IEEE Access
The journal for rapid open access publishing

IEEE Xplore Digital Library
http://ieeexplore.ieee.org/Xplore

A Generative Human-Robot Motion Retargeting Approach using a Single RGBD Sensor

SEN WANG[1,2],*Student Member, IEEE*, XINXIN ZUO[2,1],*Student Member, IEEE*, RUNXIAO WANG[1], and RUIGANG YANG[2,3,4], *Senior Member, IEEE*

***Abstract*—The goal of human-robot motion retargeting is to let a robot follow the movements performed by a human subject. Typically in previous approaches the human poses are precomputed from a human pose tracking system, after which the explicit joint mapping strategies are specified to apply the estimated poses to a target robot. However, there is not any generic mapping strategy that we can use to map the human joint to robots with different kinds of configurations. In this paper, we present a novel motion retargeting approach that combine the human pose estimation and the motion retargeting procedure in a unified generative framework without relying on any explicit mapping.**

First, a 3D parametric human-robot (*HUMROB*) model is proposed which has the specific joint and stability configurations as the target robot while its shape conforms the source human subject. The robot configurations including its skeleton proportions, joint limitations, and DoFs are enforced in the *HUMROB* model and get preserved during the tracking procedure. Using a single RGBD camera to monitor human pose, we use the raw RGB and depth sequence as input. The *HUMROB* model is deformed to fit the input point cloud, from which the joint angle of the model are calculated and applied onto the target robots for retargeting. In this way, instead of fitted individually for each joint, we will get the joint angle of the robot fitted globally so that the surface of the deformed model is as consistent as possible to the input point cloud. In the end, no explicit or pre-defined joint mapping strategies are needed.

To demonstrate its effectiveness for human-robot motion retargeting, the approach is tested under both simulations and on real robots which have quite different skeleton configurations and joint Degree of Freedoms (DoFs) as compared with the source human subjects.

***Index Terms*—motion retargeting, human robot interaction, RGBD sensor**

I. Introduction

Nowadays with the advancement of object detection and recognition as well as the development in speech understanding, social robots [1] have become more intelligent. However, there are still lots of issues that need to be concerned before getting it into daily usage. Among them, one major problem is how to get the robot perform specific movements in a natural way and interact with its surrounding objects. At present the motion generating strategies are very limited which makes it still an active research topic for which many approaches have been proposed [2]–[4]. One effective way of generating motion for the robots is to let a robot mimic the movement of a human, i.e., human-robot motion retargeting. The goal is to drive a humanoid robot to move in a natural way as provided with the joint movements or positions of human subjects. In this paper, we pay our attention on how to make robots imitate the human motion. It can be widely used in areas like robot simulation [5] and also virtual characters animation [6].

Previously, the joint position obtained from a motion capture (Mocap) system [7], [8] or Kinect skeleton tracking algorithm [9] is always considered as the input for the motion retargeting problem. Starting from that, the joint movements or the end-effector positions are applied onto robots via some predefined mapping strategies between the human subjects and robots. One straightforward way is to apply Inverse Kinematics (IK) to end-effector positions given the joint positions of human subjects [2]. Another kind of solution is to define specific mapping criteria which tries to preserve the angle of body joint [10].

These kinds of methods can perform well in some cases, however they lack the ability to generalize across various robots. The mapping between the robot and source human subjects need to be redefined whenever we want to drive a robot that has different configurations in the proportion of limbs or Degree of Freedoms (DoFs) for the joint. Another drawback for these methods is that it is difficult, if not impossible, to simultaneously enforce constraints of both joint angle and end positions. This will probably result in dissimilarity for the movements performed by the target robot and the human subjects. Moreover, the very different joint limitations and DoFs of each joint have to be taken into consideration when designing the mapping criteria, which is a tedious work and there is not any generic way to enforce these configurations into the mapping function. Stability is another essential problem that needs to be dealt with for the robots while imitating human motion. These requirements pose more challenges for human-robot motion retargeting.

To tackle the above problems, in this paper we propose a generative human robot motion retargeting approach which doesn't rely on pre-defined or explicit mapping strategies. Also, instead of dealing with the motion tracking and retargeting in two separate procedures, we intend to combine them in a unified framework using a single RGBD sensor.

In detail, for the robot model representation, it is usually modeled as joint or skeleton; in contrast we propose to use a parametric human-robot template to represent the joint together with 3D surface shape. More specifically, we propose a *HUMROB* model that has the joint configurations of the target robot while preserves the 3D shape of source human subject. The robot is modeled as 3D mesh with bones and joint embedded inside the surface. Given this *HUMROB* model, our goal of motion tracking and retargeting can be realized simultaneously by optimizing the *HUMROB* template model

Digital Object Identifier: 10.1109/ACCESS.2019.2911883

so that its surface fits as close as possible to the point cloud output by the RGBD sensor. Besides, the stability and joint limitations are enforced in the objective function naturally during the deformation. After that, we will get the joint angle of the deformed *HUMROB* model which can be applied to the robot directly since they share the same skeleton configurations. In the meantime, the deformed surface shows great pose and shape similarity with respect to the human subject. Since we do not need an explicit joint or position mapping strategies, our method can be applied to robots with different configurations quite conveniently. We classify our approach as a generative motion retargeting approach as compared with previous methods which focus on developing various skeleton mapping strategies.

As far as we know, we are the first that uses a generative unified framework to achieve motion retargeting with a single RGBD sensor. We propose a *HUMROB* model for the motion retargeting which bridges the gaps between the human subject and target robot.

This paper extends our previous work in paper [11]. Specifically, to build up the parametric human-robot template, we exploit a generative human model called Skinned Multi-Person Linear (SMPL) model, which is generated by a base mesh together with the body shape and pose parameters. The SMPL model is optimized to closely fit the body shape and poses of the human subject by conforming with the input RGBD image. In this way, we do not need any pre-scanned personalized 3D model of the human source subjects. Besides, instead of relying on any specific pose to start with, like A-pose or T-pose, we adopt the ability of deep learning based joint estimation techniques to initialize our retargeting system. Right now, our retargeting framework can work all automatically and get started immediately for any human subject in any poses. Furthermore, we exploit the facial landmarks contained in the color image to get a better estimation for the head pose.

Our algorithm is validated with both simulation and on real robots having quite different skeleton configurations from human. Satisfactory and stable motions have been generated even under extreme poses. Some examples are shown in Fig. 1.

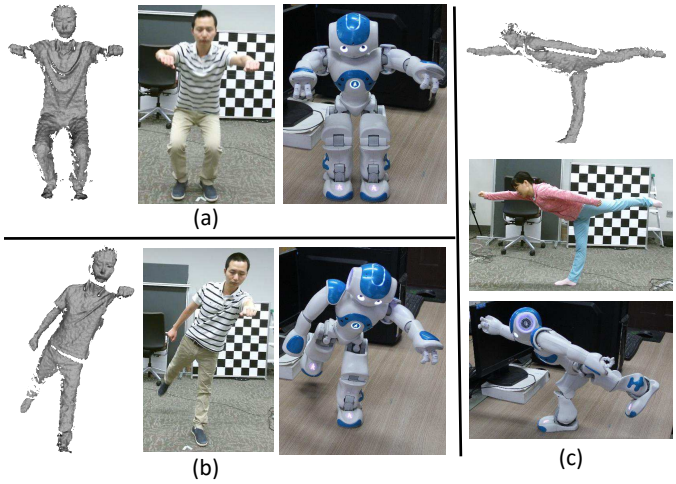


Fig. 1: Motion retargeting results of various poses. The RGBD images are captured by Kinect V2. The meshes are generated by back-projecting the depth images. We show the retargeting results on a NAO V5 robot using our proposed method.

II. Related Work

As a hot topic in robotics, motion retargeting has been widely used in areas such as gaming and motion generation. Previously, for the motion retargeting methods, two separate procedures are always involved: motion capture of the source subject and then motion retargeting to the target. In this section, we will briefly review previous work focusing on human-robot motion retargeting that is most related to ours. For a more detailed review of pose estimation and human-robot motion retargeting, we refer the reader to the survey [12] and [13] respectively.

A. Human Pose Estimation

Human pose estimation can be categorized into marker based and marker-less methods regarding to the different equipments and setups that have been used.

Marker based systems. The user is required to wear a special suit with markers attached to fixed body anchors. For example, Vicon systems [7] has been widely used in motion capture. More recently Xsens MVN motion capture system is introduced [8], [10], which consists of inertial sensors attached to the individual body segments so that we can get the marker positions more precisely. Although it can offer high accuracy and a rate of frames, the huge cost is an obstacle from letting it to get widespread used. And they are often used in lab environment.

Marker-less systems. Motion capture from color images have been studied for decades, and more recently it has got great development thanks to the deep learning techniques [14]. However, it is still an unsolved problem for computer vision community considering the inherent ill-posed condition [15]. Luckily, with the availability of consumer depth sensors, it has become easier to acquire the human skeleton. For example, the Kinect SDK [16], [17] has provided the interface to output the human skeleton, which has demonstrated its superior performance with more accurate joint estimation than the methods exploiting purely color images. Later on, apart from the default use with original SDK, researchers have focused on how to get the human pose more precisely. Lots of methods have been developed, which can be partitioned into discriminative and generative approaches. Existing discriminative approaches have either performed body part detection by identifying salient points of the human body [18], or relied on classifiers or regression machines trained offline to infer the joint locations [19]. The generative approaches intend to fit a human template to the observed data. For example, Ye et al. [20] have exploited the Gaussian Mixture Model to drive the human template to conform the observed data, without building explicit point correspondences.

B. Motion Retargeting

The challenge for motion retargeting can be summarized into two aspects: 1) Source and target has similar or same topology but with different geometry, which means the source/target is expressed as similar or even the same hierarchy link of joint, but the ratio length of bones is different from each other. 2) Source and target has different topology and also different geometry, which is a more complex situation.

