MOTION DATA BEYOND ANIMATION

by

Antonio Carlos Salzvedel Furtado Junior

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science
University of Alberta

Abstract

Motion capture (MoCap) data has always been one of the most important components in the entertainment industry, being widely employed in animated movies and games. Given technological advancements in motion capture technologies, it has been successfully applied to other areas, such as surgical assessment, elderly monitoring and surveillance systems. As capture technologies are further developed, and human kinematic models are commonly employed in different fields, new issues regarding data optimization arise.In this thesis, we present novel methods that optimize the use of MoCap data in three distinct situations: unreliable network transmission, data compression and human identification. These methods assist MoCap-based applications outside the traditional animation scope, and become relevant as data format is standardized.

Preface

Chapter 2. A version of this material has been published [Antonio Carlos Furtado, Irene Cheng, Frederic Dufaux, and Anup Basu. Robust Transmission of Motion Capture Data using Interleaved LDPC and Inverse Kinematics. In Eurographics 2016 - Short Papers, Eurographics Assoc. (2016)]. I was the lead investigator, responsible for all major areas of concept formation, data collection and analysis, as well as manuscript composition. Anup Basu and Irene Cheng were supervisory authors.

Chapter 3. A version of this material is pending publication [Antonio Carlos Furtado, Xinyao Sun, Anup Basu and Irene Cheng. Optimized Per-Joint Compression of Hand Motion Data. In Systems, Man and Cybernetics (SMC), 2016 IEEE International Conference on. IEEE, 2016]. I was the lead investigator, responsible for all major areas of concept formation, data collection data analysis, as well as manuscript composition. Xinyao Sun was involved in data collection. Anup Basu and Irene Cheng were supervisory authors.

Chapter 4. Based on ongoing work. It has not yet been published.

Acknowledgements

I would like to thank my supervisors, Dr. Anup Basu and Dr. Irene Cheng, for their help, guidance and outstanding supervision. I also want to thank my committee members, Dr. Ehab Elmallah and Dr. Joerg Sander, for taking their time to read my thesis and participate in my defense.

I would like to express my appreciation to all the members from our Multimedia Research Center (MRC). I want to thank them for their good advice and helpful tips during our group meetings and discussions. Furthermore, a special thanks to all my other colleagues that are not part of the MRC, but helped me nevertheless: Hamman Samuel, Kriti Khare, Roberto Vega and Rodolfo Wottrich.

Last, but not least, I would also like to express my gratitude and love for my family and my girlfriend, Remy Wu, for their understanding and unconditional support.

Contents

List of Tables

List of Figures

List of Abbreviations

List of commonly used abbreviations

DOF	Degree of Freedom
FK	Forward Kinematics
HMM	Hidden Markov Model
IK	Inverse Kinematics
kNN	k-Nearest Neighbors
LDPC	Low-Density Parity-Check
MoCap	Motion Capture
PCA	Principal Component Analysis
RMSE	Root Mean Squared Error
SV	Support Vector
SVM	Support Vector Machine

Glossary

Articulated Model	Skeletal model for which individual joints can be either rotated or translated.
Degree of Freedom	A direction in which independent motion can occur.
Forward Kinematics	Method in 3D computer graphics for animating models, in which positions of particular parts of the model at a specified time are calculated from the position and orientation of the object, together with any information on the joints of an articulated model.
Interleaving	Process of arranging data in a noncontiguous manner.
Inverse Kinematics	Use of the kinematics equations of a model to determine the joint parameters that provide a desired position of the end-effector.
MPEG-4 BBA	Standard that defines the interchange format for articulated models.
Syndrome bits	Error correction codes generated by multiple forward error correction techniques.

Chapter 1

Introduction

MoCap data is generally associated with the entertainment industry, since it has evolved around animation. Its origin lies on a technique known as rotoscoping, in which animators manually trace over footage. The first application of this technique dates back to 1917. Since then, advancements in motion sensors allowed to automate most of the capture process. This allowed motion capture to become much more accessible. It is now possible to extract MoCap even from low-priced depth cameras [1].

Given advancements in capture technology and motion extraction techniques, motion has been demonstrated to be useful in several areas outside its traditional scope, such as surgical skill assessment [2], elderly monitoring [3], surveillance systems [4], etc. Unfortunately, this diversification did not incur in any standardization of MoCap. Different from other multimedia data types, such as audio and video, motion data involves more than a single capture step. Generally, it relies on a feature extraction step, which consists in extracting posture and motion cues from either video or depth maps that are discriminating with respect to human pose. The extraction step can lead to different representations, ranging from complex models to simple silhouette images.

Given the lack of standardization, and the recent research interest in Mo-

Cap applications, there is still a lot of room for improvement in handling this type of data. On this paper-based thesis, we address three different issues related to MoCap: transmission, storage and biometrics. We organize the thesis as follows. Section 1.1 briefly emphasizes the motivation behind this work. In Section 1.2, we briefly describe how MoCap is generally represented. This representation is common throughout this thesis. In Chapter 2 we present our work on MoCap transmission over unreliable channels. Chapter 3 is used to describe our compression system, aimed to improve MoCap storage. Chapter 4 describes our gait recognition strategy. Finally, a general conclusion is presented in Chapter 4.

1.1 Motivation

The increasing number of applications that use MoCap, and well the advancements of commercial motion sensors, are the main motivation behind this work. As more fields benefit from MoCap, we believe this will lead to the same standardization process that took place for other multimedia data types. Unreliable transmission and encoding can already be considered a solved problem for video streams, as not only multiple research papers have addressed this particular problem, but entire standards have been created and are widely employed. Examples of such standards are the Real Time Protocol (RTP) [5] and H.264 [6], which are used to transfer streaming media and compression of video sources, respectively.

Similarly, human recognition has also been widely researched for video streams, and it is even added as an embedded feature to several computer vision software packages, such as OpenCV [7]. We consider biometrics to be one of the most promising applications for MoCap, achieved through gait recognition. This type of biometrics does not require close proximity with the subject, and it can be adapted to multiple capture systems.

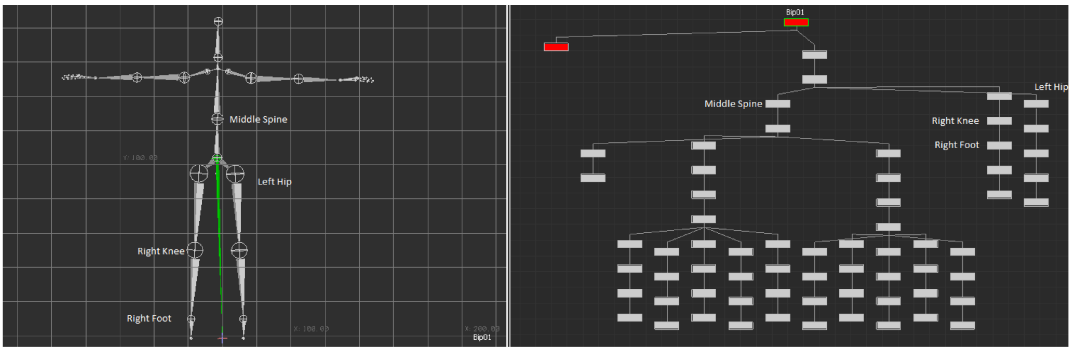

Figure 1.1: Example of skeletal model (Left) next to its hierarchical representation (Right).

1.2 Motion Data Representation

Motion can be represented in many different modeling formats [8], such as shape models, image descriptors, silhouettes, edges and kinematic models. Although previous research has exploited all of these different formats, kinematic models remain the most widely used, given their intuitiveness. Even though there is not a single industry-wide MoCap standard format, all the formats used in commercial applications are based on kinematics models.

Kinematic models consist of skeletal structures, represented by a collection of simple rigid objects connected by rotation joints. Each joint allows rotation in up to 3 orthogonal directions (i.e. DOF). In addition, they can contain multiple rotation constraints, limiting the rotation angle in any given direction.

Individual joints within the skeleton are defined in their local coordinate system, and are assembled into a single entity in a global world coordinate system, through a nested series of transformations. In Figure 1.1, we illustrate a kinematic model representing a biped. The number of joints, as well as the degrees of freedom, can easily vary from one model to another.

For each joint, we can specify its orientation by using three angles. This was demonstrated by Euler's rotation theorem [9]. Let D, C and B represent rotations about the z-axis, x-axis and y-axis, respectively. They can be written in terms of rotation matrices, as follows

$$D = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix},$$

$$B = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}.$$

A general rotation A can be written as

$$A = BCD.$$

This means that any orientation can be encoded by using the set of angles (ψ, ϕ, θ). The order in which these rotation angles are applied is important, as matrix multiplications are not commutative. For this reason, this order needs to be specified either for the MoCap file format, or for each file separately.

Rotation angles need to be encoded for each frame in a motion sequence. We refer to the time sequence of angle values for a particular joint as a channel, or as a curve. For each channel, angle values at specific time stamps are referred as keyframes. We illustrate a curve in Figure 1.2. Depending on the MoCap standard being used, keyframes may be either synchronized across multiple channels or not. They can also rely on multiple techniques to interpolate keyframes.

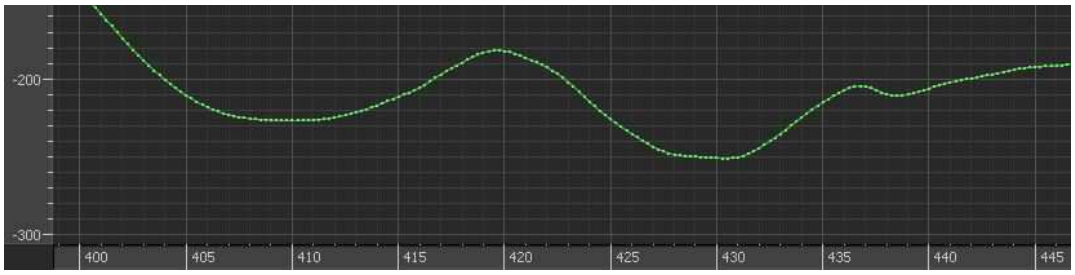

Figure 1.2: Rotation curve representing the rotation about the y-axis for a given joint. For this plot, the vertical axis represents the angle, in degrees, while the horizontal axis represents time, in milliseconds. Each white dot composing this curve represents a keyframe.

Chapter 2

Transmission

2.1 Introduction

MoCap data has been used extensively in the entertainment industry, in particular movies and games. It has also been employed in military, sports and medical applications. Given this diverse range of applications, we also have different restrictions on data transmission. We cannot assume that it is always possible to transmit data over reliable networks, especially for applications that require live-streaming of MoCap data, such as content delivery networks (CDNs). Packet loss is a major issue in wireless networks. Missing packets can lead to inaccuracies in motion, resulting in unacceptable perceptual deformations in the animation. Degradation of perceptual quality derived from packet loss is a real problem, and it affects not only motion data, but any type of multimedia data. Its effects on video streaming, for instance, are well known, and it can be caused by a variety of reasons, including network congestion. For example, Tan et al. [10] has a well documented report on the performance of congested 3G networks. It demonstrates that video-voice conversations can experience frozen video frames and one-sided conversations when network is congested, due to packet loss. That is why it is highly important to consider this scenario during transmission.

Robust transmission of multimedia data has been explored by many researchers in the past [11, 12, 13, 14]. However, little attention has been given to lossy transmission of MoCap data. In this chapter we introduce a MoCap data transmission method that attempts to minimize the errors caused by packet loss resulting from unreliable transmission. We believe this is the first work to address this issue. Our method does not rely on the retransmission of missing packets, so it does not add extra overhead to the network. Instead, we minimize perceptual loss based only on successfully transmitted packets. To achieve this goal, our proposed method relies on Forward Error Correction (FEC) techniques to optimize the transmission.

The remainder of this chapter is organized as follows. Section 2.2 reviews MoCap data transmission methods. Section 2.3 outlines potential methods that can be used to optimize coding of MoCap data for transmission. Section 2.4 presents our proposed transmission method. Preliminary results are presented in Section 2.5. Finally, conclusion and future work are present in Section 2.6.

2.2 Related Work

Data transmission methods targeted at MoCap data are still emerging. The biggest advancement in this area came with the creation of a standard interchange format by the MPEG group, called Bone Based Animation (BBA) [15], which is an extension of the MPEG-4 standard. This standard also relies on a FK representation, described in Section 1.2. It allows the representation of any articulated model. However, when it comes to encoding this representation, MPEG-4 BBA considers only the bit-stream syntax definition. Also, MPEG-4 BBA does not specify the encoding tools to be used, leaving room for improvements.

Given the opportunity to optimize the encoding of MoCap data for this standard, some studies have developed different encoding alternatives [16, 17].

Preda et al. [16] proposed the first known MPEG-4 BBA encoder implementation. It extends the default MPEG-4 compression scheme by employing keyframe reduction. It reduces the number of frames to be transmitted and interpolates intermediate frames when decoding. Chattopadhyay et al. [17] proposes another encoding method that optimizes the power consumption for the device that is decoding the motion data, making it more usable on power-constrained devices. It also employs a sparsing algorithm that improves the compression of the motion data, decreasing the bitrate necessary for transmission.

The limitation of the proposed methods is that they do not address unreliable networks, and only focus on improving the bitrate. Therefore, their method can only be applied over a reliable Transmission Control Protocol (TCP) connection. Using a reliable transport means higher retransmission rates for higher losses; something not taken into consideration by any of these previous research publications.

2.3 MoCap Coding for Robust Transmission

Till now the issue of data loss during MoCap transmission has not received much attention. Thus, we first consider some potential alternatives that can be considered for error resilience during transmission of MoCap data. These approaches are outlined below:

2.3.1 Hierarchy Protecting Transmission

Loss of data in a hierarchical skeleton can lead to errors being propagated throughout the hierarchy. A solution to address this problem is to transmit some of the more important nodes more reliably by possibly retransmitting them in case of loss. However, retransmission results in delays caused by handshake protocols, such as the ones used by TCP-IP. Thus, for real-time mobile

transmission sending duplicate information on important nodes may be a more acceptable strategy.

