Tensor Body: Real-time Reconstruction of the Human Body and Avatar Synthesis from RGB-D

Angelos Barmpoutis, *Member, IEEE*

***Abstract*—Real-time 3D reconstruction of the human body has many applications in anthropometry, telecommunications, gaming, fashion, and other areas of human-computer interaction. In this paper a novel framework is presented for reconstructing the 3D model of the human body from a sequence of RGB-D frames. The reconstruction is performed in real time while the human subject moves arbitrarily in front of the camera. The method employs a novel parameterization of cylindrical-type objects using Cartesian tensor and b-spline bases along the radial and longitudinal dimension respectively. The proposed model, dubbed tensor body, is fitted to the input data using a multi-step framework that involves segmentation of the different body regions, robust filtering of the data via a dynamic histogram, and energy-based optimization with positive-definite constraints. A Riemannian metric on the space of positive-definite tensor splines is analytically defined and employed in this framework. The efficacy of the presented methods is demonstrated in several real-data experiments using the Microsoft Kinect sensor.**

***Index Terms*—3D reconstruction, Avatar, Kinect, Tensor Basis, Positive-Definite constraints, B-Spline.**

I. Introduction

INFRARED depth cameras in conjunction with regular RGB video cameras have been widely used as low-cost peripheral devices for various applications related to virtual reality interaction using natural user interfaces. The information captured on a daily basis by these devices can also be used to extract useful information related to the tridimensional shape of the users' body, as well as track changes on its size, range of motion, and physical condition.

There are several examples in literature that present applications of RGB-D cameras [1]. A controller-free exploration of medical image data for avoiding the spreading of germs was proposed in [2]. A game-based rehabilitation system was presented in [3] using body tracking from RGB-D. Other applications include human detection [4], interactive video morphing [5], model-based 3d tracking of hand articulations [6], and real-time human pose recognition and tracking of body parts [7]. A detailed review of RGB-D applications that utilize Microsoft Kinect sensor is presented in [1].

Several of the aforementioned applications employ various well studied principles from 2D image-based computer vision in novel human computer interaction applications. It has been shown that many traditional computer vision problems can be solved more efficiently and/or accurately using RGB-D cameras. For example, there are several popular computer-vision approaches for reconstructing the 3D shape of a human face, namely shape-from-shading, shape-from-stereo, shape-from-video, and others [8], [9], [10]. However, a more efficient solution is offered in the framework presented in [11] by fitting a morphable face model to RGB-D data.

Similarly, human avatars can be reconstructed in 3D using image- or video-based approaches [12], [13], [14], [15], [16]. These methods perform various intermediate steps, such as image processing to label object pixels, calculating the volume intersection, and rendering the visual hull. However, several of these techniques require prior environmental setup and the avatars are reconstructed as non-articulated rigid objects, hence they cannot be re-rendered in new arbitrary postures.

The human bodies as articulated models have been recently studied in [17], [18]. Both techniques use RGB-D frames to reconstruct the body in 3D with [18] or without [17] an underlying parametric model of the human body; however both methods require long running times and hence are not suitable for real-time reconstruction. Real-time reconstruction of the 3D model of the human body is necessary in many applications such as gaming and teleconferencing. Furthermore, real-time measurements of the human body such as circumference and volume are useful in many medical [19], anthropological, or even fashion-related applications.

In this paper, a framework is presented for reconstructing the human body as an articulated generative 3D model that can be re-rendered in arbitrary novel postures by overcoming the aforementioned limitations of the existing techniques. The proposed method fits in real-time a novel parametric model to the data captured from a single RGB-D camera. One of the advantages of the proposed technique is that the human subjects can be reconstructed in 3D while they naturally move and interact with the system, without requiring from the users to stand in a particular posture.

The proposed parametric model employs the Cartesian tensor basis and b-spline basis, which are both well studied mathematical tools, and can be used for approximating smoothly varying fields of spherical functions [20]. The proposed body tensor model is an extension of the tensor spline framework that was used in other applications, such as for modeling diffusivity functions in MRI data [20] and bidirectional reflectance distribution functions [21]. In this paper, the proposed parameterization uses intrinsic positive-definite constraints in order to approximate cylindrical-type

A. Barmpoutis is with the Digital Worlds Institute, University of Florida, Gainesville, FL, 32611.
E-mail: angelos@digitalworlds.ufl.edu

Manuscript received October 16, 2012; revised April 10, 2013; accepted July 23, 2013.

A Java implementation of the presented algorithms along with the custom Java binding to the Microsoft Kinect SDK is available at http://www.digitalworlds.ufl.edu/angelos/lab/kinect

Patent pending.

3D objects with positive volume. This positive-definite tensor spline model is employed to approximate the arms, forearms, thighs, legs, and human torso using an energy-driven data fitting process. Several experimental results are presented that demonstrate the efficacy of the proposed framework showing a significant improvement compared to other existing techniques, specifically $\times 10^3$ improvement in running time for achieving results with similar fitting errors.

The contributions in this paper are four-fold: A) A novel framework for synthesizing avatars from RGB-D is presented with various intermediate steps that include body segmentation and dynamic robust data filtering. B) A novel parameterization of the human body, dubbed tensor body, is presented using tensor splines. C) Positive-definite constraints are imposed to the estimated tensor splines using a Riemannian metric defined on the space of positive-definite tensor splines and is also employed for interpolation/extrapolation between avatars. D) Notable improvement of 3 orders of magnitude (powers of 10) in running time is shown compared to other existing techniques.

II. Tensor Spline Framework

In this section a novel parameterization of cylindrical-type 3D shapes with positive volume is presented by employing Cartesian tensor basis with positive-definite constraints.

A. Tensors as spherical functions

There are several different known parameterizations of real-valued functions defined on the n-sphere (dubbed here *spherical functions*), $f(\mathbf{x}) : S_n \to \mathbb{R}$, where S_n denotes the space of the n-dimensional sphere that lies in the $n+1$ Euclidean space. Most of the parameterizations use a set of basis functions such as finite elements, spherical harmonics, or their Cartesian equivalent. The finite element bases have local support, which allows for local fitting to data samples, while the spherical harmonic or Cartesian tensor bases provide a global support, which allows for robust global fitting to data samples, and for this reason are employed in this work. The reader is referred to [22] for an in-depth presentation of Cartesian tensors and their use as a basis for approximating continuous real-valued spherical functions.

Spherical functions can be parameterized using a Cartesian tensor of degree d in the form of the following homogeneous polynomial:

$$T_d(\mathbf{x}) = \sum_{i_1+i_2+\cdots+i_{n+1}=d} c_{i_1,i_2,\cdots,i_{n+1}} x_1^{i_1} x_2^{i_2} \cdots x_{n+1}^{i_{n+1}} \quad (1)$$

where x_i is the i^{th} component of the $(n+1)$-dimensional unit vector $\mathbf{x} \in S_n$, and $c_{i_1,i_2,\cdots,i_{n+1}}$ are the tensor coefficients ($(n+d)!/n!d!$ in total), and the indices $i_1, i_2, \cdots, i_{n+1} \in \mathbb{N}_0$.

In the case of $n = 1$, Eq. 1 can be written as

$$T_d(\phi) = \sum_{i_1+i_2=d} c_{i_1,i_2} cos^{i_1}\phi sin^{i_2}\phi \quad (2)$$

by substituting x_1 and x_2 with $cos\phi$ and $sin\phi$ respectively, where ϕ is the angular parameter of S_1. The number of coefficients in Eq. 2 is $d+1$.

Let $\mathcal{T}_d^n$ denote the space of functions $f : S_n \to \mathbb{R}$ parameterized using tensors of degree d given by Eq. 1. It can be easily shown that $\mathcal{T}_d^n \subset \mathcal{T}_{d+2}^n \; \forall d \geq 0$, since $\exists \, T_{d+2} \in \mathcal{T}_{d+2}^n : T_{d+2}(\mathbf{x}) = \mathbf{x}\mathbf{x}^T T_d(\mathbf{x}) \; \forall$ given $T_d(\mathbf{x}) \in \mathcal{T}_d^n$. Based on the above, it can be easily shown that any spherical function can be approximated by parameterizing its symmetric and anti-symmetric component as the sum of an even and an odd degree tensor:

$$f_d(\mathbf{x}) = T_d(\mathbf{x}) + T_{d+1}(\mathbf{x}). \quad (3)$$

In the case of $n = 1$, the number of coefficients in Eq. 3 is $2d+3$.