Different geometry. For the different geometry but same topology, it is common to employ IK solvers integrated with soft constraints enforced in an object function. Also, some hard constraints are employed to define specific rules for the motion that must be maintained [8]–[10], [21]. For example, Ayusawa [4] proposed a method by solving the geometric parameter identification, motion planning, and the IK of the human motion-capture data problem simultaneously. The method can morphs the human model to the robot model according to some pre-defined virtual joints. Van de Perre et al. [44] proposed a generic method, which implements IK with a cost function for natural posture and validates on several different robots, to generate gestures.

Different topology. Source and target are identified as topologically different if they do not have the similar skeleton structure. Some methods have been proposed, among which exampled-based methods [22], [23] are commonly used. These methods are established from several typical mappings of the motion, from which some key poses and correspondences are pre-defined. Sel [28] introduce Golaem skeleton based on a hierarchy of bone chains which defines the beginning and the end of important kinematics parts of the input skeleton, however, this method still needs pre-defined limb attributes. Besides, Jin et al. [27] have preserved the spatial relationships between two characters to maintain the interactive motion. Recently, some deep learning methods have been proposed [24]. For example, Villegas [25] proposed an unsupervised model based on adversarial training with kinematics. The method utilizes cycle consistency to learn to solve the IK problem. Peng [45] and Rajeswaran [46] use deep reinforcement learning to control complex dynamical systems such as whole body or hand with the use of a small number of demonstrations. These kinds of methods are designed for character motion retargeting without considering the stability or physical constraints, therefore they cannot get applied to real robot systems directly.

Unlike the traditional motion retargeting systems which take the pose estimation and mapping procedure separately, in this paper we propose a unified framework which combines the two procedures using a single RGBD sensor. We can handle motion retargeting for both the different geometry and different topology without special treatment.

III. BACKGROUND

In this section, we would like to introduce some background knowledge that is essential for our approach.

A. SMPL model

The SMPL model [29] is a skinned vertex-based model which is able to realistically represent a wide range of human body shapes and posed with natural pose-dependent deformations. SMPL parametrizes a triangulated mesh by pose $\boldsymbol{\theta} \in \mathbb{R}^{72}$ and shape parameters $\boldsymbol{\beta} \in \mathbb{R}^{10}$. Optionally a global translation parameter $\gamma \in \mathbb{R}^3$ can be taken into account as well. SMPL incorporates a skeleton with $K = 24$ joints and for each joint it has 3 rotational Degrees of Freedom. Therefore, the 72 dimensional pose parameters represent the 72 joint angle in an axis-angle representation of the relative rotation between body parts. The shape parameters $\boldsymbol{\beta}$ are coefficients of a low-dimensional shape space, learned from a training set of thousands of registered 3D human body scans.

Basically, we have a base template mesh T_μ with $M = 6890$ vertices. Shape $B_S(\boldsymbol{\beta})$ and pose dependent deformations $B_P(\boldsymbol{\theta})$ are first applied to the base template T_μ by the following equation,

$$T_P(\boldsymbol{\beta}, \boldsymbol{\theta}) = T_\mu + B_S(\boldsymbol{\beta}) + B_P(\boldsymbol{\theta}) \tag{1}$$

where $B_S(\boldsymbol{\beta})$ is a blend shape function that takes as input a vector of shape parameters $\boldsymbol{\beta}$ and outputs a blend shape sculpting the subject identity. Similarly, $B_P(\boldsymbol{\theta})$ is a pose-dependent blend shape function that accounts for the effects of pose-dependent deformations.

Finally, a standard blend skinning function $W(\cdot)$ is applied to rotate the vertices around the estimated joint centers with smoothing defined by the blend weights. The posed body model is formulated as below with Ω denotes the skinning weights and $J(\boldsymbol{\beta})$ are the joint location which also depends on the body shapes,

$$\mathscr{M}(\boldsymbol{\beta}, \boldsymbol{\theta}) = W(T_P(\boldsymbol{\beta}, \boldsymbol{\theta}), J(\boldsymbol{\beta}), \boldsymbol{\theta}, \Omega) \tag{2}$$

To be different from the purely rigged model as used in our previous paper [11], with the SMPL model we are able to represent human subjects with various shapes more conveniently using the extra shape parameters.

B. Gaussian Mixture Model

We briefly introduce the GMM model here and its solution using EM algorithm, which are the basic techniques for our retargeting system. More details can be found in [30].

Now suppose that we have N captured points denoted as $S_{N*D} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N)^T$ and M model points, $T_{M*D} = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_M)^T$, where D is the dimensionality of the point set and it equals 3 in this paper. The basic assumption is that the captured points follows the GMM distribution whose centroids depend on the model points. Therefore, the probability of each captured point $\boldsymbol{s_n}$ can be expressed as,

$$p(\boldsymbol{s_n}) = (1-\mu)\sum_{m=1}^{M}\frac{1}{M}p(\boldsymbol{s}|\boldsymbol{t}_m) + \mu\frac{1}{N} \tag{3}$$

where

$$p(\boldsymbol{s_n}|\boldsymbol{t}_m) = \frac{1}{(2\pi\sigma^2)^{\frac{3}{2}}}\exp(-\frac{||\boldsymbol{s_n} - \boldsymbol{t}_m||^2}{2\sigma^2}) \tag{4}$$

where μ is the approximation for the percentage of outliers, which is considered to be evenly distributed.

Next the GMM centroids are re-parametrized with a set of parameters ϕ and get estimated by minimizing the negative log-likelihood function,

$$\begin{aligned} E(\phi,\sigma^2) &= -\sum_{n=1}^{N}\log\sum_{m=1}^{M+1}p(\boldsymbol{t}_m)p(\boldsymbol{s}_n|\boldsymbol{t}_m(\phi)) \\ &= -\sum_{n=1}^{N}\log(\sum_{m=1}^{M}\frac{1-\mu}{M}p(\boldsymbol{s}_n|\boldsymbol{t}_m(\phi))+\frac{\mu}{N}) \end{aligned} \tag{5}$$

The energy function is usually solved iteratively using the EM algorithm, as described in [31]. We briefly summarize the E-step and M-step here.

During the **E-step**, using the parameters estimated from the previous M-step, we can then use the Bayes' rules to calculate the posterior probability distribution for the model points given the captured data,

$$p^{old}(\boldsymbol{t}_m(\phi)|\boldsymbol{s}_n) = \frac{\exp(-\frac{||\boldsymbol{s}_n-\boldsymbol{t}_m^{old}(\phi)||^2}{2(\sigma^{old})^2})}{\sum_{i=1}^{M}\exp(-\frac{||\boldsymbol{s}_n-\boldsymbol{t}_i^{old}(\phi)||^2}{2(\sigma^{old})^2})+c}, \tag{6}$$

where $c = \left(2\pi(\sigma^{old})^2\right)^{\frac{3}{2}}\frac{1-\mu}{\mu}\frac{M}{N}$.

To simplify the expression, we let p^{old} stand for $p^{old}(\boldsymbol{t}_m(\phi)|\boldsymbol{s}_n)$ in the following sections.

During the **M-step**, we update the parameters ϕ and σ by minimizing the following complete negative log-likelihood function.

$$Q(\phi,\sigma^2) = \frac{1}{2\sigma^2}\sum_{n,m=1}^{N,M}p^{old}||\boldsymbol{s}_n-\boldsymbol{t}_m(\phi)||^2+\frac{\mathbf{3}}{2}N_{\boldsymbol{P}}\log\sigma^2, \tag{7}$$

where $N_{\boldsymbol{P}} = \sum_{n,m=1}^{N,M}p^{old}$. The optimization has been validated to get converged after several iterations.

C. Joint and Vertex transformation

To better represent overall pose of the human body, we adopt the classical twists exponential map for joint transformation and skeleton-subspace deformation for vertex transformation.

1) Joint transformation: For articulated models such as humans and robots, their poses can be represented by a set of twists $\hat{\xi}\phi \in SO(3)$, which is the exponential mapping that can be generalized to the Euclidean group ($SE(3)$) [32]. More details are given in the following.

In general, an articulated body transformation is expressed as the exponential of the twist $T \in SE(3)$,

$$T = \exp(\hat{\xi}\phi) \tag{8}$$

In homogeneous coordinates, the twist $\hat{\xi} \in se(3)$ can be represented as follows,

$$\hat{\xi} = \begin{bmatrix}\hat{\omega} & v\\ 0 & 0\end{bmatrix} \in \mathbb{R}^{4\times4}, \tag{9}$$

where $\hat{\omega} \in so(3)$ is 3×3 skewed-symmetric matrix converted from a 3D vector $\omega \in \mathbb{R}^3$, which is the direction of the rotation axis. The 3D vector v determines the location of the rotation axis and the amount of translation.

Suppose that robot has P joints and the whole body movement, namely the body posture, is controlled by these parameters $\Phi := (\phi_0, \phi_1 \ldots, \phi_P)$. Under the expression of the exponential map, the final transformation for each joint k is represented as,

$$T_k = \prod_{p=0}^{P}\exp(\kappa_{pk}\hat{\xi}_p\phi_p), \tag{10}$$

where $\hat{\xi}_0\phi_0$ is the global transformation and rotation for the root. $\kappa_{pk} = 1$ if p is the hierarchical parent of the joint k, otherwise $\kappa_{pk} = 0$.

2) Vertex transformation: With the joint transformation under any pose Φ in hand, the position of each vertex v in template mesh is calculated by skeleton-subspace deformation method (also known as "LBS") [33],

$$v_i(\Phi) = \sum_{p=1}^{P}\omega_{ip}T_p(\Phi)v_p^0, \tag{11}$$

where ω_{ip} is the skinning parameters which we use to relate the ith vertex to the underlying joint p, and v_i^0 stands for the position of each vertex in its reset pose.

