Motion Capture Data Recovery Using Skeleton Constrained Singular Value Thresholding

Cheen-Hau Tan, JunHui Hou, and Lap-Pui Chau
School of Electrical and Electronics Engineering
Nanyang Technological University

***Abstract*—Motion capture data could be missing due to imperfections during the acquisition process. Singular Value Thresholding is an effective method to recover missing motion capture data. However, its effectiveness decreases significantly when markers are missing for longer periods of time. To alleviate this problem, we utilize the fact that human bones are rigid to constrain inter-marker distances of specific sets of markers. We extend the Singular Value Thresholding method for mocap recovery to include skeleton constraints. On average, our proposed method improves on the Singular Value Thresholding method by 40%, and performs 4% better than a recent state of the art method at up to 11 times faster computation time. *Index Terms*—otion capture Mocap recovery Matrix completionotion capture Mocap recovery Matrix completionM**

I. Introduction

Human motion capture data, or mocap data, specifies the motion of a human skeletal structure, and is used to drive natural looking animations in a variety of applications such as movies, video games, virtual worlds, sports, and medical applications. New mocap data can be synthesized from existing motion data by decomposing them [30] and combining them [29], but they are initially acquired by capturing human motions. Mocap data acquisition is traditionally done using an optical capture system (e.g., Vicon System) whereby the positions of sensor markers attached to human actors are tracked while the actors perform the required motions. However, the mocap acquisition process is not perfect even for professional systems; data readings might be noisy or missing due to occluded markers. This shortcoming is more severe for mocap data acquired using off the shelf equipment such as Microsoft Kinect. Thus a number of works have been proposed to recover missing entries of mocap data.

Candès and Recht [5] showed that a low rank matrix can be accurately recovered from observations of a small fraction of its entries by solving for the matrix with the lowest nuclear norm which conforms to the observations; this approach for matrix recovery is known as matrix completion. Lai et al. [15] showed that mocap data, when arranged in the form of a matrix, is a low rank matrix. They then exploit this property to recover mocap data by using a matrix completion method known as Singular Value Thresholding (SVT) [4]. SVT formulates the missing data recovery problem as a convex optimization problem, which can be solved efficiently.

C.-H. Tan email: tanc0158@e.ntu.edu.sg
J. Hou email: houj0001@ntu.edu.sg
L.-P. Chau email: elpchau@ntu.edu.sg

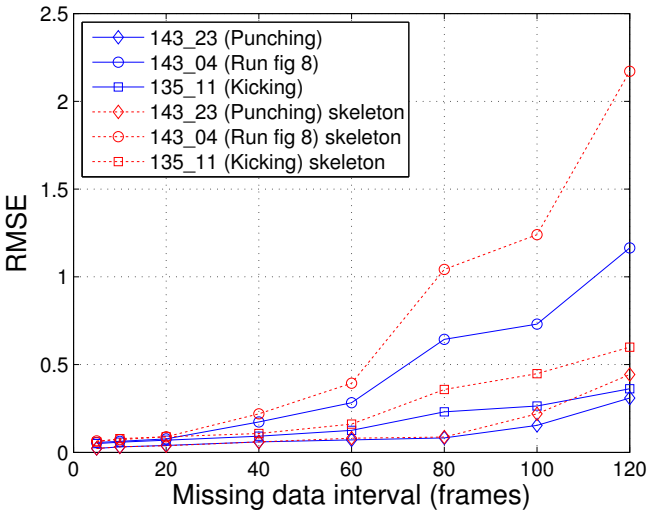


Fig. 1. RMS error (solid line) and skeleton RMS error (dotted line) of three mocap sequences recovered using Singular Value Thresholding over varying missing data intervals. The skeleton RMS error is the RMS error of inter-joint distances of all pairs of adjacent joints in the recovered mocap sequence.

Since the human skeleton is rigid, the distance between any two adjacent joints is constant throughout a motion sequence. However, this inter-joint distance is not preserved for mocap data recovered using SVT, especially when mocap entries are missing for long intervals. The bone length of the skeleton of a recovered mocap data might undergo extension and shrinkage throughout a sequence; this observation is verified quantitatively in Fig. 1. We remove consecutive entries of mocap data for certain interval lengths (details in Section V), recover the entries using SVT, and plot the RMS error for the recovered data and the RMS error of inter-joint distances of all pairs of adjacent joints from the recovered data. As shown in the figure, both errors increase when the interval length is increased; increases in fluctuation of the inter-joint distance is correlated with increases in error of the recovered mocap data.

In this paper we propose a skeleton constrained SVT method (SCSVT) which attempts to preserve inter-joint distances when recovering mocap data. We formulate the inter-joint distances as additional constraints, taking care to ensure that the problem is still convex, and solve the problem within the SVT framework. Experiments show that SCSVT outperforms the basic SVT method by a significant margin, especially for cases where mocap data is missing for long periods. SCSVT performs competitively with BOLERO [17], a mocap recovery

method modeled on linear dynamical systems which also utilizes skeleton constraints, but at a much lower computational cost. In addition, since SCSVT does not require training data, it does not suffer from drawbacks faced by methods that have such requirements.

In the following section, we will briefly describe other related work on mocap data recovery. In Section III, we will discuss the problem formulation and solution of the basic SVT mocap recovery method. After than, we extend the problem to accommodate skeleton constraints and derive the solution in Section IV. In Section V, we will compare and discuss the performance of the evaluated mocap recovery methods and wrap up with a conclusion in Section VI.

II. Related Work

A number of works have been proposed to recover missing mocap data. Wiley and Hahn [26] uses interpolation to estimate missing data. To allow for real-time missing marker recovery, Piazza et al. [22] predicts whether the immediate motion is linear, circular or both, and then applies extrapolation for recovery. Yii et al. [28] uses a form of regression by modeling all data in a time window as a linear model and predicting missing elements using linear least squares. Papadimitriou et al. [21] improves on this method by using frequency analysis to automatically set the window size.

Some works utilize local linear models extracted from prior data to predict missing data. Liu and McMillan [19] forms a global linear model and a set of locally linear models from a database. Missing values are initially recovered using the global model, then a classifier assigns a local model to each frame. By assuming that all frames lie in their respective models, a frame can be recovered by least squares estimation of model coefficients from available data, and then recovery of missing data using said coefficients. Instead of precomputing locally linear models, Chai and Hodgins [6] retrieves poses (frames) which are similar to the processed pose at runtime. These poses, which form a linear model, are used as a prior in energy minimization optimization to recover the processed pose. The pose retrieval efficiency of this framework is improved by Krüger et al. [14] by using a kd-tree to speed up retrieval and the recovery performance is improved by Baumann et al. [3] by additionally considering the velocity and acceleration of each pose.

Grochow et al. [9] learns a global nonlinear model to form a low-dimensional space using Gaussian Process Latent Variable Models (GPLVM) [16] and maximizes a likelihood function over poses. Wang et al. [25] observes that GPLVM treats each mocap pose as independent data points, and propose using Gaussian Process Dynamical Models, which considers the temporal structure of mocap data, to model mocap data instead. Lou et al. [20] extended the concept of linear model of poses to a linear model of consecutive sets of frames, i.e., a spatio-temporal model. With the aid of robust statistical methods, spatio-temporal bases from a training set are used to estimate missing values of mocap data. Taylor et al. [24] proposes that mocap data should be modeled using a non-linear model and uses Conditional Restricted Boltzmann Machine to achieve this aim. Xiao et al. [27] proposed that a sparse representation of a training set can be used to describe mocap frames. Given an incomplete frame, they solve for the sparsest representation for the frame with the constraint that the recovered frame is as similar as possible to the incomplete frame. This method is improved by Hou [13] by using a trajectory-based representation for the sparse representation.

Except for regression and interpolation-based methods, the methods described earlier require a prior motion database to work. There exists several methods with no such requirement. Li et al. [18] model the entire processed sequence as a linear dynamical system; a sequence of frames is modeled as a sequence of latent variables of lower dimension. Consecutive latent variables are related by a linear map, and each frame is related to a latent variable by a linear projection. The objective function to be maximized is the likelihood of the observed data with respect to the model. The solution is computed using expectation maximization: iteratively estimate the latent variables and model parameters, and estimate the missing values until convergence. The method BOLERO [17] improves on this system by enforcing bone length constraints for the recovered data. This is done by including bone constraints in the objective function. Since the bone constraints are formulated as nonlinear constraints, the objective function becomes nonlinear, and is solved by using either Newton's method or the coordinate descent method. In contrast, our method formulates the skeleton constraint as an inequality constraint, which is convex, and we model mocap recovery as a convex optimization problem. Hence we can utilize convex optimization to find a global optimal solution efficiently. Another method to recover mocap data is the aforementioned matrix completion by Singular Value Thresholding [15] which we build on. This method is improved in [23] for randomly distributed missing data by using a trajectory-based matrix representation.