2.3.2 Forward Error Correction

Forward Error Correction (FEC) techniques, including Reed-Solomon, Turbo-code, and Low-Density Parity-Check (LDPC), can be used to add protective bits to correct for errors during transmission. This strategy can be used to make MoCap transmission robust by intelligently creating packets so that the information received can be used to fill in the information that is lost. However, the loss rate needs to be below what can be corrected by the error corrective code.

2.3.3 Bezier Curve Interpolation

Bezier cubic splines have been used extensively to interpolate keyframes in MoCap files. They provide smooth continuous curves, which are ideal for representing joint rotations. By simply ignoring missing keyframes during transmission, and interpolating the remaining ones, Bezier cubic spline interpolation serves as an efficient candidate for several cases. Interpolation is a good alternative for motion clips that are recorded and transmitted at a high frame rate, even at moderately high loss rates.

Interpolation becomes insufficient for high packet loss at a lower frame rate. For this situation, simply interpolating close keyframes may result in perceptually visible discrepancies from the original motion sequence.

2.3.4 Interleaver

It is generally understood that the packet loss distribution in IP networks is bursty [18, 19] (i.e. errors tend to occur in bunches, as opposed to being associated with random patterns). Interleaving techniques have been widely used

to address the burstiness for multiple multimedia data types [20, 21]. Their goal is to convert "burst-like errors" to "random-like-errors." For most applications, spatially concentrated errors tend to be more harmful than a dispersed distribution of errors. This is also true for MoCap transmission. Single missing keyframes can be interpolated with Bezier spline curves without much loss of information, given that its neighboring keyframes are relatively close. On the other hand, it is harder to interpolate groups of adjacent keyframes at the same time. Therefore, interleaving helps leverage the interpolation methods.

Another benefit of applying interleaving techniques to MoCap data comes from the fact that animation data is strictly temporal. Keyframes are essentially curve values at a specific time. By simply reordering keyframes, according to their time stamp, before transmitting, it is possible to apply interleaving. The only drawback is that it is necessary to add a visualization delay, corresponding to the size of the interleaving window.

2.4 Integrated MoCap Transmission System

In Section 2.3, we listed some of the different strategies that can be used to improve the robustness of the transmission. Even though they all address relevant issues, none of them can, in isolation, offer a robust MoCap transmission method. Thus, we consider a strategy to combine multiple techniques into a single transmission system.

Our proposed transmission system is based on a Client-Server architecture. Its behavior is illustrated in Figure 2.1. At the encoder side, keyframes are interleaved, and their values are used to produce error correction codes through the LDPC method. These codes are embedded into packets, and used for keyframe reconstruction. We also refer to these codes as syndrome bits. While multiple FEC techniques are available to generate error correction codes, we decided to rely solely on LDPC, and use of other techniques were left out of

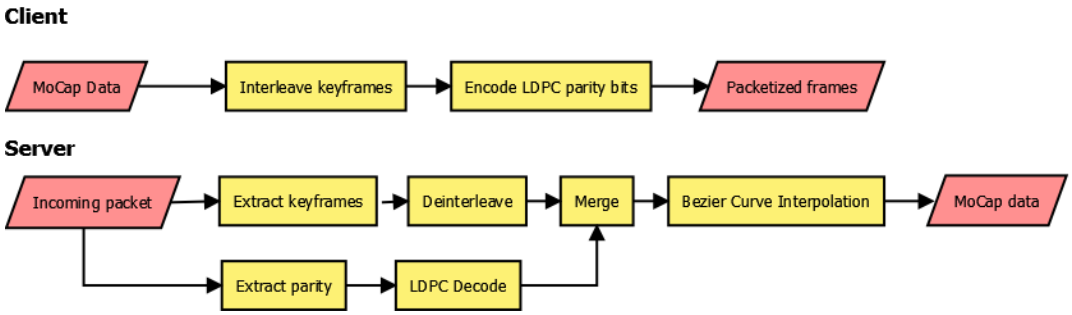

Figure 2.1: Flow chart representing the behavior of our system from decoding (client side) to decoding (server side).

the scope of this chapter. LDPC was chosen based on its efficient encoding and decoding scheme, and also its popularity, as there are multiple readily available implementations of this technique.

At the server side, a copy of the original skeleton model is assumed to be already present, since it is independent of the animation data. The server's goal is to decode the packetized data transmitted by the client. When no loss is present, the decoded data is identical to the original one. The server starts by extracting the keyframe information from their corresponding fragments, which is a simple process. These keyframes are inserted to the curves that belong to their corresponding joints. Furthermore, missing keyframes are reconstructed based on belief propagation (BP) decoding, for which we input LDPC syndrome bits together with interpolated data.

In the following subsections, we describe the behavior of our transmission system in more details. Section 2.4.1 describes our strategy to packetize MoCap data. Section 2.4.2 gives an overview on how keyframes are interleaved. An overview of the use of LDPC codes is given in 2.4.3, along with a description on how interleaving and LDPC are combined into our method.

2.4.1 Packetization

In our method, the conversion between keyframes and packet fragments is straightforward. Given that keyframes are basically composed of a time stamp and corresponding curve value, these are the only pieces of information that need to be stored in a packet fragment. Therefore, a packet fragment can be defined as:

```
struct PACK_FRAGMENT {
    int joint_id;
    long long key_time;
    TRANSFORMATION_TYPE key_type;
    float val_x,val_y,val_z;
};
```

where $joint\_id$ represents the node for which the keyframe belongs, key_time represents the keyframe time stamp, $key\_type \in \{ROTATION, TRANSLATION\}$ represents the transformation type and the set val_x, val_y, val_z represents the key value. In general key values are represented by a single floating point value. We use an optimized representation based on two assumptions. First, we assume that any rotation or translation is associated with up to 3 keyframes (i.e., any given joint has up to 3 degrees of freedom (DOFs)). We also assume that keyframes are synchronized, according to a specific frame rate. This representation may add extra unused data for some joints, since not all joints have 3 DOFs. However, this is compensated by not requiring the time stamp to be repeated for every keyframe.

2.4.2 Interleaving Keyframes

As previously stated, we consider keyframes to be aligned, which is a common property of motion clips. This means that a set of packet fragments can also

be represented as a list of pairs of joints j and time stamps t:

$$\begin{array}{llll} (j_1, t_1), & (j_2, t_1), & ..., & (j_M, t_1) \\ (j_1, t_2), & (j_2, t_2), & ..., & (j_M, t_2) \\ & \vdots & & \\ (j_1, t_N), & (j_2, t_N), & ..., & (j_M, t_N) \end{array}$$

where M represents the total number of articulated joints and N is the number of frames in the clip. Since each row in this sequence can be seen as a frame from our clip, and each column contains the set of keyframes for a particular joint, a simple way of interleaving this sequence is by shuffling the rows.

Obviously, we cannot shuffle the whole sequence (i.e., from 1 to N) without first having the entire sequence recorded. This will cause a considerable delay, and it is not feasible for real-time applications. For this reason, it is necessary to introduce a sliding window parameter. We represent this parameter as

$$1 \leq D_I \leq N,$$

which we refer as the interleaving window, and it is measured by number of frames. This sliding window delimits a a time stamp range

$$[t_i, t_{i+D_I}] \quad , 1 \leq i \leq N.$$

All the keyframes for which the times tamp belongs to this interval are considered to be part of the corresponding sliding window. The order in which these keyframes is transmitted is randomized. This parameter affects both the encoder and the decoder, as it introduces delays on both sides of the transmission. The delay at the encoding side is equal to $D_I/FrameRate$ frames, as the encoder needs D_I frames recorded before being able to shuffle it. Similarly, the decoder needs to wait for a corresponding time, as it does not know

the arrival order beforehand. The total delay D_T introduced by D_I is equal to $D_I/FrameRate + TransmissionTime(D_I) + DecodingTime(D_I)$. The parameter D_I has to be set according to how much delay is acceptable for an application.

2.4.3 Interleaved LDPC

Our strategy is to exploit LDPC codes as means of compressing redundant data to be transmitted. The use of error correcting codes for compressing binary sources has been discussed by different authors [22, 23, 22]. They all exploit the potential of the Slepian-Wolf theorem [24], which states that the compression of the output of two correlated sources that do not communicate their outputs to each other can be as efficient as in the case where they communicated their outputs. In the case of LDPC codes, the compression is achieved by taking a sequence of n input bits, multiplying by a sparse parity check matrix, and mapping it to a sequence of output syndrome bits [22].

Even though many techniques have been employed in reaching the Slepian-Wolf efficiency boundary, these studies lack a more practical application, which cannot be obtained by simply compressing synthetic data. In our work, we use LDPC codes to compress redundant keyframe data, including x, y and z coordinates, and send it in the same packet. We can define our problem as follows:

- Let $F^t = (x^t, y^t, z^t)$ represent a transformation at time t, either a rotation (in Euler Angles) or a translation. Let $F^t_{(2)} = [x^t_{(2)}|y^t_{(2)}|z^t_{(2)}]$ be the binary representation of this transformation.

- Consider $\bar{F}^t$ to be an approximation of transformation F^t, obtained through interpolation.

- H represents a sparse parity-check matrix, corresponding to a linear code (n, k).

Given our transformation F^t, we can obtain the syndrome bits $Z^t[length(n-k)]$ by multiplying our input transformation by H, such that $F_2^t \cdot H = [F_2^t|Z^t]$. The syndrome bits Z^t contain the compressed data to be transmitted. In our implementation, to add these bits to a packet fragment, we created a fragment extension to the one proposed in Section 2.4.1, shown below:

```
struct LDPC_FRAGMENT : PACK_FRAGMENT {
    std::bitset<n-k> Z;
};
```

Given that each fragment corresponds to a unique time t, our goal is to add syndrome bits for fragments corresponding to the same joint, but with different time stamps. Therefore, we can compensate the loss of some fragments by reconstructing them based on the syndrome bits, obtained in different packets. Since both the encoder and the decoder need to know which time stamp corresponds to syndrome bits Z, we created an offset parameter called D_L. This offset parameter needs to be consistent between the encoder and the decoder, this way there is no need to send any extra time stamp data. This means that transformation F^t is added to the same fragment as the syndrome Z^{t+D_L}. Similarly to D_I, described in Section 2.4.2, D_L can also cause a delay. In order to simplify the behavior of our integrated system, we ensure that $D_L \leq D_I$. By placing this constraint, we make sure that the total delay D_T remains the same as discussed in Section 2.4.2. The delay should be defined according to the requirements of each application.

Regarding the parity-check matrix H used to obtain the syndrome bits Z, we define it before the start of the transmission. This matrix is shared between the encoder and the decoder. When generating H, opting for an efficient parity-check distribution is not a trivial task. This problem has been discussed

in several papers [25, 26, 27, 28]. Unfortunately, their application in multimedia data is yet to be explored. In this chapter, we only cover regularly distributed parity-check matrices [25] (i.e. weight of columns and rows is constant). Based on initial tests, these matrices produced more stable results. A better understanding of how other distributions would affect our results is left for future work, and it is out of the scope of this thesis.

Decoding and Reconstruction

The decoding process is started by simply extracting the keyframe information from the corresponding fragments. These keyframes are inserted to the curves that belong to their corresponding joints. As previously mentioned, the joint mapping is known beforehand, as the skeleton model is available at the server.

All the curves use Bezier splines to interpolate the inserted keyframes. The order in which the keyframes are added to the curve is irrelevant. Therefore, this step is not affected by the interleaving performed by the client. However, it is important to remember that the time corresponding to the interleaving window D_I needs to be considered when displaying the animation, in case of real-time transmission applications.

When keyframes are being extracted and inserted into the curves, syndrome bits Z are also obtained from the same fragments. Once the loss of a fragment for a given joint at time t is detected, we reconstruct missing keyframes by using an estimate $\bar{F}^t$. This estimate is obtained by evaluating corresponding animation curves (i.e., curves for x, y and z coordinates) at time t.

We assume the interpolated transformation $\bar{F}^t$ to be a good approximation of the original value. This assumption relies on the fact that the keyframes are interleaved before being packetized and transmitted. This means that the distance between successfully transmitted keyframes is minimized in the event of burst losses. Even though LDPC and interleaving are able to work independently in our system, their association is able to boost results.

After obtaining $\bar{F}^t$, we proceed by feeding this information into a LDPC belief propagation decoder, along with syndrome bits Z^t, as follows:

$$F_2'^t = BPDecode(L([\bar{F}^t_{(2)}|Z^t]))$$

where L represents the log-likelihood ratio (LLR) for each bit in the sequence $[\bar{F}^t_{(2)}|Z_t]$. L is a bitwise operation, which can be defined as:

$$\ln\left(\frac{Pr(b_i = 0|y)}{Pr(b_i = 1|y)}\right)$$

where $Pr(b_i = 0|y)$ is the conditional probability of bit b_i being 0, given the received vector y. In our implementation, the LLR calculation is defined according to the representation of each bit as follows:

- For bits representing Z^t: We assume these bits to be sent through an error-free channel. Therefore, we are certain that they contain the correct value. In this case, $LLR(0) = +\infty$, whereas $LLR(1) = -\infty$;

- For bits representing $\bar{F}^t_{(2)}$: As $\bar{F}^t$ is actually represented as x^t, y^t, z^t, we can represent it as a set of 3 float values. For each float, all its bits are represented as finite values, such that $|LLR(b_i)| < |LLR(b_{i+1})|$, i representing the significance order of the bit (i.e., b_{i+1} is more significant than b_i). This representation was chosen because it is less likely that more significant bits will differ between the interpolated transformation $\bar{F}^t$ and the original F^t.

After decoding the LLR sequence, we obtain a reconstructed estimate F'^t. By decoding its value, we expect this estimate to be closer to the original value than the interpolated estimate $\bar{F}^t$. F'^t is then added to the curve as an extra keyframe.