B. Positive-definite tensors

In several applications there is the need to approximate non-negative quantities, such as distance, magnitude, and weight. If such quantities are given as a function of a unit vector, this function can be approximated by fitting the model in Eq. 3 to the data using positive-definite constraints [23], [22]. Let $\mathcal{T}_d^n \times \mathcal{T}_{d+1}^n$ denote the space of the functions given by Eq. 3. The part of the space $\mathcal{T}_d^n \times \mathcal{T}_{d+1}^n$ that corresponds to positive-definite functions is clearly a convex subspace, more precisely a hyper-cone, since any convex combination or positive scale of the elements of that subspace is also an element of the subspace. Therefore, any positive-definite function in $\mathcal{T}_d^n \times \mathcal{T}_{d+1}^n$ can be approximated by a positive-weighted sum of the elements of the boundary of the hyper-cone. Given a dense linearly independent sampling of the boundary, the non-negative elements of $\mathcal{T}_d^n \times \mathcal{T}_{d+1}^n$ can be approximated by

$$f_d(\mathbf{x}) = \sum_{i=1}^{m} w_i f_{d,i}^*(\mathbf{x}) \quad (4)$$

where $f_{d,i}^*(\mathbf{x})$ is a set of linearly independent elements of the boundary of the space of positive-definite functions in $\mathcal{T}_d^n \times \mathcal{T}_{d+1}^n$, and $w_i > 0 \; \forall \; i \in [1, m]$. The accuracy of the approximation of the hyper-cone space $\mathcal{T}_d^n \times \mathcal{T}_{d+1}^n$ by the hyper-polygon in Eq. 4 can be expressed as a function of m and d [22]. The sum $\sum_{i=1}^m w_i$ is positive, but not necessarily equal to one, since w_i also captures the scale of the modeled function $f_d(\mathbf{x})$, which is factored out of the boundary elements $f_{d,i}^*(\mathbf{x})$ due to their linear independence.

In our application we used the set of positive semi-definite functions in $\mathcal{T}_d^1 \times \mathcal{T}_{d+1}^1$ given by

$$f_{d,i}^*(\mathbf{x}) = \frac{2\pi}{m \int_0^{2\pi} cos^d \omega d\omega} [y_i(\mathbf{x})^d + y_i(\mathbf{x})^{d+1}] \quad (5)$$

where $y_i(\mathbf{x}) = x_1 cos\theta_i + x_2 \sin\theta_i$, $\theta_i = 2\pi i/m$, and $\mathbf{x} \in S_1$. Note that Eq. 5 is non-negative for even d, and $\sum_{i=1}^m f_{d,i}^*(\mathbf{x}) = 1 \; \forall \mathbf{x} \in S_1$. Besides these useful properties, this particular function behaves as a sampling kernel since $\lim_{d\to\infty} f_{2d,i}^*(\mathbf{x}) = \delta(x_1 - \cos\theta_i)\delta(x_2 - \sin\theta_i)$, where δ is the Dirac delta function (see Fig. 1). This natural property of sampling kernels associates the sampling frequency with the degree d of the tensor in our parameterization (i.e. the higher the degree of the tensor, the higher the frequencies that can be captured by this model).

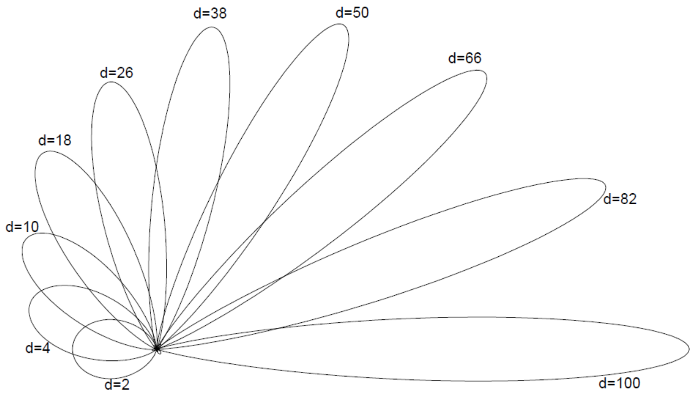

Fig. 1. Plots of Eq. 5 for various degrees d, and orientations defined by $i = 1 \cdots 10$, $m = 20$.

In the case of $d = 2$, the 7 coefficients of $f^*_{2,i}(\mathbf{x})$ are $cos^2\theta_i/m$, $sin^2\theta_i/m$, $2cos\theta_i sin\theta_i/m$, $cos^3\theta_i/m$, $sin^3\theta_i/m$, $3cos^2\theta_i sin\theta_i/m$, $3cos\theta_i sin^2\theta_i/m$ and correspond to the monomials x_1^2, x_2^2, x_1x_2, x_1^3, x_2^3, $x_1^2x_2$, and $x_1x_2^2$ respectively. Similarly, the coefficients of $f_d(\mathbf{x})$ in Eq. 4 are given by the weighted sum of the coefficients in $f^*_{2,i}(\mathbf{x})$. For example, the coefficient $\sum_{i=1}^m w_i cos^2\theta_i/m$ corresponds to the monomial x_1^2.

The degrees of freedom of the model in Eq. 4 are given by the number of the tensor coefficients ($2d+3$ in Eq. 5) and not by the number m of unknown weights w_i. This can be easily shown by rewriting Eq. 4 as $\mathbf{v}(\mathbf{x})^T\mathbf{F}\mathbf{w}$, where $\mathbf{v}(\mathbf{x})$ is a vector with all the monomials of $\mathbf{x}$ in $f^*_{d,i}(\mathbf{x})$, $\mathbf{F}$ is a 2D matrix with all the polynomial coefficients in $f^*_{d,i}(\mathbf{x})$, and $\mathbf{w}$ is an m-dimensional vector that consists of the values w_i. Similarly, the size of $\mathbf{F}$ in Eq. 5 is $(2d+3) \times m$, and its rank (that corresponds to the degrees of freedom in Eq. 4) is at most $2d+3$, assuming that $m > 2d+3$, since m was defined as the size of a dense set of linearly independent elements on the boundary of the space of positive-definite functions in $\mathcal{T}^n_d \times \mathcal{T}^n_{d+1}$.

C. Positive-Definite Tensor Splines

A continuous and smoothly varying 1-dimensional field of positive-definite spherical functions in the form of Eq. 4 can be modeled by using the B-spline basis [24], [20], [21] of degree k, denoted by $N_{j,k+1}(s)$, where j corresponds to a discretization s_j (commonly known as knots) of the domain s as follows:

$$f_d(\mathbf{x}, s) = \sum_{i=1}^{m}\sum_{j=0}^{n} w_{i,j} N_{j,k+1}(s) f^*_{d,i}(\mathbf{x}). \quad (6)$$

In Eq. 6 the weights $w_{i,j}$ are the so-called control points, which are blended across j using the B-spline basis. Furthermore, the positive-definite tensors given by $\sum_{i=1}^m w_{i,j} f^*_{d,i}(\mathbf{x})$ $\forall j \in [0, n]$ play the role of control tensors along an 1-dimensional field.

The mathematical model in Eq. 6 can be used for parameterizing cylindrical type of objects with one radial and one longitudinal dimension. The 3D coordinates of the points on the parameterized surface are given by $[x_1 f_d(\mathbf{x}, s), x_2 f_d(\mathbf{x}, s),$ $s]$, where the third dimension corresponds to the longitudinal axis s, and $\mathbf{x} = [cos\phi,\ sin\phi]$. A typical symmetric cylinder of radius ρ and height h can be parameterized using a uniform tensor spline by setting $w_{i,j} = \rho\ \forall i, j$ and $s_{j+1} - s_j = h/(n+1-k)\ \forall j$ in Eq. 6.