A related class of works deal with recovering occluded sensor markers during optical motion capture, and are usually tailored for the mocap capture system used. Herda et al. [10], [11] use a pre-specified skeleton model to estimate the position of missing neighboring markers. In the system proposed by Hornung et al. [12], sets of markers with fixed relative positions are found automatically without a manually specified skeleton structure; this information is used to aid missing marker estimation. In some works, extended Kalman filters [8], [1] and unscented Kalman filters [2] are used to track markers and to recover any missing markers.

III. Mocap Recovery Using SVT

For a mocap sequence to be recovered using matrix completion, it is first represented in the form of a matrix $\mathbf{M} \in \mathbb{R}^{n_1 \times n_2}$, $n_1 = 3j$

$$\mathbf{M} = [\mathbf{f_1}\ \mathbf{f_2}\ ...\ \mathbf{f_{n_2}}]$$
$$\text{where } \mathbf{f} = [x_1\ x_2\ ...\ x_j\ y_1\ y_2\ ...\ y_j\ z_1\ z_2\ ...\ z_j]^T$$

where each frame $\mathbf{f}$ holds the 3D coordinates $[x,y,z]$ of j joints. The mocap matrix has low rank, thus it can be effectively

recovered using matrix completion. The mocap recovery problem is posed as the following optimization problem P1.

$$\begin{aligned} \min \quad & f(\mathbf{X}) && \text{(P1)} \\ \text{s.t.} \quad & \mathscr{S}(\mathbf{X}) = \mathbf{b} && (1) \end{aligned}$$

where $\mathscr{S} : \mathbb{R}^{n_1 \times n_2} \mapsto \mathbb{R}^m$ is a linear map which extracts m entries from a $n_1 \times n_2$ matrix. $\mathbf{b}$ contains the m observed entries of the matrix $\mathbf{M}$ to be recovered, i.e., $\mathbf{b} = \mathscr{S}(\mathbf{M})$. $f(\mathbf{X}) := \|\mathbf{X}\|_*$, where $\|\mathbf{X}\|_*$ is the nuclear norm of the decision variable $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2}$. The problem can be solved using the Singular Value Thresholding (SVT) algorithm [4], which computes an approximate solution by solving for $f(\mathbf{X}) := \tau\|\mathbf{X}\|_* + \frac{1}{2}\|\mathbf{X}\|_F^2$, where $\|\mathbf{X}\|_F$ is the Frobenius norm of $\mathbf{X}$. However, it is efficient in both time and memory. A better approximation is obtained by setting a larger value for parameter τ.

The solution for Problem P1 using the SVT method is

$$\mathbf{X}^k = \mathscr{D}_\tau \mathscr{S}^*(\mathbf{y}^{k-1}) \quad (2)$$

$$\mathbf{y}^k = \mathbf{y}^{k-1} + \delta(\mathbf{b} - \mathscr{S}(\mathbf{X}^k)) \quad (3)$$

The recovered mocap matrix $\widetilde{\mathbf{X}} = \mathbf{X}^{k_{\text{final}}}$ is obtained by iteratively solving for $\mathbf{X}^k$ in Eq. (2) and $\mathbf{y}^k$ in Eq. (3) until convergence. $\mathscr{S}^* : \mathbb{R}^m \mapsto \mathbb{R}^{n_1 \times n_2}$ is the adjoint of $\mathscr{S}$, while δ is the step size parameter. $\mathscr{D}_\tau$, a characteristic operator of SVT, soft-thresholds the singular values σ_i of input $\mathbf{X}$ with parameter τ. It is defined as

$$\mathscr{D}_\tau(\mathbf{X}) := \mathbf{U}\mathscr{D}_\tau(\Sigma)\mathbf{V}^T, \quad \mathscr{D}_\tau(\Sigma) = \text{diag}((\sigma_i - \tau)_+) \quad (4)$$

where $(\sigma_i - \tau)_+ = \max(0, \sigma_i - \tau)$, and orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$, and matrix of singular values Σ are obtained using Singular Vector Decomposition (SVD), i.e., $\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^T$. In other words, $\mathscr{D}_\tau(\mathbf{X})$ factorizes $\mathbf{X}$ using SVD and sets all singular values in Σ that are less than τ to zero.

IV. Skeleton Constrained SVT

A. Problem Formulation

As mentioned earlier, the previous method of mocap recovery using SVT does not enforce preservation of skeleton constraints. We overcome this weakness by applying skeleton constraints to Problem P1. First, we rewrite $\mathscr{S}(\mathbf{X})$ in Eq. (1) as a composition of matrix $\mathbf{A}$ and a linear map vec

$$\mathscr{S}(\mathbf{X}) = \mathbf{A}\,\text{vec}(\mathbf{X}) = \mathbf{A}\mathbf{x} \quad (5)$$

$$\text{where} \quad \text{vec} : [\mathbf{f_1} \ldots \mathbf{f_{n_2}}] \mapsto \begin{bmatrix} \mathbf{f_1} \\ \vdots \\ \mathbf{f_{n_2}} \end{bmatrix}$$

vec maps $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2}$ to $\mathbf{x} \in \mathbb{R}^n$, where $n = n_1 \times n_2$, and $\mathbf{A} \in \mathbb{R}^{m \times n}$ extracts the m observed entries of $\mathbf{x}$.

The mocap skeleton used in this paper is shown in Fig. 2, where each label is a joint, and each edge is a bone connecting adjacent joints. For each joint that is missing, we insert each of its adjacent edges into a set $\mathscr{E}$. For each edge $e_i \in \mathscr{E}$, $i = 1 \ldots p$, we have its length d_i, and its connected joint pair $(\alpha, \beta)_i$. Then, for each e_i, we define skeleton constraint matrix $\mathbf{C}_i \in \mathbb{R}^{3 \times n}$ which extracts the inter-joint distance of $(\alpha, \beta)_i$ from

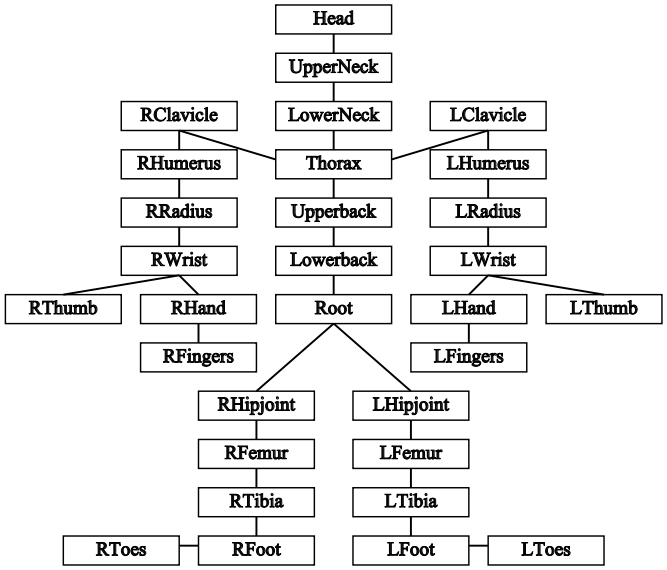


Fig. 2. Mocap skeleton used in this paper. Each label indicates a joint, and 'L' and 'R' prefixes denote left-sided and right-sided respectively.

$\mathbf{x}$, i.e., $\mathbf{C}_i\mathbf{x} = [x\ y\ z]_{\alpha_i}^T - [x\ y\ z]_{\beta_i}^T$. Applying constraints $\mathbf{C}_i$ to Problem P1, we get Problem P2 as follows

$$\begin{aligned} \min \quad & f(\mathbf{X}) && \text{(P2)} \\ \text{s.t.} \quad & \mathbf{x} = \text{vec}(\mathbf{X}) && (6) \\ & \mathbf{A}\mathbf{x} = \mathbf{b} && (7) \\ & \|\mathbf{C}_i\mathbf{x}\|_2 \le d_i \qquad i = 1 \ldots p && (8) \end{aligned}$$

d_i can be obtained in a preprocessing step by simple averaging of inter-joint distances of non-missing joints.