2.5 Experimental Results

In this section, we describe some results from our integrated system. Since we are not aware of any existing method that addresses unreliable transmission of motion capture data, we did not compare our result with any prior work. Instead, we considered three different strategies for transmission, and compared their results to the original sequence:

- **Simplified Serialized Transmission:** This is the most straightforward approach. A traditional hierarchical skeleton has its keyframes packetized and transmitted in the same order as they are displayed. No interleaving, nor LDPC is applied in this case;

- **Interleaved Transmission:** Similar to the simplified case, a traditional skeleton is also transmitted. However, interleaving is applied before transmission takes place. The interleaving window D_I is not fixated, and it is specified for each test case;

- **Interleaved LDPC (I-LDPC) Transmission:** Here we apply the method described in Figure 2.1. Interleaving window D_I and LDCP offset D_L are specified for each test case.

The remainder of this section is organized as follows. In Section 2.5.1, we describe how packet loss is simulated in our experiments. Next, in Section 2.5.2 we provide details for each test case. Finally, in Section 2.5.3 we discuss evaluation results.

2.5.1 Packet Loss Simulation

As previously described in Section 2.4.1, keyframes are represented as packet fragments, which are subsequently concatenated and inserted into packets. For our experiments, we fixated the UDP payload size to 512 bytes. The reason

for this choice comes from the behavior of IP networks. The IPv4 standard specifies that every host must be able to reassemble packets of 576 bytes or less [29]. Considering both IPv4 and UDP headers, this gives us a 512-byte payload limit. By restricting the packet size, we ensure that it will not be re-fragmented on the network. Thus, it gives us more control over packet loss simulation.

Regarding the packet drop simulation, it is known that loss in IP networks is bursty. As previously mentioned, bursty channels are characterized by errors occurring in bunches, or bursts [18]. Furthermore, the burst length is variable, and it is not known beforehand. Given that we would like to simulate the loss rate as close as possible to a real loss scenario, we used the following mathematical equation to determine the probability P_n of a given packet n to be lost:

$$\begin{cases} P_n = Rand() & , n = 1 \\ P_n = P_{n-1} \cdot co + Rand() \cdot (1 - co) & , n > 1 \end{cases}$$

where the correlation parameter $0 \leq co \leq 1$ dictates how dependent P_n is to its predecessor, and $Rand()$ represents a random number in the range $[0, 1]$. Package P_n is dropped if $P_n < ch$, where $ch \in [0, 1]$ is simply a threshold.

The total loss rate is directly proportional to both ch and co, if the value of any of these parameters is increased, the loss rate will also be affected. These parameters allow us to simulate losses with a slightly variable burst length, which is closer to a real scenario. In Section 2.5.3, we indicate the average burst length produced for each test case, besides indicating the total loss rate.

2.5.2 Experimental Details

We performed tests on three motion clips extracted from the CMU Graphics Library[30], namely 03_03, 60_01 and 85_14. A short description for each clip is given below.

- **Clip 03_03:** subject walks on an uneven surface;

- **Clip 60_01:** salsa dance;

- **Clip 85_14:** breakdance.

These clips were selected based on how dynamic the animation is. Clip 03_03 presents a slow transition between poses, and there is not much movement for most of the joints. Whereas clip 85_14 is the exact opposite. For this clip the transitions are fast and joints are highly articulated. Clip 60_01 falls between these two. By selecting these clips, our goal is to demonstrate how loss affects motion in different scenarios, and also how much can be improved for each case. The most dynamic clips should be the most sensitive to data loss. Consequently, improvements should be clearer for them.

Before transmitting, all the clips were resampled. Tests were performed at $30fps$, which is a common framerate for low-budget sensors. Our goal is to demonstrate how the loss impacts clips at lower resolutions, which tend to be more sensitive to loss.

2.5.3 Discussion

To evaluate our transmission method, it is necessary to determine how "close" the decoded MoCap data is to the original data. A widely used metric for distortion is RMSE [31, 32, 33], which is also referred as the average joint error and can be expressed as

$$RMSE(P, \bar{P}) = \frac{1}{FJ} \sum_{i=1}^{F} \sum_{j=1}^{J} dist(p_i^j, \bar{p}_i^j)$$

$$p_i^j \in P, \bar{p}_i^j \in \bar{P}$$

where $P = \{p_1^1, ..., p_J^1, ..., p_J^F\}$ represents the set of original joint positions, $\bar{P} = \{\bar{p}_1^1, ..., \bar{p}_J^F\}$ is the set of distorted positions, F represents the total number of frames, J represents the total number of joints, and $dist(p_i^j, \bar{p}_i^j)$ is the Euclidean distance between joint positions. Joint positions can easily be calculated based on the motion data and FK model.

Objective results, based on $RMSE$ evaluation, are presented in Table 2.1. Transmission strategies are evaluated at different packet loss rates. The interleaving window D_I and the LDPC offset D_L were set to 30 and 15 frames, respectively. These offset parameters address the average burst length used for the test cases, equal to 12 packets. For this experiment, LDPC was not used to compress data. This means that the number of syndrome bits Z^t is equal to the number of bits used to represent a transformation $F_{(2)}^t$. The usage of less syndrome bits requires further testing, and it is out of the scope of this thesis. Although results will certainly be affected by increasing D_I, we believe the interleaving window is sufficient to address average burst. Based on presented results, it is possible to observe that while interleaving itself does not produce considerable gain in measured $RMSE$, I-LDPC results are greatly improved, especially for higher packet loss ratios.

Even though the calculation of the RMSE is straightforward, it is argued that it provides very little information about the perceptual closeness of the distorted motion from the original [31]. Even though other metrics have been introduced in order to achieve a higher correlation with human perception [34], this is still not a mature research area.

For this reason, multiple authors rely on user studies during evaluation [35, 36]. Given the issues regarding RMSE, we provide additional results based on user evaluation. Our evaluation follows the approach taken by Firouzmanesh et al. [36]. Essentially, we sign up a group of participants, and introduce them to the concept of quality degradation in motion data. Then, we present multiple versions of the same motion clip to the user. These versions are obtained from

Sequence	**Loss Ratio**	**Method**	$RMSE$ (in cm)
03_03	20%	Simplified	0.049
		Interleaved	0.036
		I-LDPC	0.021
	40%	Simplified	0.063
		Interleaved	0.054
		I-LDPC	0.034
	60%	Simplified	0.105
		Interleaved	0.101
		I-LDPC	0.063
60_01	20%	Simplified	0.073
		Interleaved	0.068
		I-LDPC	0.034
	40%	Simplified	0.114
		Interleaved	0.092
		I-LDPC	0.048
	60%	Simplified	0.239
		Interleaved	0.209
		I-LDPC	0.089
85_14	20%	Simplified	0.111
		Interleaved	0.071
		I-LDPC	0.042
	40%	Simplified	0.163
		Interleaved	0.142
		I-LDPC	0.063
	60%	Simplified	0.372
		Interleaved	0.323
		I-LDPC	0.064

Table 2.1: $RMSE$ results for different clips, at different packet loss ratios. An average burst length of 12 packets is used for all test cases.

clips that were transmitted and decoded under different conditions (i.e., using different methods). Furthermore, we ensure that the order of these clips is randomized before being presented to an user. Each user is asked to rank how close each clip looks, when compared to the original. The scores range from 1 to 5 (5 representing the best case), varied in steps of 0.5.

The set of motion clips used for this study is the same as before. However, we rely on higher packet loss rates and higher average burst lengths. We observed that deformations at smaller loss rates are not easily distinguishable by the human eye, even though they can be easily measured through $RMSE$. To address higher loss rates and larger bursts, simulation parameters were modified. The interleaving window D_I and the LDPC offset D_L were set to 90 and 15 frames, respectively. The number of syndrome bits Z^t remained unchanged for the user evaluation.

Perceptual differences for clip 60_01 are illustrated in Figure 2.2. These results represent a transmission affected by a 78% packet loss rate and with an average burst length of 28 packets. The burst length chosen for our experiments can be considered typical for such a loss rate [20]. Furthermore, this length also allows us to explore situations for which simple interpolation may be insufficient. Figure 2.2 is used to compare the transition between various poses for this clip. It can be observed that the I-LDPC algorithm produces poses that are closer to the original clip, compared to simple serialized transmission and interleaved transmission.

Perceptual differences for clip 85_14, with 74% loss and average burst length of 35 packets, are shown in Figure 2.3. As with the previous example, it can be seen that the I-LDPC algorithm produces transitions that are closer to the original clip, compared to the other approaches.

The MoCap clip 03_03 was transmitted with 74% loss and an average burst length of 28 packets, as we wanted to simulate a similar loss rate for all the clips. Illustrations are omitted for this clip. Given that 03_03 does not have

Method / Clip	03_03	60_01	85_14
Simple	3.6	3.5	2.3
Interleaved	3.6	3	3
I-LDPC	**4**	**4**	**4.1**

Table 2.2: Average of perceptual evaluation scores for different clips, transmitted with three distinct approaches. Ratings used for this evaluation ranged from 1 to 5. Best performing algorithm for each clip is highlighted in bold.

fast transitions between poses, and also the fact that it is harder to notice perceptual loss from image sequences, we only show the evaluation results for this clip.

These 3 motion clips were evaluated by the 12 participants from our user study. Results are illustrated in Table 2.2. Our user evaluation results present a high correlation with our previous $RMSE$ evaluation. As we can observe, I-LDPC has a higher rating for all the test cases, especially when the animation is considered more dynamic, like 85_14. This demonstrates that animations with faster transitions between poses are more sensitive to data loss, and benefit more from an optimized transmission.

Another important information that can be extracted from these results is that the difference between the simple and the interleaved transmission is not significant. Given that we have an extremely high loss rate, around 74%, only interleaving keyframes is not sufficient to mitigate loss. Although interleaving window D_I is much higher than the average burst length, it is possible that for such loss rates the bursts are considerably close to each other. The inability to address this issue by only interleaving reinforces the importance of integrating LDPC into the method.

2.6 Conclusion

In this chapter we presented a new method to optimize transmission of MoCap data over unreliable channels with loss. We studied various alternatives for

protection against data loss. Subsequently, we introduced an Interleaved LDCP (I-LDPC) method to make the transmission more robust. Our approach does not require retransmission; thus, it adds no overhead on the network during transmission.

Conducted studies demonstrate quantitative and perceptual gain in the quality of the motion using I-LDPC over alternative approaches, especially for animations that contain faster transition between poses. In future work, we intend to perform a more thorough evaluation, in order to validate the gain in perceptual quality attained by our method. We also plan on expanding the proposed method to cover more FEC techniques.

In this work we show results incorporating burst errors. However, a more thorough analysis of how the average burst length affects the performance of various algorithms needs to be studied in future work.

Figure 2.2: Illustration of perceptual quality for clip 60_01 at 78% loss. For the simplified transmission, both the upper and lower body are severely affected. The interpolation between poses does not look realistic. For most of the clip, including this illustrated segment, the feet are affected by an unnatural sliding effect. The interleaved transmission offers some gain on transition between poses, especially for the lower body, as seen on the third row. The interleaved LDPC transmission, shown on the last row, is the closest to the original, as the individual poses are the most similar.

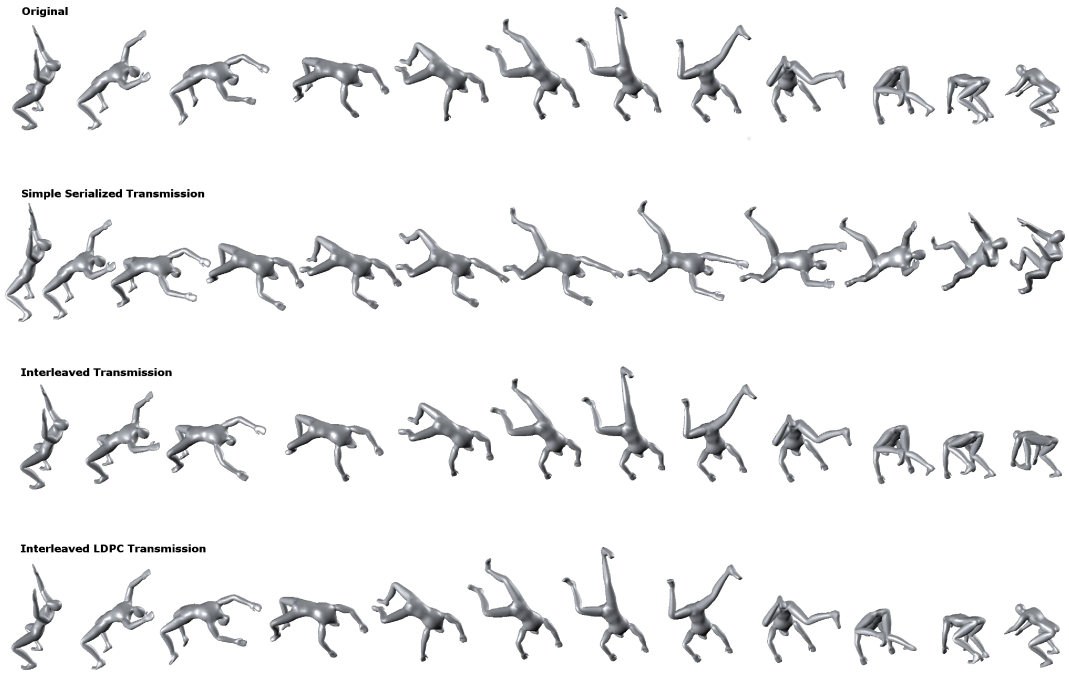

Figure 2.3: Illustration of perceptual quality for clip 85_14 for 74% loss. For the simple serialized transmission of a skeleton, rotation information from the root joint is severely affected during the backflip, causing major distortion. Quality is improved when interleaving is introduced, on the third row. However, the last pose still contains some distortion for the leg. The I-LDPC method, shown on the last row, is the one that is closest to the original.

Chapter 3

Compression

3.1 Introduction

Another challenge for applications using motion sensor technology is a large volume of MoCap data. Similar to other multimedia data types, raw motion data is associated with redundancy and noises. For visualization-centric applications, robust compression algorithms should be able to eliminate or reduce data irrelevant to the application and preserve satisfactory perceptual quality of the displayed data after decompression.