IV. Approach

In this paper, we want to retarget the motion performed by the source human subjects to target robots in a generative way without using any pre-defined joint mapping. The overall pipeline is shown in Fig. 2 and we have mainly two steps in our framework. First, we develop a novel approach to create a parametric *HUMROB* model in section IV-A. The parametric model is assumed to keep the body shapes as close as possible to the source human subject and embedded with the specific skeleton configurations of the target robot. In this way, the parametric model bridges the gaps between the human subject and target robot to realize our goal for human-robot motion retargeting. As for the second step, the parametric model is deformed to fit the captured RGBD image of the source human subject as described in section IV-B. We have enforced joint limitations and stability constraints of the target robot during the optimization. Finally, the joint angle computed after the deformation can be applied directly onto the robots for the motion retargeting purpose.

A. HUMROB Model

In this section, we will describe our method of building up the *HUMROB* model, which is essential for our retargeting system. As a personalized 3D Human-Robot model, the *HUMROB* model is supposed to closely fit to the source human subject in body shapes for the tracking purpose. In the meantime, for the retargeting purpose, the model is parametrized by the skeleton of the target robot, which is usually different from the humans. As a parametric model, the *HUMROB* model is able to deform according to the joint angle and positions.

At the first step, we need to build a personalized 3D human model for the source human subject. Instead of relying on the

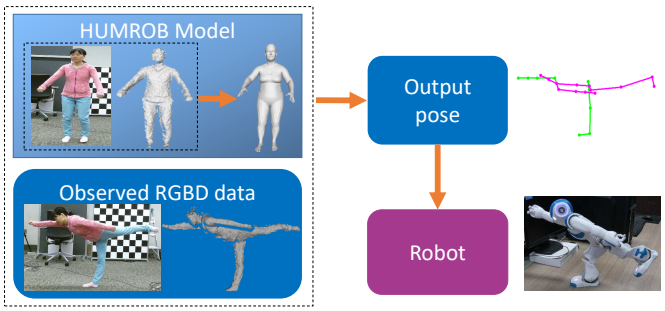


Fig. 2: System Pipeline of our approach. We use captured images from a single RGBD sensor as input. First, we build up the *HUMROB* model, which is a parametric model that has same skeleton configurations as the target robot and also fits closely to the source human subject in 3D shapes. This parametric model is deformed to fit the captured RGBD image, from which the joint angle of the target robots are computed. Finally, the joint angle can be easily applied onto the robot to get the final retargeting results with the robots performing similar motion with respect to the human subject.

fusion system which uses multiple depth sensors or a depth sequence [34] as in our previous paper [11], we exploit the SMPL model and try to build up a parametric human model for the human subject from a single RGBD image, which is more efficient and convenient for our retargeting system. This is achieved in the following two steps.

1) Model initialization: First, we take advantage of the information contained in the color image to optimize the human body shape and pose directly, so that we could get an approximate estimation for human body. To be more specific, we estimate 2D joint from the input color image using a deep learning based joint detection approach called OpenPose [35]. Then an objective function is formulated to get the projected joint of the SMPL model to be close to the 2D joint output from the network by optimizing the pose $\boldsymbol{\theta}$ and shape parameters β in the SMPL model. Mathematically, the objective function is defined as below,

$$E_{data}(\boldsymbol{\beta},\boldsymbol{\theta}) = \sum_{i\in|J|} \omega_i \rho(\Pi_K(f(J(\boldsymbol{\beta})_i,\boldsymbol{\theta})) - \hat{J}_{est,i}) \quad (12)$$

where $\hat{J}_{est}$ is the estimated joint positions in 2D space with its confidence represented by ω. $f(\cdot)$ is the function that transforms the joint from its rest pose to current positions as controlled by the pose parameters $\boldsymbol{\theta}$ using the chain rule defined by the human skeleton. Π_K is the projection operation. A differentiable Geman-McClure penalty function ρ is used here to be more robust to noisy estimates.

However, it is not sufficient to constrain the 3D human body only by its 2D joint as lots of configurations could result in the exact same 2D joint. Most methods for 3D pose estimation use some sort of pose priors to favor probable poses over improbable ones. We adopt the pose prior trained from the CMU dataset to constrain the poses. The pose prior is defined by a mixture of Gaussians, and the objective function is formulated to minimize the negative logarithm of a sum [36],

$$E_p(\boldsymbol{\theta}) = -\log \sum_j (g_j \mathrm{N}(\boldsymbol{\theta}; \mu_{\boldsymbol{\theta},\mathrm{j}}, \Sigma_{\boldsymbol{\theta},\mathrm{j}})) \quad (13)$$

We put those both data and pose prior terms together and minimize the objective function as below,

$$E_{color}(\boldsymbol{\beta},\boldsymbol{\theta}) = E_{data}(\boldsymbol{\beta},\boldsymbol{\theta}) + \lambda_p E_p(\boldsymbol{\theta}) \quad (14)$$

More details on how to solve the objective function can be found in paper [36].

2) Model refinement: After the above step, we will get an approximated human body that fits the 2D joint in the input color image. However, due to the inherent ambiguity caused by the perspective projection, the recovered 3D model still haven't fitted very well to the real surface as shown in Fig. 3(b). To deal with this problem, we further optimize the shape and pose of the SMPL model to let it conform with the depth image. Using the approximated 3D model from the previous step as a starting point, we propose an EM based optimization framework to finally get our personalized 3D human model as in Fig 3(d).

The overall objective function is defined as,

$$E(\boldsymbol{\beta},\boldsymbol{\theta},\sigma^2) = -\sum_{n=1}^{N} \log(\sum_{m=1}^{M} \frac{1-\mu}{M} p(\boldsymbol{s_n}|\mathscr{M}_m(\boldsymbol{\beta},\boldsymbol{\theta})) + \frac{\mu}{N}) \quad (15)$$

$$p(\boldsymbol{s_n}|\mathscr{M}_m(\boldsymbol{\beta},\boldsymbol{\theta})) = \frac{1}{(2\pi\sigma^2)^{\frac{3}{2}}} \exp(-\frac{||\boldsymbol{s_n} - \mathscr{M}_m(\boldsymbol{\beta},\boldsymbol{\theta})||^2}{2\sigma^2}), \quad (16)$$

where $\boldsymbol{s_n}$ is a sampled vertice from the captured depth data and $\mathscr{M}_m(\boldsymbol{\beta},\boldsymbol{\theta})$ is a vertice of the deformed SMPL model as controlled by the shape and pose parameters. The vertices of the SMPL model are taken as the centroids of GMM model and they are optimized to fit the observed depth image.

To be noticed, we haven't enforced any priors here as the current approximation is already relatively close to the real surface, and the depth image provides sufficient constraints to resolve the ambiguity.

The above function is minimized under the EM based optimization framework. In the **E-step**, we compute the posteriors with the following equation,

$$p_{mn}(\mathscr{M}_m(\boldsymbol{\beta},\boldsymbol{\theta})|\boldsymbol{s_n}) = \frac{\exp(-\frac{||\boldsymbol{s_n}-\mathscr{M}_m(\boldsymbol{\beta},\boldsymbol{\theta})||^2}{2\sigma^2})}{\sum_{i=1}^{M} \exp(-\frac{||\boldsymbol{s_n}-\mathscr{M}_i(\boldsymbol{\beta},\boldsymbol{\theta})||^2}{2\sigma^2}) + c}, \quad (17)$$

In the above equation, σ^2 is the updated value from the previous **M-step** and it is initialized to be 0.05 at the first iteration.

During the **M-step**, the shape and pose parameters of the SMPL model as well as σ^2 get updated by minimizing the following equation,

$$Q(\boldsymbol{\beta},\boldsymbol{\theta},\sigma^2) = \frac{1}{2\sigma^2} \sum_{n,m=1}^{N,M} p_{mn} ||\boldsymbol{s_n} - \mathscr{M}_m(\boldsymbol{\beta},\boldsymbol{\theta})||^2 + \frac{\mathbf{3}}{2} N_{\boldsymbol{P}} \log\sigma^2, \quad (18)$$

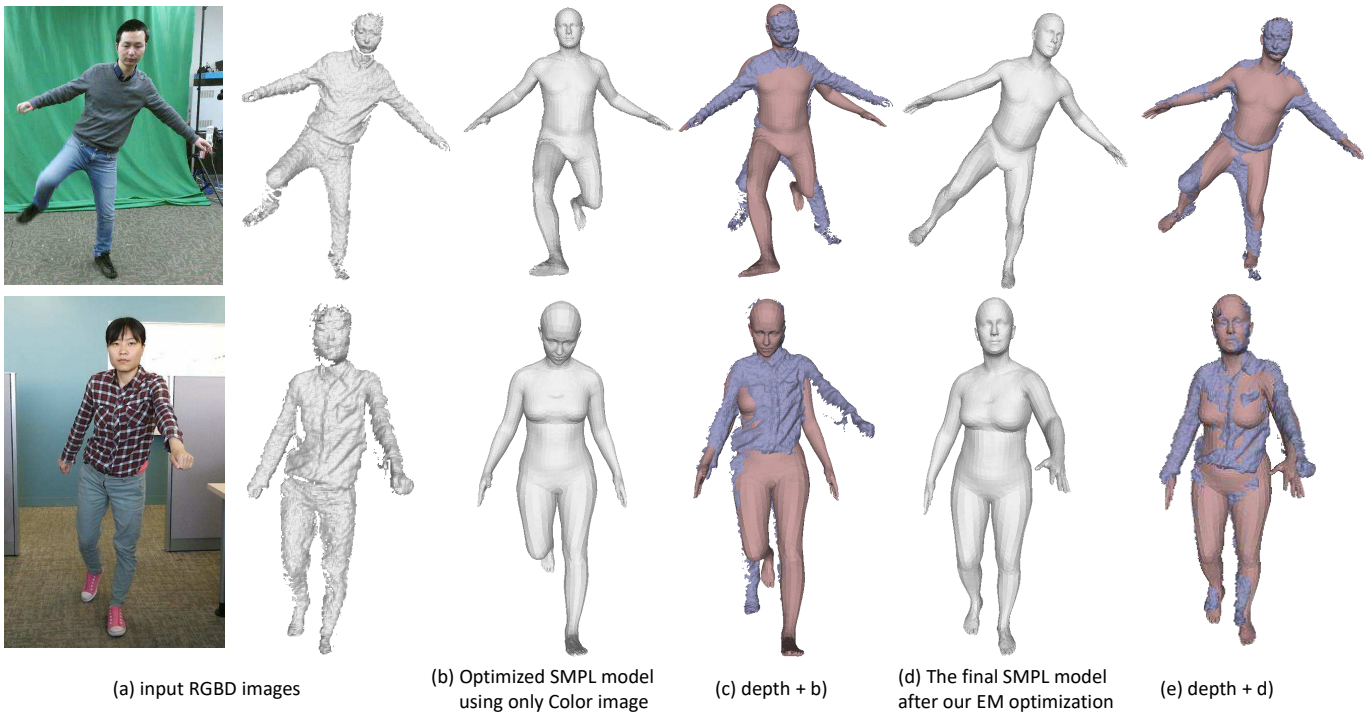


Fig. 3: Generating a personalized model with single RGBD image. (a) shows the input RGBD image. (b) is the optimized SMPL model using only the joint information in the color image. In (c), we put the optimized model from (b) together with the captured surface. (d) and (e) shows our optimized SMPL model after the EM optimization which aligns well to the real surface.