B. Solution of Problem

In a similar way to [4], we derive the solution to Problem P2 by stating the solution given by the gradient algorithm applied to the Lagrangian, and then recasting it into the solution obtained using SVT. The gradient algorithm applied to the Lagrangian for a problem with Lagrangian $\mathscr{L}(\mathbf{X}, \mathbf{y})$ can be expressed as

$$\mathbf{X}^k = \arg\min_X \mathscr{L}(\mathbf{X}, \mathbf{y}^{k-1}) \quad (9)$$

$$\mathbf{y}^k = \mathbf{y}^{k-1} + \delta\, \partial_{\mathbf{y}} \mathscr{L}(\mathbf{X}^k, \mathbf{y}) \quad (10)$$

where $\mathbf{y}$ is the Lagrange multiplier of $\mathscr{L}(\mathbf{X}, \mathbf{y})$, and δ is the step size parameter. The solution proceeds by solving for $\mathbf{X}$ which minimizes $\mathscr{L}(\mathbf{X}, \mathbf{y})$ and moving $\mathbf{y}$ in the direction of the gradient iteratively. Upon convergence, the solution $\widetilde{\mathbf{X}} = \mathbf{X}^{k_{\text{final}}}$ is obtained.

To obtain the Lagrangian for Problem P2, we first express the quadratic skeleton constraints in Eq. (8) in a convex form, i.e., in the form of a second order cone $\mathscr{K}$ (see [4]).

$$\begin{bmatrix} \mathbf{C}_i\mathbf{x} \\ d_i \end{bmatrix} \in \mathscr{K}_i \quad (11)$$

Following that, the Lagrangian is

$$\begin{aligned}\mathscr{L}(\mathbf{X},\mathbf{y}_a,\mathbf{y}_c,\mathbf{s}) =& f(\mathbf{X}) + \langle \mathbf{y}_a, \mathbf{b} - \mathbf{A}\,\mathrm{vec}(\mathbf{X})\rangle \\ &+ \sum_i^p (\langle \mathbf{y}_{c_i}, -\mathbf{C}_i\,\mathrm{vec}(\mathbf{X})\rangle - s_i d_i) \qquad (12)\\ =& f(\mathbf{X}) + \langle \mathbf{y}_a, \mathbf{b}\rangle - \langle \mathbf{y}_a, \mathbf{A}\,\mathrm{vec}(\mathbf{X})\rangle \\ &- \sum_i^p (\langle \mathbf{y}_{c_i}, \mathbf{C}_i\,\mathrm{vec}(\mathbf{X})\rangle + s_i d_i) \qquad (13)\\ =& f(\mathbf{X}) + \langle \mathbf{y}_a, \mathbf{b}\rangle - \left\langle \mathrm{vec}^*(\mathbf{A}^T\mathbf{y}_a), \mathbf{X}\right\rangle \\ &- \sum_i^p \left(\left\langle \mathrm{vec}^*(\mathbf{C}_i^T\mathbf{y}_{c_i}), \mathbf{X}\right\rangle + s_i d_i\right) \qquad (14)\end{aligned}$$

where $<\mathbf{X},\mathbf{Y}>$ is the standard inner product between matrices $\mathbf{X}$ and $\mathbf{Y}$. Taking $f(\mathbf{X}) \coloneqq \tau\|\mathbf{X}\|_* + \frac{1}{2}\|\mathbf{X}\|_F^2$, we have

$$\begin{aligned}&\arg\min_X \mathscr{L}(\mathbf{X},\mathbf{y}_a,\mathbf{y}_c,\mathbf{s})\\ &= \left\{ \mathbf{X} \;\middle|\; \tau\,\partial_X\|\mathbf{X}\|_* + \mathbf{X} \right. \qquad (15)\\ &\quad \left. - \mathrm{vec}^*(\mathbf{A}^T\mathbf{y}_a) - \sum_i^p \mathrm{vec}^*(\mathbf{C}_i^T\mathbf{y}_{c_i}) = 0 \right\} \qquad (16)\\ &= \left\{ \mathbf{X} \;\middle|\; \tau\,\partial_X\|\mathbf{X}\|_* + \mathbf{X} - \mathrm{vec}^*\left(\begin{bmatrix}\mathbf{A}^T & \mathbf{C}^T\end{bmatrix}\begin{bmatrix}\mathbf{y}_a\\ \mathbf{y}_c\end{bmatrix}\right) = 0 \right\} \qquad (17)\end{aligned}$$

$$\text{where}\quad \mathbf{C} = \begin{bmatrix}\mathbf{C}_1\\ \vdots\\ \mathbf{C}_p\end{bmatrix}, \mathbf{y}_c = \begin{bmatrix}\mathbf{y}_{c_1}\\ \vdots\\ \mathbf{y}_{c_p}\end{bmatrix}$$

It has been shown in [4] that

$$\begin{aligned}\mathscr{D}_\tau(\mathbf{Y}) &= \arg\min_X\ \tau\|\mathbf{X}\|_* + \frac{1}{2}\|\mathbf{X}-\mathbf{Y}\|_F^2 \qquad (18)\\ &= \{\mathbf{X} \mid \tau\,\partial_X\|\mathbf{X}\|_* + \mathbf{X} - \mathbf{Y} = \mathbf{0}\} \qquad (19)\end{aligned}$$

By comparing with Eq. (19), we can solve Eq. (17) using the singular value soft thresholding operator $\mathscr{D}_\tau$

$$\arg\min_X \mathscr{L}(\mathbf{X},\mathbf{y}_a,\mathbf{y}_c,\mathbf{s}) = \mathscr{D}_\tau\,\mathrm{vec}^*\left(\begin{bmatrix}\mathbf{A}^T & \mathbf{C}^T\end{bmatrix}\begin{bmatrix}\mathbf{y}_a\\ \mathbf{y}_c\end{bmatrix}\right) \qquad (20)$$

Thus we update $\mathbf{X}^k$ in Eq. (9) using Eq. (20). To update $\mathbf{y}_a$, we substitute the Lagrangian $\mathscr{L}(\mathbf{X},\mathbf{y}_a,\mathbf{y}_c,\mathbf{s})$ in Eq. (12) with $\mathbf{X} = \mathbf{X}^k$ into Eq. (10)

$$\begin{aligned}\mathbf{y}_a^k =& \mathbf{y}_a^{k-1} + \delta\,\partial_{\mathbf{y}_a}\mathscr{L}(\mathbf{X}^k,\mathbf{y}_a,\mathbf{y}_c,\mathbf{s}) \qquad (21)\\ =& \mathbf{y}_a^{k-1} + \delta(\mathbf{b} - \mathbf{A}\,\mathrm{vec}(\mathbf{X}^k)) \qquad (22)\end{aligned}$$

Each $\mathbf{y}_{c_i}$ is updated by orthogonal projection $\mathscr{P}_i$ onto their respective cones $\mathscr{K}_i$ (see [4])

$$\begin{bmatrix}\mathbf{y}_{c_i}^k\\ s_i^k\end{bmatrix} = \mathscr{P}_i\left(\begin{bmatrix}\mathbf{y}_{c_i}^{k-1}\\ s_i^{k-1}\end{bmatrix} + \delta\begin{bmatrix}-\mathbf{C}_i\,\mathrm{vec}(\mathbf{X}^k)\\ -d_i\end{bmatrix}\right), i = 1...p \qquad (23)$$

$$\text{where}\quad \mathscr{P}_i : (\mathbf{y}_c, s)_i \mapsto \begin{cases}(\mathbf{y}_c, s), & \|\mathbf{y}_c\| \le s\\ \frac{\|\mathbf{y}_c\|+s}{2\|\mathbf{y}_c\|}(\mathbf{y}_c, s), & -\|\mathbf{y}_c\| \le s \le \|\mathbf{y}_c\|\\ (\mathbf{0}, 0), & s \le -\|\mathbf{y}_c\|\end{cases}$$

The proposed method, an algorithm to solve Problem P2 and recover the mocap sequence, is summarized in Algorithm 1. The inputs are the m observed entries of a mocap matrix, $\mathbf{b} \in \mathbb{R}^m$, and the inter-joint distance of p joint pairs, $\mathbf{d} \in \mathbb{R}^p$. Matrix $\mathbf{A}$ and map vec are described in Eq. (5), and matrix $\mathbf{C}$ was described when discussing Eq. (8). A larger τ improves the accuracy of the solution at the cost of increased number of iterations to convergence. A higher δ reduces the number of iterations to convergence, but if the value is too large, the solution will diverge. In this paper, we empirically set τ to $8\sqrt{n_1 n_2}$ for a reasonable compromise between performance and speed. The value of δ is discussed in Section V-B. The RMSE operator in the condition statement of the while loop is defined in Eq. (24), and the tolerance tol value used is 0.01. The basic SVT algorithm without skeleton constraints can be obtained by removing $\mathbf{C}$, $\mathbf{y}_c$, and s_i from Algorithm 1.