TABLE I
LIST OF TENSOR COEFFICIENTS IN EQ. 7 FOR $d = 2$

$i_1 + i_2$	$c_{i_1,i_2,j}$
2	$c_{2,0,j} = \sum_{i=1}^m w_{i,j} cos^2(2\pi i/m)/m$
2	$c_{0,2,j} = \sum_{i=1}^m w_{i,j} sin^2(2\pi i/m)/m$
2	$c_{1,1,j} = \sum_{i=1}^m w_{i,j} 2cos(2\pi i/m)sin(2\pi i/m)/m$
3	$c_{3,0,j} = \sum_{i=1}^m w_{i,j} cos^3(2\pi i/m)/m$
3	$c_{0,3,j} = \sum_{i=1}^m w_{i,j} sin^3(2\pi i/m)/m$
3	$c_{2,1,j} = \sum_{i=1}^m w_{i,j} 3cos^2(2\pi i/m)sin(2\pi i/m)/m$
3	$c_{1,2,j} = \sum_{i=1}^m w_{i,j} 3cos(2\pi i/m)sin^2(2\pi i/m)/m$

By substituting Eqs. 2 and 5 into Eq. 6 the following positive-definite tensor spline model can be derived for S_1:

$$f_d(\phi, s) = \sum_{j=0}^{n}\sum_{i_1,i_2} c_{i_1,i_2,j} N_{j,k+1}(s) cos^{i_1}\phi sin^{i_2}\phi \quad (7)$$

where the second sum is over all pairs of indices (i_1, i_2) : $i_1 + i_2 \in \{d, d+1\}$, $i_1, i_2 \in \mathbb{N}_0$. In the case of $d = 2$, there are 7 tensor coefficients $c_{i_1,i_2,j}$, which are listed in table I.

Eq. 7 is positive-definite $f_d(\phi, s) > 0\ \forall\phi \in [0, 2\pi]$, and $s \in [s_0, s_{n+1-k}]$. Note that there are no additional constraints imposed on the range of the values of the tensor coefficients $c_{i_1,i_2,j}$, besides the fact that $w_{i,j} > 0$. The degrees of freedom of the models in Eqs. 6 and 7 are given by the number of tensor coefficients $c_{i_1,i_2,j}$. In the particular case of Eq. 7 the number of coefficients is $(2d+3) \times (n+1)$, i.e. it depends linearly on the degree of the tensor, as well as the number of control points of the B-spline.

D. Tensor Spline Distance Measure

Let $a_d(\mathbf{x}, s)$ and $b_d(\mathbf{x}, s)$ be two positive-definite tensor splines (defined as in Eq. 7), with coefficients $a_{i_1,i_2,j}$ and $b_{i_1,i_2,j}$ respectively. There are several possible metrics that can be used to define the distance between a_d and b_d, such as the Euclidean distance $dist(a_d, b_d) = \sqrt{\sum_{j=0}^n \sum_{i_1,i_2}(a_{i_1,i_2,j} - b_{i_1,i_2,j})^2}$, or the L_2 norm given by $dist(a_d, b_d) = \sqrt{\sum_{j=0}^n \int_{S_1}(a_d(\mathbf{x}, s) - b_d(\mathbf{x}, s))^2 d\mathbf{x}}$. In the latter case, the integrals can be analytically computed as powers of trigonometric functions by parameterizing the vectors in S_1 as $\mathbf{x} = [cos\phi\ sin\phi]$. Such metrics are useful not only for computing the distances between tensor splines, but also for atlas construction, as well as for interpolation and extrapolation, and for defining energy functionals in optimization methods.

In the case of the two aforementioned metrics, the tensor splines $a_d(\mathbf{x}, s)$ and $b_d(\mathbf{x}, s)$ can be treated as elements of a Euclidean space, and be represented in this space by vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{(2d+3)\times(n+1)}$ that consist of the coefficients $a_{i_1,i_2,j}$ and $b_{i_1,i_2,j}$ respectively. However, tensor splines that are not necessarily positive-definite can also be mapped to the same Euclidean space, hence there is no guarantee that the result of extrapolation given by $\mathbf{a} + \lambda(\mathbf{b} - \mathbf{a}) : \lambda \in (-\infty, 0) \cup (1, \infty)$ will correspond to a positive-definite tensor spline. This may produce shapes of negative volume that are unnatural in many applications, including the one presented in Sec. III for modeling the 3D shape of human body parts. To overcome this problem the positive-definite parameterization that was introduced in Sec. II-B will be employed.

E. Riemannian metric

Let the coefficients $a_{i_1,i_2,j}$ and $b_{i_1,i_2,j}$ be parameterized, as in table I, using the positive weights $w^a_{i,j}$ and $w^b_{i,j}$ respectively (the table lists the formulas for the 2^{nd} and 3^{rd} degree coefficients only, but it can be easily extended to higher degrees by expanding the terms in Eq. 5). The corresponding tensor splines can be treated as elements of the $\mathbb{R}^{m\times(n+1)}_{*+}$, and be represented in this space by stacking the weights $w^a_{i,j}$ and $w^b_{i,j}$ in the form of vectors $\mathbf{w}_a$, $\mathbf{w}_b \in \mathbb{R}^{m\times(n+1)}_{*+}$, where $\mathbb{R}_{*+}$ denotes the space of positive real numbers.

The distance measure in this space can be defined using the Riemannian metric on $\mathbb{R}_{*+}$ that utilizes its tangent space (defined by the log mapping [25], [26], [27]): $dist(a_d, b_d) = ||Log(\mathbf{w}_a) - Log(\mathbf{w}_b)||$, where the function $Log()$ is the natural logarithm applied individually to every element of the input vector. The same Riemannian metric can be used for interpolation/extrapolation using the exp projection from the tangent space to $\mathbb{R}_{*+}$ as follows: $Exp(Log(\mathbf{w}_a) + \lambda(Log(\mathbf{w}_b) - Log(\mathbf{w}_a)))$, where the function $Exp()$ is the natural exponential applied individually to every element of the input vector. The computed vectors are guaranteed to correspond to positive-definite tensor splines $\forall\lambda \in \mathbb{R}$. The Riemannian metric assigns infinite distance between positive-definite tensor splines and semi-definite tensor splines, hence the boundary of the space of positive-definite tensor splines can be approached by extrapolating towards the boundary using $\lim_{\lambda\to\infty}$. Examples of interpolation and extrapolation of positive-definite tensor splines using the Riemannian metric are shown in Fig. 7 in the *Experimental Results* section.

III. AVATARS AS TENSOR BODIES

Most parts of the human body can be modeled as a set of positive-definite tensor splines that approximate the shape of the arms, forearms, legs, thighs, and torso. These segments of the human body can be approximated by rigid tridimensional models, since there are no large deformations in their structure during a natural human motion, unlike the hands, the head (for 3D face reconstruction from RGB-D see [11]), and the elbows and knees, which can be easily rendered by interpolating the adjacent tensors. The coefficient vector $\mathbf{w}$ of each tensor spline can be estimated from real data captured by RGB-D cameras. In this section, a novel method is presented for real-time human avatar synthesis by fitting a tensor body model, i.e. a set of positive-definite tensor-splines, to point-sets collected from a sequence of RGB-D frames. The proposed framework consists of several steps depicted in Fig. 2.

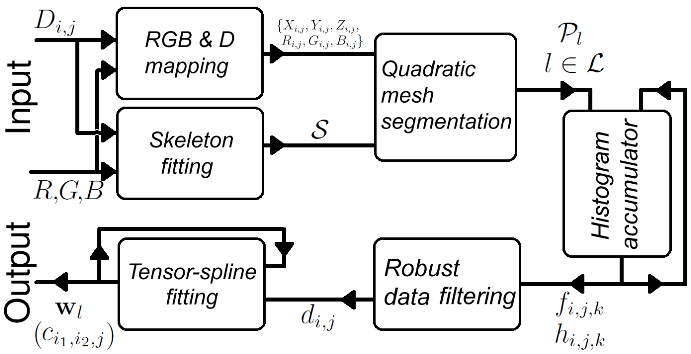

Fig. 2. Flow chart of the proposed framework for avatar reconstruction from RGB-D frames.

A. RGB-D data acquisition and skeleton fitting

Depth cameras generate sequences of discrete depth frames in the form of 2D arrays $D_{i,j}$, which can be equivalently expressed as quadratic meshes given by $X_{i,j} = (i - i_c)D_{i,j}c_d^{-1}$, $Y_{i,j} = (j - j_c)D_{i,j}c_d^{-1}$, and $Z_{i,j} = D_{i,j}$, where i_c, j_c denote the coordinates of the central pixel in the depth frame, and c_d is the focal length of the depth camera.

