UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Rodolfo Luis Tonoli

Motion Retargeting Preserving Spatial Relationship

Transferência de Movimentos Preservando Relação Espacial

Campinas
2019

Rodolfo Luis Tonoli

Motion Retargeting Preserving Spatial Relationship

Transferência de Movimentos Preservando Relação Espacial

Dissertation presented to the School of Electrical and Computer Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Electrical Engineering, in the area of Computer Engineering.

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica, na área de Engenharia de Computação.

Supervisor: Profa. Dra. Paula Dornhofer Paro Costa

Co-supervisor: Prof. Dr. José Mario De Martino

Este exemplar corresponde à versão final da dissertação defendida pelo aluno Rodolfo Luis Tonoli, orientado pela Profa. Dra. Paula Dornhofer Paro Costa e coorientado pelo Prof. Dr. José Mario De Martino.

Campinas
2019

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

T616m

Tonoli, Rodolfo Luis, 1991-
Motion retargeting preserving spatial relationship / Rodolfo Luis Tonoli. – Campinas, SP : [s.n.], 2019.

Orientador: Paula Dornhofer Paro Costa.
Coorientador: José Mario De Martino.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Animação por computador. 2. Avatares. 3. Computação gráfica. 4. Movimento corporal. 5. Análise de movimento. I. Costa, Paula Dornhofer Paro, 1978-. II. De Martino, José Mario, 1958-. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Transferência de movimentos preservando relação espacial
Palavras-chave em inglês:
Computer animation
Avatars
Computer graphics
Body motion
Motion analysis
Área de concentração: Engenharia de Computação
Titulação: Mestre em Engenharia Elétrica
Banca examinadora:
Paula Dornhofer Paro Costa [Orientador]
Hélio Pedrini
Léo Pini Magalhães
Data de defesa: 20-08-2019
Programa de Pós-Graduação: Engenharia Elétrica

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0002-5800-654X
- Currículo Lattes do autor: http://lattes.cnpq.br/3199907158975409

Comissão Julgadora – Dissertação de Mestrado

Candidato: Rodolfo Luis Tonoli **RA:** 105652

Data da defesa: 20 de agosto de 2019

Título da Tese: "Motion Retargeting Preserving Spatial Relationship."

Profa. Dra. Paula Dornhofer Paro Costa (Presidente, FEEC/UNICAMP)
Prof. Dr. Hélio Pedrini (IC/UNICAMP)
Prof. Dr. Léo Pini Magalhães (FEEC/UNICAMP)

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no SIGA (Sistema de Fluxo de Dissertação/Tese) e na Secretaria de Pós-Graduação da Faculdade de Engenharia Elétrica e de Computação.

Acknowledgements

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

I acknowledge the motion capture infrastructure from TALES project supported by CNPq/MCTI-SECIS, Grant Number 458691/2013-5, and by Capes/SDH/MCTI, Grant Number 88887.091672/2014-0.

I acknowledge Professor Paula Dornhofer Paro Costa and Professor José Mario De Martino for their guidance, their patience, and for taking the time to share their thoughts and knowledge with me. I want to thank the colleagues of the department for the discussions and conversations regarding this work, science, and other shared interests. Finally, I am grateful to my parents, Débora Carina Tonoli and Márcio Luis Tonoli, and Isadora Bugni Ribeiro for all the support and patience.

