M.S. THESIS

A Pseudo-Sample Based Low Cost 3D Human Motion Capture System Using Wireless Camera Sensor Network

카메라 센서 네트워크를 이용한 pseudo-sample 기반 저비용 모션캡쳐 시스템

BY

Hyeongseok Oh

FEBRUARY 2013

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

M.S. THESIS

A Pseudo-Sample Based Low Cost 3D Human Motion Capture System Using Wireless Camera Sensor Network

카메라 센서 네트워크를 이용한 pseudo-sample 기반 저비용 모션캡쳐 시스템

BY

Hyeongseok Oh

FEBRUARY 2013

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

A Pseudo-Sample Based Low Cost 3D Human Motion Capture System Using Wireless Camera Sensor Network

카메라 센서 네트워크를 이용한 pseudo-sample 기반 저비용 모션캡쳐 시스템

지도교수 오 성 회

이 논문을 공학석사 학위논문으로 제출함

2013년 2월

서울대학교 대학원

전기 컴퓨터 공학부

오 형 석

오형석의 공학석사 학위 논문을 인준함

2013년 2월

위 원 장: _____

부위원장: _____

위 원: _____

Abstract

There is a growing interest in 3D contents in the global market following the recent developments in 3D movies, 3D TVs, and 3D smartphones. However, the 3D content creation is still dominated by professionals due to the high cost of 3D motion capture instruments. The availability of a low-cost motion capture system will promote 3D content generation by general users and accelerate the growth of the 3D market. In this paper, we propose a real-time low-cost motion capture system, called Samba. The Samba motion capture system is based on a portable low-cost wireless camera sensor network and it can reconstruct accurate 3D full-body poses at 16 frames per second using only eight markers on the subject body. The processing times on camera motes are significantly reduced by using a smaller number of markers. A pseudo-pose based data-driven 3D human pose reconstruction method is proposed to reduce the required computation time and to improve the 3D reconstruction accuracy. The performance of Samba is evaluated extensively in experiments.

keywords: Camera sensor networks, Pseudo-samples, Motion capture system
student number: 2011-20878

Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

We are witnessing rapidly growing 3D industries, such as 3D movies, 3D TVs, and 3D smartphones. It is expected that the consumption of 3D contents will be further increased as more 3D products are introduced into the market. While there are many ways to consume 3D contents, 3D content generation is still an expensive task and remained in the hands of professionals. As we have seen from the analog to digital camera conversion in early 2000, an inexpensive way to create photos and videos has revolutionized the industry and the society. As common users can create multimedia contents with inexpensive cameras, the demand for digital cameras has become greater, further lowering prices of hardware for content generation and consumption. In addition, the introduction of digital cameras has also revolutionized the Internet as creation and sharing of photos and videos have been accelerated with help from social network sites such as YouTube, Flickr, and Facebook. Based on these facts, it can be concluded that the success of the 3D market depends on the availability of an inexpensive tool, with which a common user can easily create 3D contents. The present paper proposes a low-cost approach to 3D content generation.

There are two major approaches to create 3D contents: stereoscopy and motion capture systems. Stereoscopy uses a stereo camera to take images at different view points of the scene. Two sets of images are separately projected to the left and the right

eye of the viewer, giving an illusion of depth in the brain. Since stereoscopy does not reconstruct the scene in 3D, it cannot be used as a general 3D content generation tool. On the other hand, motion capture systems, such as the Vicon MX motion capture system [2], can provide a full 3D pose of the subject using a set of markers. They are commonly used in the film and gaming industry for creating realistic and complex 3D motions. While highly accurate 3D motions can be generated, motion capture systems are still too expensive for common users. For example, a typical entry-level motion capture system set with a minimum of three cameras costs more than $ 12,000. Hence, there is no accessible instrument for general 3D content creation for common users yet.

A wireless camera sensor network consists of inexpensive camera motes with wireless connectivity and has been applied to a number of applications, such as surveillance, environmental monitoring, traffic modeling, human behavior understanding, human pose estimation, assisted living, and sports (see [28] and references therein). In this paper, we propose a human motion capture system using a wireless camera sensor network. An overview of the proposed system is shown in Figure 1.1.

The name of the proposed motion capture system is *Samba*. The Samba motion capture system consists of a wireless camera sensor network, a server, and an optional display. Note that a server can be a notebook computer or a smart device as in [28] to make the overall system more portable. A human subject wears markers on her joints and performs a series of motions in front of a wireless camera sensor network. Each camera mote in a wireless camera sensor network detects markers and wirelessly transmits marker locations to the Samba server. The server then reconstructs the 3D motion using received marker positions and a human pose database. By using a wireless camera sensor network, we have developed a portable personal human motion capture system. A user can easily carry and deploy camera motes to collect 3D contents. Also a minimal number of markers are used to make motion capture more convenient for common users. The Samba motion capture system can track moving marker sets and

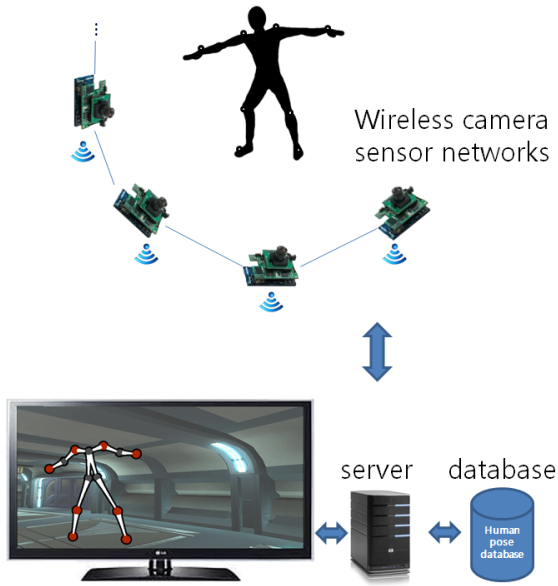


Figure 1.1: An overview of the Samba motion capture system.

display 3D motion in real-time. Currently, the system runs at 16 frames per second.

Samba is a data-driven motion capture system which utilizes a human pose database to reliably reconstruct human poses from noisy and missing observations from camera motes. A human pose is reconstructed using the local pose space which is consistent with the current observation. We propose a pseudo-pose based method to better represent the local pose space with a dense set of data points. In experiments, the reconstruction error is reduced by 14% with a reduced computation time when pseudo-poses are used to represent the local pose space.

A newly introduced Kinect depth camera from Microsoft is another motion capture system. Kinect uses an infrared laser projector and a camera to construct a depth image. However, the depth information from Kinect is not accurate and it can easily fail to correctly construct a human pose. The proposed data-driven approach can be applied to Kinect to improve reconstruction accuracy.

The remainder of this paper is organized as follows. Related work in 3D motion capture is discussed in Section 2. The overview of the proposed Samba motion cap-

ture system is described in Section 3. In Section 4, the data-driven 3D human pose estimation method is presented. The system implementation of Samba is described in Section 5. The experimental results of the Samba motion capture system are presented in Section 6.

Chapter 2

Related works

A motion capture system is used for a wide range of applications, including sports, medicine, advertising, law enforcement, human-robot interaction, manufacturing, surveillance, and entertainment [22, 24]. A number of different methods have been developed for capturing human motions.

Wei and Chai reconstructed 3D human poses from uncalibrated monocular images in [5]. They assumed that all positions of joints from images were known and the camera was placed far from the human subject. These assumptions are based on the reconstruction method for an articulated object by Taylor [4]. In [4], the author formulated the 3D human pose reconstruction problem as an optimization problem using three sets of constraints: bone projection, bone symmetry, and rigid body constraints. Magnus Burenius et al. [6] developed a new bundle adjustment method for 3D human pose estimation using multiple cameras. Their method is similar to [5] but temporal smoothness constraints are added and spline regression is used to impose weak prior assumptions on human motion. All three methods require known positions of joints and lengths between pairs of joints. Since it is not possible to reliably detect all markers autonomously from images due to self-occlusion, they are not applicable for a practical motion capture system.