Finally, we will get a SMPL model(as shown in Fig. 3(e)) that fits well to the input surface when the optimization converges after several iterations.

To better validate the reconstruction accuracy of the result SMPL model, in Fig. 4 we have compared the generated models with 3D models reconstructed by some fusion system [34]. The mean error distance from our generated models to the 3D models is 18.9mm for the Male and 15.5mm for the Female, which is relatively small with respect to the current state-of-art human tracking system. Therefore, we believe that the generated SMPL model is sufficient for the following tracking and retargeting system.

3) Parametric model retargeting: Right now, we have got a personalized 3D human model for the source human subject. To realize our goal of human-robot motion retargeting, the next step is to embed the skeleton configurations of the target robot to this 3D model to finally get our *HUMROB* model. For the limbs, we relocate the joint positions according to the limb proportion of the robot. The skinning weights are recomputed and the joint limitations are redefined based on the target robot as shown in Fig. 5. The DoFs of the robot might also be different from the human subject, so they are also adjusted with respect to the robot. For the joint of the upper torso, the robot we are using has not any DoFs, therefore they are locked.

B. Motion Retargeting

Now we have the *HUMROB* model, we can then perform the motion retargeting by fitting this parametric model to the captured RGBD data. Some of the basic techniques used in the tracking or retargeting framework has been described in Sec. III-B and Sec. III-C.

1) Initialization: Instead of making additional constraints of specifying the initial starting pose, like T-pose or A-pose, of our tracking and retargeting system, we have exploited the deep learning based pose estimation techniques for initialization. In detail, we detect the 2D joint in the color image using the OpenPose and back-project the 2D joint into 3D space using the corresponding depth value. We take those 3D joint positions as the end-effect constraints and use Inverse-Kinematics(IK) to compute the joint angle of the target robot. For the NAO ROBOT used in this paper, we only use the joint positions of the left/right shoulders, left/right waists, left/right hips, left/right ankles.

After the initialization, the poses will get refined by our following retargeting method.

2) Energy formulation: Before delving into our approach, we want to clarify that in this paper the tracking and motion retargeting is conducted continuously along the sequence rather than selecting keyframes and performing retargeting frame by frame, which is often used in some previous work.

Now suppose that the robot has J joints and we have got the pose (the joint angle in our case) at time t, expressed as $\Theta^t := (\theta_0^t, \theta_1^t \ldots, \theta_J^t)$, and we intend to compute the changes for the pose $\Delta\Theta$ from time t to $t+1$,

$$\Theta^{t+1} = \Theta^t + \Delta\Theta \tag{19}$$

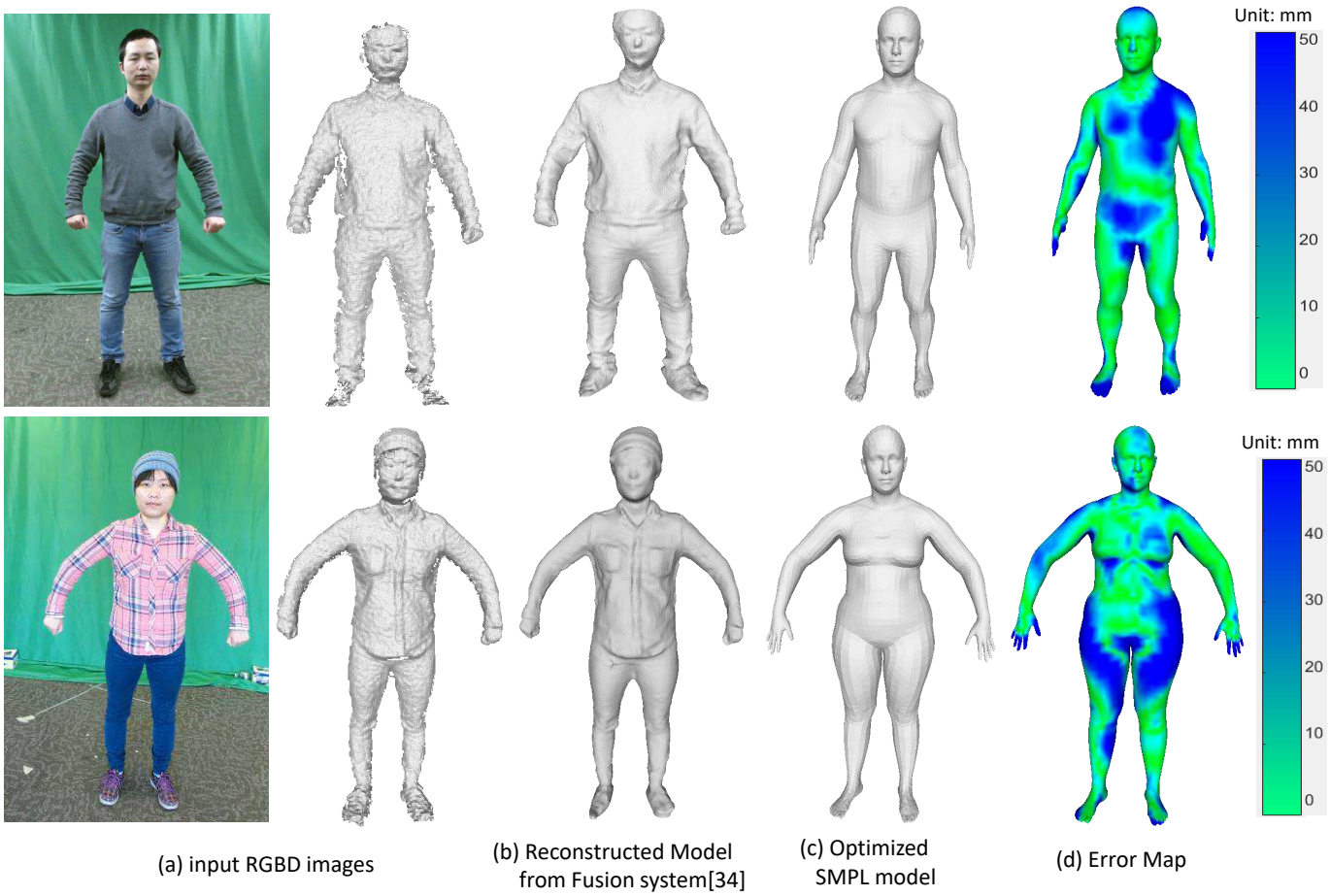


Fig. 4: Validation of the optimized SMPL model. (a) are the input RGBD images. (b) are the 3D models recovered from fusion system [34]. (c) shows our optimized SMPL models. (d) is the error map showing the distance from our optimized SMPL model to the reconstructed 3D models in (b).

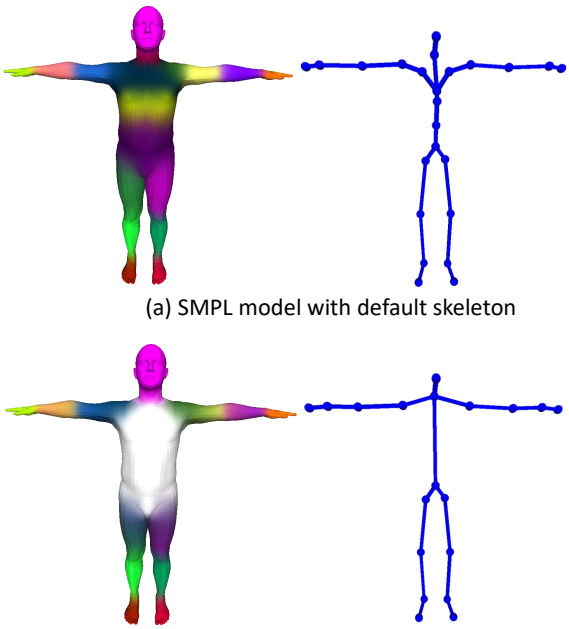


Fig. 5: Parametric model retargeting.

Under twist-based parametrization, the transformation for any joint k at time $t+1$ can be expressed as,

$$T_k^{t+1} = \prod_{j=0}^{J} \exp(\kappa_{jk}\hat{\xi}_k(\theta_j^t + \Delta\theta_j)) \tag{20}$$

To compute the pose expressed by Θ, we formulate the energy function as below,

$$E = E(\Theta, \sigma^2) + \lambda_\perp E_\perp(\Theta, \sigma^2) + \lambda_r E_r(\Delta\Theta) \tag{21}$$

The **first term** is the fitting term that penalizes the distance of deformed *HUMROB* template model from the captured depth map, which is enforced to maintain pose similarity. In details, suppose we have the observed points from the capture human pose which are denoted as $(S_{N*\mathbf{3}} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N)^T)$, and also the template points sampled from the *HUMROB* model which are represented as a vertex set $(V_{M*\mathbf{3}} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_M)^T)$. As similar to the GMM model described in Sec.III-B, $S_{N*\mathbf{3}}$ can be considered as the observed data which is supposed to be generated from the GMM centroids $V_{M*\mathbf{3}}$.