The video frames captured by an RGB camera can be associated with the 3D quadratic meshes by using a UV texture mapping given by the coordinates $U_{i,j} = X'_{i,j}{Z'_{i,j}}^{-1}c_v$, $V_{i,j} = Y'_{i,j}{Z'_{i,j}}^{-1}c_v$, where the coordinates of the vector $[X'\ Y'\ Z']^T$ are related to $[X\ Y\ Z]^T$ via a known rigid transformation (rotation and translation), and c_v is the focal length of the video camera [8]. The aforementioned transformation corresponds to the mapping between the locations of the focal points and orientations of the two cameras (RGB and D).

Each frame of the RGB-D sequence can be considered a set of arrays $\{X_{i,j}, Y_{i,j}, Z_{i,j}, R_{i,j}, G_{i,j}, B_{i,j}\}$, where R, G, B correspond to the red, green, and blue color channels of the video frame at the image coordinates $U_{i,j}, V_{i,j}$. This sequence of data frames can be used to detect the presence of a particular skeletal geometry, such as human skeletal geometry, and fit to each frame a skeletal model that consists of the following set of parameters:

$$\mathcal{S} = \{\mathbf{a}_l \in \mathbb{R}^3, \mathbf{b}_l \in \mathbb{R}^3, \mathbf{R}_l \in SO(3) : l \in \mathcal{L}\} \tag{8}$$

where $\mathcal{L}$ is a set of indices of line segments defined by the end-points $\mathbf{a}_l$ and $\mathbf{b}_l$, and its orientation in the 3D space is given by the rotation matrix $\mathbf{R}_l$. There are several algorithms that compute $\mathcal{S}$ from RGB-D, or just D, such as those implemented in the Microsoft Kinect SDK [28], in OpenNI library [29] (see detailed discussions in [30], [31] and comparison of these two libraries in [1]), and others [4], [7], any of which could be used as a module in the proposed framework (Fig. 2).

B. RGB-D Segmentation

The parameters in the skeletal model $\mathcal{S}$ can be used in order to segment the quadratic mesh that corresponds to a frame of the RGB-D sequence into different body regions. For every vertex $\mathbf{p} = [X_{i,j}\ Y_{i,j}\ D_{i,j}]^T$ in the quadratic mesh we compute the index l of the closest line segment in the skeletal

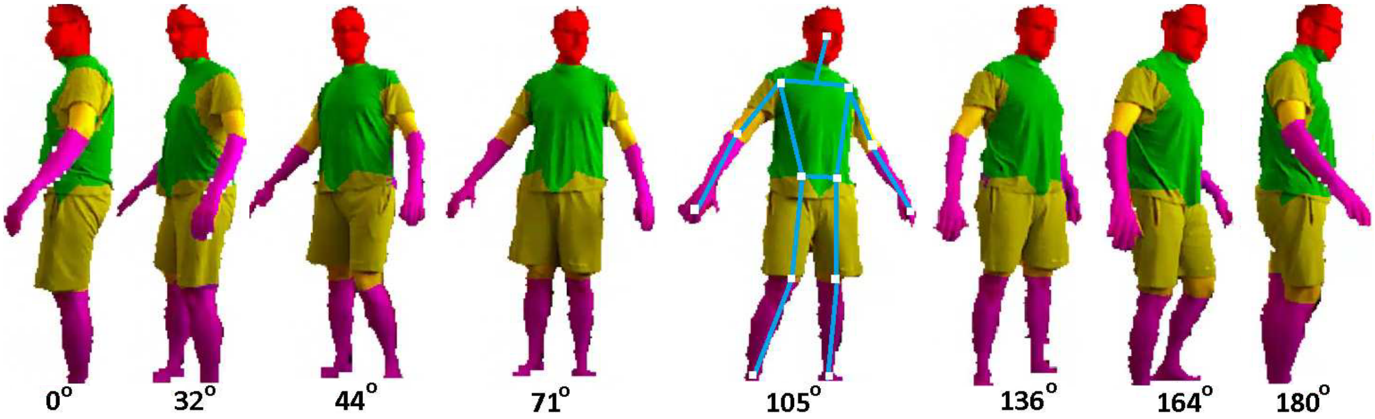

Fig. 3. Examples of the quadratic mesh segmentation results obtained from different RGB-D frames depicting various orientations of the body. The fitted skeleton is shown on the fifth plate.

model as follows:

$$l(\mathbf{p}) = argmin_{l\in\mathcal{L}}||\mathbf{a}_l + s_l(\mathbf{p})(\mathbf{b}_l - \mathbf{a}_l) - \mathbf{p}|| \quad (9)$$

where $\mathbf{a}_l, \mathbf{b}_l \in \mathbb{R}^3$ are vertices/joints that define a particular line segment in the skeletal model (Eq. 8), and $s_l(\mathbf{p})$ is the projection of $\mathbf{p}$ onto the l^{th} line segment given by:

$$s_l(\mathbf{p}) = max\{min\{\frac{(\mathbf{b}_l - \mathbf{a}_l)^T(\mathbf{p} - \mathbf{a}_l)}{||\mathbf{b}_l - \mathbf{a}_l||^2}, 1\}, 0\} \quad (10)$$

The max and min functions in Eq. 10 guarantee that, if the projection falls outside the line segment, the distance given as the argument of $argmin$ in Eq. 9 will be equal to the Euclidean distance between $\mathbf{p}$ and the closest end-point of the line segment (i.e $min\{||\mathbf{a}_l - \mathbf{p}||, ||\mathbf{b}_l - \mathbf{p}||\}$). Using Eq. 9 every vertex $\mathbf{p}$ in the quadratic mesh is assigned to the closest body segment. This process segments the quadratic mesh into several body regions and is performed for every frame of the RGB-D sequence. The points of the deformable areas around the elbows and knees, whose projections fall outside the line segment will be intentionally mapped to the boundary of the closest body part, and consequently will be ignored by the robust data fitting algorithm (see Sec. III-D). This useful property of Eq. 10 guarantees that the deformable areas around joints will not be explicitly reconstructed by the proposed tensor body parameterization as it was discussed in the beginning of Sec. III. Instead, elbows and knees are rendered by interpolating the adjacent tensors in the tensor body model (Fig. 10). Note that the points that do not belong to the depicted human subject can be easily thresholded across $Z_{i,j}$, since the background objects usually have larger $D_{i,j}$ values. This is an implicit assumption of many skeletal fitting algorithms including the one employed in our experiments (provided by Microsoft Kinect SDK).

The points that belong to a particular body region form the point-set $\mathcal{P}_l = \{\mathbf{p} \in \mathbb{R}^3 : l(\mathbf{p}) = l, 0 < s_{l(\mathbf{p})}(\mathbf{p}) < 1\}$, which will be used as our data source in the positive-definite tensor spline fitting algorithm described in the next sections. Results from the quadratic mesh segmentation are shown in Fig. 3 and are discussed in detail in Sec. IV.

C. Tensor Spline Estimation

In order to fit a positive-definite tensor spline (Eq. 6) to a pointset $\mathcal{P}_l$ that consists of points on the surface of the l^{th} body region, we first need to map each point in $\mathcal{P}_l$ to the domain of the function in Eq. 6. In our particular application, the domain is $S_1 \times \mathbb{R}$ and corresponds to the relative orientation and location of each point with respect to the central axis of the tensor spline.

Every point $\mathbf{p} \in \mathcal{P}_l$ can be uniquely mapped to $\mathbb{R}^2$ (i.e. the 2D plane of the unit circle S_1) by

$$\mathbf{x}_p = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{R}_l^{-1}(\mathbf{p} - \frac{\mathbf{a}_l + \mathbf{b}_l}{2}) \quad (11)$$

where $\mathbf{a}_l$, $\mathbf{b}_l$, and $\mathbf{R}_l$ are the parameters of the l^{th} segment of the skeleton modeled by Eq. 8. The role of the matrix on the left is to project the result to a 2D plane that is perpendicular to the central axis of the tensor spline. Without loss of generality, the central axis is assumed here to be parallel to the y-axis of the Cartesian space hence the first (x) and the third (z) components of the rotated vector are used as the elements of $\mathbf{x}_p$.