Multiple cameras have been used to reconstruct 3D human poses using markers.

In [31], four colored markers are used for extracting joints from two cameras. The locations of other joints are estimated using four marker positions and a silhouette of the subject. While it provides a low-cost solution, it cannot be run in real-time and the reconstruction error is large to be used in practice. In [32], an optical motion capture system with pan-tilt cameras is proposed. The proposed motion capture system runs in real-time and labels markers automatically. However, the system requires a set of pan-tilt cameras and computers. While the system is cheaper than commercial optical motion capture systems, it is still too expensive for common users.

It is also possible to estimate 3D human poses from images without markers. For example, markerless methods include silhouettes [15, 19], an articulated deformable model [12], articulated surfaces [21], and the volume data representation [30]. In [15], silhouettes are extracted from images and the likelihood of joint angles is computed. The human pose is constructed by estimating joint angles using the maximum a posteriori criterion. In [12], a 3D volume of an object is reconstructed from silhouettes extracted from multiple video streams. Then parameters such as rotations and translations between cameras with respect to the center of the human subject are estimated, along with joint angles, in order to reconstruct the 3D human pose. In [21], human motion is tracked by fitting articulated surfaces to 3D cloud points and surface normals. In [30], voxels are computed from multiple images and skeleton poses are estimated from voxels. They fitted a skeleton model to voxels by fitting each bone of a skeleton to a corresponding set of voxels. However, a markerless method is not appropriate for real-time motion capture system since it is not accurate compared to marker-based methods and it requires the computation of silhouettes from multiple images, which cannot be done reliably in real-time.

The depth information can be used for 3D human pose estimation [8, 9]. In [8], the authors developed a nonlinear optimization method based on the relationship between joint angles and an observed pose from a depth image for human motion estimation. In [9], notable image features, such as a head and a torso, are extracted from depth and

color images and 3D human poses are estimated using constrained optimization based on constraints among extracted image features.

In [20], a 3D human posture reconstruction system using a wireless camera network was presented. The method detected five-upper body parts (a head, two shoulders, and two wrists) and reconstructed a upper body in 3D. While our proposed system is similar to [20], in our system, the motion capture data is used to improve the accuracy without compromising computational cost and 3D reconstruction of a full body is possible in Samba.

Chapter 3

An Overview of the Samba Motion Capture System

The Samba motion capture system is based on a wireless camera sensor network (see Figure 1.1). A wireless camera sensor network consists of a number of camera motes (e.g., a CITRIC camera mote [27]). Each camera mote has a camera, on-board processor for image processing, and wireless connectivity. We assume that a wireless camera sensor network has been deployed and cameras are calibrated in advance as described in [28].

The goal of the Samba motion capture system is to reconstruct 3D poses of a human subject based on images taken from multiple camera motes with different view points. We represent a human pose using 13 joints as shown in Figure 3.1. The 13 joints include a left shoulder, a right shoulder, a left elbow, a right elbow, a left wrist, a right wrist, a left hip joint, a right hip joint, a left knee, a right knee, a left foot, and a right foot.

The Samba motion capture system is a data-driven 3D human pose reconstruction system to improve the accuracy and robustness. A 3D human pose is reconstructed based on input images (i.e., 2D marker positions) and the human pose database, which consists of publicly available human motion data sets and human poses collected using Samba. Each pose in our database is represented as a 39 dimensional pose vector (13 joints $\times$ 3 dimensions).

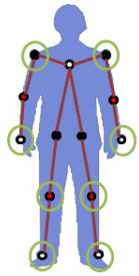


Figure 3.1: A 13-joint human skeleton model. The eight markers used by the Samba motion capture system are indicated with larger circles around the joint. Red colored joints have one degree of freedom and black colored joints have two degrees of freedom.

The overall computational flow of the Samba motion capture system is given in Figure 3.2. While a human pose is reconstructed using 13 joint positions, the system uses only eight markers and the locations of those eight markers are circled in Figure 3.1. The use of a reduced number of markers makes it more convenient for general users and we can also reduce the computation time on camera motes. From an extensive experimental study, we have found that eight is the minimum number of marker locations which can provide the full 3D reconstruction of 13 joints with almost no loss.

The Samba motion capture system operates as follows. Each camera mote tracks eight markers in its image frame and transmits locations of markers to a server at fixed time intervals. When the server receives marker locations from camera motes, it computes 3D positions of markers using multi-view geometry based on the camera network calibration parameters computed during the calibration step (for more detail, see [28]). The computed 3D markers are aligned to be compared with 3D poses in the human pose database. From the human pose database, K poses which are close to the computed 3D marker locations are selected. We then generate pseudo-poses from the K selected poses to form a local pose space. We first reduce the dimensionality of the local pose space using the principal component analysis (PCA) to reduce the

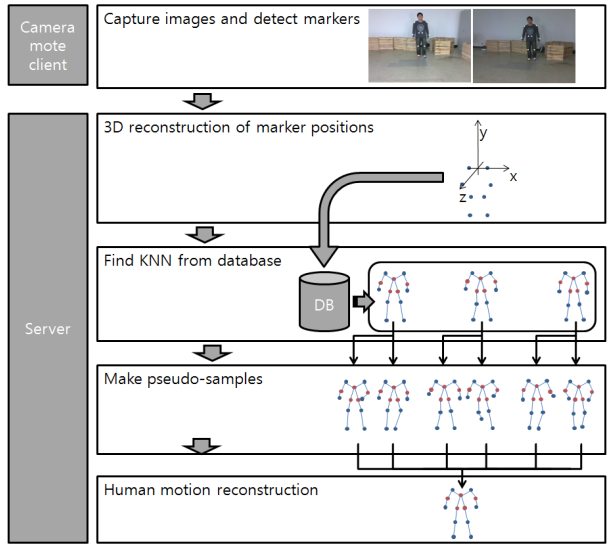


Figure 3.2: A flowchart of the Samba motion capture system based on a wireless camera sensor network.

computation time and noise.

Chapter 4

Data-Driven Human Pose Estimation

The use of a local, lower dimensional representation of a human pose gives a better pose estimation by eliminating ambiguity [1]. The local pose space can be constructed using neighboring poses in the human pose database. In addition, the computation time required for the optimization routine can be reduced since lower dimensional representations of human poses are used. However, the method suffers when data points in the human pose database are clustered or sparse. Since we cannot have a complete human pose database with dense data points in practice due to a vast number of possible poses, it is difficult to use the method from [1] for diverse human poses. We address this issue in Samba by generating pseudo-poses around the current pose in order to better represent the local human pose space. Pseudo-poses are randomly generated based on poses in the database while maintaining human joint constraints.

We first describe the local pose space approach which is adapted from [1] and then explain how pseudo-poses are generated in Samba to better represent local pose spaces.

4.1 Local Pose Space Method

Suppose that i is the current frame number. From images of at least two different view points, a 3D joint marker vector $x_i \in \mathbb{R}^{24}$ (8 markers $\times$ 3 dimensions) is computed and K poses $\{q_i^{(k)} | 1 \leq k \leq K\}$ that are close to the computed marker positions are selected from the database. Note that not all eight markers are required to select K neighboring poses. For each pose $q_i^{(k)}$, we randomly generate N pseudo-poses as described in the next section. Let $Q_i = \{q_i^{(k,n)} | 1 \leq k \leq K, 1 \leq n \leq N\}$ be a set of KN pseudo-poses. Pose vectors in Q_i form a local pose space for the current pose.