Compression of full-body MoCap data has been studied extensively [31, 32, 33]. However, for many applications, such as surgical skill assessment and sign language recognition, full body tracking is not necessary. It may even be unavailable, depending on the type of sensor being used. In this chapter, we investigate a lossy compression technique designed for hand motion data. Even though hand motion data follows the same model structure as a full-body model, there is a notable difference, which can be exploited. Hand models have a significant smaller number of degrees of freedom per joint. Channels that represent these degrees of freedom tend to be highly correlated. For most hand gestures, fingers present some type of synchronization regarding their movements and can easily be spotted.

We introduce a novel method to exploit correlation in hand motion. The method is an effective variable compression method at per-joint level, based on the fact that each joint has a different impact on the overall motion distortion. Furthermore, we demonstrate the quantitative gain of our approach, compared to a fixed compression method, for different compression ratios.

The remainder of this chapter is organized as follows. In Section 3.2 we present related motion compression methods. Section 3.3 describes the distortion measure that we aim to minimize during compression. Then, in Section 3.4 we propose our compression approach, which is followed by evaluation results, presented in Section 3.5. Finally, we give the conclusion in Section 3.6.

3.2 Related Work

MoCap compression has been a main focus in research studies. In this section, we compare our approach with some main works in this area. Arikan [31] represents body joints as virtual markers, which are derived from joint orientation data. These markers are fitted into splines. The control points are later compressed by using clustered Principal Component Analysis (PCA) [37]. We refer his method as $Clip-PCA$. Tournier et al. [32] propose a method that uses Principal Geodesic Analysis (PGA) to build a descriptive model of the pose data of a motion, keeping only the leading principal geodesics. This model yields a reduced pose parametrization that is used by an Inverse Kinematics (IK) algorithm. This algorithm is used to recover original poses, solely based on end-joints positions. We refer this method as $PGA-IK$. Following a different approach, Lin et al. [33] introduce a more sophisticated method that relies on a pre-processing step, which classifies and indexes body poses in three distinct categories: unique, primary and repeated. Repeated poses are compressed more compactly, by making use of the variable variance characteristic in PCA and adaptive-bit quantization. We refer his method as $RepeatedMotionAnalysis$.

All of the aforementioned methods target full-body compression. Therefore, none of them presents results for hand motion data, which suggests inadequate research on hand motion compression given the usefulness of this type of data for many applications, such as surgical simulation assessment. To the best of our knowledge, this chapter is the first work to present compression results for hand motion data. We will compare our approach with related work, based on a modified version of our method targeting full-body compression.

3.3 Distortion Metric

The goal of lossy compression methods is to maximize the compression ratio that is applied to the original motion, while minimizing the perceptual distortion for the decoded motion. Just as lossy transmission methods, the $RMSE$ is a also traditional distortion metric for compression methods, and it has already been presented in Section 2.5.3.

In the case of hand motion data, we propose a slightly different metric, based on $RMSE$. Our metric only evaluates fingertip positions and thus is computationally less expensive. When assessing manual task skills, the deviation in the position of the hand root is not considered as relevant compared to the deviation in the position of the fingertips. Furthermore, fingertips are more susceptible to errors, because movement errors tend to propagate down the human skeleton hierarchy. Therefore, fingertips provide a better error estimation in term of visualization. Our metric, named $RMSE_{End}$, is defined as follows:

$$RMSE_{End}(P, \bar{P}) = \frac{1}{FE} \sum_{i=1}^{F} \sum_{j=1}^{E} dist(p_i^j, \bar{p}_i^j)$$

$$p_i^j \in P, \bar{p}_i^j \in \bar{P}$$

where $E \subset J$ represents the set of end joints.

In practice, the computation of $RMSE_{End}$ is not viable for real-time ap-

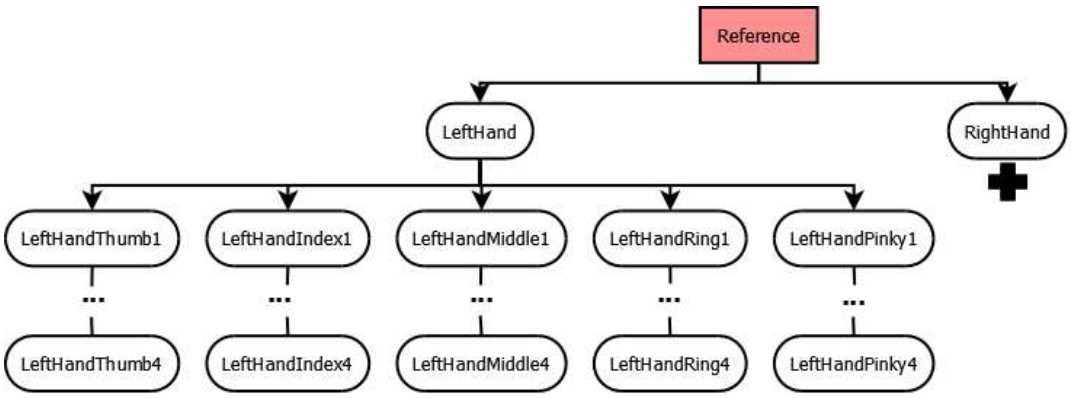

Figure 3.1: A traditional Forward Kinematics (FK) model used to represent a pair of hands. The right hand hierarchy has been collapsed, and it follows the same structure as the left hand. Each palm joint (i.e. LeftHand or RightHand) contains 6 degrees of freedom (DOFs). First joint of each finger has 3 DOFs, while second and third joints can only be rotated around 1 axis.

plications because as F gets larger, the cost of measuring the distance for every joint at every frame becomes expensive. Besides, when minimizing the error, the metric likely needs to be calculated multiple times. For this reason, we only sample $RMSE_{End}$ measures, where sampled neighboring frames are equidistant.

3.4 Proposed Computational Model

In our computation, we assume that hands are represented in a traditional Forward Kinematics (FK) model (i.e. hierarchical skeletal representation). The skeletal structure used in our computation is illustrated in Figure 3.1.

Motion data applied to a skeleton model is known to display both spatial and temporal coherence [31]. In other words, compression methods can remove redundancy by exploiting: (1) the spatial correlation between neighboring joints and, (2) the temporal correlation of poses between neighboring frames. For most hand gestures, fingers present some type of synchronization regarding their movements and can easily be spotted. This spatial coherence for finger movements is illustrated in Figure 3.2.

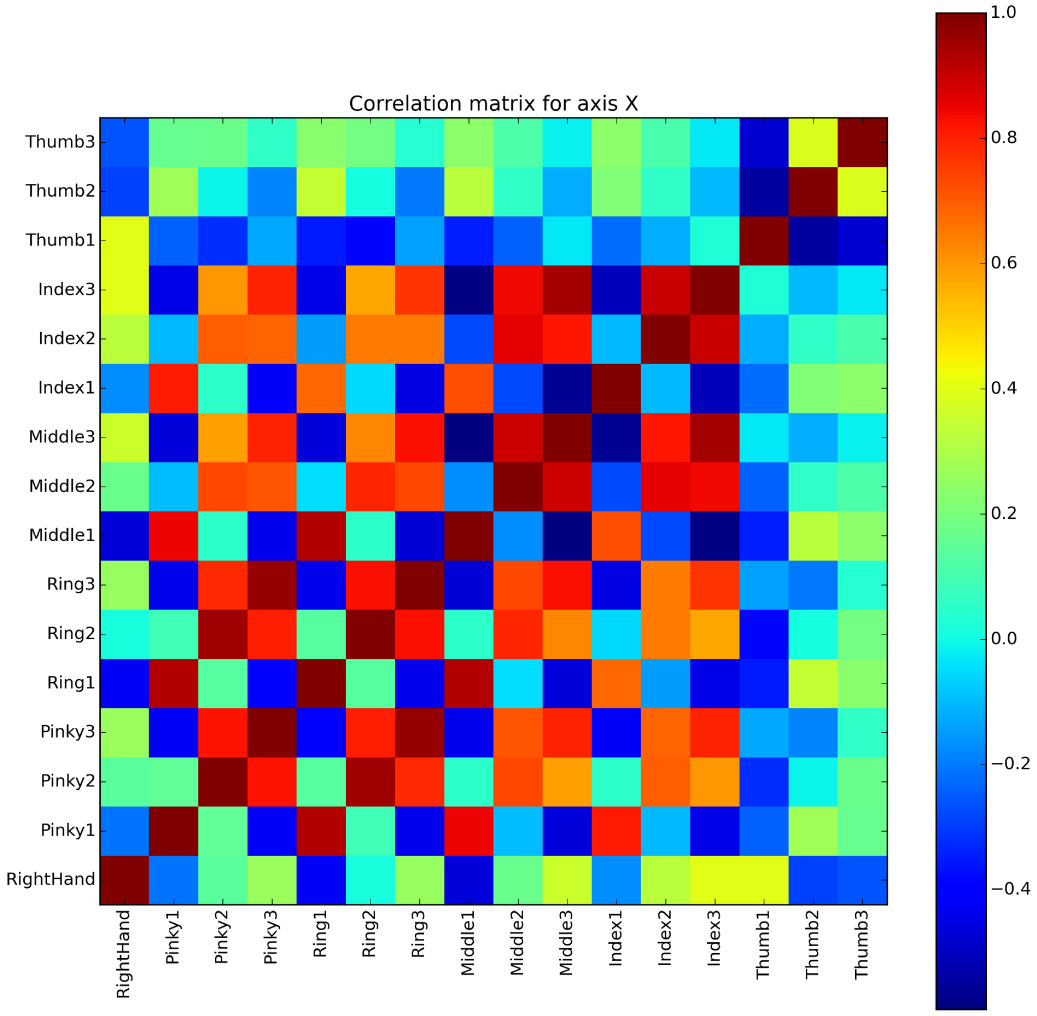

Figure 3.2: Pearson correlation plot for rotation curves around the x axis. Curves were extracted from a simulated line knotting motion clip. For better visualization, left hand joints were omitted. It is possible to notice that finger joints at the same level in the hierarchy (e.g. Index3, Middle3 and Ring3) are noticeably correlated.

Given that motion data for fingers is correlated, and that most finger joints have a single degree of freedom (DOF), we can summarize them into a set of principal components, which can later be represented as channels. We achieve this through Principal Component Analysis (PCA) [37]. The number of components to be used depends on the acceptable distortion. In this work, we apply a PCA reduction that retains at least 95% of the total variance.

After the component reduction, our method proceeds to compress each channel through keyframe reduction. The goal is to exploit temporal coherence in motion data. Channels are commonly represented by using Bezier cubic splines. For each frame, splines will contain a corresponding keyframe, used as a control point. Since Bezier splines are known to be robust interpolation techniques, it is possible for us to reduce the amount of control points being used without losing too much perceptual information. Channel compression can now be handled similar to curve simplification. Curve simplification is a well-studied field [38, 39]. In our implementation, we use the curve simplification algorithm proposed by Dierckx [40]. The algorithm allows us to fine-tune a non-negative smoothing parameter, namely *smo*, which can be used to regulate the amount of control points. Given a curve c, containing n keyframes, and a Bezier spline that fits c, we can define *smo* as follows

$$\sum_{i=1}^{m}(c(i) - s(i))^2 \leq smo.$$

Basically, *smo* represents an upper bound between the sum of differences between the original curve and a fitting spline.

We observed that distortions added to different channels cause a different overall distortion, measured by $RMSE_{End}$. This happens because motion error added to a higher level joint is propagated down the hierarchy and amplified. A major contribution of our work lies in proposing an optimization step, which searches for the optimal joint smoothing configuration, for any desired com-

pression ratio. In other words, our algorithm can find the optimal distortion for each joint and compress each channel appropriately.

3.4.1 Joint Smoothing Configuration

We start by defining an initial joint smoothing configuration. We can represent any configuration through the set $S = \{smo_1, smo_2, ..., smo_{|M|}\}$, for which M represents the motion data that is applied to a given FK model H, $|M|$ returns the the number of channels from M and smo_i represents the smoothing applied to channel i. This set is used by the simplification function $Simplify(M, S)$, which returns a set M_s of simplified channels. Our method starts by taking an initial fixed configuration S_0, for which all the channels are subjected to the same distortion, therefore

$$S_0 = \{smo_i = k : 1 \leq i \leq |M|\}$$

In this configuration, k is directly proportional to the compression ratio. Our goal is to find a better configuration, resulting in a smaller overall distortion, for the same compression ratio. Our problem can be formalized as follows

$$\begin{aligned} \underset{S}{\text{minimize}} \quad & RMSE_{End}(P, P_s) \\ \text{subject to} \quad & CRatio(M, S) = CRatio(M, S_0) \\ \text{where} \quad & P_s = Eval(H, Simplify(M, S)). \end{aligned}$$

where $CRatio$ measures the compression ratio achieved by the simplification, and $Eval(H, M)$ is simply the evaluation function for a FK model, returning joint positions. The solution for this problem is not trivial, given that the set S is composed of several variables. Furthermore, this optimization problem is not linear, since it is affected by rotational data.

Given the definition of smoothing parameters, we propose an heuristic

method to solve the minimization problem, following an iterative approach similar to gradient descent. We start with an initial smoothing configuration $S \leftarrow S_0$. At each step, we both add and subtract a small disturbance α to each element in the set. Then we measure the difference in the distortion using $RMSE_{End}$. These measures are used to decide which channel causes the least amount of distortion by increasing its smoothness, and also which channel causes the most gain by decreasing its smoothness. The values for these channels are updated, and the algorithm proceeds to the next iteration.

Our iterative optimization approach has two stop conditions. First, it stops if it reaches a predefined number of iterations. It also stops if no optimization could be achieved for the last $maxCount$ iterations, which is also predefined value. Our heuristic optimization is given in Figure 3.3. Some associated processes are defined as follows:

- $Distortion(H, M, S)$: $RMSE_{End}$ evaluation;

- $LocalOptimization(S, i^+, i^-, M, \alpha)$: responsible for defining $\alpha^-, \alpha^+ \leq \alpha$, used to decrement and increment S at positions i^- and i^+, respectively. α^- and α^+ are defined in a way, such that compression ratio $CRatio$ remains the same.

After temporal coherence is exploited through our curve simplification process, keyframe data is undergone an arithmetic encoder for further compression. Our compression scheme is given in Figure 3.4.