The positive-definite tensor spline model (Eq. 6) can be fitted to the magnitude $||\mathbf{x}_p||$ by minimizing the following energy function with respect to the coefficient vector $\mathbf{w}_l$:

$$E(\mathbf{w}_l) = \sum_{p\in\mathcal{P}_l} (f_l(\mathbf{x}_p/||\mathbf{x}_p||, s_{l(\mathbf{p})}(\mathbf{p})) - ||\mathbf{x}_p||)^2. \quad (12)$$

The data value $||\mathbf{x}_p||$ in Eq. 12 corresponds to the unit vector $\mathbf{x}_p/||\mathbf{x}_p||$ in the angular domain of the tensor spline model and the point $s_{l(\mathbf{p})}(\mathbf{p})$ along the longitudinal dimension. The unknown vector $\mathbf{w}_l$ can be estimated by any gradient-based optimization method [32] using the analytically computed gradients of Eq. 12. Additionally, positive-definite constraints can be applied to the elements of $\mathbf{w}_l$ by updating their values using gradients computed in the Riemannian space discussed in Sec. II-E.

Finally, the fitting process can be easily extended to accommodate multiple point-sets $\mathcal{P}_l$ that correspond to several RGB-D frames.

D. Robust data fitting

The least-squares fitting process described in Sec. III-C performs averaging over the data values $||\mathbf{x}_p||$ that correspond to the same angular and longitudinal coordinates $(\mathbf{x}, s)$ of the tensor spline domain in Eq. 6. If the corresponding data values vary across multiple frames due to misfit of the skeletal model, or due to deformations in the areas around joints, then the result of the least-square fit is equivalent to the result obtained by fitting the tensor spline model to the mean of the corresponding data values. The average value, or $L2$-norm statistical quantities in general, are significantly affected by the presence of outliers in the data, causing in our case erroneous tensor spline estimates.

This problem can be solved by introducing a robust energy function based on the distribution of the data values computed in the form of a histogram as follows:

$$h(f,\mathbf{x},s;\mathcal{P}) = \sum_{p\in\mathcal{P}} N(f;||\mathbf{x}_p||,\sigma_f^2)N(s;s_{l(\mathbf{p})}(\mathbf{p}),\sigma_s^2)V(\mathbf{x};\frac{\mathbf{x}_p}{||\mathbf{x}_p||},\kappa)$$

where $f \in \mathbb{R}$, $\mathbf{x} \in S_1$, $s \in \mathbb{R}$, and the functions $N()$ and $V()$ denote the Normal and von Mises probability density functions respectively. The parameters σ_f^2, σ_s^2, and κ are the variances and concentration of the probability functions.

For a given pair $(\mathbf{x}, s)$ the histogram $h(f,\mathbf{x},s;\mathcal{P})$ shows the distribution of the data values $||\mathbf{x}_p||$ in the space of real numbers, parameterized here by f. The peak of the histogram corresponds to the most dominant data value for a given $(\mathbf{x}, s)$ and it is robust to outliers. The robust data estimate is given by

$$d(\mathbf{x},s) = argmax_{f\in\mathbb{R}} h(f,\mathbf{x},s;\mathcal{P}), \tag{13}$$

and can be used for robust positive-definite tensor fitting in the following energy function

$$E(\mathbf{w}_l) = \int_{S_1}\int_0^1 (f_l(\mathbf{x},s) - d(\mathbf{x},s))^2 ds d\mathbf{x}. \tag{14}$$

The integrals in Eq. 14 are over the unit circle S_1 and the $[0,1]$ interval of the longitudinal axis of the tensor spline. Note that $s=0$ and $s=1$ correspond to two 2D sections of the tensor spline that are perpendicular to the line segment $(\mathbf{a}_l, \mathbf{b}_l)$ and pass through $\mathbf{a}_l$ and $\mathbf{b}_l$ respectively. The energy function in Eq. 14 can be optimized with respect to the unknown vector $\mathbf{w}_l$ using any gradient-based method.

E. Implementation details

For real-time ($\sim$ 25 frames/second) 3D body reconstruction, the histogram $h(f,\mathbf{x},s;\mathcal{P})$ discussed in Sec. III-D can be implemented by discretizing the domains of f, $\mathbf{x}$, and s. The unit circle can be divided into M sections represented by $\mathbf{x}_i = [cos(2\pi i/M)\ sin(2\pi i/M)]$, $i = 1\cdots M$ and the longitudinal axis can be similarly divided into N line segments represented by $s_j = (j-1)/(N-1)$, $j = 1\cdots N$. For every new data pair $(\mathbf{x}_p/||\mathbf{x}_p||, s_{l(\mathbf{p})}(\mathbf{p}))$ the closest bin $(\mathbf{x}_i, s_j)$ in the discretized histogram will be used.

The domain of f is dynamically discretized in the form of an on-line K-means clustering algorithm. For each of the K clusters the mean value of the cluster f_k is stored, as well as the number of data points assigned to this cluster h_k, $k = 1\cdots K$, without explicitly storing the individual data points. For every new data value $||\mathbf{x}_p||$ in the bin $(\mathbf{x}_i, s_j)$, the closest cluster is found (i.e. $argmin_{k=1\cdots K}|f_{i,j,k} - ||\mathbf{x}_p|||$), and if the distance from this cluster is smaller than σ_f^2, the cluster is properly updated (i.e. $f_{i,j,k} \leftarrow (f_{i,j,k}h_{i,j,k}+||\mathbf{x}_p||)/(h_{i,j,k}+1)$ and $h_{i,j,k} \leftarrow h_{i,j,k}+1$). Otherwise, the cluster with the smaller population is found (i.e. $argmin_{k=1\cdots K}h_{i,j,k}$) and is updated as follows: $f_{i,j,k} \leftarrow ||\mathbf{x}_p||$, and $h_{i,j,k} \leftarrow 1$.

The discretized version of Eq. 13 is given by

$$d_{i,j} = f_{i,j,argmax_{k=1:K}h_{i,j,k}} \tag{15}$$

and can be used for robust positive-definite tensor fitting in the following energy function

$$E(\mathbf{w}_l) = \sum_{i=1}^{M}\sum_{j=1}^{N}(f_l(\mathbf{x}_i,s_j) - d_{i,j})^2. \tag{16}$$

In our experiments we used $N=64$, $M=64$, $K=21$, and $\sigma_f^2 = 10^{-2}$. Note that the histogram in Eq. 15 does not use a point-set $\mathcal{P}$ as one of its arguments, because the histogram $h_{i,j,k}$ is updated on-line by one data point at a time, in contrast to Eq. 13. Finally, Eq. 16 is a polynomial and its derivatives with respect to $\mathbf{w}_l$ can be easily computed analytically.

After estimating the coefficient vectors $\mathbf{w}_l$ $\forall l \in \mathcal{L}$, the human avatar can be rendered in any arbitrary posture given in the form of a skeleton structure $\mathcal{S}$. For the purpose of rendering, each tensor-spline model is scaled by the magnitude of $||\mathbf{a}_l - \mathbf{b}_l||$ along the longitudinal axis, its center is translated to the point $(\mathbf{a}_l + \mathbf{b}_l)/2$ and rotated by $\mathbf{R}_l$. In the next section, the proposed method is demonstrated through several experiments using real RGB-D datasets.

Finally, the areas around the knees and elbows are rendered by smoothly interpolating between the two boundary tensors of the adjacent body parts using the Riemannian framework discussed in Sec. II-E. Figure 10 shows an example of a fitted tensor body model with and without filling the gap between the reconstructed tensor-spline segments of the legs.

IV. Experimental Results

The results presented in this section were obtained by applying the proposed framework to real-time data acquired using the PrimeSense$^{\text{TM}}$depth sensor as well as the video camera of Microsoft Kinect$^{\text{TM}}$device. The device was connected (via a USB 2.0 port) to a 64-bit computer with Intel Core i5 (quad core) CPU at 2.53GHz and 4GB RAM. The resolution of the depth camera was 320×240 pixels with a viewing range from 0.8m to 4.0m and horizontal field-of-view angle (FoV) angle of 57^o. The resolution of the video camera was 640×480 pixels with horizontal FoV of 62^o. The proposed framework was implemented solely in Java using custom bindings to OpenGL and Kinect SDK libraries, and the implementation is available at http://www.digitalworlds.ufl.edu/angelos/lab/kinect.