Algorithm 1 Skeleton Constrained SVT

Input: $\mathbf{b}$, $\mathbf{d}$, $\mathbf{A}$, $\mathbf{C}$
Parameter: δ, τ, k_{max}, tol
Output: Recovered mocap sequence $\widetilde{\mathbf{X}}$
Initialization:
$\mathbf{y}_a \leftarrow \mathbf{b}$
$\mathbf{y_c} \leftarrow \mathbf{0}$
$\mathbf{s_c} \leftarrow 0$
$k \leftarrow 1$

while $k < k_{max}$ **and** $\mathrm{RMSE}(\mathbf{A}\,\mathrm{vec}(\mathbf{X}^k), \mathbf{b}) > tol$ **do**
  // update $\mathbf{X}$
  $Y \leftarrow \mathrm{vec}^*\left(\begin{bmatrix}\mathbf{A}^T & \mathbf{C}^T\end{bmatrix}\begin{bmatrix}\mathbf{y}_a^{k-1}\\ \mathbf{y}_c^{k-1}\end{bmatrix}\right)$
  where $\mathbf{C} = \begin{bmatrix}\mathbf{C}_1\\ \vdots\\ \mathbf{C}_p\end{bmatrix}, \mathbf{y}_c = \begin{bmatrix}\mathbf{y}_{c_1}\\ \vdots\\ \mathbf{y}_{c_p}\end{bmatrix}$
  $\sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^T \leftarrow \mathbf{Y}$ // singular value decomposition
  $\mathbf{X}^k \leftarrow \sum_i \max(\sigma_i - \tau, 0)\mathbf{u}_i\mathbf{v}_i^T$

  // update $\mathbf{y}_a, \mathbf{y}_c$
  $\mathbf{y}_a^k = \mathbf{y}_a^{k-1} + \delta(\mathbf{b} - \mathbf{A}\,\mathrm{vec}(\mathbf{X}^k))$
  $\begin{bmatrix}\mathbf{y}_{c_i}^k\\ s_i^k\end{bmatrix} = \mathscr{P}_i\left(\begin{bmatrix}\mathbf{y}_{c_i}^{k-1}\\ s_i^{k-1}\end{bmatrix} - \delta\begin{bmatrix}\mathbf{C}_i\,\mathrm{vec}(\mathbf{X}^k)\\ d_i\end{bmatrix}\right)$ // Eq. (23)
  $k \leftarrow k+1$
end while
$\widetilde{\mathbf{X}} \leftarrow \mathbf{X}^k$

V. Results and Discussion

In this section we evaluate the performance of the proposed method in recovering missing mocap data. For our experiments, we use mocap sequences from the CMU mocap database [7]; each sequence contains 31 joints and 30 skeleton constraints (see Fig. 2), and is sampled at 60 fps. We generate the 3D coordinates of each joint from joint angle data (amc/asf), so the length of each bone d_i is known exactly.

We compare the proposed skeleton constrained SVT mocap recovery method (SCSVT) against the previous basic SVT mocap recovery method (BSVT) [15]. We also compare against BOLERO[1] [17], a method which models the problem as a linear dynamical system with skeleton constraints.

[1]Mocap data values are scaled down for algorithm stability as recommended, but are not transformed to local coordinates. All parameters are left at default values

We use the BOLERO code provided publicly by the authors and the implementation which enforces hard constraints (BOLERO-HC). However, for our test data, since the algorithm only terminates long after the result has practically converged, for better fairness, we remove the stricter termination criterion that checks for a decrease in a likelihood funtion (likelihood increases logarithmically during optimization), and retain the criterion which checks whether the rate of change of likelihood has dropped below a threshold value. Modifying the code in this way decreases computation time substantially with very little change in performance. Implementations of all the compared algorithms are written in matlab and the experiments are run on a computer with an Intel i7-3770 CPU and 8GB memory.

To simulate missing data, we partition a sequence into time intervals of the same length and for each interval we randomly remove a fixed number of joints. The number of removed joints determines the error rate of the corrupted mocap, e.g., removing three joints induces an error rate of $\frac{3}{31} \approx 10\%$. We also vary the interval length from 15–120 frames (0.25–2 seconds). Increasing this length allows us to judge the robustness of mocap recovery towards such 'bursty' errors under the same error rate. We use Root Mean Square Error[2] (RMSE) to quantify the distortion of a recovered matrix.

$$\mathrm{RMSE}(\widetilde{\mathbf{X}}, \mathbf{M}) = \sqrt{\frac{1}{n_1 \times n_2} \sum_{j=1}^{n_2} \sum_{i=1}^{n_1} \left(\widetilde{\mathbf{X}}_{ij} - \mathbf{M}_{ij}\right)^2} \qquad (24)$$

where $\widetilde{\mathbf{X}}$ and $\mathbf{M}$ are the matrices of the recovered and original mocap respectively.

A. Performance Evaluation

We first show the effectiveness of the proposed method in constraining the inter-joint distance of connected pairs of joints. We compute the skeleton RMS error

$$\mathrm{RMSE}_{\mathrm{skel}} = \sqrt{\frac{1}{k} \sum_{i}^{30} \left(\left\| \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{\alpha_i} - \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{\beta_i} \right\|_2 - d_i \right)^2} \qquad (25)$$

where for each ith bone its joints are α_i and β_i, and its length is d_i. There are 30 bones as shown in Fig. 2. As can be seen in Fig. 3, the skeleton RMSE for BSVT and SCSVT generally increases with increasing interval length. However, the error for SCSVT is lower and increases at a much lower rate.

Table I shows a comparison of mocap recovery performance. First, we observe that in most cases the performance of BSVT decreases substantially when the time interval increases, while the performance of SCSVT degrades more slowly. This shows that skeleton constraints improves the resistance of BSVT towards long bursts of marker data loss. We can also observe that SCSVT outperforms BSVT even at the shortest missing joint interval of $l = 15$, which indicates that skeleton constraints improves the performance of BSVT under all missing data patterns. Comparing SCSVT with BOLERO, SCSVT outperforms BOLERO in a slight majority of our tests with a 4% improvement in RMSE on average.

While in general the RMSE of all algorithms increase with increased interval lengths, there are cases where RMSE fluctuates. For example, for sequence 85_02 at $n = 3$, the RMSE for SCSVT rises from 0.191 to 0.230 at $l = 60$, and then falls to 0.204 at $l = 90$. This is not surprising since the simulated missing data entries are not the same for different l, and certain data entries might be more important for recovering the mocap sequence. For example, for $l = 60$, important mocap entries might be missing, but these entries are available for $l = 90$, hence in this case $l = 90$ might result in lower RMSE. The fact that this phenomenon occurs more frequently for SCSVT and BOLERO shows that skeleton constraints are effective in reducing the performance degradation caused by longer interval lengths.