Abstract

One application of motion capture technology is the recording of the movement of a human being to animate virtual characters. This technology allows the production of realistic animations, able to reproduce the dynamics of a real person's movement. However, the process of transferring the movement to a virtual character, called motion retargeting, can cause distortions when the performer and the character have different body proportions. These distortions may generate artifacts in the animation, such as self-penetration and ill-conditioned poses, that cause strangeness and hinder movement perception. This work proposes the motion retargeting of motion capture data to 3D characters preserving the hands' spatial relationship with the body surface. This process ensures the hands' interaction with the body surface, such as covering the eyes or mouth. The body surfaces of the actor and the virtual character are calibrated in a semi-automatic process. A mesh composed with triangles is constructed to represent each body surface and the limbs, upper and lower arms and legs, are modeled as capsules. For the captured motion, the distance of the hands and feet to the performer's body surface components, i.e., triangles and capsules, are normalized by the body proportions. Then, the virtual character's body proportions and body surface are used to compute its hands and feet positions. The character pose is adjusted by inverse kinematics for each frame of the animation. To evaluate the implemented method, a perceptual evaluation of recorded movements through motion capture was conducted. Participants were presented to three stimuli: video recordings of the performer during the motion capture session; and two 3D characters performing the motions, one had motions retargeted through the implemented method, and the other had not. Participants were asked which animated character performed the movements most similar to the ones in the videos. The results indicate that the implemented motion retargeting process improved the videorealism of animations recorded through motion capture by preserving the interaction of the hands with the body surface. Among the contributions of this work is the automatic retargeting of captured movements from human beings to digital characters with different body proportions and shapes.

Keywords: Motion Retargeting; Digital Animation; 3D Virtual Characters; Motion Capture.

Resumo

Dentre as aplicações da tecnologia de captura de movimentos, está a gravação de movimentos de um ser humano para animação de personagens virtuais. Esta tecnologia permite gerar animações realistas, capazes de reproduzir a dinâmica dos movimentos de uma pessoa real. Porém, o processo de transferência do movimento para um personagem virtual, chamado retargeting de movimento, pode resultar em distorções quando o ator e o personagem têm proporções corporais distintas. Essas distorções podem gerar artefatos na animação, como auto-penetração e poses incorretas, causando estranhamento e percepção distorcida do movimento. Este trabalho propõe a transferência dos dados da captura de movimento para um personagem 3D, preservando a relação espacial das mãos com a superfície do corpo. Este processo garante a interação das mãos com a superfície do corpo, como cobrir os olhos ou a boca. As superfícies do corpo do ator e do personagem virtual são calibradas em um processo semiautomático. Uma malha composta por triângulos é construída para representar cada superfície corporal e os membros, braços, antebraços, coxas, e pernas, são modelados por cápsulas. No movimento capturado, as distâncias entre as mãos e pés até os componentes da superfície, triângulos e cápsulas, são normalizadas pelas proporções corporais do ator. Então, as proporções corporais e superfície corporal do personagem virtual são utilizadas para calcular a posição de seus pés e suas mãos. A pose do personagem é ajustada por cinemática inversa para cada frame da animação. Para avaliar o método implementado, foi aplicada uma avaliação perceptual de movimentos registrados através de captura de movimentos. Foi apresentado três estímulos para participantes voluntários: vídeos gravados do ator durante a captura de movimentos; e dois personagens 3D executando os movimentos, um com movimentos resultantes do processo implementado e o outro não. Os participantes deveriam responder qual personagem animado executou os movimentos mais parecido com aqueles do vídeo. Os resultados indicam que o processo de retargeting implementado contribuiu para um maior videorrealismo das animações gravadas por captura de movimentos ao preservar a interação das mãos com a superfície do corpo. Dentre as contribuições deste trabalho, está a transferência automática de movimentos capturados de seres humanos para personagens digitais com diferentes proporções e formatos corporais.

Palavras-chave: Transferência de movimentos; Animação Digital; Personagens Virtuais 3D; Captura de Movimentos.

List of Figures

1.1 Snapshot of two characters performing the captured motion of covering the ears. The 3D model on the right has arms fifteen percent longer than the model on the left. 20