Let $p_i \in \mathbb{R}^{39}$ (13 marker positions $\times$ 3 dimensions) be the mean of Q_i. The PCA is applied to the covariance matrix Λ_i of Q_i to obtain an estimate of the current pose vector $\bar{q}_i \in \mathbb{R}^{39}$ as follows:

$$\bar{q}_i = p_i + U_i w_i, \tag{4.1}$$

where $w_i \in \mathbb{R}^n$ is a low dimensional representation of $\bar{q}_i$ and $U_i \in \mathbb{R}^{39 \times n}$ is a projection matrix. The vector w_i is constructed from n largest eigenvalues of the covariance matrix Λ_i and columns of U_i are the corresponding eigenvectors.

In order to recover the low dimensional current pose, we solve the following energy minimization problem.

$$w_i^* = \arg\min_{w_i} \; E_p(w_i) + \alpha E_e(w_i) + \beta E_s(w_i), \tag{4.2}$$

where E_p, E_e, and E_s are energy terms and α and β are weighting parameters. E_p measures the deviation from the distribution of local poses, E_e measures the reconstruction error, and E_s measures the smoothness of motion. These three energy terms are described below.

The prior term (E_p): We assume that a pose in the local pose space is distributed according to a multivariate Gaussian distribution as follows:

$$P(\bar{q}_i | Q_i) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Lambda_i|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} (\bar{q}_i - p_i)^T {\Lambda_i}^{-1} (\bar{q}_i - p_i) \right), \tag{4.3}$$

where $d = 39$, the dimension of $\bar{q}_i$, and p_i and Λ_i are the mean and covariance matrix of pseudo-poses in Q_i. We desire to find $\bar{q}_i$ which is consistent with KN pseudo samples, hence, $\bar{q}_i$ maximizes the density function. Maximizing the density function is equivalent to minimizing the following energy function.

$$\begin{aligned} E_p(w_i) &= (\bar{q}_i - p_i)^T \Lambda_i^{-1} (\bar{q}_i - p_i) \\ &= (U_i w_i)^T \Lambda_i^{-1} (U_i w_i) \end{aligned} \tag{4.4}$$

The reconstruction error term (E_e): The reconstruction error term measures the distance between detected marker locations and marker locations from the estimated pose.

$$E_m(w_i) = \|x_i - f(\bar{q}_i)\|, \tag{4.5}$$

where $f(q)$ is a function which extracts eight marker positions from the pose q.

The smoothness term (E_s): E_s measures the smoothness of the estimated pose from two previous estimated poses $\bar{q}_{i-1}$ and $\bar{q}_{i-2}$ to make the current pose consistent with previous estimates and defined as follows:

$$E_s(w_i) = \|\bar{q}_i - 2\bar{q}_{i-1} + \bar{q}_{i-2}\|. \tag{4.6}$$

Combining all the energy terms from 4.4, 4.5, and 4.6 and substituting $\bar{q}_i$ using 4.1, the energy minimization problem 4.2 becomes:

$$\begin{aligned} w_i^* &= \arg\min_{w_i} \ (U_i w_i)^T {\Lambda_i}^{-1} (U_i w_i) \\ &+ \alpha \|x_i - f(p_i + U_i w_i)\| \\ &+ \beta \|p_i + U_i w_i - 2\bar{q}_{i-1} + \bar{q}_{i-2}\|. \end{aligned} \tag{4.7}$$

The above optimization problem is solved for w_i^* using the Levenberg-Marquardt method [3] and the current pose $\bar{q}_i$ is computed using 4.1.

4.2 Pseudo-Pose Generation

The success of the local pose space method depends on how well a local pose space is represented by data points from the database. Since it is not practical to collect every possible human pose and store them in a database, we propose the use of pseudo-poses to better represent local pose spaces. Each pseudo-pose is generated randomly using an existing pose from the database while maintaining human motion and joint constraints.

A pseudo-pose is generated by adding a little noise to a joint angle. Since each entry in our human pose database is a joint position vector representing positions of 13 joints in 3D, in order to generate a pseudo-pose, we need to compute joint angles from the joint position vector. Since it is faster to use joint position vectors when estimating the current pose vector and we do not need to compute all joint angles, a joint position vector is a more convenient representation for our human pose database.

There are two types of joints: joints with one degree of freedom (DOF) and joints with two degrees of freedom (see Figure 3.1). In a general human body model, shoulders, hip joints and a back bone have three degrees of freedom. However, we do not consider the twist along the direction of the bone for joints with three DOFs since such twisting causes a large variation. Hence, we treat the joints with three DOFs as joints with two DOFs when generating pseudo-poses. In what follows, we describe how a pseudo-pose is generated for each type of joints.

4.2.1 One-DOF Joints

Let us consider a joint with one DOF formed by joints p_1, p_2, and p_3 (see Figure 4.1). The corresponding joint angle is the angle between two lines $\overrightarrow{p_1p_2}$ and $\overrightarrow{p_2p_3}$. Let $J_1 = \overrightarrow{p_2p_3}$ and $J_2 = \overrightarrow{p_1p_2}$. Then the angle $\angle p_1p_2p_3$ can be computed as follows:

$$\theta = \pi - \arccos\left(\frac{J_1^T J_2}{\|J_1\|\|J_2\|}\right). \tag{4.8}$$

The rotation axis R is

$$R = \frac{J_1 \times J_2}{\|J_1\|\|J_2\|}, \tag{4.9}$$

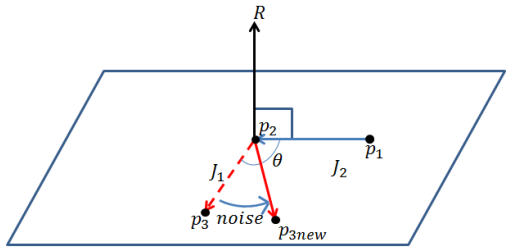


Figure 4.1: Generation of a new joint angle for an one-DOF joint. The joint p_3 is moved to p_{3new} by adding a noise to the joint angle $\angle p_1p_2p_3$. The axis of rotation is R.

where $\times$ is a cross product between two vectors. By rotating p_3 by a small noise with respect to R, we assign a new joint angle $\angle p_1p_2p_3$ to the joint p_2 and obtain p_{3new}.

4.2.2 Two-DOF Joints

Let us consider a joint with two DOFs. J_1, J_2, p_1, p_2 and p_3 are defined the same as Section 4.2.1. We add a small noise to each degree of freedom and obtain a new joint angle for the two-DOF joint.

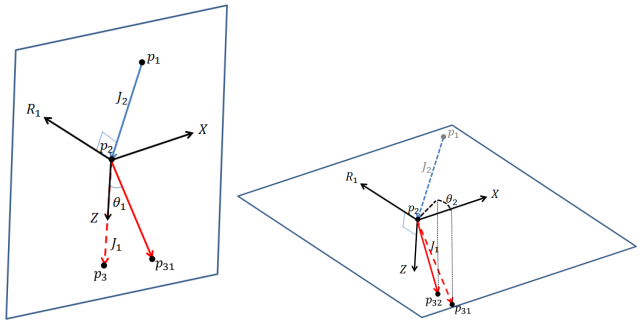
Let Z be a unit vector in the direction of $\overrightarrow{p_2p_3}$ (see Figure 4.2). Let v be a vector from the right shoulder to the left shoulder. We obtain the x-axis of the local coordinate for J_1 based on v by the following equation.

$$X = \frac{v - (v^T Z)Z}{\|v - (v^T Z)Z\|_2}. \tag{4.10}$$

We also obtain the axis of rotation R_1 by $R_1 = Z \times X$.