Therefore, we formulate the fitting cost between the deformed template model and captured depth map as below,

$$E(\Theta,\sigma^2)=\frac{1}{2\sigma^2}\sum_{n,m=1}^{N,M}p_{mn}||\boldsymbol{s}_n-\boldsymbol{v}_m(\Theta)||^2+\frac{\mathbf{3}}{2}N_{\boldsymbol{P}}\log\sigma^2, \tag{22}$$

Similar to p^{old} in Eq. 6, the p_{mn} can be seen as the probability of $\boldsymbol{s}_n$ relating to $\boldsymbol{v}_m$, we give more details in section IV-B4. $v_m(\Theta)$ is calculated by substituting the $T_j(\Theta)$ in Eq. 11 with Eq. 20, and then we get the position for each vertex computed as,

$$v_m(\Theta)=\sum_{j=1}^{J}\omega_{mj}\big(\prod_{j=0}^{J}\exp(\kappa_{jk}\hat{\xi}_k(\theta_j^t+\Delta\theta_j))\big)v_m^0, \tag{23}$$

The **second term** in Eq. 21 is another fitting term that ensures the normal consistency between deformed template model and the observed depth data. The cost function for this term is given as,

$$E_\perp(\Theta,\sigma^2)=\frac{1}{2\sigma^2}\sum_{n,m=1}^{N,M}p_{mn}||\boldsymbol{s}_n^\perp-\boldsymbol{v}_m^\perp(\Theta)||^2+\frac{\mathbf{3}}{2}N_{\boldsymbol{P}}\log\sigma^2, \tag{24}$$

where $\boldsymbol{s}_n^\perp$ and $\boldsymbol{v}_m^\perp(\Theta)$ are the normal vector for the $\boldsymbol{s}_n$ and $\boldsymbol{v}_m(\Theta)$ respectively.

To reduce the computation cost, we use same σ^2 for Eq. 22 and Eq. 24.

In addition to the above two fitting terms, we also have the **third term** in Eq. 21 that penalizes big changes of the poses between neighboring frames. This term is incorporated to preserve temporal consistency and smooth motion transition along the sequence.

$$E_r(\Delta\Theta)=\sum||\Delta\Theta|| \tag{25}$$

Another important constraint we must consider while performing motion retargeting is the **joint limitation** with respect to the target robot. The joint limitation for the robots usually differs from humans, which is handled naturally in our framework by specifying the lower (θ_{min}) and upper bound (θ_{max}) for each joint while performing the optimization for Eq. 21. It can be seen as a hard constraint on the cost function. Mathematically, the overall energy function is defined as below,

$$\begin{aligned}E=E(\Theta,\sigma^2)+\lambda_\perp E_\perp(\Theta,\sigma^2)+\lambda_r E_r(\Delta\Theta)\\ \Theta\in[\Theta_{min},\Theta_{max}]\end{aligned} \tag{26}$$

We present two examples in Fig. 6 showing the deformation results. As in the left column of Fig. 6, the *HUMROB* model under the initial poses with respect to the first frame is overlaid with the captured meshes. And the right column shows our deformed *HUMROB* template achieved from the above optimization.

3) Robot stability: In this paper we assume that the robot is motionless as it is driven at a very low motor speed. This indicates that it only experiences gravitational forces. Therefore instead of using the complex ZMP criteria, we turn to use floor projection of center of mass ($FCoM$) criteria to ensure the stability of the robot. Next we will show how to integrate this stability constraint into our framework.

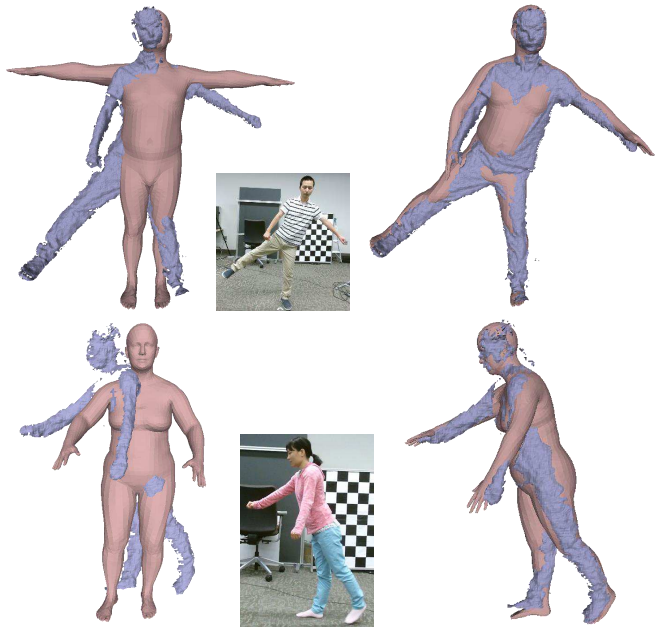


Fig. 6: Deformed *HUMROB* model results after applying our framework.

Suppose the robot has L links or bones and the vectors $\boldsymbol{p}_l$ are the vectors pointing from the CoM of each individual bones to coordinate origin. The CoM for the whole body, which denoted as $\boldsymbol{p}_{CoM}$, is computed by,

$$\boldsymbol{p}_{CoM}=\frac{\sum_{l=1}^{L}m_l\boldsymbol{p}_l}{\sum_{l=1}^{L}m_l}, \tag{27}$$

where m_l is the mass of each bone.

The floor projection of the CoM ($\boldsymbol{p}_{FCoM}$) is calculated by taking x and y component of vector $\boldsymbol{p}_{CoM}$ as its x and y component respectively and setting its z component to be zero.

The main criteria is: the motionless humanoid should be able to maintain its balance if $\boldsymbol{p}_{FCoM}$ is in the support polygon (SP), which is formed by the convex hull about the floor support points. In this paper, we use the inscribed circle of SP as the more strict support polygon, denoted as C_SP. Therefore, we introduce another term E_b to express the robot stability, which is computed as,

$$E_b=\begin{cases}0 & if\,\boldsymbol{p}_{FCoM}\ \ inside\ \ C\_SP\\ \infty & if\,\boldsymbol{p}_{FCoM}\ \ outside\ \ SP\\ \boldsymbol{o}-\boldsymbol{p}_{FCoM} & otherwise\end{cases} \tag{28}$$

where $\boldsymbol{o}$ is center of C_SP.

For each frame we will first find the optimal pose configuration by minimizing Eq. 26. Then we can compute E_b which reflects the current stability status. If E_b equals zero, it means we can safely apply the current pose into the robot with stability already maintained. However, if E_b equals ∞, we will give the opposite offset (0.3 rad) to the hips' pitch/roll angle. Otherwise, we will record the $\overrightarrow{op}_t=\boldsymbol{o}-\boldsymbol{p}_{FCoM}$ for the current frame t. Suppose for the the next frame $t+1$, we have

got the vector $\overrightarrow{op}_{t+1}$. Then if $cos(\overrightarrow{op}_t, \overrightarrow{op}_{t+1}) > cos(30°)$ & $||\overrightarrow{op}_{t+1}|| > ||\overrightarrow{op}_t||$, which indicates that the stability violation will probably occur in successive frame, we will give an opposite offset (0.05 rad) to the hips' pitch/roll angle in this case.

4) Implementation details: Alg. 1 shows the overall procedure to compute pose parameters Θ and σ^2 of the current frame. First, we adopt a linearized surface deformation strategy [37] to simplify the constrained nonlinear optimization into a linear one. Since the normal fitting term Eq. 24 cannot be linearized easily, it is neglected in this stage. The computed linear solution is then used to initialize the overall nonlinear optimization. We choose to use trust-region-reflective algorithm to find a optimal solution for the nonlinear minimization of function Eq. 26.

In more details, in the Linear solver part, we only consider the fitting term with respect to the vertex position of the template mesh(Eq. 22), and the the posteriors p_{mn} is computed as,

$$p_{mn} = \frac{\exp(-\frac{||\boldsymbol{s}_n - \boldsymbol{v}_m^{old}(\Theta)||^2}{2(\sigma^{old})^2})}{\sum_{i=1}^{M} \exp(-\frac{||\boldsymbol{s}_n - \boldsymbol{v}_i^{old}(\Theta)||^2}{2(\sigma^{old})^2}) + c}, \tag{29}$$

In the Nonlinear solver part, the constraints of both the vertex position of the template mesh and its surface normal are incorporated together (Eq. 22 and Eq. 24), and we have the the posteriors p_{mn} computed as below,

$$p_{mn} = \frac{\exp(-\frac{||\boldsymbol{s}_n - \boldsymbol{v}_m^{old}(\Theta)||^2 + \lambda_f ||\boldsymbol{s}_n^{\perp} - \boldsymbol{v}_m^{\perp old}(\Theta)||^2}{2(\sigma^{old})^2})}{\sum_{i=1}^{M} \exp(-\frac{||\boldsymbol{s}_n - \boldsymbol{v}_i^{old}(\Theta) + \lambda_f ||\boldsymbol{s}_n^{\perp} - \boldsymbol{v}_m^{\perp old}(\Theta)||^2}{2(\sigma^{old})^2}) + c}, \tag{30}$$

And c in Eq. 29 and Eq. 30 can be computed by Eq. 6.

initial pose: linear third order auto-regression
begin Step One: Linear solver for initialization
  while *Pose not converged* **do**
    E-step: Compute posteriors via Eq. 29.
    M-Step:
    - Minimize the linearized cost function of Eq. 22 for (Θ, σ^2)
    - Update template vertices via Eq. 23
    - parameters are adjusted by enforcing robot stability
  end
end
begin Step Two: Nonlinear solver for global solution
  E-step: Compute posteriors via Eq. 30.
  M-Step:
  - Minimize Eq. 26 using trust-region-reflective algorithm
  - Apply stability to adjust the parameters
end

Algorithm 1: Our pose optimization and retargeting procedure.