3.5 Evaluation Results

Any smart sensor can be used to record hand motion data, but in our current setup, we used the LEAP motion sensor to record hand motion data to evaluate our method. The orientation data obtained from the sensor has been fitted into

Require: Initial smoothing configuration S_0,
maximum number of iterations $MaxIter$,
maximum number of optimization trials $maxCount$,
tolerance ε,
disturbance α, hand model H
and motion data M {End of Requirements}
$S \leftarrow S_0$
$exitCount \leftarrow 0$
for $1..MaxIter$ **do**
  $CurCost \leftarrow Distortion(H, M, S)$
  $\Delta Cost^- \leftarrow ZeroArray(|S|)$
  $\Delta Cost^+ \leftarrow ZeroArray(|S|)$
  for $i \leftarrow 1..|S|$ **do**
    $S^+ \leftarrow S$, $S^- \leftarrow S$
    $S^+[i] \leftarrow S^+[i] + \alpha$
    $\Delta Cost^+[i] \leftarrow CurCost - Distortion(H, M, S^+)$
    $S^-[i] \leftarrow S^-[i] - \alpha$
    $\Delta Cost^-[i] \leftarrow CurCost - Distortion(H, M, S^-)$
  end for
  $i^+ \leftarrow index(\max(\Delta Cost^+))$
  $i^- \leftarrow index(\max(\Delta Cost^-))$
  $S' \leftarrow LocalOptimization(S, i^+, i^-, M, \alpha)$
  if $|CurCost - Distortion(H, M, S')| < \varepsilon$ **then**
    $exitCount \leftarrow exitCount + 1$
    if $exitCount > maxCount$ **then**
      return S'
    end if
  end if
  $S \leftarrow S'$
end for
return S

Figure 3.3: Heuristic that finds an optimal smoothing configuration S.

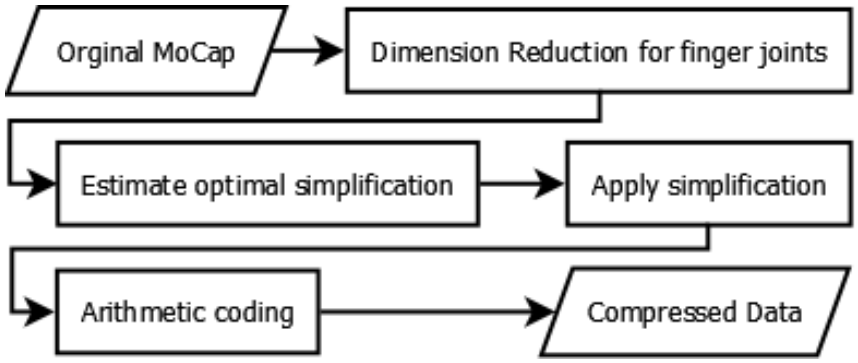

Figure 3.4: Flow diagram of our variable compression scheme

our proposed hand skeletal model. In this experiment, a subject is asked to perform a line knotting task involving the use of both of his hands.

In Figure 3.5, we illustrate the execution of our algorithm for a clip compressed at a 21 : 1 ratio. When compressing this file under a fixed compression scheme, the $RMSE_{END}$ measure at this ratio is approximately 0.00843 cm. At the first iteration of our method, we are able to reduce the distortion to approximately 0.0084 cm, which represents a 0.5% gain. At the 10^{th} iteration, the $RMSE_{END}$ is further reduced to about 0.00598 cm, indicating a 29% relative gain. Besides demonstrating the iterative gain of our method, we use a colormap to encode pairs of channels that have their smoothness configurations modified. Given that we have a large number of channels, 50 for this experiment, a colormap allows us to visualize how our method explores its search space. We can observe that, even though we only run our method for few iterations, pairs in different areas of our search space are considered by our optimization process.

We can also observe that $RMSE_{END}$ is not constantly decreased at every step. For some iterations, some loss is allowed by our method. This happens because our method does not guarantee that $|\Delta Cost^{+}| < |\Delta Cost^{-}|$. The lack of a constant improvement did not seem to pose an issue during our evaluation, as the method ended up converging to a minimized distortion at the end of the 10^{th} iteration. However, we cannot guarantee that this method will always

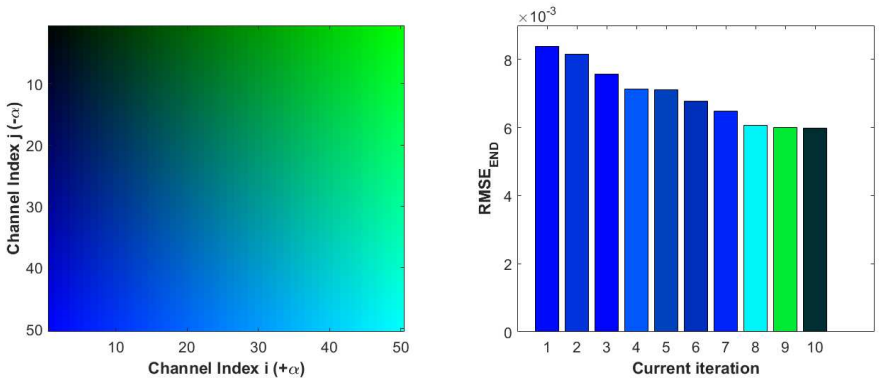

Figure 3.5: We illustrate the iterative resource reallocation $RL(i, j)$ performed by our heuristic approach. In each iteration, a given pair of channels i and j have their smoothness configuration modified (i.e. $i \leftarrow i + \alpha$ and $j \leftarrow j - \alpha$). On the left plot, pairs of channels (i, j) are encoded by using a colormap. On the right plot, we demonstrate the distortion minimization performed by our heuristic after 10 iterations. Each iteration is represented by a different color, depending on the pair of channels that had resources reallocated.

converge. This is a natural consequence of its gradient-based approach, as the method may end up getting stuck at local minimums.

Additionally, we illustrate the perceptual gain of our approach in Figure 3.6. For this illustration, we selected a frame sequence from our motion clip, for which frames are equally spaced. This time, we used a $27 : 1$ compression ratio, to make results easier to be visualized. The distorted view is overlaid on the original motion (in red). We can observe that our variable approach has a resulting motion closer to the original, especially for the indicator, which is being raised. Even though both results are not very discrepant, some applications, such as medical assessment tools, may have strict requirements regarding the quality of the compressed motion.

To the best of our knowledge, this chapter is the only compression method specially targeted at hand motion data. Unfortunately, this limits comparison with previous methods, which are mostly tailored towards full-body compression. Therefore, we implemented and tested a limited version of our approach for full-body compression. In this implementation, we focus on testing the ef-

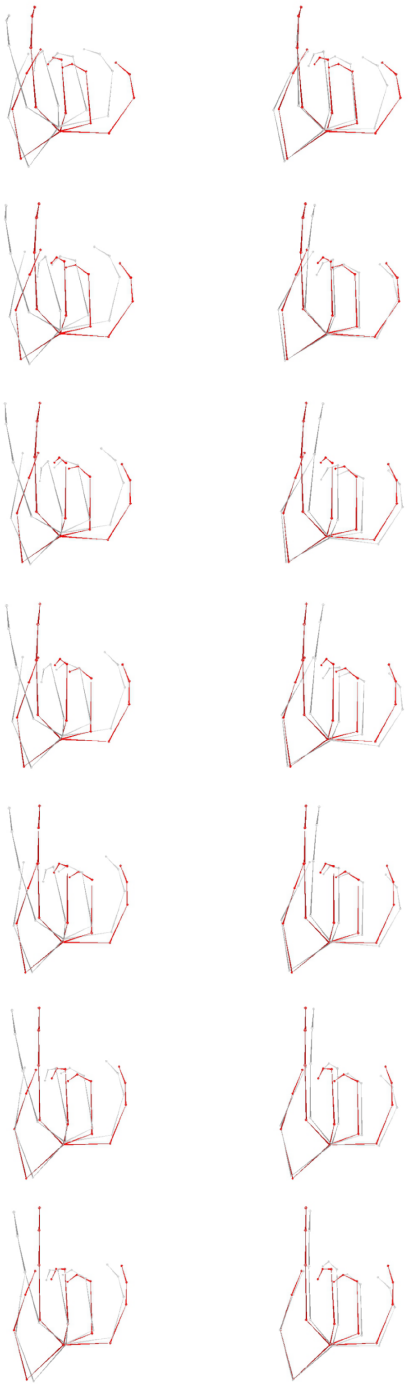

Figure 3.6: Perceptual difference between a fixed compression scheme (on the left side) and our proposed optimized scheme (on the right side). Original motion (in red) is displayed along its distorted version (in gray). Note that the distortion generated from our method is visually smaller than that of a fixed compression scheme.

Method	**Compression Ratio**	$RMSE$
Fixed Compression	62 : 1	3.65 cm
Proposed heuristic	62 : 1	2.6 cm
Repeat Motion Analysis[33]	62.6 : 1	2.65 cm
Clip-PCA[31]	9.2 : 1	3.16 cm
PGA-IK[32]	61.1 : 1	3.15 cm

Table 3.1: Results obtained for clip 17_10 (Boxing).

Method	**Compression Ratio**	$RMSE$
Fixed Compression	58 : 1	4.72 cm
Proposed heuristic	58 : 1	4.48 cm
Repeat Motion Analysis [33]	60.1 : 1	5.01 cm
Clip-PCA [31]	11.1 : 1	4.78 cm
PGA-IK [32]	97.1 : 1	4.02 cm

Table 3.2: Results obtained for clip 85_12 (Breakdancing).

ficiency of our heuristic in minimizing motion deviation error. For this reason, dimension reduction step is not applied.

Results are presented for two distinct clips from the CMU Graphics Library [30], namely 17_10 and 85_12. Results for clip 17_10, which contains a subject performing a boxing movement, are presented in Table 3.1. In this comparison, we resort to the more traditional $RMSE$ metric, given the data type and the ability to compare with other methods. Results for clip 85_12, which contains a subject performing a breakdancing movement, are presented in Table 3.2.

Given the results presented, it is possible to observe that our method is able to find an optimized configuration for full-body compression as well, even in its limited version. Given that full-body motion clips contain a higher number of joints and more redundancy, there is more room for improvement. These comparison results suggest that our heuristic can deliver good compression ratio and preserve good visual quality in other types of motion data, such as full-body motion. Additionally, we observe that our results are comparable to the state-of-the-art.

3.6 Conclusion

We proposed a lossy compression technique for hand motion data. Our method takes advantage of the different impact of each joint on the overall motion distortion, and is a per-joint compression level technique. We demonstrated that our approach outperforms the fixed compression method, which treats all joints with the same importance. At least a 10% gain is observable for different compression ratios when adopting our algorithm. Our method follows a heuristic iterative approach, and it is demonstrated that it can minimize overall distortion after only a few steps.

In future work, we intend to use our algorithm for compressing medical training data. We intend to demonstrate the capability of our method to compress a larger amount of motion data for surgical training.

Chapter 4

Biometrics

4.1 Introduction

Although efficient storage and transmission are important to consider when dealing with any type of multimedia data, such as MoCap, they only serve to assist other applications. This chapter is used to demonstrate one of the most promising, yet commercially unexploited applications of MoCap data, which is gait recognition. This recognition process aims to identify users based on their walking style.

There is considerable evidence in biomechanics, psychology and literature supporting the notion that gait is biometric [41]. Gait can also be considered one of newest identification methods, as early work was first developed in the 1990s, when computer memory and processing became sufficient to process sequence of images. Since then several other gait methods have been introduced, as more processing power became available and image and depth sensors have been improved.

Gait identification could play an important role in surveillance systems. There has always been a need for efficient security architectures in public access areas. In areas of high pedestrian activity, authentication and verification is practically impossible. Other available biometric features, such as face [42],

hand geometry [43], fingerprints [44], iris [45] and voice [46], can only be captured at a close distance, and may require user interaction. Furthermore, they are prone to obstruction (e.g. use of face masks). Gait, on the other hand, has the advantage of being easier to capture in a controlled environment and harder to obstruct.

We present a computationally simple and practical gait recognition method, which relies only on joint flexion angle information. Our method is based on a rank-aggregation technique, and it is intended for general identification purposes. We organize this chapter as follows. Section 4.2 is used to review related gait recognition approaches. Section 4.3 reviews currently available gait databases, and describes their major differences. In Section 4.4, we describe our method. Section 4.5 presents evaluation results. Finally, Section 4.6 offers a conclusion.

4.2 Related Work

Although gait information can easily be derived from kinematic models, the use of such models is not unanimous in research work. Basically, gait recognition methods can be divided in two main categories: appearance-based and model-based [47]. While model-based approaches identify individuals based on kinematic characteristics, as seen in Section 1.2, appearance-based approaches consider the body as an integral part, in which it is usually represented by silhouettes or contours. The latter is considered the most popular in gait recognition, since it is traditionally less computationally expensive to estimate. In most cases, silhouettes are extracted from images acquired by a single camera, through background subtraction, as illustrated in Figure 4.1.

In this section, we briefly describe the most relevant appearance-based methods [48, 49, 50]. Kale et al. [48] define a HMM to model the individual gait. A single HMM will be trained for each individual in the dataset. Five

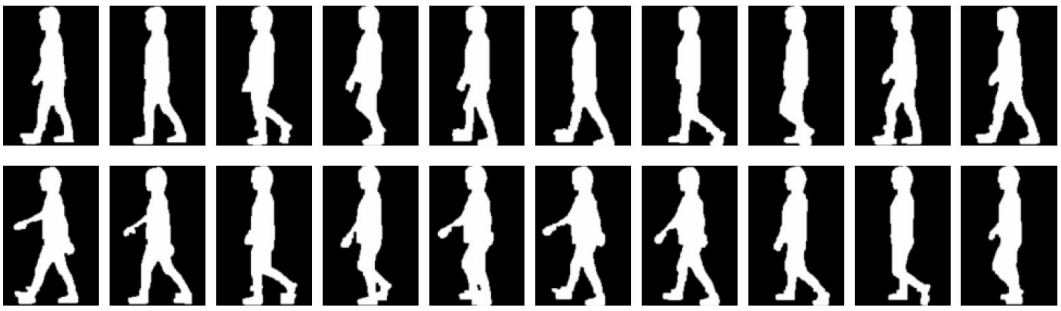

Figure 4.1: Typical silhouette images used by appearance-based methods.

representative silhouettes are used as hidden states, in order to train transition probabilities and observation likelihoods. During recognition, the HMM with the largest probability will be identified as a successful match. BenAbdelkader et al. [49] proposed an approach that relies on self similarity and structural stride parameters. Initially, self similarity plots are derived from differentiating, they encode both the phase and frequency of the gait. Afterwards, PCA is applied to these plots and kNN is used for classification in the reduced space. Sakar et al. [50] relies on temporal correlation of silhouettes for recognition. After silhouette extraction, gait detection is performed through a simple strategy, in which foreground pixels are counted. The similarity score is computed between all gallery and probe gait sequences.