For simplicity, let us assume that $X = (1, 0, 0)$, $R_1 = (0, 1, 0)$, and $Z = (0, 0, 1)$. Then, $p_3 = (0, 0, L)$, where L is length of $\overrightarrow{p_2p_3}$. We now rotate p_3 with respect to R_1 by θ_1 using the following equation. The resulting new position is p_{31}.

$$p_{31} = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 \\ 0 & 1 & 0 \\ -\sin\theta_1 & 0 & \cos\theta_1 \end{bmatrix} p_3. \tag{4.11}$$



(a) First rotation (b) Second rotation

Figure 4.2: Generation of a new joint angle for a two-DOF joint. The joint p_3 is first moved to p_{31} by adding a noise to the joint angle $\angle p_1 p_2 p_3$ with respect to R_1 (a) and then moved to p_{32} by rotating with respect to Z (b).

Next, we rotate p_{31} by θ_2 with respect to Z using the following equation to obtain p_{32}.

$$p_{32} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} p_{31}. \tag{4.12}$$

For a pose q from the human pose database, we generate N pseudo-poses by adding a small noise to each joint angle as described above. Examples are shown in Figure 4.3. By adding noises to the existing pose, we obtain a set of pose variations which is useful for representing the local pose space. In experiments, we have found that $N = 10$ gives the best tradeoff between the reconstruction error and the computation time.

4.2.3 Initial Values for Optimization

For the faster convergence of the nonlinear optimization problem given in 4.7, we initialize w_i using the linear least squares solution of following problem. Given $Q_i = \{q_i^{(k,n)}\}$, a set of pseudo-samples for the i-th pose, let $r_i^{(k,n)} = f\left(q_i^{(k,n)}\right)$ for $1 \leq k \leq K$ and $1 \leq n \leq N$, where $f(q)$ is a function which extracts eight marker positions

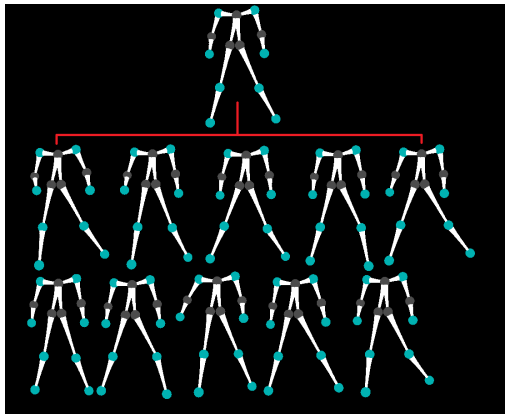


Figure 4.3: Examples of pseudo-poses. The top pose is from the human pose database and the poses in the second and third rows are pseudo-poses generated by the proposed algorithm.

from the pose q. We concatenate $r_i^{(k,n)}$ and construct a matrix R_i as follows.

$$R_i = \left[r_i^{(1,1)}, r_i^{(1,2)}, \ldots, r_i^{(K,N)} \right].$$

R_i is a $24 \times KN$ matrix of eight marker positions of pseudo-poses. Let $\gamma_i \in \mathbb{R}^{KN}$ be a weight vector. We want to find γ_i which minimizes $\|x_i - R_i\gamma_i\|^2$, where x_i is the observed pose vector. The linear least squares solution $\gamma_i^* = (R_i^T R_i)^{-1} R_i^T x_i$ minimizes the error of representing x_i using a linear combination of pseudo-poses. Given γ_i^*, the initial pose is computed as $\bar{q}_i = R_i\gamma_i^*$. We then project $\bar{q}_i$ on to the low dimensional space to obtain the initial value for w_i, which will be used in the nonlinear optimization routine.

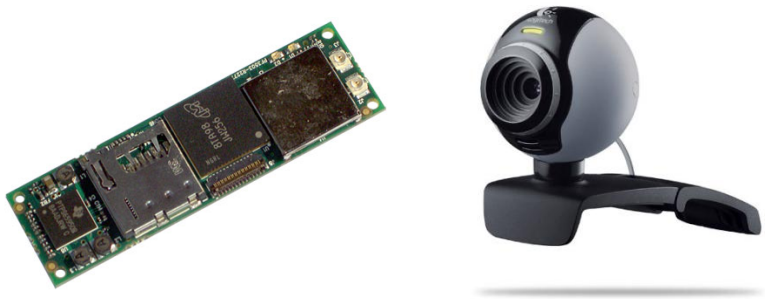
Chapter 5

System Implementation

Our implementation of the Samba motion capture system consists of a server and a wireless camera sensor network with two camera motes. While a larger number of camera motes is possible, we used the minimum possible number of camera motes, which is two, to show the feasibility of Samba. Each camera mote is equipped with a Gumstix Overo Fire computer-on-module [25] and a Logitech C250 webcam. A camera mote runs a client program written using the OpenCV library which detects and tracks markers and transmits marker positions to the Samba server regularly.

5.1 Camera Mote

A combination of a Gumstix Overo Fire computer-on-module and a webcam is used as a camera mote (see Figure 5.1). Gumstix Overo Fire has an ARM Cortex-A8 processor which can be run up to 720 MHz with 512 MB of RAM and 512 MB of ROM. The module can provide wireless connectivity such as Bluetooth and Wi-Fi and it weighs less than 50 g. Compared to other existing camera motes, such as CITRIC [27], the developed Gumstix based mote has the higher processing power and supports floating-point operations, which is required to run computer vision programs using OpenCV. But it is still challenging to process high resolution images using Gumstix in real-time.



(a) (b)

Figure 5.1: (a) Gumstix Overo Fire computer-on-module. (b) Logitech C250 webcam.

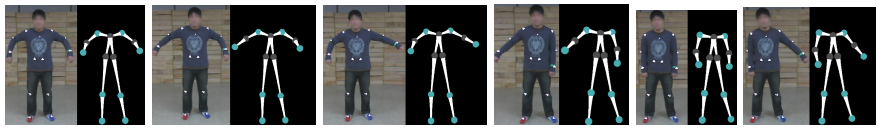
Hence, we reduced the image resolution to 160×120 for real-time operations. Since Gumstix does not support OpenCV 2.x, we have ported OpenCV 1.0. While a Logitech C250 webcam is used for Samba, other cameras with Linux drivers can be used. Wi-Fi is used to communicate between camera motes and the Samba server.

5.2 Samba Server

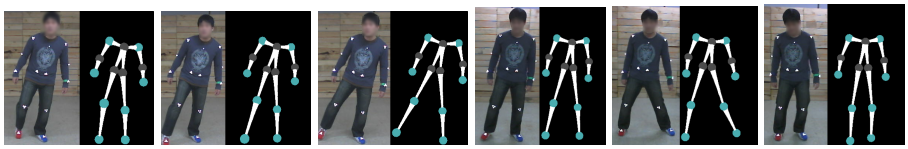
The Samba server, which is a notebook computer, receives 2D marker positions from camera motes. After camera initialization, camera motes start taking images and report marker positions. The server reconstructs 3D marker positions using camera calibration parameters and performs data-driven human pose estimation using the human pose database as discussed above.

5.3 Human Pose Database

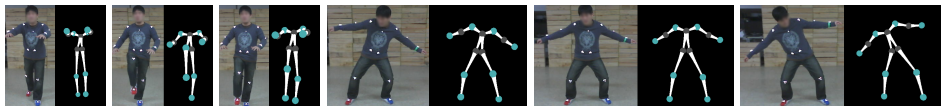
The human pose database is constructed from two sources: a local database and the CMU motion capture data [26]. A local database is a collection of 3D human poses collected using Samba. For the collection of 3D human poses, a human subject wears 13 markers at joint positions. A total of eight different actions are collected: two upper body actions, two lower body actions, and four whole body actions. See Figure 5.2 for



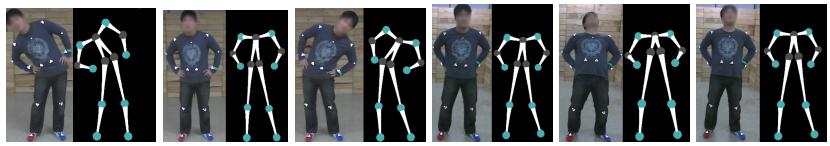
(a) Arm swinging (horizontal) (b) Arm swinging (c) Arm swinging (d) Arm lifting (e) Arm lifting (f) Arm lifting



(g) Fake motion (soccer) (h) Fake motion (i) Fake motion (j) Zig-zag (k) Zig-zag (l) Zig-zag



(m) Knee-Knee-kick (n) Knee-kick (o) Knee-kick (p) Surfing (q) Surfing (r) Surfing



(s) Stretching 1 (t) Stretching 1 (u) Stretching 1 (v) Stretching 2 (w) Stretching 2 (x) Stretching 2

Figure 5.2: Examples of actions from the local human pose database.

examples. The CMU motion capture data is converted to align the 13 joint positions and included in the Samba human pose database. The incorporation of an existing human pose database, which has been collected using expensive motion capture systems, provides better 3D reconstruction results while a local database can be used to customize the motion capture system with actions which are not typical in publicly available databases.