The parameters are further adjusted in every M-step to satisfy the stability constraints as described in section IV-B3.

Finally, in order to speed up the convergence speed and reduce the computation cost, we have performed sub-sampling on both the *HUMROB* template and the captured point cloud. We have tested different numbers of sampled points. Without losing key components and good performance, we identified the template and point maps sample number should be 6890 and 3000 respectively.

5) Head pose estimation: As compared to the limbs and torso of the human body, the head is more difficult to track with only depth information, especially when the human is standing relatively far way from the camera and there is not much detailed geometry captured in the depth image. Also there is great noise around the head in the depth image as affected by the hair. Therefore, in this paper we exploit the facial landmarks in the color image to overcome those difficulties. We assume the facial expressions haven't changed much during this procedure.

First, we detect the facial landmarks in the color image using the method [38]. Given those detected landmarks, we find their corresponding depth values in the depth image. We neglect the landmarks around the boundary of face as they probably fall outside the face part as caused by the displacement between the color and depth sensor. In this way, we can get 3d correspondences between the current frame and the previous frame around the head part. We compute the transformations of the head with those correspondences with a RANSAC based technique to filter out possible outliers. In Fig. 7 we show the comparison results with and without using the color information for the head pose estimation.

V. EXPERIMENTS

A. Data Preprocessing

In this paper, the RGBD sequences of the source human subject are captured using Microsoft Kinect V2. For the depth image, it has the resolution at 512×424 and 1920×1080 for color image. The time resolution is 30fps. The RGBD sensor is first calibrated so that the depth and color images are aligned to the same coordinates. We use background subtraction to have the human subject segmented from the images.

The pose parameters (Θ) are all computed as rad to be consistent with the real NAO robot. The algorithm is implemented in Matlab and takes about 3 seconds per frame on a regular PC with an Intel i7 3.6 GHz processor and 32GB RAM. The target pose is enforced on the real robot with Python interface through Choregraphe™.

B. Parameter settings

In Eq. 22 and Eq. 24, the initial $\sigma^2 = 0.02^2$. In Eq. 30, $\lambda_f = 0.5$. In Eq. 26, $\lambda_\perp = 0.5$ and $\lambda_r = 1000$. In Alg. 1, the pose converges when the maximum movements for all joints are less than $0.002rad$ since the precision for the real NAO robot sensor is $0.1°$.

We have tuned those parameters empirically, which then remain fixed for all the experiments.

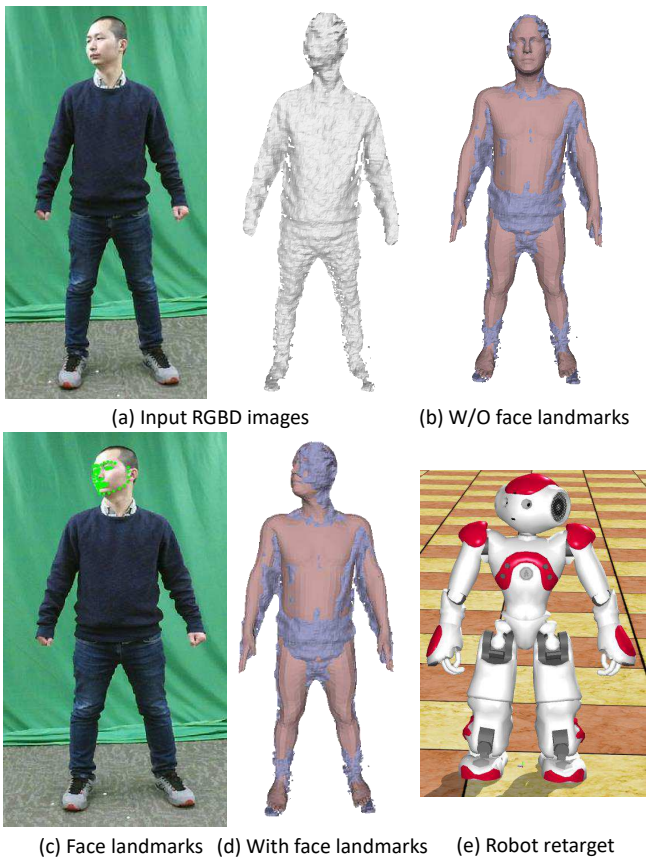


Fig. 7: Head pose estimation.

C. Robot test

We have verified our method on a real robot, namely a NAO V5 robot which is manufactured by Softbank. The robot is 58cm tall and weighs 5.2kg. The robot has 23 DoFs in total for its whole body (excluding the open/close hands DoFs), which is quite different from the generic human skeleton that has 72 DoFs overall.

We have validated our approach on both male and female human subjects performing various and even some extreme poses. We demonstrate our retargeting results on the NAO robot under several different kinds of postures in Fig. 8.

As we can see in Fig. 8a, the human subjects are waving his/her hands and the robot can mimic the hands up postures successfully showing great resemblance to the human pose. In Fig. 8b, we show the retargeting results of bow and squat postures which involve both hands and upper body motion. As we can see, they can also be executed by the robot with its poses very similar to the source human subjects. These postures mainly focus on upper/lower body posture with limited stability constraints.

We present more challenging cases in Fig. 8c and Fig. 8d with the human subjects leaning forward and kicking. In those cases, in addition to maintaining the pose similarity, stability needs to be considered as well. Especially in one foot support, stability has the highest priority to prevent the robot from falling down. As we can see from Fig. 8c and Fig. 8d, although some postures are not executed exactly as same as the human subjects, they still have large pose similarities on a global scale.

D. Stability

While we try to let the robot mimic the posetures performed by the human subjects, we have to take its stability into consideration. We want to prevent the robot from falling down especially when performing any extreme poses. In this section, we would like to further demonstrate the effectiveness of our method in terms of stability, as the stability check plays an important role in the motion retargeting. We present two cases in Fig. 9 showing our retargeting results with and without our stability constraints. The middle columns of Fig. 9 are the simulation results without the stability constraints enforced, which means they are not subjected to gravity force. In this case, the robot acts just like the human subjects. However, if we apply those poses directly onto the real NAO robot, it will actually fall down. After the stability constraints are enforced, we will get the results as shown in the right columns of Fig. 9. As you can see, we are able to achieve the most similar posture while maintaining stability.

When testing on the real NAO robot, we also restrict that at least one foot fully touches the ground. Besides, the supporting leg of the NAO robot needs to bend a little when it reaches the maximum motor torque, as shown in the third posture of Fig. 8d.

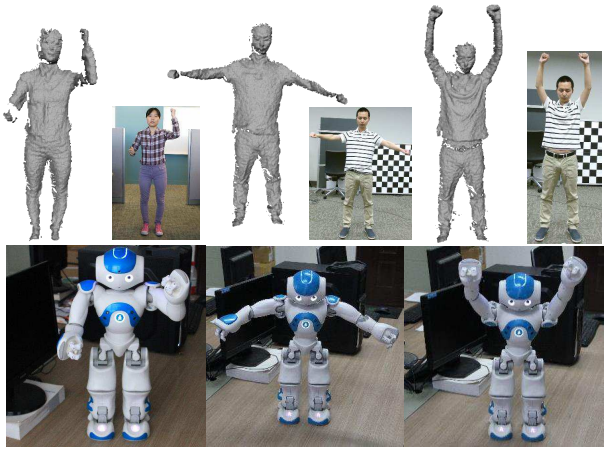
E. Comparison

1) Compare with poses from Kinect SDK: In this section, we show some qualitative comparisons (shown in Fig. 10) that we have conducted on motion tracking between our methods and using Kinect SDK [40] .

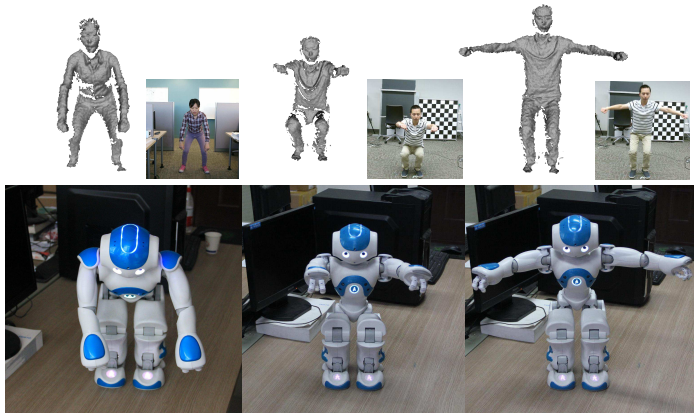
We show the skeletons computed from our approach and output by Kinect SDK. In order to show the results more clearly, we have colored the skeletons of the left and right part of the body in green and magenta respectively. In the first case, the human subject leans forward with one leg severely occluded. As demonstrated in Fig. 10a, the skeleton we obtained from Kinect SDK is obviously wrong for the leg, while we can still get reasonable results through our tracking system. For the second case as shown in Fig. 10b, when the human subject turns around, the Kinect SDK cannot distinguish the front and back of the human body, resulting in the flipping of left and right sides of the body. Instead, we are able to handle these situations, since instead of frame-by-frame tracking we have taken the depth sequence into consideration and prevented any abrupt pose changes in successive frames.

2) Quantitative evaluation on tracking: We have compared with some traditional state-of-the-art algorithms on human pose tracking. In this section, we show the quantitative evaluation of our algorithm on the SMMC-10 dataset [47] and Personalized Depth Tracker(PDT) dataset [48] with mean joint distance metrics. First, we give a brief introduction to the two datasets in Table I.