Next, we examine the individual errors of recovered joints with three missing joints and missing interval lengths of 120 frames as shown in Fig. 4 and Fig. 5. We define joint error as the euclidean distance between the position of a recovered joint and the ground truth, i.e.,

$$\text{Error} = \left\| \begin{bmatrix} x & y & z \end{bmatrix}^T_{\mathrm{reco}} - \begin{bmatrix} x & y & z \end{bmatrix}^T_{\mathrm{gnd}} \right\|_2 \qquad (26)$$

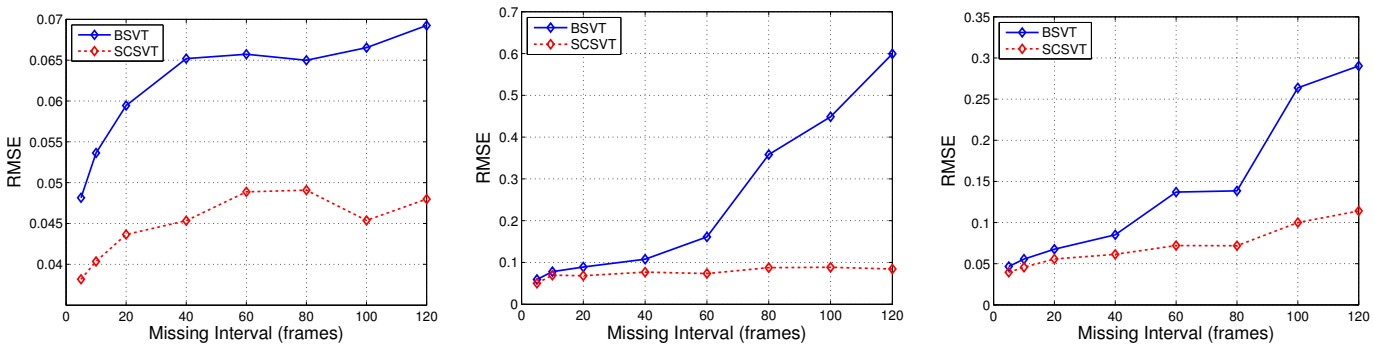
In both figures, subfigures (a)-(c) show end joints, which are joints with one neighboring joint, while subfigures (d)-(f) show inner joints, which are joints with more than one neighboring joint. We can see that in nearly every case, SCSVT performs better than BSVT. In certain cases, the joints recovered by BSVT deviate drastically from the correct positions, as seen in Fig .4(a) and Figs. 5(a)-(b). Since skeleton constraints limit the maximum error of each joint, SCSVT shows a very large improvement over BSVT in these cases. Overall, SCSVT shows a slight improvement over BOLERO.

B. Processing Time Evaluation

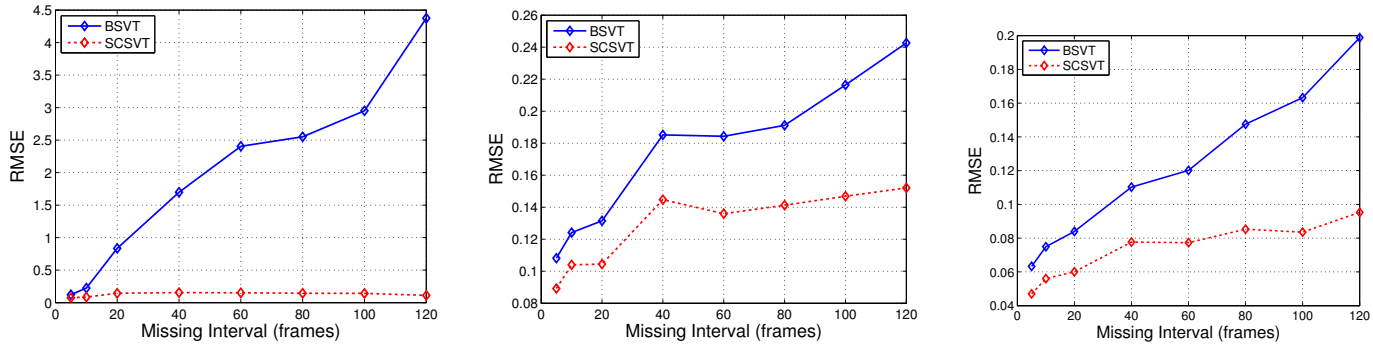
The processing time for SCSVT is strongly dependent on the number of frames of the mocap sequence, weakly dependent on the error rate, and generally not dependent on the missing interval length, as shown in Fig. 6 and Fig. 7. In both figures, the error rate is $\frac{n}{31}$, where n is the number of missing joints. This observation can be explained from the operations listed in Algorithm 1. When the error rate $1 - \frac{m}{n_1 n_2}$ is increased, the dimension of $\mathbf{y_c} \in \mathbb{R}^{3p}$ and $\mathbf{C} \in \mathbb{R}^{3p \times n_2}$ increases while the dimension of $\mathbf{y} \in \mathbb{R}^m$ decreases. Thus the resulting processing time increase is largely due to the projection operation (step 6 in Algorithm 1). The most time consuming operation by far is the singular value decomposition with a complexity of $O(n_2 n_1^2)$. It scales linearly with the number of frames n_2, and is independent of the error rate. The fluctuations across varying interval lengths, especially for a very short sequence like 85_02, can be attributed to the experimental setup; for each experiment, each sequence is truncated to multiples of missing interval length.

Compared to BSVT, the computation of SCSVT requires the projection operation and operations associated with $\mathbf{y_c}$ and $\mathbf{C}$. Since the singular value decomposition operation is the same for both BSVT and SCSVT, the complexity for each

[2]All mocap data values in this paper are expressed in units from the CMU mocap database. To obtain values in millimetres, multiply them by 56.44



(a) '14_01' (Boxing) (b) '135_11' (Kicking) (c) '49_02' (Jumping)



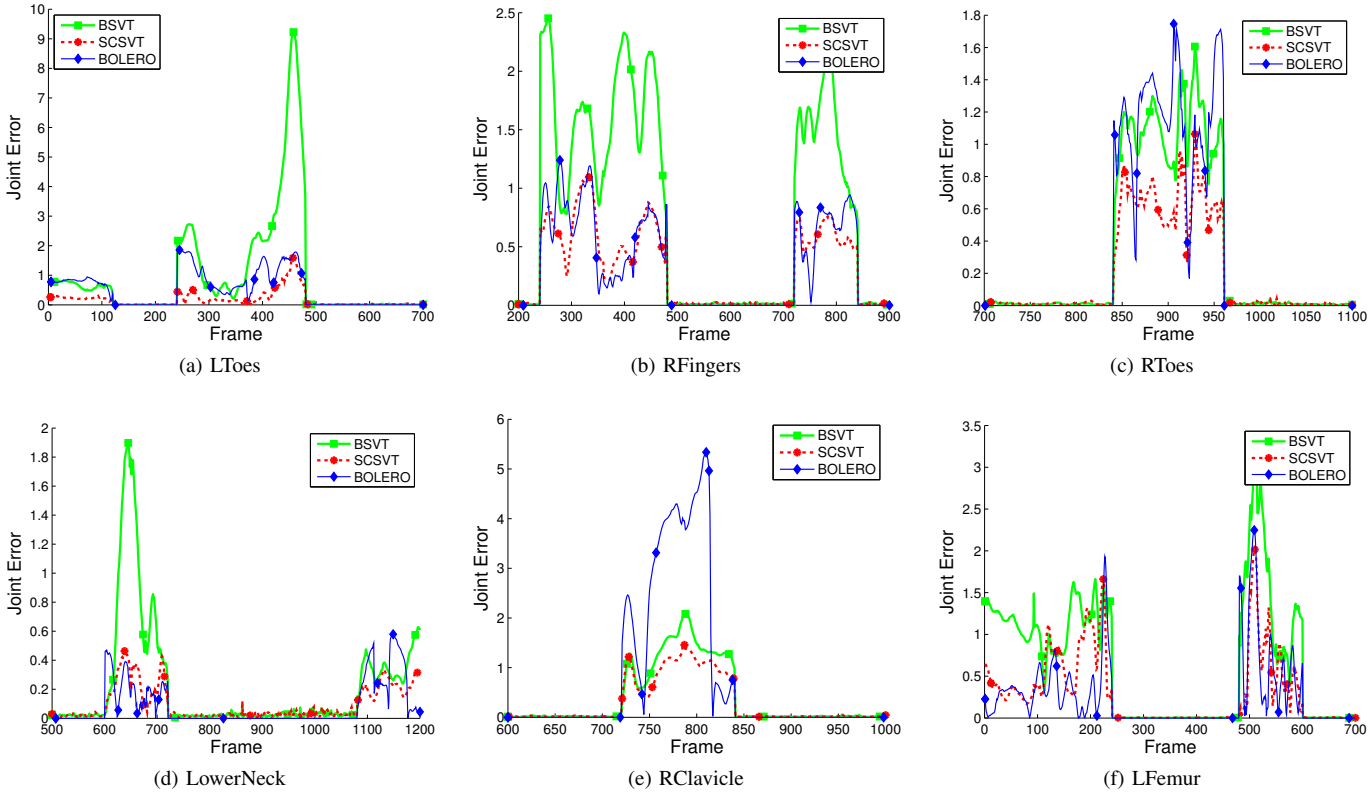
(d) '85_02' (Jumptwist) (e) '135_02' (MartialArts) (f) '61_05' (Salsa)

Fig. 3. Skeleton RMS error for BSVT and SCSVT for 3 missing joints.