For model-based approaches, each frame of a walking sequence is fitted into a model, from which different gait parameters are extracted. These methods tend to be easier to understand, however the computational complexity is generally higher. Bobick [51] developed a method that recovers static body and stride parameters from video sequences. Two sets of parameters are proposed for classification, one being a subset of the other. Among included parameters, they rely on the distance between head and foot, distance between left foot and right foot, etc. The within-class and between-class variation were analyzed to determine their discrimination power. Whang et al. [52] presented a method that starts by extracting several contours, and a mean contour is computed to represent the static body information. Dynamic information is extracted by

using a detailed model composed of 14 body parts. Particle filtering is used to compute the likelihood of a pose from an image. Classification is performed through kNN. Krzeszowski et al. [53] proposes a recognition method achieved through dynamic time warping on the normalized joint-angle information, followed by a kNN classifier on euclidean distances. The joint-angle information is extracted from the estimation of a 3D human model. The body pose estimation takes place on video sequences obtained from 4 different calibrated and synchronized cameras. They rely on additional subject height information to strengthen their results.

All the reviewed model-based gait recognition methods also propose their own body pose estimation approaches from video sources [51, 52, 53]. Some also include additional results for data derived from MoCap systems [51, 53], although none of them are from public repositories. Even though body pose estimation and gait recognition are two distinct problems, current approaches tend to span all of them. Given that many recent commodity depth sensors are able to provide out-of-the-box body pose recognition functionality, such as the Microsoft Kinect [54], we believe that it is no longer necessary for research papers to address these problems altogether. Therefore, our main focus on this chapter is address the gait recognition, regardless of the source of the MoCap. Furthermore, we restrict our evaluation to open MoCap databases, so results are reproducible and comparable.

Additionally, our approach does not rely on information that is not directly related to gait to improve accuracy, such as subject height [51, 53]. Even though many body parameters are useful to distinguish subjects, it does not demonstrate the classification potential of gait patterns.

4.3 Motion Database

The success of biometric applications relies heavily on the dataset used for evaluation. Multiple differences are found in existing public gait databases, given the year they were collected and the different goals. In Table 4.1, we summarize the main existing gait databases.

The first available databases: CMU [55],SOTON [56],USF [50] and CASIA [57]; consist of multi-view video datasets. Videos are post-processed, and silhouettes are extracted based on different techniques. While these databases share a similar capture system setup, they use different sets of control parameters.

In the CMU [55], subjects are only evaluated on a treadmill. Videos are recorded at 6 viewpoints. Unfortunately, calibration data is no longer made available for these viewpoints. Control parameters include different walking speeds and whether an user is carrying an object or not. The SOTON [56] database not only evaluates the subject on a treadmill, but also on walking on the floor, both outdoors and indoors. They only rely on two uncalibrated cameras to record each trial. Similarly, the USF [50] database also uses two cameras. However, it provides a calibration board to assist body estimation methods. Control parameters include different walking surfaces and shoes.

Different from the previous multi-view databases, TUM GAID [58] relies on a single RGB-D camera and an audio stream. This database addresses the emergence of low-priced consumer depth cameras. Control parameters include use of different shoes and backpacks.

The CSU [59] is a database aimed to test the effect of mechanical perturbations on human gait. It can be considered the most distinct database among those being presented. Besides being open-access, it is the only to offer high-precision MoCap data. While it is not adequate for body pose estimation methods, it can be used for recognition purposes, despite its smaller size.

Database	Year	Subjects	Data type	Access
CMU [55]	2001	25	Video/Silhouettes	Under Request
SOTON [56]	2004	118	Video/Silhouettes	Open (Silhouettes only)
USF [50]	2005	71	Video/Silhouettes	Open (Silhouettes only)
CASIA [57]	2011	124	Video/Silhouette	Open (Silhouettes only)
TUM GAID [58]	2014	305	RGB-D/Audio	Under Request
CSU [59]	2015	15	Positional MoCap	Open

Table 4.1: List of main gait databases

Given that our chapter is aimed strictly at gait recognition, we only rely on this database for our evaluation. In Section 4.4, we give more details on how it is organized.

4.4 Method

Apart from the vast majority of related work on gait recognition, we do not extend our work to body pose estimation, as we believe these are two distinct problems. As previously mentioned, our gait recognition is directly applied to MoCap data.

For experimental purposes, we use the CSU MoCap database [59]. This database includes 11 males and 4 females with the following averaged features:

- Age: 24 ± 4 years;
- Height: 1.75 ± 0.09 meters;
- Mass: 74 ± 13 kilograms.

Each subject was subjected to at least 3 trials. In each trial, subjects were required to walk on a split-belt treadmill, under perturbed and unperturbed conditions, for 10 minutes. During the exercise, a set of 47 highly accurate 3D positional markers have their trajectories recorded. An illustration of the treadmill is given in Figure 4.2. The subjects walk along the $-Z$ axis.

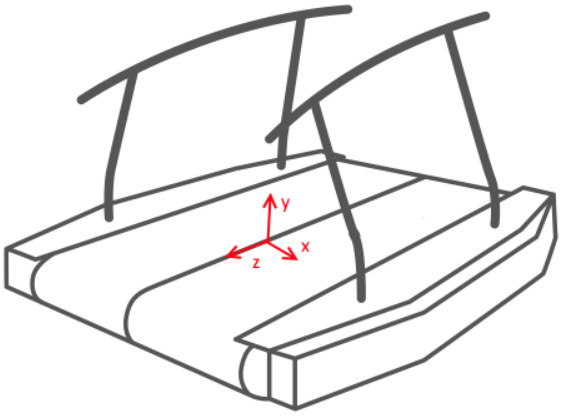

Figure 4.2: Illustration of split-belt treadmill used for data collection, along with coordinate system.

Our method relies heavily on joint angle data, For this reason, our first step is to project the human body into a 2D plane in order to simplify the analysis. Simplified 2D kinematic models are common in gait analysis [60]. However, in many cases they are used simply because of a monocular capture system, which is subjected to self-occlusion issues. In our approach, we rely on IK to obtain 2D joint flexion angles from 3D marker positions. Therefore, our approach is not affected by the same occlusion issues. Our simplified model is illustrated in Figure 4.3.

As illustrated, we rely only on flexion angles from limbs belonging to the lower-body. At total, there are six angles used for analysis. We ignore upper-body limbs, as they may become irrelevant in real use case scenarios. The simple act of holding a box with your arms is sufficient to inhibit the natural arm swinging movement. We believe that any gait recognition method should be robust enough to extract information only from lower-body limbs.

Additionally, our model-based method treats each flexion angle independently. This means that information derived from the right side of the body is handled independently form the left side. Even though this may appear redundant, we believe this is a differential part of our method. Multiple recognition methods, especially appearance-based ones, assume the gait movement to be

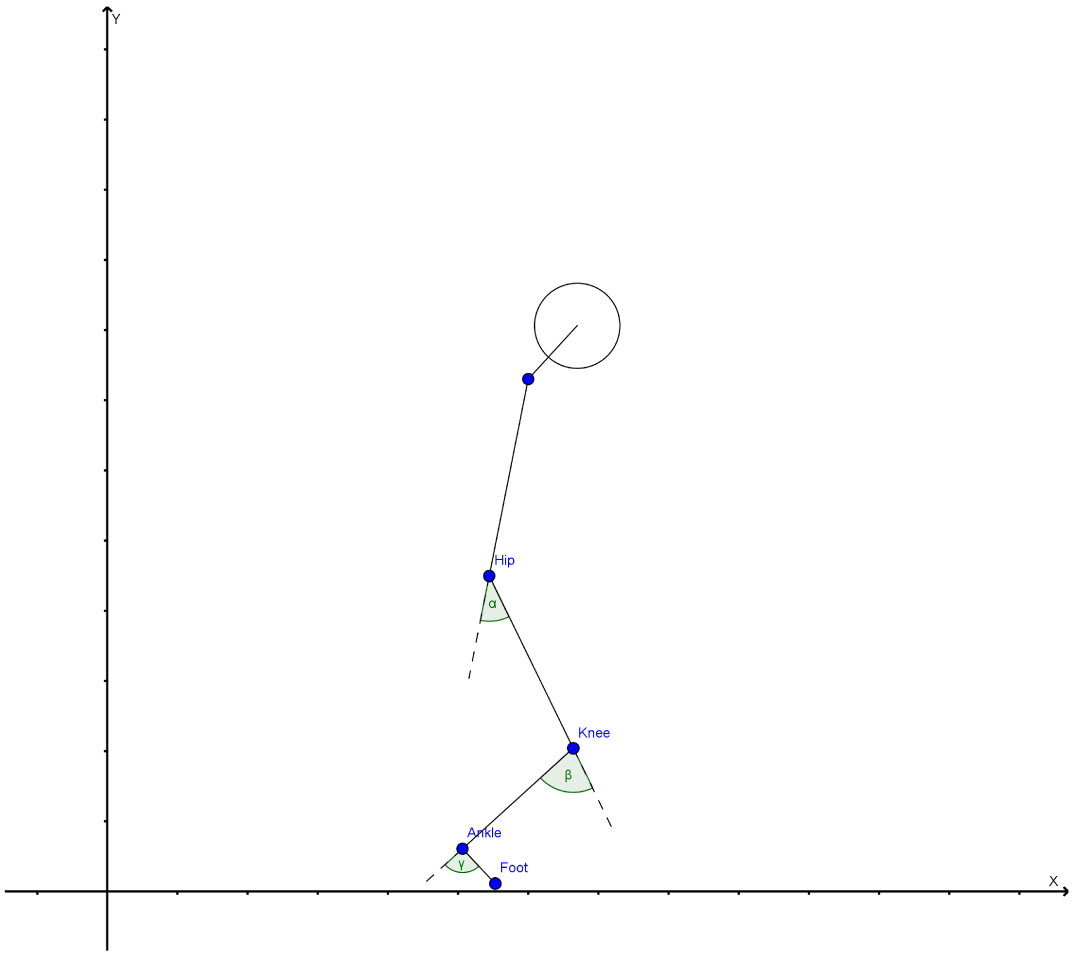

Figure 4.3: Illustration of 2D kinematic model, for which subject is assumed to walk along the X axis. For simplification purposes, only lower-body limbs from one side are displayed. Our method relies on three types of extracted angles: hip flexion (α), knee flexion (β) and the ankle plantar flexion (γ).

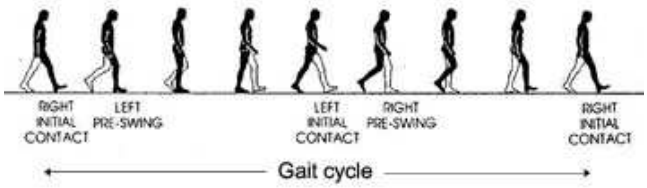

Figure 4.4: Illustration of a gait cycle (i.e. partitioning of a gait sequence that depicts a complete walking period)

symmetric. However, gait asymmetry is a well-known characteristic, even for able-bodied subjects [61].

Besides extracting flexion information, another important part of a gait analysis process is the gait cycle detection. The gait cycle is illustrated in Figure 4.4. Even though there is not a single way to extract the cycle, almost all approaches use a time series corresponding to a measure extracted from the sequence (e.g. number of pixels present in a silhouette). In our approach, we rely on the force plate system information provided by the CSU database [59]. The treadmill used in their experiments is equipped with two force plates, one for each belt. Each plate records three ground reaction forces (i.e. F_x, F_y, F_z). We simply threshold the ground force on the vertical axis (i.e. when $F_y < \tau$, foot is not touching the ground).

This straightforward cycle detection approach gives accurate results. However, we acknowledge that ground force data may not be available in real use case scenarios. For other databases, thresholding can be applied to other data sources, such as joint angle acceleration.

4.4.1 Gait Cycles

Once cycles are segmented, angle data can be scaled according to relative gait percentage, instead of absolute time. This allow us to compare cycles at different speeds, besides making it easier to sample the data. In Figure 4.5, we illustrate flexion data for 3 different joints. A set of 43 gait cycles is represented, all from the same trial, recorded in sequence. For clarity purposes, not every

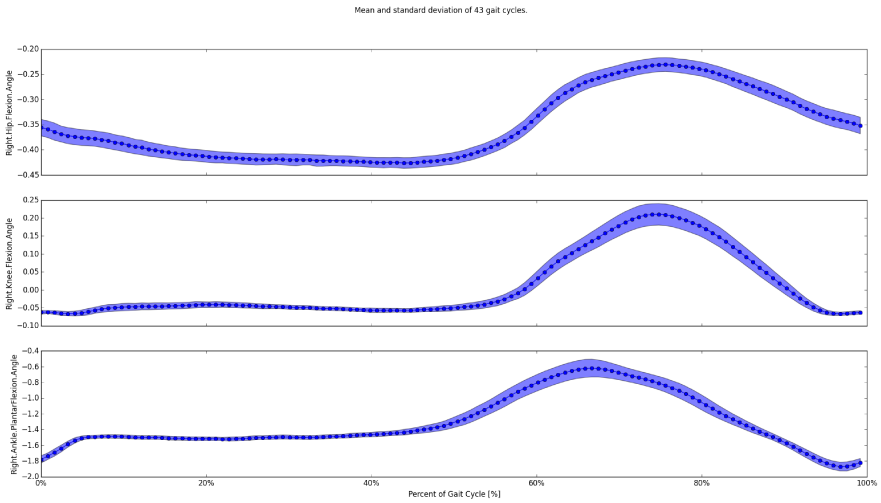

Figure 4.5: Mean and standard deviation plot for flexion data of 3 different joints. Mean is represented as the main dark blue line, while standard deviation is represented by the light blue area. Samples are uniform, and represented by dots. It is possible to observe that angle variation is small.

cycle is plotted. Instead, the mean of all cycles (dark blue line) is displayed along the standard deviation (light blue area), for every sample. Each cycle contains 106 samples, marked by each dot.