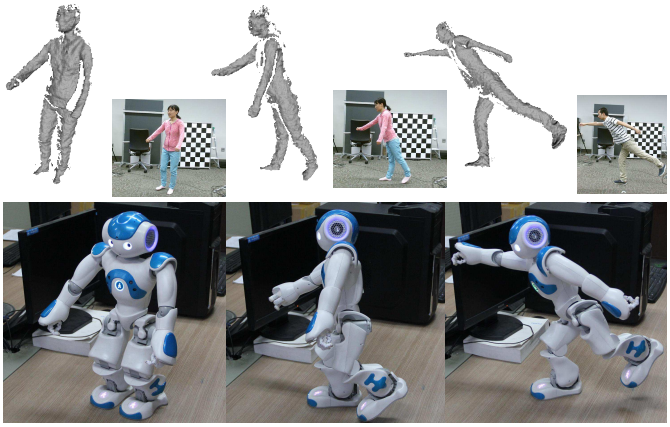
The datasets and their corresponding groundtruth are used for joint tracking evaluation, all the procedures and parameters are same as real data. For the two datasets, We select 11 joints



(a) Hands up



(b) Bow and Squat



(c) Roatation and Lean forward

(d) Kick

Fig. 8: Results in different kinds of postures. For each case, the first row shows the original mesh and image, and the second row shows the robot executing the pose computed with our proposed approach.

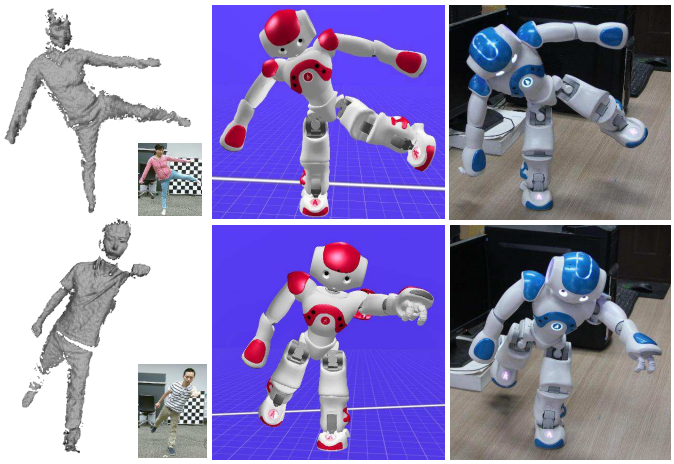


Fig. 9: The effect of stability check. The leftmost column shows the original mesh from the depth map. The middle column is simulation result without enforcing stability constraints. Finally, the rightmost column displays the retargeting result on real NAO robot with stability check executed.

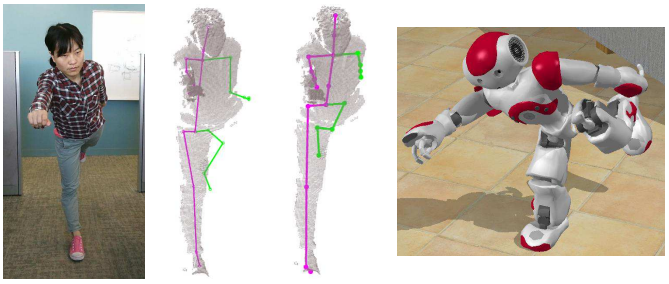
	SMMC-10	PDT
subject	one female	Three males, one female
Data	28 sequences 100 or 400 frames for each	4 sequences per subject, around 1500 frames per sequence
Posture	simple motion	jumping, sitting on floor, dancing, etc.
Groudtruth	markers	Joints + Transformations

TABLE I: Brief introduction to SMMC-10 and PDT dataset

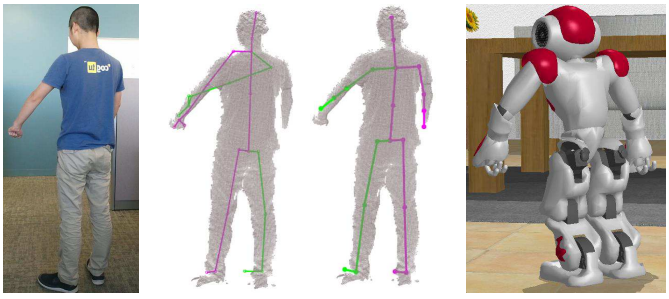
Algorithms	distance error
Ganapathi et al. [47]	73.10
Baak et al. [50]	62.15
Helten et al. [48]	61.15
Ye et al. [37]	34.02
ours	**32.67**

TABLE II: Comparison of joint distance errors (unit = millmeter) on SMMC-10 dataset

(head, waist, left/right shoulders, left/right elbows, left/right wrist, left/right knees and left/right ankles) to calculate the accuracy. As shown in the Table II and III, we can see that with our EM based tracking method which is more robust on handling outliers, we have achieved better results with small errors on the joint than other approaches.



(a) Lean forward



(b) Rotation

Fig. 10: Visual comparisons of our approach and the results from Kinect SDK on two cases. In each case, the leftmost is the captured image, the middle left is the pose captured from the Kinect SDK and the middle right one is result using our framework. The rightmost one is our simulation result for motion retargeting.

Algorithms	distance error
KinectSDK V1	87.03
Andriluka et al. [49]	71.04
Helten et al. [48]	61.32
Ye et al. [37]	46.42
ours	**42.23**

TABLE III: Comparison of joint distance errors (unit = millmeter) on PDT dataset

3) Compare with IK on motion retargeting: We also have done some qualitative comparisons with IK based motion retargeting approaches. We use Naoqi IK solver [41], which writes the generalized inverse kinematics problem as a quadratic program and solved using the qpOases library [43]. The simulation are worked on Webots [42] platform.

As shown in Fig. 11, the motion around the elbow is not quite similar to the human subject for the IK approach, since it has only considered the end-effectors position. In contrast, we are able to take both end positions and joint angle into account which makes our retargeting results more similar to the source human subject.

VI. Conclusion

In this paper, we have presented a novel generative approach for human-robot motion retargeting from which we have motion tracking and retargeting performed in an unified framework. A parametric *HUMROB* template has got proposed and built up by taking advantage of the SMPL model using a single RGBD image. The robot configurations are embedded with the template. Next, we have developed an energy function which

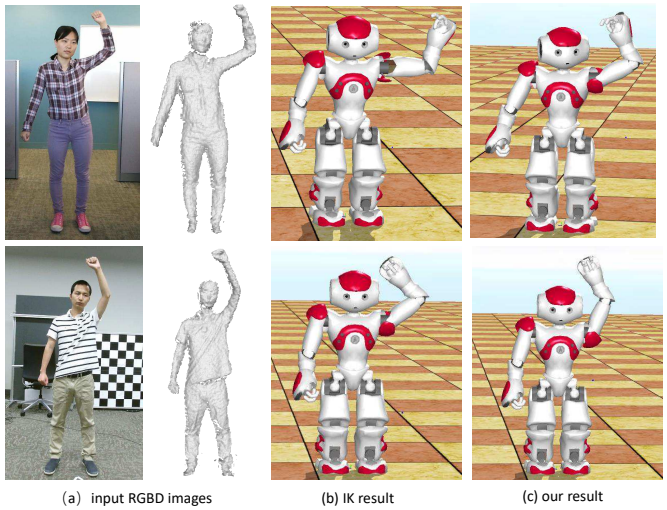


Fig. 11: Visual comparisons of our approach and the results from IK on two cases. In each case, the leftmost are the captured RGBD images, the middle ones are the results from the IK approach and the rightmost are the results from our framework.

penalizes the distance between deformed template model and the captured data from a RGBD sensor. In this way, we have considered the similarity of both joint angle and the end position simultaneously. The joint limitation of the target robots and stability constraints have also got enforced in the optimization. Our method has been verified on both simulated and real robots to demonstrate its ability of motion retargeting under a variety of posture.

Limitations: As a future work, we would like to incorporate interaction constraints into our system. Right now, for some self-interacting actions, such as touching the head by hand, our system may not be able to maintain the interaction because we have not explicitly enforced the interaction constraints in our formula.

References

[1] C. Stahl, D. Anastasiou, and T. Latour, "Social Telepresence Robots: The role of gesture for collaboration over a distance," in *Proceedings of the 11th PErvasive Technologies Related to Assistive Environments Conference,* Jun. 2018, pp. 409-414.

[2] D. Rakita, B. Mutlu, and M. Gleicher, "A motion retargeting method for effective mimicry-based teleoperation of robot arms," in *ACM/IEEE International Conference on Human-Robot Interaction(HRI),* Aug. 2017, pp. 361-370.

[3] M. Al Borno, L. Righetti, M. J. Black, S. L. Delp, E. Fiume, and J. Romero, "Robust Physics-based Motion Retargeting with Realistic Body Shapes," *Comput. Graph. Forum.*, vol. 37, no. 8, pp. 81-92, 2018.

[4] K. Ayusawa, and E. Yoshida, "Motion retargeting for humanoid robots based on simultaneous morphing parameter identification and motion optimization," *IEEE Transactions on Robotics*., vol. 33, no. 6, pp. 1343-1357, 2017.

[5] M. J. Gielniak, C. K. Liu, and A. L. Thomaz, "Generating human-like motion for robots," *Int. J. Robot. Res.*., vol. 32, no. 11, pp. 1275-1301, 2013.

[6] S. Guo, R. Southern, J. Chang, D. Greer, and J. Zhang, "Adaptive motion synthesis for virtual characters: a survey," *The Visual Computer*., vol. 31, no. 5, pp. 497-512, 2015.

[7] O. E. Ramos, N. Mansard, O. Stasse, C. Benazeth, S. Hak, and L. Saab, "Dancing Humanoid Robots: Systematic Use of OSID to Compute Dynamically Consistent Movements Following a Motion Capture Pattern," *IEEE Robot. Autom. Mag.*., vol. 22, no. 4, pp. 16-26, 2015.