TABLE I

PERFORMANCE COMPARISON IN TERMS OF RMSE OF RECOVERED MOCAP DATA BEWEEN (A) BSVT, (B) THE PROPOSED SCSVT MOCAP RECOVERY, AND (C) BOLERO. n AND l DENOTE THE NUMBER OF MISSING JOINTS AND INTERVAL LENGTH RESPECTIVELY.

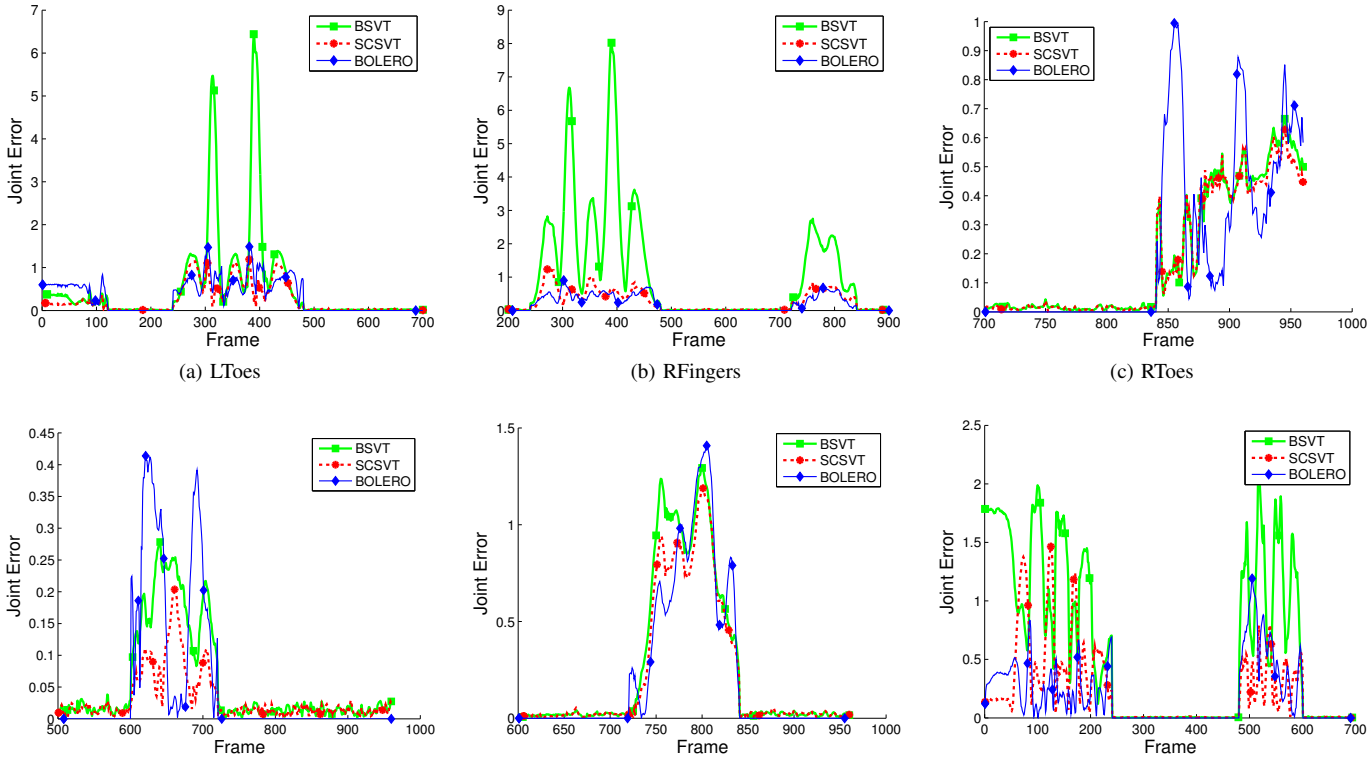
Sequence		n=3, l=15	n=3, l=30	n=3, l=60	n=3, l=90	n=3, l=120	n=6, l=15	n=6, l=30	n=6, l=60	n=6, l=90	n=6, l=120
14_01	A	0.052	0.051	0.053	0.056	0.061	0.098	0.118	0.138	0.128	0.140
(Boxing)	B	**0.046**	**0.043**	**0.045**	**0.045**	**0.048**	**0.085**	**0.104**	**0.118**	**0.102**	**0.105**
	C	0.068	0.066	0.068	0.063	0.072	0.100	0.112	0.143	0.112	0.130
85_02	A	0.281	0.613	1.563	2.038	2.499	0.716	1.119	2.204	3.060	3.745
(Jumptwist)	B	**0.120**	0.191	0.230	**0.204**	**0.249**	**0.248**	**0.315**	**0.354**	**0.485**	**0.321**
	C	0.145	**0.154**	**0.214**	0.266	0.326	0.290	0.340	0.366	0.538	0.397
143_04	A	0.075	0.136	0.510	0.380	1.261	0.156	0.340	1.148	1.634	3.280
(Run fig 8)	B	0.059	0.088	**0.124**	**0.128**	0.155	0.117	0.193	0.220	**0.270**	**0.290**
	C	**0.057**	**0.062**	0.155	0.135	**0.141**	**0.105**	**0.143**	**0.174**	0.314	0.403
49_02	A	0.046	0.065	0.111	0.091	0.238	0.128	0.182	0.239	0.255	0.423
(Jumping)	B	**0.039**	**0.053**	0.086	0.068	0.093	0.119	0.140	**0.132**	0.146	**0.153**
	C	0.057	0.058	**0.074**	**0.064**	**0.079**	**0.103**	**0.099**	0.139	**0.133**	0.233
135_02	A	0.097	0.120	0.168	0.166	0.209	0.201	0.246	0.404	0.428	0.453
(MartialArts)	B	**0.083**	**0.096**	**0.139**	**0.131**	**0.151**	**0.181**	**0.211**	0.333	**0.247**	**0.280**
	C	0.138	0.144	0.144	0.161	0.195	0.216	0.228	**0.287**	0.266	0.282
135_11	A	0.057	0.071	0.144	0.261	0.288	0.132	0.180	0.326	0.500	0.608
(Kicking)	B	**0.052**	0.065	0.095	**0.095**	0.104	0.125	0.150	**0.184**	0.160	0.157
	C	0.054	**0.058**	**0.081**	0.099	**0.074**	**0.092**	**0.102**	0.383	**0.154**	**0.114**
61_05	A	0.070	0.086	0.102	0.167	0.174	0.159	0.206	0.233	0.390	0.344
(Salsa)	B	**0.060**	**0.063**	**0.080**	**0.100**	**0.102**	0.137	0.168	**0.144**	0.185	**0.179**
	C	0.073	0.085	0.086	0.100	0.112	**0.123**	**0.145**	0.156	**0.151**	0.209
88_04	A	0.131	0.202	0.274	0.342	0.347	0.363	0.617	0.576	1.036	0.949
(Acrobatics)	B	**0.101**	**0.123**	**0.146**	**0.147**	**0.141**	**0.265**	0.432	**0.317**	0.314	**0.261**
	C	0.155	0.143	0.201	0.178	0.177	0.270	**0.286**	0.341	**0.302**	0.297