In every iteration of the proposed framework cycle (illustrated in Fig. 2) the most recent pair of frames is used as input data. The data are converted to a colored quadratic mesh $\{X,Y,Z,R,G,B\}_{i,j}$, which is then segmented into several

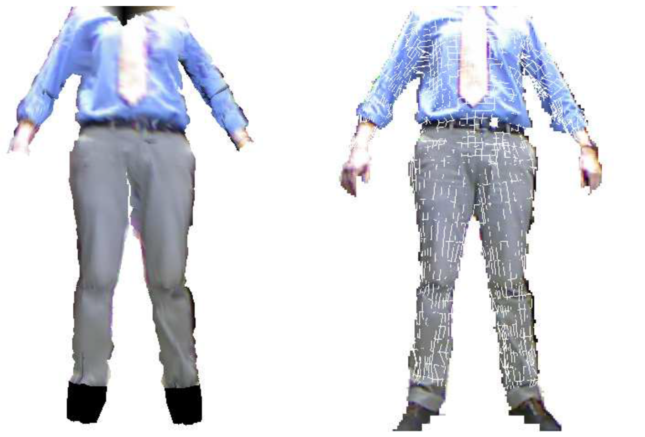

Fig. 4. Left: An intermediate state of the 3D reconstructed model before convergence. Right: The rectangular grid made of the current peaks of the data histograms superimposed on the current input frame in 3D.

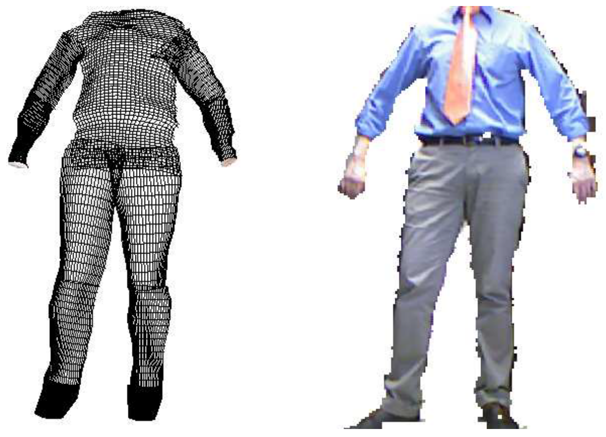

Fig. 5. Example of an estimated tensor body. The fitted tensor-splines are shown as quadratic meshes on the left. An image of the corresponding human subject is shown on the right.

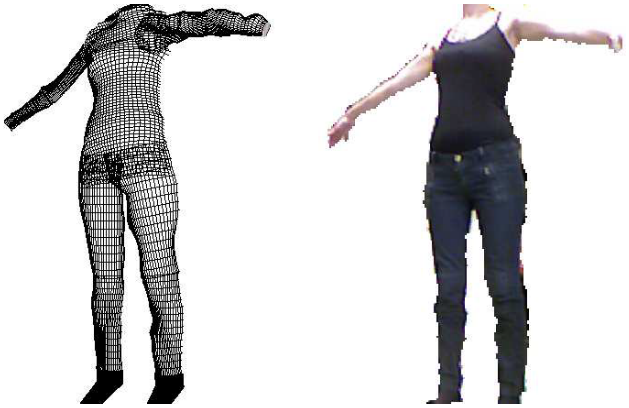

Fig. 6. Another example of a tensor body computed from a female human subject.

body regions using the parameters of the skeletal model $\mathcal{S}$ computed from the input data. In our implementation we used a skeletal model with 13 joints connected via 13 line segments ($\mathcal{L} = 1 \cdots 13$ in Eq. 8) shown on the fifth plate of Fig. 3. Each line segment corresponds to a different body region with the only exception of the torso, which is made out of 4 line segments. The proposed method divides the data into 11 point-sets $\mathcal{P}_l$ in total (background, head, torso, 2 arms, 2 forearms, 2 thighs, and 2 legs) as discussed in Sec. III-B.

Figure 3 shows the obtained quadratic mesh segments in different colors. Each plate shows the results produced in real time from various frames during a natural body motion corresponding to body orientations in $[0^o - 180^o]$. The presented results show that even in extreme angles the segmentation is visually accurate. As shown in Fig. 2, the fitted skeleton $\mathcal{S}$ is one of the two input sources of the body segmentation module, hence the quality of the segmentation depends on the skeletal tracking method. In the case of an erroneous skeletal fitting, the quality of the segmentation drops without though compromising the results of the overall 3D body reconstruction, because such outliers are rejected by the proposed robust data filtering method.

The proposed method uses the obtained point-sets to fit 9 positive-definite tensor-spline models to the torso, arms, forearms, thighs, and legs. A discussion regarding the head, hands and joints can be found in the beginning of Sec. III. The data flow diagram in Fig. 2 shows that the data histograms are updated in every frame using the incoming point-sets and then the robust data computed from the peaks of the histograms are fed to the proposed tensor fitting method (Sec. III-D). The tensor fitting is performed by minimizing the energy function in Eq. 16 in an on-line fashion, i.e. one iteration of the minimization algorithm is executed per frame. The cycle of the proposed framework (shown in Fig. 2) has linear computational complexity with respect to the size of the input data ($O(n)$) and runs in real time ($\sim$ 25 frames/second) using the computer configuration described earlier.

Figure 4 shows an example of an intermediate state of the real-time process, i.e. before the fitting algorithm converges. The right plate shows a frame of the input data with the current peaks of the data histograms ($d_{i,j}$ in Eq. 15) superimposed as a quadratic grid. The left plate shows an intermediate state of the 3D reconstructed body model.

Figures 5 and 6 show the computed positive-definite tensor-spline models after convergence. The tensor spline models are visualized as quadratic meshes obtained by evaluating Eq. 7 at a predefined discrete set of points in the input domain (ϕ, s). A picture of the corresponding person is also shown on the right for visual comparison. In both cases all tensor-splines use tensor bases of degrees $d = 2, 3$ with cubic B-splines, i.e. the number of unknown tensor coefficients are 7 per control point. This configuration produces realistic approximation of the shape of the body segments, based on visual comparison with the images of the depicted human subjects.

The use of the Riemannian metric on positive-definite tensor splines (Sec. II-E) is demonstrated in Fig. 7. The third avatar from the left (A) and from the right (B) correspond to the positive-definite tensor-spline models in Figs. 5 and 6 respectively. The 9 avatars in Fig. 7 lie on the geodesic defined in the Riemannian space of positive-definite tensor-splines that passes through the two aforementioned avatars at $\lambda = 0$ and $\lambda = 1$ respectively. Other avatars on this geodesic are shown for various values of λ in the range $[-0.5, 1.5]$ and

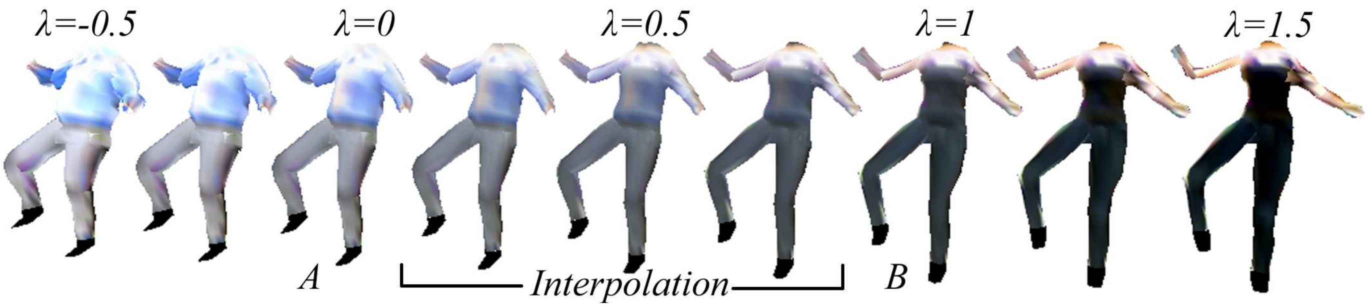

Fig. 7. Avatars on a geodesic defined in the Riemannian space of positive-definite tensor splines. The results of extrapolation and interpolation between the two data points show natural transitions in the appearance of the body, such as the body fat added in the extrapolant on the left ($\lambda = -0.5$).

correspond to results of interpolation or extrapolation using the Riemannian metric presented in Sec. II-E. By observing the avatar on the left ($\lambda = -0.5$), one can see that the shape of the body shows natural-looking body fat in the torso and thighs. It should be emphasized that, although the proposed algorithm does not model special parameters of the body, such as body fat, the result of the extrapolation follows a natural increment of the body fat while transitioning from the right (thinner body type) to the left (bulkier body type).