[8] J. Koenemann, F. Burget, and M. Bennewitz, "Real-time imitation of human whole-body motions by humanoids," in *IEEE International Conference on Robotics and Automation (ICRA),* Jun. 2014, pp. 2806-2812.

[9] L. P. Poubel, S. Sakka, D. Cehajic, and D. Creusot, "Support changes during online human motion imitation by a humanoid robot using task specification," in *IEEE International Conference on Robotics and Automation (ICRA),* Jun. 2014, pp. 1782-1787.

[10] L. Penco, B. Clément, V. Modugno, Valerio, and et al., "Robust Real-time Whole-Body Motion Retargeting from Human to Humanoid," in *IEEE-RAS Int. Conference on Humanoid Robots (HUMANOIDS),* Nov. 2018, pp. 1-8.

[11] S. Wang, X. Zuo, R. Wang, Runxiao, F. Cheng and R. Yang, "A generative human-robot motion retargeting approach using a single depth sensor," in *IEEE International Conference on Robotics and Automation (ICRA),* Jun. 2017, pp. 5369-5376.

[12] E. Marchand, H. Uchiyama, and F. Spindler, "Pose estimation for augmented reality: a hands-on survey," *IEEE transactions on visualization and computer graphics*., vol. 22, no. 12, pp. 2633-2651, 2016.

[13] J. P. Bandera, and J. A. Rodríguez, L. Moltina-tanco, and A. Bandera, "A survey of vision-based architectures for robot learning by imitation," *Int. J. of Hum. Robot..*, vol. 9, no. 1, pp. 1-40, 2012.

[14] J. Martinez, R. Hossain, J. Romero, and J. J. Little. "A simple yet effective baseline for 3d human pose estimation," in *IEEE Conference on Computer Vision (ICCV),* Oct. 2017, pp. 2659-2668.

[15] M. Lee, and I. Cohen, "Proposal maps driven mcmc for estimating human body pose in static images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Jun. 2004, pp. 334.

[16] T. Tosun and R. Mead and R. Stengel, "A General Method for Kinematic Retargeting: Adapting Poses Between Humans and Robots," in *Proc. ASME 2014 Int. Mech. Eng. Congress Expo.,* Jun. 2013, pp. 213-221.

[17] Y. Ou, J. Hu, Z. Wang, Y. Fu, X. Wu, and X. Li, "A real-time human imitation system using kinect," *International Journal of Social Robotics*., vol. 7, no. 5, pp. 587-600, 2015.

[18] C. Plagemann and V. Ganapathi and D. Koller and S. Thrun, "Real-time identification and localization of body parts from depth images," in *IEEE International Conference on Robotics and Automation (ICRA),* May 2010, pp. 3108-3113.

[19] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Jun. 2011, pp. 1297-1304.

[20] M. Ye, and R. Yang, "Real-time Simultaneous Pose and Shape Estimation for Articulated Objects Using a Single Depth Camera," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Jun. 2014, pp. 2353-2360.

[21] Y. Yang and V. Ivan and S. Vijayakumar, "Real-time motion adaptation using relative distance space representation," in *IEEE International Conference on Advanced Robotics (ICAR),* Jul. 2015, pp. 21-27.

[22] Y. Seol, C. O'Sullivan, and J. Lee, "Creature features: online motion puppetry for non-human characters," in *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA),* Jul. 2013, pp. 213-221.

[23] K. Yamane, Y. Ariki, and J. Hodgins, "Animating non-humanoid characters with human motion data," in *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA),* Jul. 2010, pp. 169-178.

[24] H. Jang, B. Kwon, M. Yu, and J. Kim, Jongmin, "A Deep Learning Approach for Motion Retargeting," in *Proceedings of the 2018 ICRL workshops,* Apr. 2018.

[25] R. Villegas, J. Yang, D. Ceylan, and H. Lee, "Neural Kinematic Networks for Unsupervised Motion Retargetting," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Jun. 2018, pp. 8639-8648.

[26] Z. Liu, and A. Mucherino, and L. Hoyet, and F. Multon,, "Surface based motion retargeting by preserving spatial relationship," in *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games (MIG),* Nov. 2018, pp.1-11.

[27] T. Jin, M. Kim, and S. Lee, "Aura Mesh: Motion Retargeting to Preserve the Spatial Relationships between Skinned Characters," *Comput. Graph. Forum.*, vol. 37, no. 2, pp. 311-320, 2018.

[28] D. Sel, Y. Pinczon, N. Chaverou, and M. Rouillé, "Motion retargeting for crowd simulation," in *Proceedings of the 2015 Symposium on Digital Production,* Aug. 2015, pp. 9-14.

[29] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, Gerard and M. J. Black, "SMPL: A skinned multi-person linear model," *ACM Transactions on Graphics (TOG).*, vol. 34, no. 6, pp. 248, 2015.

[30] A. Myronenko, and X. Song, "Point set registration: Coherent point drift," *IEEE Trans. Pattern Anal. Mach. Intell..*, vol. 32, no. 12, pp. 2262-2275, 2010.

[31] 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*, pp. 1-38, 1977.

[32] R. M. Murray, Z. Li, and S. S. Sastry, "A Mathematical Introduction to Robotic Manipulation," CRC Press, Inc., Boca Raton, FL, USA, 1994.

[33] J. P. Lewis, M. Cordner, and N. Fong, "Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation," in *Proc. ACM SIGGRAPH,* Jul. 2000, pp. 165-172.

[34] S. Wang, X. Zuo, C. Du, R. Wang, Runxiao, J. Zheng and R. Yang, "Dynamic Non-Rigid Objects Reconstruction with a Single RGB-D Sensor," *Sensors*, vol. 18, no. 3, pp. 886, 2018.

[35] Z. Cao, G. Hidalgo, T. Simon, S. Wei, and Y. Sheikh, "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields," *arXiv preprint arXiv:1812.08008*, 2018.

[36] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image," in *European Conference on Computer Vision (ECCV),* Oct. 2016, pp. 561-578.

[37] M. Ye and Y. Shen and C. Du and Z. Pan and R. Yang, "Real-time Simultaneous Pose and Shape Estimation for Articulated Objects Using a Single Depth Camera," *IEEE Trans. Pattern Anal. Mach. Intell..*, vol. 38, no. 8, pp. 1517-1532, 2016.

[38] D. E. King, "Dlib-ml: A machine learning toolkit," *J. Mach. Learn. Res.*, vol. 10, no. Jul., pp. 1755-1758, 2009.

[39] D. Anguelov and P. Srinivasan and D. Koller and S. Thrun and J. Rodgers and J. Davis, "SCAPE: shape completion and animation of people," *ACM Transactions on Graphics (TOG).*, vol. 24, no. 3, pp. 408-416, 2005.

[40] Microsoft. Kinectsdk [Online]. Available: https://dev.windows.com/en-us/kinect, Accessed on: March. 2, 2019

[41] Softbank. NAO Inverse Kinematics [Online]. Available: http://doc.aldebaran.com/2-1/naoqi/motion/control-wholebody.html, Accessed on: March. 2, 2019

[42] Cyberbotics. Webots robot simulator [Online]. Available: https://github.com/omichel/webots, Accessed on: March. 2, 2019

[43] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit MPC," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 8, pp. 816-830, 2008.

[44] G. V. de Perre, and A. De Beir, H. Cao, and et al., "Reaching and pointing gestures calculated by a generic gesture system for social robots," *Robotics and Autonomous Systems*, vol. 83, pp. 32–43, 201.

[45] X. Peng, P. Abbeel, S. Levine, M. V. de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions on Graphics (TOG)*, vol. 37 , no. 4, pp. 143, 2018.

[46] Rajeswaran A, Kumar V, Gupta A, and et al. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations Robotics: Science and Systems Conference. arXiv preprint arXiv:1709.10087, 2018.

[47] V. Ganapathi, C. Plagemann, D. Koller, and et al, "Real time motion capture using a single time-of-flight camera,"in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Jun. 2010, pp. 755-762.

[48] T. Helten, A. Baak, G. Bharaj, M. M uller, H.-P. Seidel, and C. Theobalt. "Personalization and evaluation of a real-time depthbased full body tracker,". in *International Conference on 3D Vision(3DV)*, 2013, pp. 279-286.

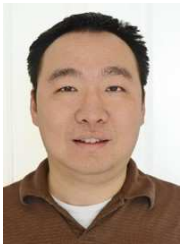
[49] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In CVPR, 2014.

[50] A. Baak, M. M uller, 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

Sen Wang received the B.E in 2011 from Northwestern Polytechnical University, Xi'an, China. He is currently pursuing his PhD degree in Northwestern Polytechnical University, Xi'an, China. During 03/2015 to 10/2016, he was a visiting PhD student at the University of Kentucky. His research interests inculde robotics and computer vision.

Xinxin Zuo received her B.E and M.E degree in 2011 and 2014 from Northwestern Polytechnical University, Xi'an, China. She is currently pursuing her PhD degree in the University of Kentucky. Her research interests include computer vision and graphics, especially on 3D reconstruction and Human modeling.

Runxiao Wang received his M.E. and PhD degree from Northwestern Polytecnical University in 1982 and 1998. Currently he is a full professor with school of Mechanical Engineering, Northwestern Polytechnical University. His research interests include robotics and management. He has published over 100 papers in the related research area.



Ruigang Yang (SM'13) is currently a full professor of Computer Science at the University of Kentucky. He received his PhD degree from University of North Carolina at Chapel Hill and his M.S. degree from Columbia University. His research interests span over computer graphics and computer vision, in particular in 3D reconstruction and 3D data analysis. He has published over 100 papers, which, according to Google Scholar, has received close to 10000 citations with an h-index of 48 (as of 2017). He has received a number of awards, including US NSF Career award in 2004 and the Dean's Research Award at the University of Kentucky in 2013. He is currently an associate editor of IEEE TPAMI.