(a) LToes (b) RFingers (c) RToes

(d) LowerNeck (e) RClavicle (f) LFemur

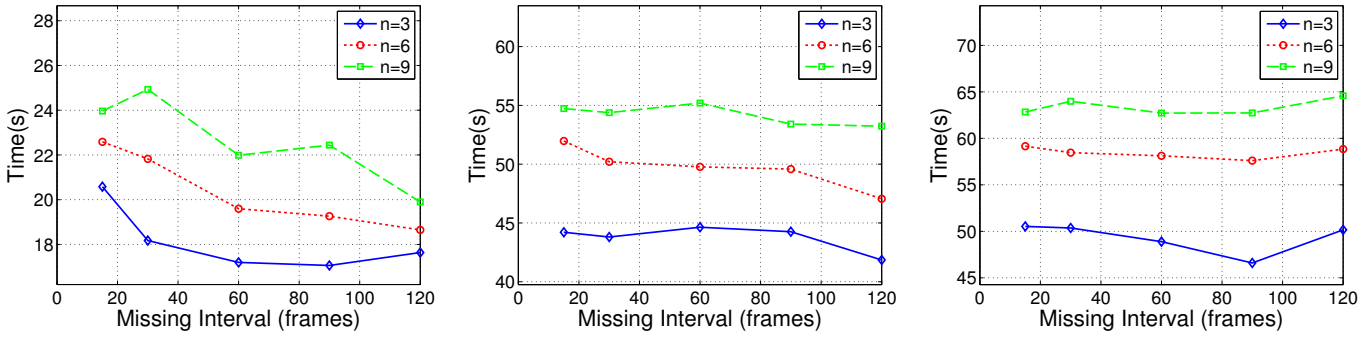
Fig. 4. Joint error for sequence 88_04



(a) LToes (b) RFingers (c) RToes

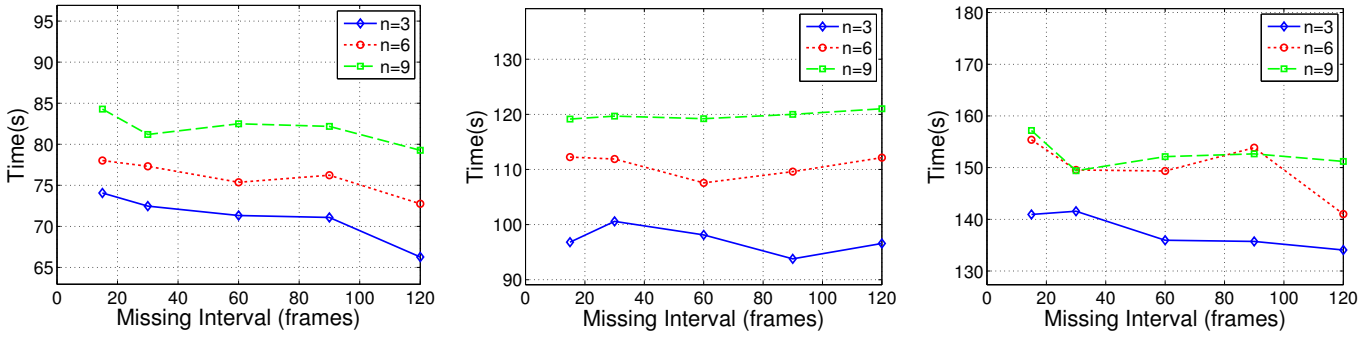
(d) LowerNeck (e) RClavicle (f) LFemur

Fig. 5. Joint error for sequence 49_02



(a) '85_02' (Jumptwist) (b) '49_02' (Jumping) (c) '135_11' (Kicking)

Fig. 6. Processing time of SCSVT for varying interval lengths and number of missing joints n for three short sequences.



(a) '61_05' (Salsa) (b) '135_02' (MartialArts) (c) '14_01' (Boxing)

Fig. 7. Processing time of SCSVT for varying interval lengths and number of missing joints n for three long sequences.

iteration is the same for both algorithms; the processing time is somewhat higher for SCSVT, but not greatly so. However, SCSVT has a lower δ threshold for convergence than BSVT. With a lower value of δ, SCSVT requires many more iterations to converge and hence requires a longer processing time.

We compare the processing times of SCSVT($\delta = 0.75$) against that of BSVT($\delta = 0.75$), BSVT($\delta = 1.5$) and BOLERO in Table II for various sequences for three missing joints. The average processing time over interval lengths of {15, 30, 60, 90, 120} are shown. BSVT($\delta = 1.5$) is twice as fast as BSVT($\delta = 0.75$), which is in turn up to 25% faster than SCSVT. Nevertheless, SCSVT is 3 to 11 times faster than BOLERO.

Fig. 8 and Fig. 9 show how the compared algorithms converge for three missing joints. Both BSVT and SCSVT converge smoothly, and the convergence rate is independent of interval length l. BSVT converges faster than SCSVT, hence it has a lower RMS error initially, but eventually the RMS error of SCSVT decreases to a smaller value. BOLERO converges more slowly than both BSVT and SCSVT. Generally, convergence time of BOLERO is also longer for longer l.

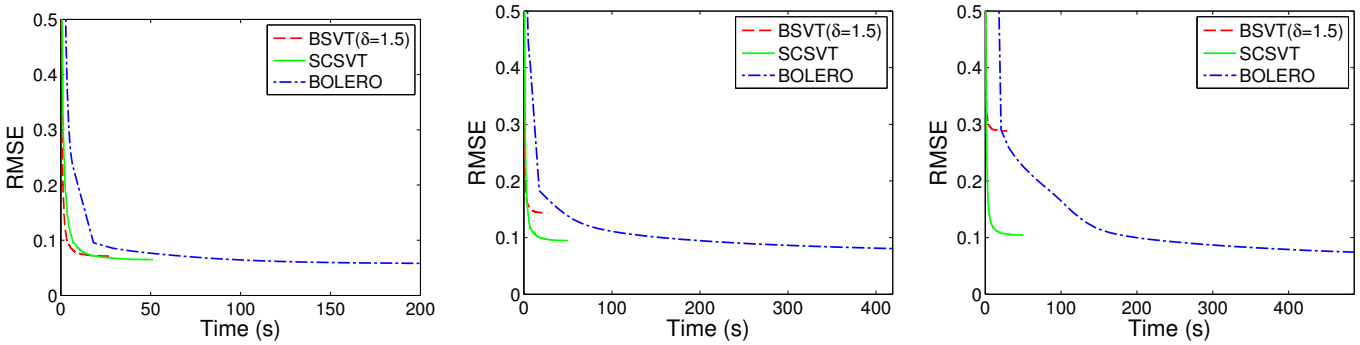
TABLE II
AVERAGE PROCESSING TIME (SECONDS) OF (A1) BSVT($\delta = 1.5$), (A2) BSVT($\delta = 0.75$), (B) THE PROPOSED SCSVT, AND (C) BOLERO FOR 3 MISSING JOINTS. N DENOTES THE NUMBER OF FRAMES IN A SEQUENCE.

Sequence	N	A1	A2	B	C
85_02	405	8	15	18	164
143_04	543	10	21	25	291
49_02	1043	19	38	44	156
135_11	1223	21	42	49	438
88_04	1240	21	40	46	266
61_05	1678	29	57	71	241
135_02	2601	39	78	97	367
14_01	2797	55	110	138	480

C. Discussion

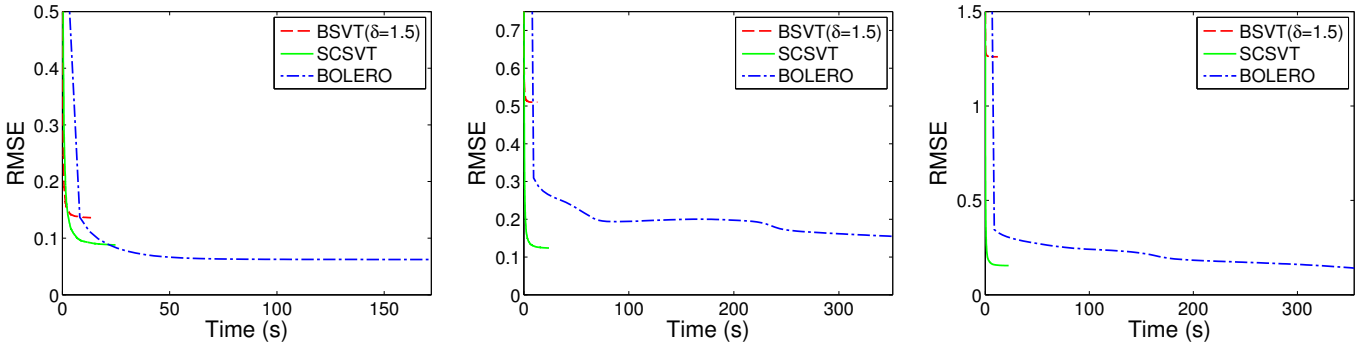
It is worth noting that SCSVT, as well as BSVT and BOLERO, do not rely on prior motions or pre-trained dictionaries. Hence, compared to machine learning based methods such as [20], [27], [3], our proposed method has fewer limitations. Machine learning based methods usually require the processed motion and prior motions to have the same skeleton model. The performance of these methods are also reliant on the similarity of prior motions and processed motions. If the processed motion is captured from an actor with a different size, and if there are no prior motions of the same type as the processed motion, recovery performance might be degraded significantly.