Another useful application of the proposed tensor body reconstruction is shown in Fig. 8. The body of a female subject was scanned using the proposed method two times between a 3-month period during pregnancy. The difference between the two models can be computed by subtracting the corresponding tensor splines (Eq. 7) for every point in the (ϕ,s) domain.

After having reconstructed the 3D shape of a human body using positive-definite tensor-splines, it can be rendered in any arbitrary posture given in the form of a parametric skeleton $\mathcal{S}$. The avatars shown in Figs. 7, 9, 8 and 10 are examples of tensor-spline models rendered in various postures. The 3D models are colored using the R,G,B values at the corresponding projection of the points in the video frames. Although texture reconstruction was not discussed in this paper, it can be simply done by collecting R,G,B values in the K-mean clusters along with the data values in the dynamic histogram method discussed in Sec. III-E.

The proposed technique was validated using anthropometric measurements from a group of four male volunteers. Standard protocols for anthropometry were followed as described in the ISAK guide [33], in order to measure the circumference of the legs of the participants in five distinct zones identified by their distance from the maximum girths of the calfs and thighs. The results were compared with those computed from the 3D models using the proposed method, and the absolute errors are reported in Fig. 11. According to the results, the median errors are in the range of 1.5-2cm, which are similar to the errors reported in [18]. This observation, although it cannot lead to precise scientific comparisons between the proposed method and the one presented in [18] due to differences in the pool of participants and potential errors introduced by the anthropometry procedures, it shows a clear indication of similarities between the reported results, in terms of the overall order of magnitude of the reported errors. A comparison between the running time of these two techniques shows a notable difference of 3 orders of magnitude (i.e. 10^3). Specifically the method in [18] requires more than 60 minutes for a single body reconstruction, while the proposed technique converges in about 2 seconds ($\sim$ 50 frames @25fps) using computer configurations with similar computational power. This conclusively demonstrates the efficiency of the presented method.

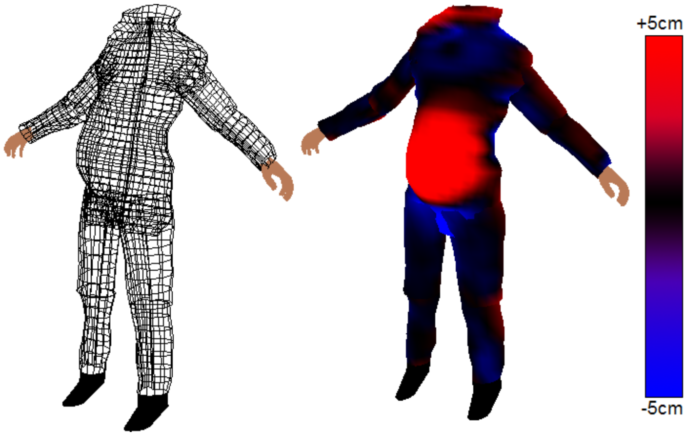

Fig. 8. Example of 3D body reconstruction of a female pregnant model. Visualization of body changes measured by the proposed method in a 3-month period during pregnancy.

Finally, the same validation procedure was followed to compare the 3D models computed from the proposed method and those obtained using the Kinect Fusion algorithm included in the Microsoft Kinect SDK [28]. The latter algorithm does not work when the body moves in front of the camera, unlike the proposed method. Furthermore, the camera collected RGB-D images from a close distance from the subjects (partially depicted in the images), which resulted to $\sim$ 10 times more precise data compare to those collected using the Tensor Body reconstructions, in which case the camera was placed far from the subjects so that they are fully depicted in the recorded images. Due to this significant difference in the quality of the input data the results from the Kinect Fusion algorithm was treated as the ground truth and was compared with the estimated Tensor Bodies (Fig. 12) using the same metric and format as in Fig. 11. The reported errors were around 1.5cm, which is within the range of errors reported in Fig. 11.

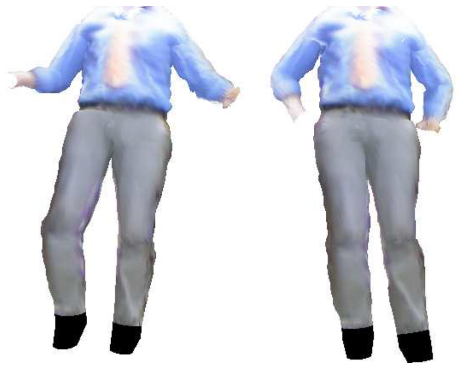

Fig. 9. A reconstructed Tensor-Body avatar rendered in two different natural postures.

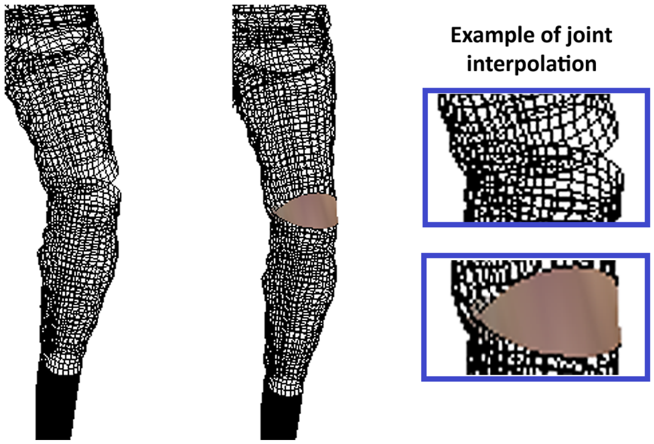

Fig. 10. Example of tensor interpolation in the deformable area around the knee.

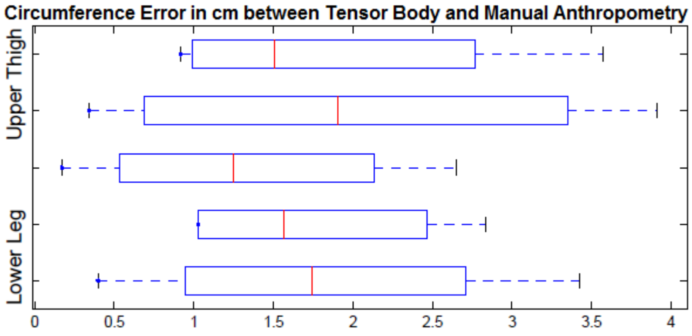

Fig. 11. Absolute errors between manual anthropometric measurements and those computed using the proposed tensor body method.

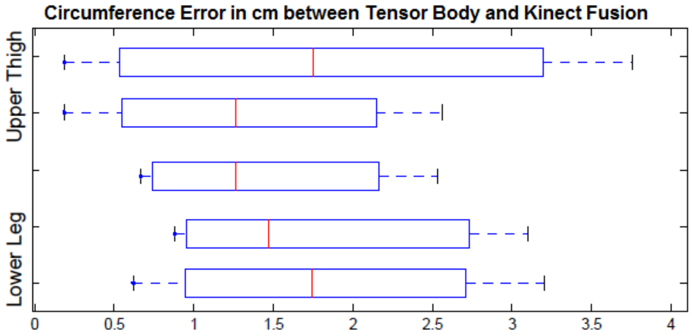

Fig. 12. Absolute errors between anthropometric measurements using the Kinect Fusion algorithm [28] and those computed using the proposed tensor body method.

V. Discussion and Conclusion

In this paper a novel framework for real-time 3D reconstruction of the human body was presented, dubbed *Tensor Body*. A novel algorithm for estimating positive-definite tensor-splines from RGB-D data was introduced. The proposed algorithm uses a mathematical model for parametrizing the space of positive-definite tensors using a convex approximation of the space, which guarantees that the estimated tensors lie within the positive-definite side of the space. Furthermore, a Riemannian metric on the space of positive-definite tensor-splines was presented and employed for interpolation, extrapolation, and for computing geodesics between 3D reconstructed avatars.

One of the benefits of the proposed method is that it runs in real-time and it does not require the human subjects to be on a specific posture. The 3D reconstruction can be performed while the user plays a game or in general interacts with a natural user interface environment, and hence is depicted in the RGB-D frames on a variety of postures.