2.1 (a) Eadweard Muybridge's galloping horse photographs. (b) The Marey's precursor of a mocap suit and the record of a movement dynamics. (c) *Rotoscope* by Fleischer. Source: (a), (b) Wikimedia Commons. (c) Extracted from (PARENT, 2008, p. 10) 24
2.2 (a) Wired polygons of the virtual character surface in black. (b) Skeleton structure of the virtual character in yellow, the circles and the lines represent the joints and bones, respectively. 25
2.3 Straight (top) and rotated (bottom) left hand of a virtual character. (a) Wired polygons of the hand. (b) Weights of the left hand bone, blue and green colors indicate regions with lower and higher weights. 26
2.4 Heat map of capture volume covered by mocap infrared cameras. Regions visible to at least two, three, and four or more cameras are colored in green, yellow, and red, respectively. (a), (b) and (c) volumes contain 4, 8 and 16 cameras. Source: Extracted from <https://www.vicon.com/visualization>. Visualization simulated using Vicon Vantage™ V16 cameras with 18mm lens. 27
2.5 Motion capture performer in the T-Pose with the Vicon Shōgun™ standard 53 marker setup. 28
2.6 Forward and inverse kinematics. Source: Extracted and adapted from (KUCUK; BINGUL, 2006). 29
2.7 (a) The motion of an adult to grasp a glass of water on a table. (b) The motion retargeting to a smaller character preserving the pose of the arm. (c) The motion retargeting to the same smaller character but constraining the position of the hand. Source: Extracted and adapted from (SHIN *et al.*, 2001). 30
2.8 Correspondence between two skeletons with distinct topology. Source: Extracted from (MONZANI *et al.*, 2000). 31
2.9 Representation of the skeleton by Kulpa *et al.* (2005), the dotted lines represent limbs with variable length. Group of joints, highlighted in red, are adapted separately. Source: Extracted from (KULPA *et al.*, 2005). . . 32
2.10 Results presented by Al-Asqhar *et al.* (2013) of a motion retargeting to different sizes characters. Source: Extracted and adapted from (AL-ASQHAR *et al.*, 2013) 33

2.11 Illustration of the methodology proposed by Molla *et al.* (2017). From left to right: mocap performer; left-hand spatial relationship with the surface for the performer; spatial relationship transferred to a child character; final result. The yellow and red lines illustrate the calibrated surfaces and spatial relationships, respectively. Source: Extracted and adapted from (MOLLA *et al.*, 2017) 35

3.1 Workflow overview of the motion retargeting process that preserves the spatial relationship of joints with the body surface. It receives the mocap data (source animation), the virtual character skeleton (target skeleton), a joints correspondence between both skeletons (skeleton map), and the calibrated surface from both the mocap performer (source surface) and the virtual character (target surface). (*) refers to the retargeted animation sequence disregarding surface relationship. 38
3.2 The *Initial Motion Retargeting* process transfers the mocap data, the source animation, to the target skeleton. 39
3.3 On the left part of the image, two skeletons of virtual humans with different topologies. The second skeleton is more complex: the root and the hips joints are not the same, the limbs have intermediate joints, and it has clavicle, feet, hands, and facial joints that are not present in the first skeleton. On the right part of the image, a visual representation of the *Skeleton Map*: a minimal set of named joints (root, spine, neck, etc.) that are required to animate the body of a virtual human and the source and target joints associated to them. 42
3.4 The left arm of the source skeleton and the target skeleton on the left and right, respectively. Both palms of the hand are facing the ground with the arms stretched. The hand joints are highlighted in green, and the axes represent their local orientation. Although the skeletons are in the same pose, the local orientation of matching joints is not guaranteed to be equal due to the distinct topology of each skeleton. 43
3.5 The topology of two skeletons. The black dots and lines represent joints and bones, respectively. The colored lines represent the current orientation of the mapped joints, where red, green, and blue represent the local x, y, and z axes, respectively. 44
3.6 Two target characters: Talita, on the left, from project TAS; and Aragor, on the right, available in Autodesk® MotionBuilder®. 46
3.7 Right-hand joint distance from the origin during an action sequence, in centimeters, of the performer, Talita, and Aragor skeletons represented by the black, red, and blue lines, respectively. The motion was retargeted to the characters Talia and Aragor through the *Initial Motion Retargeting* process. 47
3.8 Pose animations with motion retargeting artifacts: (a) self-penetration of the arms; (b) lack of contact of the right hand with the chest; (c) and (d), front and side view of the same pose, wrong hand placement. 48