Pose Alignment: All human pose vectors are aligned to a fixed coordinate system with a fixed scale. The center of two shoulders is used as the origin $(0, 0, 0)$. Let RS

be the 3D position of the right shoulder and LS be the 3D position of the left shoulder. The x-axis is chosen as the direction of $LS - RS$. The y-axis is determined by the direction of the vector y:

$$y = \frac{(RS - RW) - a(LS - RS)}{\|(RS - RW) - a(LS - RS)\|_2}, \tag{5.1}$$

where $a = (LS - RS)^T(RS - RW)$ and RW is the right wrist. Lastly, the z-axis is chosen using the direction of $z = y \times x$. Each 3D pose is scaled such that the length of a lower leg has a unit length. An example of pose alignment is shown in Figure 5.3.

To further reduce the time required to search for neighboring poses, a neighborhood graph is constructed for the entire human pose database.

5.4 Marker Detection and Tracking

Each camera mote takes images at a resolution of 160×120 due to its computational cost of image acquisition and processing. Moreover, storing images in a Gumstix local memory causes a large overhead for the NAND flash memory. For this reason, images are stored temporarily in the RAM when markers are detected and removed from the RAM before taking the next image.

In Samba, a human subject wears retro-reflective markers, which reflect light when illuminated, since they are easier to detect with a simple intensity based marker detection algorithm. Given a gray-scale image, the intensity value of each pixel is thresholded. Then connected component analysis is used to extract blobs. The center of each blob is declared as a marker position. In order to light the retro-reflective markers, we have placed light stands behind camera motes. Currently, the use of retro-reflective markers and light stands is a limitation of Samba but, with additional computation, it is possible to detect and track non-reflective colored markers with a more sophisticated image processing algorithm.

Samba tracks eight markers as shown in Figure 3.1 and they are placed on the following joints: a left shoulder, a right shoulder, a left wrist, a right wrist, a left knee,

a right knee, a left foot and a right foot. We assume that a human subject starts with a standard pose, in which all marker positions are known, and all marker positions are initialized based on the standard pose. Then each marker is tracked using a nearest neighbor filter. Since the frame rate is high for low resolution images, we can reliably track all markers using nearest neighbor filtering.

5.5 3D Reconstruction of Markers

As the Samba server receives two sets of marker positions from camera motes, it performs 3D reconstruction of markers using the extrinsic camera parameters found during the calibration step based on the method described in [13]. An example is shown in Figure 5.3.

Since the reconstructed 3D marker positions are aligned with respect to the camera coordinate system, they cannot be directly compared to human pose vectors in the database. We rescale and align 3D marker positions as shown in Figure 5.3 such that an observed pose is consistent with the coordinate system used in the human pose database. The alignment step follows the pose alignment method described in Section 5.3.

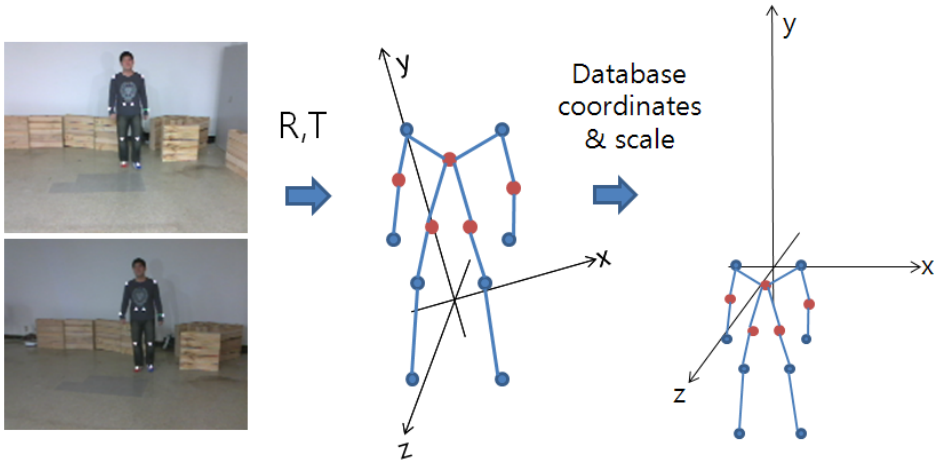


Figure 5.3: 3D reconstruction of markers and alignment. Input images are shown on the left. In the middle, the computed 3D positions of eight markers (blue dots) are shown. The 3D positions of markers are based on the camera coordinate system. Then the pose vector is aligned and scaled according to the coordinate system used in the database as shown on the right.

Chapter 6

Experimental Results

An extensive set of experiments is performed to validate the performance of the Samba motion capture system. A notebook computer is used as a server, which receives data wirelessly from camera motes. In order to measure reconstruction errors, we have defined the length of the tibia (lower leg) as 45 cm. We first examine processing times of the developed camera mote at different image resolutions. Then we measure reconstruction errors and computation times with different numbers of neighbors and pseudo-poses. Lastly, the performance of the overall system is examined.

6.1 Image Processing Times at a Camera Mote

Table 6.1 shows image processing times at a camera mote for different image sizes. The server program runs at 30 ms per frame, hence, a camera mote does not need to run faster than 30 ms per frame. When the image resolution is 640×480, it takes 423 ms on average to process one image while a 320×240 image takes 131 ms on average. Hence, we have chosen to use images at 160×120 resolution, which takes about 64 ms per frame on average. Due to the processing time at a camera mote, the overall frame rate of the Samba motion capture system is 16 frames per second.

Table 6.1: Processing times for different image sizes by a camera mote.

Image size	160×120	320×240	640×480
Average	64 ms	131 ms	423 ms
Max	102 ms	360 ms	925 ms
Min	62 ms	75 ms	324 ms

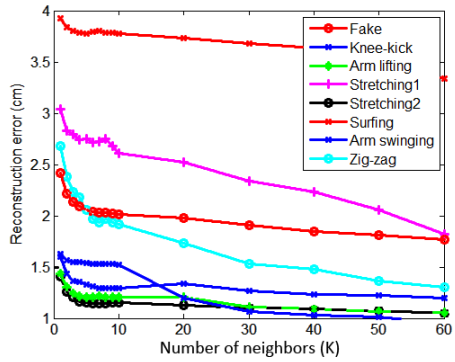


Figure 6.1: Reconstruction errors of eight actions at different number of neighbors.

6.2 Number of Neighbors (K)

A local pose space is constructed using neighboring poses. In order to determine the effect of the number of neighbors, we varied the number of neighbors, K, and measured the reconstruction errors and required computation times. The results are based on image sequences of eight actions which are not in the database. Each action sequence contains about 900 frames. For all eight actions, we computed reconstruction errors at different K and they are shown in Figure 6.1. The surfing action gave the worst error due to a large variation when both arms are swung.

