T. B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *CVIU*, 2006.

[18] R. M. Murray, S. S. Sastry, and L. Zexiang. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1994.

[19] A. Myronenko and X. B. Song. Point-set registration: Coherent point drift. *TPAMI*, 2010.

[20] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun. Real-time identification and localization of body parts from depth images. In *ICRA*, pages 3108–3113, 2010.

[21] G. Pons-Moll, J. Taylor, J. Shotton, A. Hertzmann, and A. Fitzgibbon. Metric regression forests for human pose estimation. In *BMVC*, 2013.

[22] R. Poppe. Vision-based human motion analysis: An overview. *CVIU*, 108(1-2):4–18, Oct. 2007.

[23] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, 2011.

[24] M. Straka, S. Hauswiesner, M. Rüther, and H. Bischof. Simultaneous shape and pose adaption of articulated models using linear optimization. In *ECCV*, volume 7572, pages 724–737. 2012.

[25] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *CVPR*, 2012.

[26] D. Vlasic, I. Baran, W. Matusik, and J. Popović. Articulated mesh animation from multi-view silhouettes. In *SIGGRAPH)*, pages 97:1–97:9. ACM, 2008.

[27] X. Wei, P. Zhang, and J. Chai. Accurate realtime full-body motion capture using a single depth camera. *SIGGRAPH ASIA*, 31(6):188:1–188:12, Nov. 2012.

[28] A. Weiss, D. Hirshberg, and M. J. Black. Home 3d body scans from noisy image and range data. In *ICCV*. IEEE, 2011.

[29] G. Ye, Y. Liu, N. Hasler, X. Ji, Q. Dai, and C. Theobalt. Performance capture of interacting characters with handheld kinects. In *ECCV*, pages 828–841, 2012.

[30] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys. Accurate 3d pose estimation from a single depth image. In *ICCV*, 2011.

[31] M. Zeng, J. Zheng, X. Cheng, and X. Liu. Templateless quasi-rigid shape modeling with implicit loop-closure. In *CVPR*, June 2013.