With regard to the skeleton constraints in SCSVT, it is likely that the performance of SCSVT can be improved by tightening



(a) $l = 30$ (b) $l = 60$ (c) $l = 120$

Fig. 8. RMS error for sequence 135_11 (Kicking) at different interval lengths l for 3 missing joints.



(a) $l = 30$ (b) $l = 60$ (c) $l = 120$

Fig. 9. RMS error for sequence 143_04 (Run fig 8) at different interval lengths l for 3 missing joints.

the bound of the constraint in Eq. (8), $\|\mathbf{C}_i\mathbf{x}\|_2 \leq d_i$, to

$$d_i - \varepsilon \leq \|\mathbf{C}_i\mathbf{x}\|_2 \leq d_i + \varepsilon \tag{27}$$

In other words, each bone is constrained to a certain length d_i with tolerance ε, instead of being constrained to have length of less than d_i. However, the constraint in Eq. (8) is convex; this means that the optimization problem is still convex and so retains the advantages of convex optimization, the most pertinent is that the global optimum can be found efficiently. Applying the constraint in Eq. (27) changes the problem into a non-convex and non-linear problem which is generally not tractable. In contrast to SCSVT, BOLERO enforces the constraint in Eq. (27) which is tighter, but requires a much longer processing time than SCSVT. Nevertheless, even with skeleton constraints with looser bound, SCSVT still improves on BSVT by a large extent, and its overall performance is better than BOLERO.

VI. Conclusion

Low rank matrix completion methods such as SVT works well in recovering mocap data with missing values. However, when mocap joints are missing for long intervals, the performance of SVT decreases substantially and the positions of certain recovered joints deviate drastically from their true positions. We alleviate this issue by constraining the inter-joint distances of adjacent joint pairs in the SVT method. The constraints limit the maximum error of recovered joints and also reduce the error of recovered joints on average. Experiments show that our proposed skeleton constrained SVT method (SCSVT) improves on the basic SVT method (BSVT) by 40% on average, with higher gains at longer missing joint intervals. Compared to the recent state of the art algorithm [17], on average our method offers 4% better performance and is 3 to 11 times faster.

References

[1] Aristidou, A., Cameron, J., Lasenby, J.: Real-time estimation of missing markers in human motion capture. In: The 2nd International Conference on Bioinformatics and Biomedical Engineering, 2008. ICBBE 2008, pp. 1343–1346 (2008)

[2] Aristidou, A., Lasenby, J.: Real-time marker prediction and CoR estimation in optical motion capture. The Visual Computer **29**(1), 7–26 (2013)

[3] Baumann, J., Krüger, B., Zinke, A., Weber, A.: Data-driven completion of motion capture data. Proceedings of Workshop on Virtual Reality Interaction and Physical Simulation (VRIPHYS) (2011)

[4] Cai, J.F., Candès, E.J., Shen, Z.: A singular value thresholding algorithm for matrix completion. SIAM J. on Optimization **20**(4), 1956–1982 (2010)

[5] Candès, E., Recht, B.: Exact matrix completion via convex optimization. Foundations of Computational mathematics **9**(6), 717–772 (2009)

[6] Chai, J., Hodgins, J.K.: Performance animation from low-dimensional control signals. ACM Trans. Graph. **24**(3), 686–696 (2005)

[7] CMU: CMU graphics lab motion capture database. http://mocap.cs.cmu.edu/ (2012)

[8] Dorfmüller-Ulhaas, K.: Robust optical user motion tracking using a kalman filter. In: 10th ACM Symposium on Virtual Reality Software and Technology (2003)

[9] Grochow, K., Martin, S.L., Hertzmann, A., Popović, Z.: Style-based inverse kinematics. In: ACM SIGGRAPH 2004 Papers, SIGGRAPH '04, p. 522–531. ACM, New York, NY, USA (2004)

[10] Herda, L., Fua, P., Plankers, R., Boulic, R., Thalmann, D.: Skeleton-based motion capture for robust reconstruction of human motion. In: Computer Animation 2000. Proceedings, pp. 77 –83 (2000)

[11] Herda, L., Fua, P., Plänkers, R., Boulic, R., Thalmann, D.: Using skeleton-based tracking to increase the reliability of optical motion capture. Human Movement Science **20**(3), 313–341 (2001)

[12] Hornung, A., Sar-Dessai, S., Kobbelt, L.: Self-calibrating optical motion tracking for articulated bodies. In: IEEE Virtual Reality, 2005. Proceedings. VR 2005, pp. 75–82 (2005)

[13] Hou, J., Chau, L.P., He, Y., Chen, J., Magnenat-Thalmann, N.: Human motion capture data recovery via trajectory-based sparse representation. In: Proc. IEEE International Conference on Image Processing (ICIP) (2013)

[14] Krüger, B., Tautges, J., Weber, A., Zinke, A.: Fast local and global similarity searches in large motion capture databases. In: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '10, p. 1–10. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2010)

[15] Lai, R., Yuen, P., Lee, K.: Motion capture data completion and denoising by singular value thresholding. In: Eurographics 2011-Short Papers, pp. 45–48 (2011)

[16] Lawrence, N.: Gaussian process latent variable models for visualisation of high dimensional data. In: S. Thrun, L. Saul, B. {schölkopf} (eds.) Advances in Neural Information Processing Systems 16. MIT Press (2004)

[17] Li, L., McCann, J., Pollard, N., Faloutsos, C.: BoLeRO: a principled technique for including bone length constraints in motion capture occlusion filling. In: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '10, p. 179–188. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2010)

[18] Li, L., McCann, J., Pollard, N.S., Faloutsos, C.: DynaMMo: mining and summarization of coevolving sequences with missing values. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 507–516. ACM, New York, NY, USA (2009)

[19] Liu, G., McMillan, L.: Estimation of missing markers in human motion capture. The Visual Computer **22**(9-11), 721–728 (2006)

[20] Lou, H., Chai, J.: Example-based human motion denoising. IEEE Transactions on Visualization and Computer Graphics **16**(5), 870 –879 (2010)

[21] Papadimitriou, S., Brockwell, A., Faloutsos, C.: Adaptive, hands-off stream mining. In: Proceedings of the 29th international conference on Very large data bases - Volume 29, VLDB '03, pp. 560–571. VLDB Endowment (2003)

[22] Piazza, T., Lundström, J., Kunz, A., Fjeld, M.: Predicting missing markers in real-time optical motion capture. In: Proceedings of the 2009 international conference on Modelling the Physiological Human, p. 125–136. Springer-Verlag (2009)

[23] Tan, C.H., Hou, J., Chau, L.P.: Human motion capture data recovery using trajectory-based matrix completion. Electronics Letters **49**(12), 752–754 (2013)

[24] Taylor, G.W., Hinton, G.E., Roweis, S.: Modeling human motion using binary latent variables. In: Advances in Neural Information Processing Systems, p. 1345–1352. MIT Press (2007)

[25] Wang, J., Fleet, D., Hertzmann, A.: Gaussian process dynamical models for human motion. IEEE Transactions on Pattern Analysis and Machine Intelligence **30**(2), 283–298 (2008)

[26] Wiley, D., Hahn, J.: Interpolation synthesis for articulated figure motion. In: Virtual Reality Annual International Symposium, 1997., IEEE 1997, pp. 156–160 (1997)

[27] Xiao, J., Feng, Y., Hu, W.: Predicting missing markers in human motion capture using $l1$-sparse representation. Computer Animation and Virtual Worlds **22**(2-3), 221–228 (2011)

[28] Yi, B.K., Sidiropoulos, N., Johnson, T., Jagadish, H., Faloutsos, C., Biliris, A.: Online data mining for co-evolving time sequences. In: 16th International Conference on Data Engineering, 2000. Proceedings, pp. 13–22 (2000)

[29] Yi, P., Zhang, Q., Wei, X.: Laplacian coordinates-based motion transition for data-driven motion synthesis. IET Image Processing **6**(9), 1331–1337 (2012)

[30] Zhou, F., De la Torre, F., Hodgins, J.: Hierarchical aligned cluster analysis for temporal clustering of human motion. Pattern Analysis and Machine Intelligence, IEEE Transactions on **35**(3), 582–596 (2013)