The average reconstruction error of all eight actions as a function of the number of neighbors is shown in Figure 6.2. As expected, the reconstruction error decreases as the number of neighbors K is increases. The computation times for different K are shown in Figure 6.3. The computation time increases rapidly and decreases when

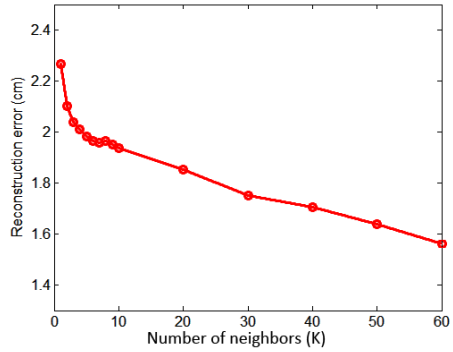


Figure 6.2: Average reconstruction errors of all eight actions at different number of neighbors.

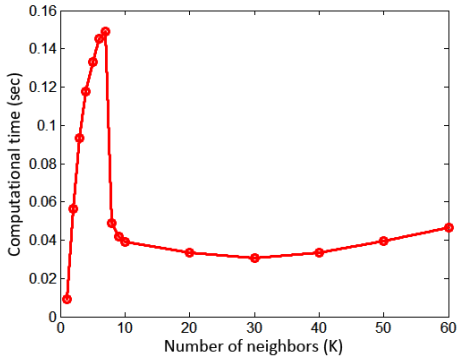


Figure 6.3: Computation times at different number of neighbors.

$K = 8$. Then it slowly decreases until $K = 30$ and then slowly increases again. This is due to the fact that optimization routine requires more time for finding a solution when the number of data is small (the optimization problem becomes numerically unstable).

6.3 Number of Pseudo-Poses (N)

We have also evaluated the effect of pseudo-poses in terms of the reconstruction error and computation time. Figure 6.4 shows reconstruction errors as more pseudo-poses are used to represent the local pose space. We have tested three different values of

K (10, 20, and 30). Again, the results are based on image sequences of eight actions which are not in the database and each action sequence contains about 900 frames. For all three cases, the reconstruction error decreases rapidly until $N = 10$ and then decreases slowly. Note that, when $N = 1$, no pseudo-pose is used and the local pose space is constructed using K neighboring poses. The result clearly indicates the benefit of using pseudo-poses. The reduction of the reconstruction error from using pseudo-poses is about 14% when $K = 10$ and $N = 10$. Figure 6.6 shows some examples where the 3D reconstruction of a pose is improved by the use of pseudo-poses. The required computation times at different values of N are shown in Figure 6.5. We see that the computation time increases linearly with N when N is larger than two. It is interesting to note that the computation time is significantly reduced when pseudo-poses are used. When $K = 10$ and $N = 10$, the average computation time is 0.0235 sec and the optimization routine requires about 49 iterations on average. But, when $K = 100$ and pseudo-poses are not used, it requires about 0.0886 sec and 78 iterations on average. The use of pseudo-poses has reduced the computation time by 73%. When $K = 10$ and $N = 1$, it still requires about 0.04 sec. It illustrates that pseudo-poses can better represent the local pose space with a dense set of data points, which results in faster convergence. Hence, we can better reconstruct a 3D human pose using pseudo-poses given a limited amount of time.

We have found that the best trade-off between the reconstruction error and computation time is when $K = 10$ and $N = 10$. These values are used for experiments described below.

6.4 CMU Motion Capture Database

The human pose database in Samba is a combination of the CMU motion capture data and locally collected motions as described above. See Figure 5.2 for examples of locally collected motions. The CMU motion capture data used in the human pose

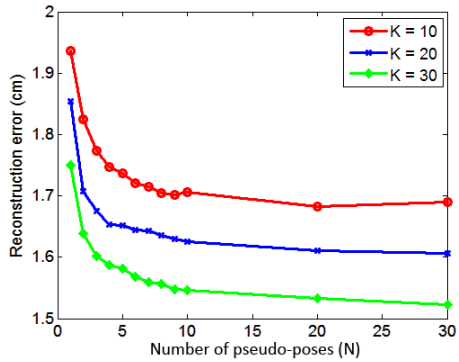


Figure 6.4: Reconstruction errors at different number of pseudo poses for $K = 10$, $K = 20$, and $K = 30$.

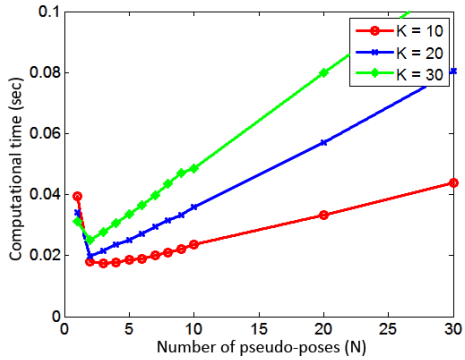


Figure 6.5: Computation times at different number of pseudo poses for $K = 10$, $K = 20$, and $K = 30$.

database consists of five motions: march, guard, jump, run, and walk. As shown in Figure 6.7, the Samba motion capture system can reconstruct poses that are from the CMU motion capture database as well.

6.5 Demonstration of the Samba Motion Capture System

Some snapshots from a demonstration of the Samba motion capture system are shown in Figure 6.8. For each photo in Figure 6.8, the left half of the photo shows a human

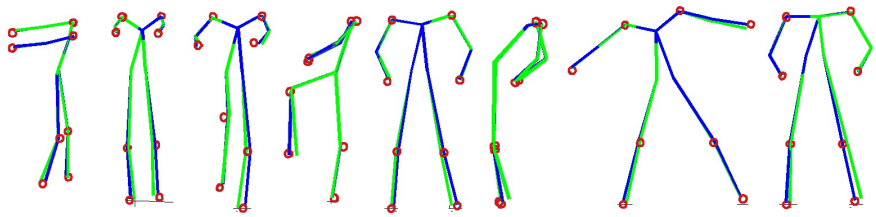


Figure 6.6: Examples of reconstructed poses. The red dots indicate ground-truth positions of eight markers. The green lines are reconstruction results without pseudo-poses and the blue lines are reconstruction results with pseudo-poses.

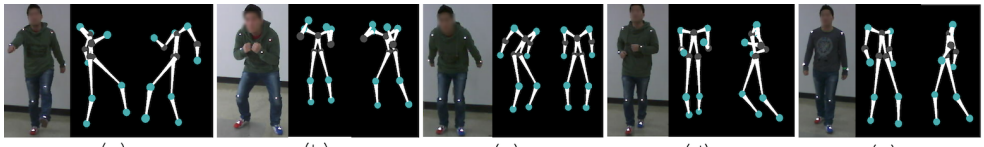
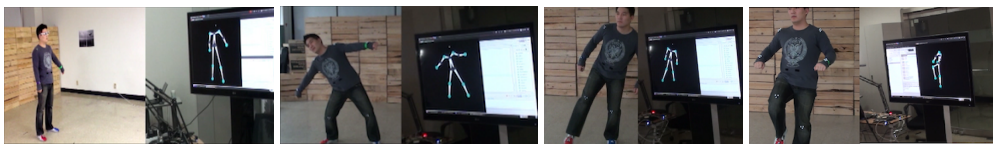


Figure 6.7: Reconstruction of poses from the CMU motion capture database. (a) March. (b) Guard. (c) Jump. (d) Run. (e) Walk.



(a) Arm lifting (b) Surfing (c) Fake motion (d) Knee-kick motion

Figure 6.8: A real-time demonstration of the Samba motion capture system.

subject performing a motion and the right half of the photo shows the reconstructed 3D pose displayed on a monitor, demonstrating a real-time operation of Samba.

Figure 6.9 shows reconstruction results from a kneekick action. While only eight markers are used by Samba, Samba reconstructs the entire human pose reliably throughout the action sequence. We have also tested the system on a new subject. As shown in Figure 6.10, we can reliably reconstruct a motion by another user. Note that the surfing motion has the worst reconstruction error. But reconstruction results are quite impressive.