As seen on the plot, flexion gait data has a clear "signature", as the standard deviation is low for most of the samples. Since our goal is to base the recognition of an individual solely on flexion data, this section is used to demonstrate its classification potential. We achieve this by using SVMs.

SVMs are well known in statistical learning theory. They are able to perform binary classification and estimation tasks, and have been shown to correctly classify gender based on gait [62]. Let a training dataset of n points

$$(\overrightarrow{x}_1, y_1), ..., (\overrightarrow{x}_n, y_n),$$

where $y = \pm 1$ and $\overrightarrow{x}_i$ is a p-dimensional vector. SVM learns a separating $(p-1)$-dimensional hyper-plane $\overrightarrow{w} \cdot \overrightarrow{x} + b = 0$ that represents the largest separation between two classes. Therefore, the distance between the hyperplane

Samples	Classif. Rate
106	0.946 ± 0.009
53	0.931 ± 0.018
36	0.929 ± 0.0125

Table 4.2: Experimental results for a 10-fold cross validation test.

and the nearest point $\overrightarrow{x}_i$ is maximized.

Since SVM is a binary classifier, a decomposition strategy is necessary for multi-class classification. In our case, the number of classes is equal the number of subjects S. We rely on a decomposition strategy known as one-vs-rest, where S classifiers are trained to classify samples into their corresponding class against all others.

Our results for a 10-fold cross validation test are illustrated in Table 4.2. We ran our method with a different number of features, which is directly related to the number of samples being used. The number of features used is calculated as follows

$$N_{features} = Samples \times |J|,$$

where $Samples$ is the number of samples, uniformly collected for each gait, and J is the set of joints used. For this experiment, we are not excluding abnormal gaits, and not relying on averaged results. It is possible to observe that classification is high, even when number of samples, and consequently features, is reduced by one third.

4.4.2 Classification

While SVM is sufficient to correctly classify gait data, given a substantial amount of data, we propose a simple classification strategy based on rank aggregation. This technique is straightforward, and it can be used in small datasets, or just for simple matching. It is also computationally inexpensive, and requires no training.

Let $\bar{c}_s^j$ represent the average flexion angle curve for a given subject $s \in S$

and a given joint $j \in J$. This is the same curve that was depicted in Figure 4.5. For each joint j, we can compute the distance between any two given subjects, as follows

$$d^j_{s,t} = dist(\bar{c}^j_s, \bar{c}^j_t),$$

such that $s, t \in S$, $s \neq t$ and $dist$ is a generic distance function. Given a distance function, we can easily generate a rank-ordered list τ^j_s, by calculating the pairwise distance between s and all other subjects, as follows

$$\tau^j_s = RankOrderedList(\{d^j_{s,t} | t \in S \text{ and } t \neq s\}).$$

In a rank-ordered list, the rank of an item $t \in S$ is denoted by $\tau(t)$ [63]. The highest ranked item is assigned 1, while the lowest ranked item is assigned $|S - 1|$. This allows us to order similar curves.

Given a candidate subject s and a subject database, it is possible to compute a rank-ordered list for each one of our joints: $\tau^1_s, ...\tau^{|J|}_s$. A rank-aggregation method is able to combine these ranking-results from different sources [64]. Rank aggregation methods have been widely explored in the past. While there are multiple aggregation methods, exploring multiple methods is beyond the scope of this work. In our implementation, we rely on a traditional voting system, known as the Borda Count method [65]. Given a particular ranking τ_j, this method works by assigning a score to each member of the list, according to their relative position. Once it is applied to all rankings, the final aggregated ranking is a sorted list in a decreasing order of the sum of scores of each element.

While rank-aggregation methods do not output a class for each ranked item, they are able to order results, given our selection criteria. Therefore, we can easily compare the gait from a candidate subject against a predefined database of known subjects. This approach can be seem as similar to kNN-based classifiers, as both do not try to model each user. While both approaches are computationally simple, the accuracy of both methods has not been compared,

to the best of our knowledge. Bouchrika et al. [66] and Ahad et al. [67] are examples of papers that rely on kNN to classify human motion. They achieve an overall 73% and 93% recognition rate, however, results are not comparable, as papers rely on private datasets.

4.5 Evaluation Results

In this section, we evaluate our rank-aggregation based classification method, discussed in Section 4.4, on the CSU database [59]. We use flexion information from all 6 joints. Our evaluation approach is a straightforward rank-1 based comparison. For each trial, we rank all other trials in the dataset, according to our proposed method. The closest trial is considered to belong to a matching subject. For clarity purposes, we present our results in a normalized confusion matrix, in which trials are grouped by subject. There are at least three trials for each subject.

As our method relies on a generic distance function, we also use this section to evaluate the most promising measures. Given two curves c and d, equally sampled M times, we use the following metrics

- **Euclidean**: $\sqrt{\sum_{i=1}^{M}(c_i - d_i)^2}$;

- **Manhattan**: $\sum_{i=1}^{M}|c_i - d_i|$;

- **Correlation:** $1 - \frac{(c-\bar{c})\cdot(d-\bar{d}))}{||(c-\bar{c})||_2||(d-\bar{d})||_2}$, where $\bar{c}$ is the mean of elements of vector c, and $c \cdot d$ is the cross-product between vectors c and d.

Results are illustrated in figs. 4.6 to 4.8. It is possible to observe that while all distance measures have high accuracy rates, Manhattan and Eucliean clearly performs better than correlation. This demonstrates that while there is

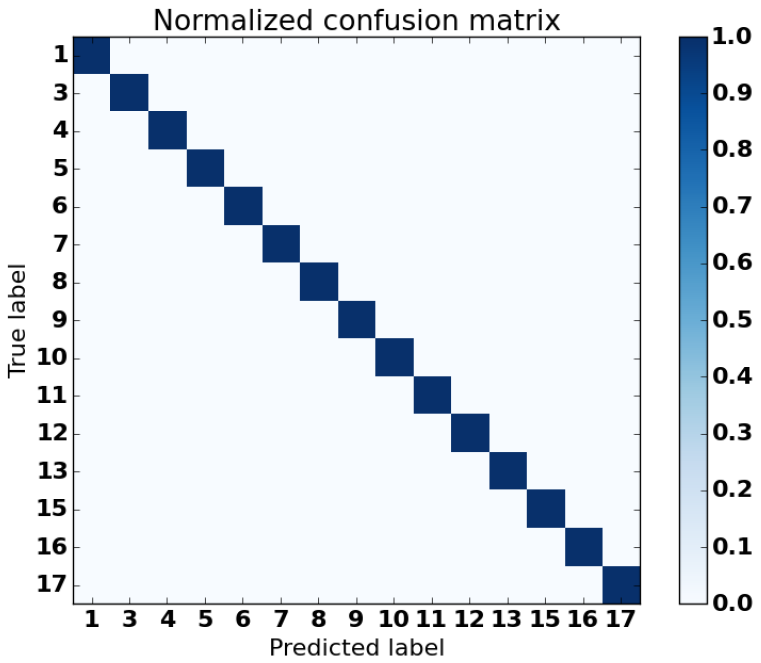

Figure 4.6: Normalized confusion matrix when using Manhattan distance. Overall accuracy is 100%. Labels represent expected and obtained subject IDs for each comparison.

a clear statistical dependence between gait movements for the subject, the best comparison measure is the simple geometrical distance. By ignoring the absolute difference between vectors, a considerable amount of information about the gait is lost.

As the time of writing, the publicly available CSU database[59] can still be considered recent. For this reason, no previous work on gait recognition has been attempted on this database, to the best of our knowledge. This limits the comparison of our results.

Additionally, we demonstrate the importance of not relying on the symmetry between left and right limbs. The experiment illustrated in Figure 4.6 was repeated, but only flexion angles derived from right side limbs have been used during the aggregation process. Results are illustrated in Figure 4.9. Only the Manhattan distance has been used for this experiment, since it is the most accurate distance. The results illustrate the asymmetry for gait movement. If

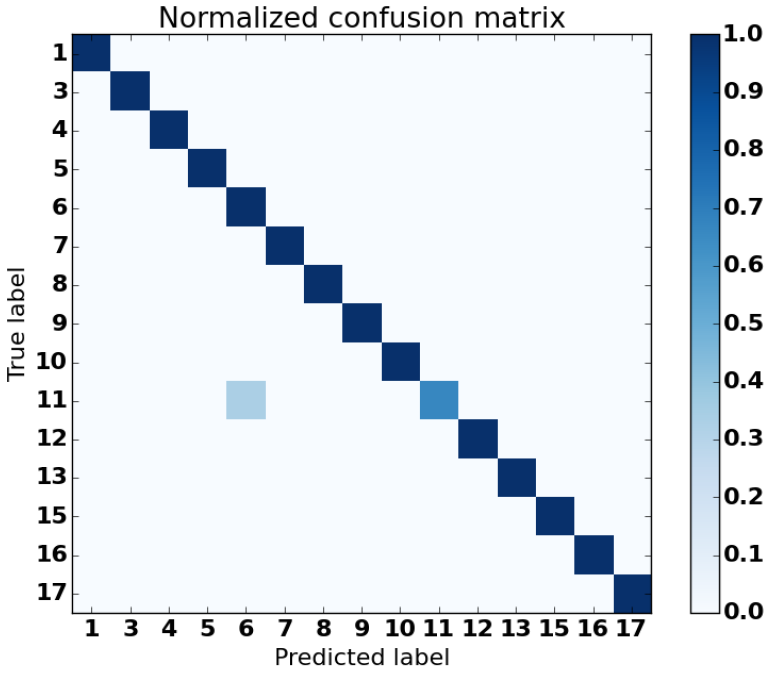

Figure 4.7: Normalized confusion matrix when using Euclidean distance. Overall accuracy is 97.8%. Labels represent expected and obtained subject IDs for each comparison.

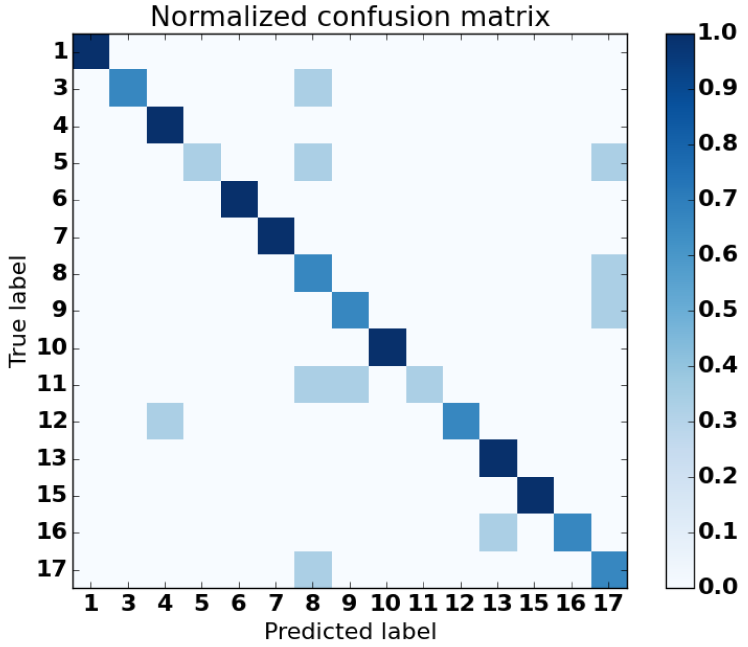

Figure 4.8: Normalized confusion matrix when using correlation. Overall accuracy is 78%. Labels represent expected and obtained subject IDs for each comparison.

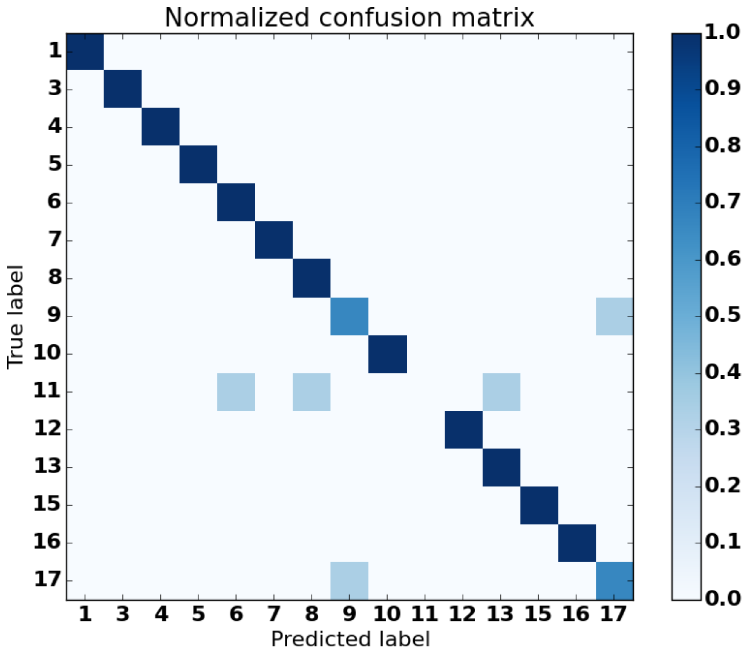

Figure 4.9: Normalized confusion matrix when using Manhattan distance for right side flexion angles. Overall accuracy is 89%. Labels represent expected and obtained subject IDs for each comparison.

gait data was indeed symmetric, accuracy should remain the same, regardless whether right or left side limbs are used.

4.6 Conclusion

We presented a simple gait recognition method based on a rank-aggregation technique. Our method relies exclusively on flexion angle data, and it is computationally inexpensive. We demonstrated satisfactory classification results for the CSU database [59] by using a rank-1 based evaluation. This method can be easily used for pairwise comparison of subjects.