The presented framework has a robust mechanism that filters the incoming 3D points (input depth measurements). It should be noted that the magnitude of errors reported in Figs. 11 and 12 is very close to the resolution of the depth camera, which recorded 1 pixel per $\sim 1cm$ on the bodies of the human subjects. More specifically, when the subject is fully depicted in the acquired pictures, ~ 200 depth measurements are recorded along the subject's height (assuming that 40 out of the 240 pixels are not utilized due to natural motion of the subject in front of the camera). Therefore, the camera records 1 depth measurement per $\sim (h/200)cm$, where h is the height of the human subject in centimeters (i.e. $\sim 0.95cm$ sampling frequency for $h = 190cm$). Hence, it is natural to expect anthropometric errors in the magnitude reported in Figs. 11 and 12 due to the resolution limit of the depth sensor.

The proposed method for real-time 3D reconstruction of the human body has the potential to be employed in several applications in the areas of anthropometry, communications, psychology, tele-medicine, and other areas of human-computer interaction. Furthermore, it can be used as a module for frequency-based shape compression of human bodies depicted in holographic videos. Future improvements on the resolution of the depth sensor will also allow the proposed method to be used in other areas that require higher quality graphics such as motion pictures.

In the future, the author plans to apply the proposed framework to monitor changes in the shape of human bodies and perform quantitative analysis of body shapes in specific age/gender groups, which could potentially be proven to be a significant tool against obesity, or other related diseases, such as heart disease [19]. Furthermore, the Tensor Body framework can be used as a tool for indirect anthropometry in order to compute body shape atlases from healthy subjects of various ages, genders, and ethnicities. Such an atlas could be used for analyzing quantitatively the shape differences of the bodies across population groups and derive various useful statistical results.

Acknowledgment

The author would like to thank Mrs. Brittany Osbourne from the UF Department of Anthropology for many fruitful discussions about anthropometry. The author would also like to thank the anonymous volunteers who participated in the data collection process, as well as the anonymous reviewers for their insightful comments.

References

[1] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with Microsoft Kinect sensor: A review," *Cybernetics, IEEE Transactions on*, p. in press, 2014.

[2] A. P. Placitelli and M. Ciampi, "Controller-free exploration of medical image data: Experiencing the Kinect," *24th International Symposium on Computer-Based Medical Systems*, pp. 1–6, 2011.

[3] B. Lange et al., "Interactive game-based rehabilitation using the Microsoft Kinect," *IEEE Virtual Reality Workshops*, pp. 171–172, 2012.

[4] L. Xia et al., "Human detection using depth information by Kinect," *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 15–22, 2011.

[5] M. Richter, K. Varanasi, N. Hasler, and C. Theobalt, "Real-time reshaping of humans," in *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, 2012, pp. 340–347.

[6] I. Oikonomidis et al., "Efficient model-based 3d tracking of hand articulations using Kinect," *In Proceedings of the British Machine Vision Association Conference*, 2011.

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

[8] O. Faugeras, *Three-Dimensional Computer Vision*. MIT press, 1993.

[9] D. Forsyth and J. Ponce, *Computer Vision: A modern approach*. Prentice Hall, 2003.

[10] B. Horn, *Robot Vision*. MIT press, Cambridge, Massachusetts, 1986.

[11] M. Zollhfer, M. Martinek, G. Greiner, M. Stamminger, and J. Smuth, "Automatic reconstruction of personalized avatars from 3d face scans," *Computer Animation and Virtual Worlds*, vol. 22, no. 2-3, pp. 195–202, 2011.

[12] V. Uriol and M. Cruz, *Video-based avatar reconstruction and motion capture*. California State University at Long Beach, 2005.

[13] M.-C. Villa-Uriol, F. Kuester, and N. Bagherzadeh, "Image-based avatar reconstruction," *In Proceedings of the NSF Workshop on Collaborative Virtual Reality and Visualization*, 2003.

[14] A. Hilton, D. Beresford, T. Gentils, R. Smith, and W. Sun, "Virtual people: capturing human models to populate virtual worlds," in *Computer Animation, 1999. Proceedings*, 1999, pp. 174 –185.

[15] B. Lok, "Online model reconstruction for interactive virtual environments," in *Proceedings of the 2001 symposium on Interactive 3D graphics*, ser. I3D '01. ACM, 2001, pp. 69–72.

[16] S.-Y. Lee, I.-J. Kim, S. C. Ahn, H. Ko, M.-T. Lim, and H.-G. Kim, "Real time 3d avatar for interactive mixed reality," in *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, ser. VRCAI '04. ACM, 2004, pp. 75–80.

[17] J. Tong, J. Zhou, L. Liu, Z. Pan, and H. Yan, "Scanning 3d full human bodies using kinects," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 4, pp. 643–650, 2012.

[18] A. Weiss, D. Hirshberg, and M. Black, "Home 3d body scans from noisy image and range data," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011, pp. 1951–1958.

[19] A. Barmpoutis, "Automated human avatar synthesis for obesity control using low-cost depth cameras," *Stud. Health Technol. Inform.*, vol. 184, pp. 36–42, 2013.

[20] A. Barmpoutis, B. C. Vemuri, T. M. Shepherd, and J. R. Forder, "Tensor splines for interpolation and approximation of DT-MRI with applications to segmentation of isolated rat hippocampi," *TMI: Transactions on Medical Imaging*, vol. 26, no. 11, pp. 1537–1546, 2007.

[21] R. Kumar, A. Barmpoutis, A. Banerjee, and B. C. Vemuri, "Non-lambertian reflectance modeling and shape recovery for faces using anti-symmetric tensor splines," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 533–567, 2011.

[22] A. Barmpoutis, J. Ho, and B. C. Vemuri, "Approximating symmetric positive semi-definite tensors of even order," *SIAM Journal on Imaging Sciences*, vol. 5, no. 1, pp. 434–464, 2012.

[23] A. Barmpoutis, M. S. Hwang, D. Howland, J. R. Forder, and B. C. Vemuri, "Regularized positive-definite fourth-order tensor field estimation from DW-MRI," *NeuroImage*, vol. 45, no. 1 sup.1, pp. 153–162, 2009.

[24] C. de Boor, "On calculating with b-splines," *J. Approx. Theory*, vol. 6, pp. 50–62, 1972.

[25] S. Helgason, *Differential geometry, Lie groups, and symmetric spaces*. American Mathematical Society, 2001.

[26] P. Fletcher, C. Lu, S. Pizer, and S. Joshi, "Principal geodesic analysis for the study of nonlinear statistics of shape," *IEEE Transactions on Medical Imaging*, vol. 23, no. 8, pp. 995–1005, 2004.

[27] X. Pennec, P. Fillard, and N. Ayache, "A Riemannian framework for tensor computing." *International Journal of Computer Vision*, vol. 65, 2005.

[28] "Microsoft Kinect SDK," *http://www.microsoft.com/enus/kinectforwindows/*.

[29] "OpenNI," *http://www.openni.org/*.

[30] M. A. Livingston et al., "Performance measurements for the Microsoft Kinect Skeleton," *IEEE Virtual Reality Workshops*, pp. 119–120, 2012.

[31] A. Davison, *Kinect Open Source Programming Secrets: Hacking the Kinect with OpenNI, NITE, and Java*. McGraw-Hill, 2012.

[32] C. Lawson and R. Hanson, *Solving Least Squares Problems*. Prentice-Hall, 1974.

[33] I. S. for the Advancement of Kinanthropometry, *International Standards for Anthropometric Assessment*. Australia, 2001.

Angelos Barmpoutis Angelos Barmpoutis is an Assistant Professor and the coordinator of research and technology in the Digital Worlds Institute at the University of Florida. He is also an affilate faculty of the Computer and Information Science and Engineering Department and the Biomedical Engineering Department at the University of Florida. He received the B.Sc. in Computer Science from the Aristotle University of Thessaloniki in 2003, the M.Sc. in Electronics and Electrical Engineering from the University of Glasgow in 2004, and the Ph.D. degree in Computer and Information Science and Engineering from the University of Florida in 2009. He has coauthored numerous journal and conference publications in the areas of computer vision, machine learning, biomedical imaging, and digital humanities.