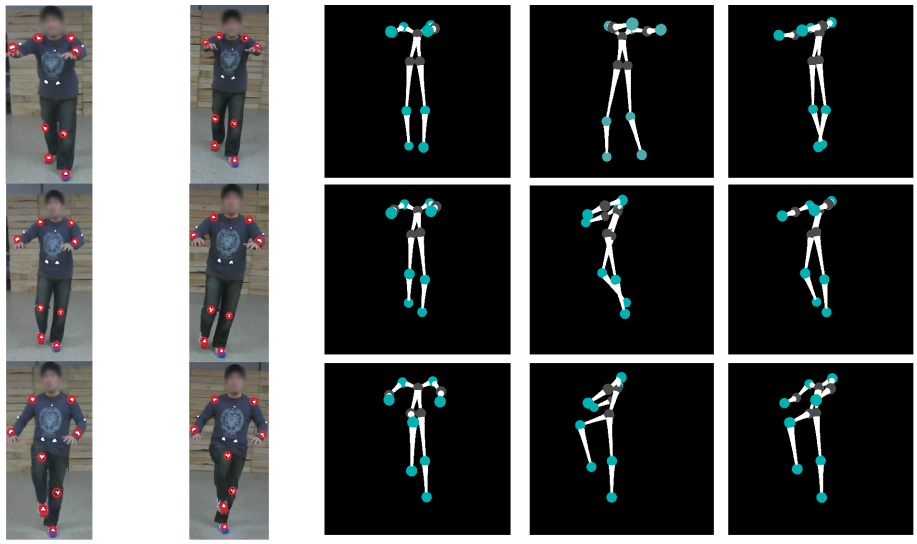


Figure 6.9: 3D reconstruction results from a kneekick motion. (a) and (b) are images taken by webcam connected to laptop. We use 8 markers in red circles as input. (c), (d), and (e) are reconstructed poses from different view points.

We have performed eight actions from the local pose database in front of the Samba system. The resulting reconstruction errors are shown in Figure 6.11. The reconstruction errors from the Samba system are slightly larger than reconstruction errors reported in Section 6.2. This is due to the fact that the local database is constructed using images at 640×480 resolution, which results in better localization of markers. However, the proposed system reconstructs various motions with an average error of 2.37 cm at 16 frames per second, which is small enough to represent a wide range of motions reliably. It demonstrates the feasibility of the Samba system as a low-cost alternative for a real-time motion capture system.

One limitation of the data-driven human pose estimation method is that it may not be able to reconstruct a pose which is significantly different from poses in the database. This problem can be addressed by adding more poses to the database as allowed by the Samba system. The next issue is whether the system can make a smooth transition

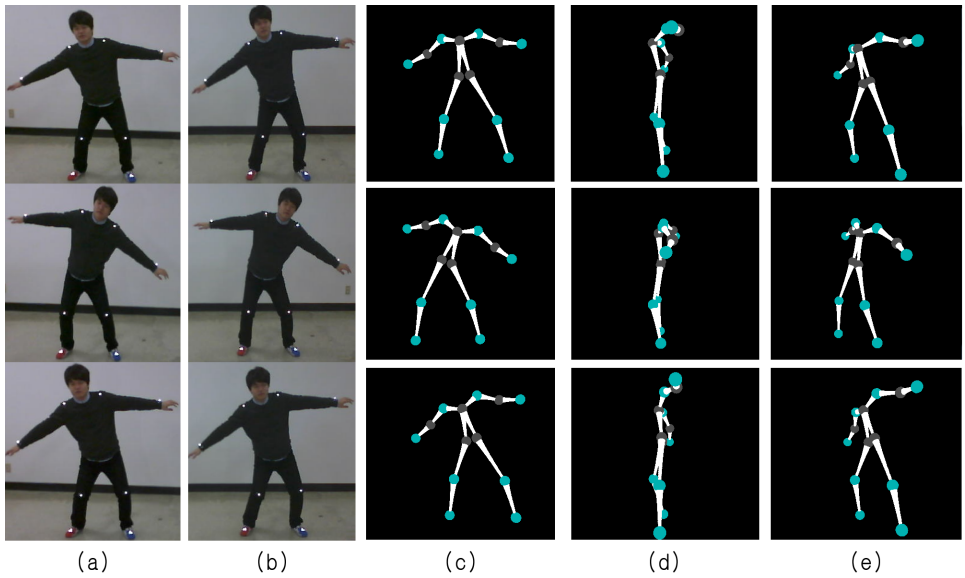


Figure 6.10: 3D reconstruction results from a surf motion by another user. (a) and (b) are images taken by camera motes. (c), (d), and (e) are reconstructed poses from different view points.

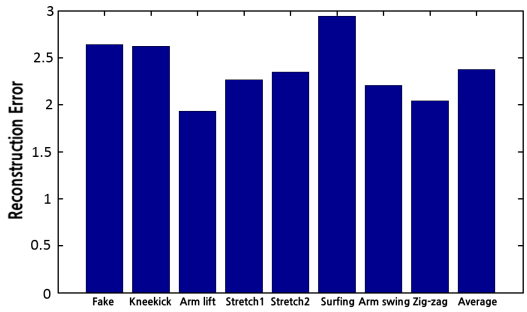


Figure 6.11: Reconstruction errors of eight actions using the Samba motion capture system. The average error is 2.37 cm which is slightly larger than 1.7 cm from Section 6.2, which is based on higher resolution images.

from a pose from one database to a pose from another database. We can make a transition between databases by introducing more importance to the smoothness term in

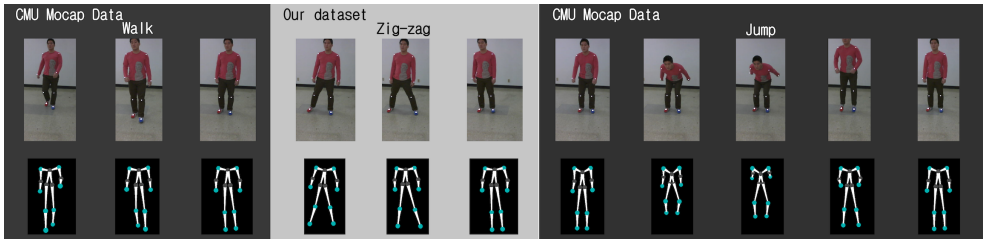


Figure 6.12: Reconstruction of a composite action: walk (CMU), zigzag (local), and jump (CMU).

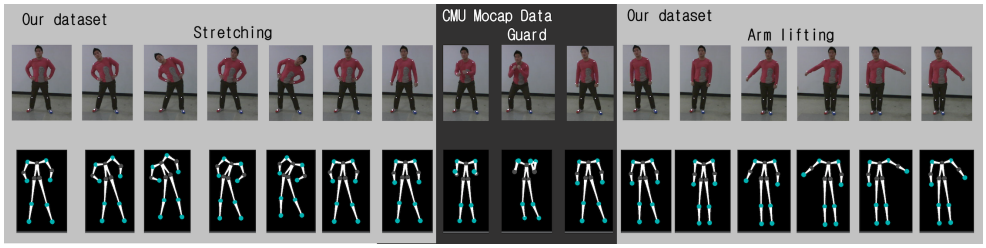


Figure 6.13: Reconstruction of a composite action: Stretching (local), Guard (CMU), and arm lifting (local).

4.7 and increasing the number of pseudo-poses and noises added to pseudo-poses. Figure 6.12 and Figure 6.13 show transitions between different databases, demonstrating that Samba can reliably capture complex composite actions in real-time.