For future work, it would be necessary to extend the experimental parts to other gait databases, to make comparison more realistic. In addition, it is necessary to run experiments with other rank aggregation approaches, as there are approaches that offer an aggregation closer to optimal [64].

Chapter 5

Conclusion

In this thesis, we addressed three different topics related to MoCap data: robust transmission, compression and gait recognition. The research in this area becomes important, as motion sensing technology becomes widely available, and MoCap starts being useful outside animation studios. In Chapter 2, we proposed a MoCap transmission technique aimed to minimize perceptual loss when data is lost. Our method incurs in minimal delay, and requires no retransmission of packets. In Chapter 3, we presented a MoCap compression technique. Even though our method can be applied on any type of MoCap data, it was focused on hand data, given their applications in surgical analysis and the lack of previous research on this area. Finally, in Chapter 4 we targeted biometrics through gait recognition. This is a popular research topic, even though it is commercially unexploited. Our method is computationally efficient, and it can be applied to smaller databases, or for simple subject matching.

5.0.1 Contributions

Below we list contributions made during this Master's program:

- Antonio Carlos Furtado, Irene Cheng, Frederic Dufaux, and Anup Basu. Robust Transmission of Motion Capture Data using Interleaved LDPC

and Inverse Kinematics. In Eurographics 2016 - Short Papers, Eurographics Assoc. (2016)

- Antonio Carlos Furtado, Irene Cheng, Eric Fung, Bin Zheng and Anup Basu. "Low Resolution Tool Tracking for Microsurgical Training in a Simulated Environment." 2016 Annual International Conference of the IEEE Engineering in Medicine and Biology. IEEE, 2016.

- Antonio Carlos Furtado, Xinyao Sun, Anup Basu and Irene Cheng. Optimized Per-Joint Compression of Hand Motion Data. In Systems, Man and Cybernetics (SMC), 2016 IEEE International Conference on. IEEE, 2016

Bibliography

[1] Zhengyou Zhang. Microsoft kinect sensor and its effect. *MultiMedia, IEEE*, 19(2):4–10, 2012.

[2] Vivek Datta, Sean Mackay, Mirren Mandalia, and Ara Darzi. The use of electromagnetic motion tracking analysis to objectively measure open surgical skill in the laboratory-based model. *Journal of the American College of Surgeons*, 193(5):479–485, 2001.

[3] Kalvina Reddy, Andrews Samraj, Maheswari Rajavel, and Nikos Mastorakis. Suitability analysis of gestures for emergency response communication by patients, elderly and disabled while using data gloves. In *1st WSEAS International Conference on Information Technology and Computer Networks (ITCN12)*, 2012.

[4] Weiming Hu, Tieniu Tan, Liang Wang, and Steve Maybank. A survey on visual surveillance of object motion and behaviors. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 34 (3):334–352, 2004.

[5] Henning Schulzrinne, Stephen Casner, Ron Frederick, and Van Jacobson. Rtp: A transport protocol for real-time applications. Technical report, 2003.

[6] Joint Video Team. Advanced video coding for generic audiovisual services. *ITU-T Rec. H*, 264:14496–10, 2003.

[7] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[8] Ronald Poppe. Vision-based human motion analysis: An overview. *Computer vision and image understanding*, 108(1):4–18, 2007.

[9] Gregory G Slabaugh. Computing euler angles from a rotation matrix. *Retrieved on August*, 6(2000):39–63, 1999.

[10] Wee Lum Tan, Fung Lam, and Wing Cheong Lau. An empirical study on the capacity and performance of 3g networks. *IEEE Transactions on Mobile Computing*, 7(6):737–750, 2008.

[11] Irene Cheng, Lihang Ying, Kostas Daniilidis, and Anup Basu. Robust and scalable transmission of arbitrary 3d models over wireless networks. *Journal on Image and Video Processing*, 2008:1, 2008.

[12] Irene Cheng and Anup Basu. Perceptually optimized 3-d transmission over wireless networks. *Multimedia, IEEE Transactions on*, 9(2):386–396, 2007.

[13] Ghassan Alregib, Yucel Altunbasak, and Jarek Rossignac. Error-resilient transmission of 3d models. *ACM Transactions on Graphics (TOG)*, 24(2): 182–208, 2005.

[14] Simone Milani and Giancarlo Calvagno. A cognitive approach for effective coding and transmission of 3d video. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 7(1):23, 2011.

[15] ISO. Information technology—coding of audio-visual objects— part 2: Visual. ISO ISO 14496-2:2001, International Organization for Standardization, Geneva, Switzerland, 2001.

[16] Marius Preda, Blagica Jovanova, Ivica Arsov, and Françoise Prêteux. Optimized mpeg-4 animation encoder for motion capture data. In *Proceedings*

of the twelfth international conference on 3D web technology, pages 181–190. ACM, 2007.

[17] Siddhartha Chattopadhyay, Suchendra M Bhandarkar, and Kang Li. Model-based power aware compression algorithms for mpeg-4 virtual human animation in mobile environments. *Multimedia, IEEE Transactions on*, 9(1):1–8, 2007.

[18] Aura Ganz, Zvi Ganz, and Kitti Wongthavarawat. *Multimedia Wireless Networks: Technologies, Standards and QoS*. Pearson Education, 2003.

[19] Yi J Liang, John G Apostolopoulos, and Bernd Girod. Analysis of packet loss for compressed video: Does burst-length matter? In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 5, pages V–684. IEEE, 2003.

[20] Irene Cheng, Lihang Ying, and Anup Basu. Perceptually coded transmission of arbitrary 3d objects over burst packet loss channels enhanced with a generic jnd formulation. *Selected Areas in Communications, IEEE Journal on*, 30(7):1184–1192, 2012.

[21] Yun Q Shi, Xi Min Zhang, Zhi-Cheng Ni, and Nirwan Ansari. Interleaving for combating bursts of errors. *Circuits and Systems Magazine, IEEE*, 4 (1):29–42, 2004.

[22] Angelos D Liveris, Zixiang Xiong, and Costas N Georghiades. Compression of binary sources with side information at the decoder using ldpc codes. *IEEE communications letters*, 6(10):440–442, 2002.

[23] Anne Aaron and Bernd Girod. Compression with side information using turbo codes. In *Data Compression Conference, 2002. Proceedings. DCC 2002*, pages 252–261. IEEE, 2002.

[24] David Slepian and Jack K Wolf. Noiseless coding of correlated information sources. *Information theory, IEEE Transactions on*, 19(4):471–480, 1973.

[25] Robert G Gallager. Low-density parity-check codes. *Information Theory, IRE Transactions on*, 8(1):21–28, 1962.

[26] Hllatthew C Davey and David JC MacKay. Low density parity check codes over gf (q). In *Information Theory Workshop, 1998*, pages 70–71. IEEE, 1998.

[27] Michael G Luby, Michael Mitzenmacher, M Amin Shokrollahi, Daniel Spielman, et al. Improved low-density parity-check codes using irregular graphs. *Information Theory, IEEE Transactions on*, 47(2):585–598, 2001.

[28] David JC MacKay. Good error-correcting codes based on very sparse matrices. *Information Theory, IEEE Transactions on*, 45(2):399–431, 1999.

[29] Jon Postel et al. Rfc 791: Internet protocol. 1981.

[30] Jessica Hodgins. CMU Graphics Lab Motion Capture Database. http://mocap.cs.cmu.edu, 2015. URL `http://mocap.cs.cmu.edu`.

[31] Okan Arikan. Compression of motion capture databases. *ACM Transactions on Graphics (TOG)*, 25(3):890–897, 2006.

[32] Maxime Tournier, Xiaomao Wu, Nicolas Courty, Elise Arnaud, and Lionel Reveret. Motion compression using principal geodesics analysis. In *Computer Graphics Forum*, volume 28, pages 355–364. Wiley Online Library, 2009.

[33] I Chen Lin, Jen Yu Peng, Chao Chih Lin, and Ming Han Tsai. Adaptive motion data representation with repeated motion analysis. *Visualization and Computer Graphics, IEEE Transactions on*, 17(4):527–538, 2011.

[34] L Váša and G Brunnett. Rate-distortion optimized compression of motion capture data. In *Computer Graphics Forum*, volume 33, pages 283–292. Wiley Online Library, 2014.

[35] Irene Cheng and Walter Bischof. A perceptual approach to texture scaling based on human computer interaction. *Short Paper of Eurographics*, 2006: 1–6, 2006.

[36] Amirhossein Firouzmanesh, Irene Cheng, and Anup Basu. Perceptually guided fast compression of 3-d motion capture data. *Multimedia, IEEE Transactions on*, 13(4):829–834, 2011.

[37] K Peason. On lines and planes of closest fit to systems of point in space. *Philosophical Magazine*, 2:559–572, 1901.

[38] David G Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial intelligence*, 31(3):355–395, 1987.

[39] Onur Onder, Ugur Güdükbay, Bülent Özguc, Tanju Erdem, Çiğdem Erdem, and Mehmet Özkan. Keyframe reduction techniques for motion capture data. In *3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video, 2008*, pages 293–296. IEEE, 2008.

[40] Paul Dierckx. An algorithm for smoothing, differentiation and integration of experimental data using spline functions. *Journal of Computational and Applied Mathematics*, 1(3):165–184, 1975.

[41] Mark S Nixon, Tieniu Tan, and Ramalingam Chellappa. *Human identification based on gait*, volume 4. Springer Science & Business Media, 2010.

[42] Matthew A Turk and Alex P Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings*

CVPR'91., IEEE Computer Society Conference on, pages 586–591. IEEE, 1991.

[43] Anil K Jain and Nicolae Duta. Deformable matching of hand shapes for user verification. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, volume 2, pages 857–861. IEEE, 1999.

[44] Anil K Jain, Lin Hong, Sharath Pankanti, and Ruud Bolle. An identity-authentication system using fingerprints. *Proceedings of the IEEE*, 85(9): 1365–1388, 1997.

[45] John G Daugman. High confidence visual recognition of persons by a test of statistical independence. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(11):1148–1161, 1993.

[46] Lawrence R. Rabiner and B. H. Juang. *Fundamentals of speech recognition*. PTR Prentice Hall, united states ed edition, April 1993. ISBN 0130151572.

[47] Mark S Nixon and John N Carter. Automatic recognition by gait. *Proceedings of the IEEE*, 94(11):2013–2024, 2006.

[48] Amit Kale, Aravind Sundaresan, AN Rajagopalan, Naresh P Cuntoor, Amit K Roy-Chowdhury, Volker Krüger, and Rama Chellappa. Identification of humans using gait. *Image Processing, IEEE Transactions on*, 13 (9):1163–1173, 2004.

[49] Chiraz BenAbdelkader, Ross Cutler, Harsh Nanda, and Larry Davis. Eigengait: Motion-based recognition of people using image self-similarity. In *Audio-and Video-Based Biometric Person Authentication*, pages 284–294. Springer, 2001.

[50] Sudeep Sarkar, P Jonathon Phillips, Zongyi Liu, Isidro Robledo Vega, Patrick Grother, and Kevin W Bowyer. The humanid gait challenge prob-

lem: Data sets, performance, and analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(2):162–177, 2005.

[51] Aaron E Bobick and Amos Y Johnson. Gait recognition using static, activity-specific parameters. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–423. IEEE, 2001.

[52] Liang Wang, Huazhong Ning, Tieniu Tan, and Weiming Hu. Fusion of static and dynamic body biometrics for gait recognition. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(2):149–158, 2004.

[53] Tomasz Krzeszowski, Adam Switonski, Bogdan Kwolek, Henryk Josinski, and Konrad Wojciechowski. Dtw-based gait recognition from recovered 3-d joint angles and inter-ankle distance. In *Computer Vision and Graphics*, pages 356–363. Springer, 2014.

[54] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[55] Ralph Gross and Jianbo Shi. The cmu motion of body (mobo) database. 2001.

[56] Jamie D Shutler, Michael G Grant, Mark S Nixon, and John N Carter. On a large sequence-based human gait database. In *Applications and Science in Soft Computing*, pages 339–346. Springer, 2004.

[57] Shuai Zheng, Junge Zhang, Kaiqi Huang, Ran He, and Tieniu Tan. Robust view transformation model for gait recognition. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 2073–2076. IEEE, 2011.

[58] Martin Hofmann, Jürgen Geiger, Sebastian Bachmann, Björn Schuller, and Gerhard Rigoll. The tum gait from audio, image and depth (gaid) database: Multimodal recognition of subjects and traits. *Journal of Visual Communication and Image Representation*, 25(1):195–206, 2014.

[59] Jason K Moore, Sandra K Hnat, and Antonie J van den Bogert. An elaborate data set on human gait and the effect of mechanical perturbations. *PeerJ*, 3:e918, 2015.

[60] Shanon X Ju, Michael J Black, and Yaser Yacoob. Cardboard people: A parameterized model of articulated image motion. In *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on*, pages 38–44. IEEE, 1996.

[61] Heydar Sadeghi, Paul Allard, and Morris Duhaime. Functional gait asymmetry in able-bodied subjects. *Human Movement Science*, 16(2):243–258, 1997.

[62] Jang-Hee Yoo, Doosung Hwang, and Mark S Nixon. Gender classification in human gait using support vector machine. In *Advanced concepts for intelligent vision systems*, pages 138–145. Springer, 2005.

[63] M Elena Renda and Umberto Straccia. Web metasearch: rank vs. score based rank aggregation methods. In *Proceedings of the 2003 ACM symposium on Applied computing*, pages 841–846. ACM, 2003.

[64] Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622. ACM, 2001.

[65] Jean C de Borda. Mémoire sur les élections au scrutin. 1781.

[66] Imed Bouchrika, Michela Goffredo, John N Carter, and Mark S Nixon.

Covariate analysis for view-point independent gait recognition. In *International Conference on Biometrics*, pages 990–999. Springer, 2009.

[67] Md Ahad, Atiqur Rahman, JK Tan, HS Kim, and S Ishikawa. Temporal motion recognition and segmentation approach. *International Journal of Imaging Systems and Technology*, 19(2):91–99, 2009.