Chapter 7

Conclusion

In this paper, we have described the development of Samba, a low-cost real-time motion capture system, which is based on a portable wireless camera sensor network. A new class of camera motes is constructed in order to process images in real-time. We have also reduced the required number of joint markers to eight to make it easier for general users. The proposed system provides accurate reconstruction of 3D human poses at 16 frames per second using the data-driven pose estimation method augmented with pseudo-poses. An ability to combine existing human pose databases with a locally generated pose database makes Samba scalable and a practical solution for general motion capture applications.

Bibliography

[1] Chai, Jinxiang and Hodgins, Jessica K., "Performance animation from low-dimensional control signals," in *ACM Trans. Graph.: NY*, 2005.

[2] VICON MX motion capture system, [online]. Available: http://www.vicon.com/

[3] levmar : C/C++ implementation of the Levenberg-Marquardt, [online]. Available: http://www.ics.forth.gr/ lourakis/levmar/

[4] Taylor, C.J., "Reconstruction of articulated objects from point correspondences in a single uncalibrated image," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on.*, 2000.

[5] Wei, Xiaolin K., "Modeling 3D human poses from uncalibrated monocular images," in *Computer Vision, 2009 IEEE 12th International Conference on.*, 2009.

[6] Burenius, Magnus and Sullivan, Josephine and Carlsson, Stefan and Halvorsen, Kjartan, "Human 3D Motion Computation from a Varying Number of Cameras," in *Image Analysis*, 2011.

[7] Yuanqiang Dong and DeSouza, G.N., "A new hierarchical particle filtering for markerless human motion capture," in *Computational Intelligence for Visual Intelligence, 2009. CIVI '09. IEEE Workshop on*, 2009.

[8] Grest, Daniel and Woetzel, Jan and Koch, Reinhard, "Nonlinear Body Pose Estimation from Depth Images," in *Pattern Recognition*, 2005.

[9] Zhu, Youding and Fujimura, Kikuo, "Constrained Optimization for Human Pose Estimation from Depth Sequences," in *Computer Vision ACCV 2007*, 2007.

[10] Ferrari, V. and Marin-Jimenez, M. and Zisserman, A., "Progressive search space reduction for human pose estimation," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008.

[11] Moeslund, T.B. and Granum, E., "3D human pose estimation using 2D-data and an alternative phase space representation," in *Procedure Humans 2000*, 2000.

[12] Ogawara, K. and Li, X. and Ikeuchi, K., "3D human pose estimation using 2D-data and an alternative phase space representation," in *Robotics and Automation, 2007 IEEE International Conference on*, 2007.

[13] Yi Ma and Stefano Soatto and Jana Košecká and S. Shankar Sastry, "An invitation to 3-D vision: From images to geometric models, 2004.

[14] Rogez, G. and Rihan, J. and Ramalingam, S. and Orrite, C. and Torr, P.H.S., "Randomized trees for human pose detection," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008.

[15] Sminchisescu, Cristian and Telea, Alexandru, "Human Pose Estimation from Silhouettes. A Consistent Approach Using Distance Level Sets," in *10th International Conference on Computer Graphics,Visualization and Computer Vision (WSCG '02)*, 2002.

[16] Agarwal, A. and Triggs, B., "3D human pose from silhouettes by relevance vector regression," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004.

[17] Lee, MunWai and Nevatia, Ram, "Human Pose Tracking Using Multi-level Structured Models," in *Computer Vision ECCV 2006*, 2006.

[18] Gall, J. and Stoll, C. and de Aguiar, E. and Theobalt, C. and Rosenhahn, B. and Seidel, H.-P., "Motion capture using joint skeleton tracking and surface estimation," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009.

[19] Franco, J.-S. and Lapierre, M. and Boyer, E., "Visual Shapes of Silhouette Sets," in *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, 2006.

[20] Chen Wu, Hamid Aghajan, Richard Kleihorst, "Real-Time Human Posture Reconstruction in Wireless Smart Camera Networks," in *IPSN '08 Proceedings of the 7th international conference on Information processing in sensor networks*, 2008.

[21] Horaud, R. and Niskanen, M. and Dewaele, G. and Boyer, E., "Human Motion Tracking by Registering an Articulated Surface to 3D Points and Normals," in *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2009.

[22] King, B.A. and Paulson, L.D., "Motion Capture Moves into New Realms" in *Computer*, 2009.

[23] Moeslund, T.B. and Hilton, A. and Krüger, V., "A survey of advances in vision-based human motion capture and analysis,"in*Computer Vision and Image Understanding*, 2006.

[24] Berman, S. and Stern, H., "Sensors for Gesture Recognition Systems," in*IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2012.

[25] Gumstix, [online]. Available: http://www.gumstix.com/

[26] CMU Graphics Lab Motion Capture Database, [online]. Available: http://mocap.cs.cmu.edu/

[27] hoebus Wei-Chih Chen and Parvez Ahammad and Colby Boyer and Shih-I Huang and Leon Lin and Edgar J. Lobaton and Marci Lenore Meingast and Songhwai Oh and Simon Wang and Posu Yan and Allen Yang and Chuohao Yeo and Lung-Chung Chang and Doug Tygar and S. Shankar Sastry, "CITRIC: A low-bandwidth wireless camera network platform," in *Proc. of the ACM/IEEE International Conference on Distributed Smart Cameras*, 2008.

[28] Sangseok Yoon and Hyeongseok Oh and Donghoon Lee and Songhwai Oh, "PASU: A personal area situation understanding system using wireless camera sensor networks," in *Personal and Ubiquitous Computing (online)*, 20 12.

[29] Jianbo Shi and Jitendra Malik, "Normalized cuts and image segmentation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[30] Aravind Sundaresan and Rama Chellappa, "Markerless motion capture using multiple cameras," in *Proc. of the Computer Vision for Interactive and Intelligent Environment*, 2005.

[31] Hiroyuki Ukida and Seiji Kaji and Yoshio Tanimoto and Hideki Yamamoto, "Human Motion Capture System using Color Markers and Silhouette," in *Proc. of the IEEE Instrumentation and Measurement Technology Conference*, 2006.

[32] Kazutaka Kurihara and Shinichiro Hoshino and Katsu Yamane and Yoshihiko Nakamura, "Optical motion capture system with pan-tilt camera tracking and real time data processing," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2002.

국문 초록

3D TV, 3D 스마트폰, 3D 영화등의 3D 디스플레이 산업의 발달로인해 3D 컨텐츠에 대한 관심이 증가하고있다. 3D 산업의 발달로 인해 3D 컨텐츠의 소모량이 증가할 것으로 예상된다. 만약 일반 사용자가 쉽게 3D contents를 제작 및 배포할 수 있다면 3D 디스플레이가 좀 더 유용하게 쓰이며 3D 산업의 성장으로 이어질 것이다. 그러나 기존의 모션 캡쳐 시스템은 일반 사용자들이 사용하기에 너무 비싸고 어렵다. 본 연구에서는 SAMBA Mocap system이라고 하는 pseudo-sample을 기반으로 한 wireless camera sensor network를 사용한 저비용의 모션 캡쳐 시스템을 제안한다. Wireless sensor mote를 webcam과 연동하여 이동가능한 일반 사용자용 모션 캡쳐 시스템을 제작하였다. 모션 캡쳐를 하기위한 마커를 최소한으로 줄였기 때문에 일반 사용자들이 쉽게 이용할 수 있다. SAMBA Mocap system은 움직이는 마커를 실시간으로 감지, 추적하며 3D motion capture 데이터를 실시간으로 구한다. Camera sensor mote로 마커를 감지하고 추적하는 것이 실시간으로 이루어지기 때문에, 추적 알고리즘을 쉽게 디자인할 수 있었다. 또한 현재의 동작을 optimization하는 데에 30ms밖에 걸리지 않았기 때문에 전체 시스템이 실시간으로 돌아갈 수 있다.

주요어: Camera sensor networks, Pseudo-samples, Motion capture system

학번: 2011-20878