3.9 The relationship between the joints and the body surface of the mocap performer is encoded in the egocentric coordinates $\mathbf{E}_j$. The source animation provides the position of the joints as well as the body proportions, the bone lengths. The performer body surface is calibrated during mocap. An egocentric coordinate details the relationship between a single joint and the surface components in one frame. 49
3.10 Skeletons, meshes components, and limbs capsules composing the body surface of the (a) mocap performer and the target characters (b) Talita and (c) Aragor. The red dots indicate surface points calibrated. The gray capsules represent each modeled limb. 50
3.11 A 3D character and the sampled points and limb radius that the yellow markers and lines indicate. The points are the edges of the triangles composing two meshes, head and body mesh, that represents the surface of the virtual human. Capsules represent the limbs. Capsule radius and length are equal to half the thickness and the bone segment length. 51
3.12 The process to obtain the target and surface CSV files, that specifies a static representation of the body surface for the performer and 3D character. The surface of the target character is acquired manually while the body surface of the mocap performer is extracted from three mocap takes. 51
3.13 Head surface calibration poses (second BVH take). Surface points extracted from each pose, from left to right: forehead, ear, and chin on the first line; cheek, mouth, and forehead middle on the second line; and nape and nape middle at the bottom. 52
3.14 Capsule with length $2L$ and radius R. 53
3.15 The front and side view of a skeleton on the left and right, respectively. The circumferences with crosses inside represent some of the surface points, and the lines show their attached joint. 54
3.16 (a) Pose with the hands pressed together to compute the position of the hand surface. The distance between the hand joints, the middle hand joints, in this case, is given by $2\mathbf{d}_s$. (b) The recalculation of the surface point position $\mathbf{p}_s$ given the position $\mathbf{p}$ of the attached joint. 55
3.17 Head and body surface meshes at different angles. The edges of the triangles, the red dots, represent the surface points from the surface calibration. Black dots and lines represent the joints and bones, respectively. 56
3.18 The decomposition of the position a joint n, the right hand joint in the figure, relative to the ith surface component. The position of the joint $\mathbf{p}_n$ is the sum of the reference point $\mathbf{x}_i$, the red dot, and the displacement vector $\mathbf{v}_i$, the blue arrow. The pink arrow represents the surface normal of the mesh component, and θ is the angle between the normal and the displacement vector. 57
3.19 The skeleton pose during an animation that the right hand passes in front of the face (a) and the waist (b), and the sum of the importance factor during the animation (c). In (a) and (b), the displacement vectors are represented by blue lines, components with higher importance have darker intensities of blue. Also, the reference points are shown as red dots. In (c), the sum of the importance factor for the head and body components are presented in green and blue, respectively. The vertical red lines correspond to the moment of the poses in (a) and (b). 59

3.20 The kinematic path, in orange, from the hand joint to two surface components. On the left, a triangle of the body mesh, where the reference point $\mathbf{x}_i$ is the centroid. On the right, a limb capsule of the left forearm, with $\mathbf{x}_i$ as the intersection of the line from the right hand to the center of the bone with the capsule surface. The vector $-\mathbf{x}_{j0}$ starts on the root joint of the kinematic path and ends on the reference point. 60
3.21 Aragor's skeleton pose during an animation that the right hand passes in front of the face (a) and the waist (b). The red cross and green dot represent the target and current position of the right hand, respectively. 67
3.22 Right-hand joint distance from the origin during an action sequence of the performer (black) and the virtual character Aragor after the *Initial Motion Retargeting* (red) and after the motion retargeting process proposed (blue). 67
3.23 Pose animations without (left) and with (right) the motion retargeting preserving spatial relationship: (a) and (b) the self-penetration of the arms was diminished; (c) and (d) the right hand was attracted to the chest; (e) and (f) the mouth was covered by the hand; (g) and (h) the hand contact with the mouth was preserved. 68
3.24 On the left, the optical mocap system setup consisting of nine Vicon Vantage™ cameras (V1 to V9 and on the top right), and a conventional HD video camera (VC). Passive markers (on the bottom right) were placed on the performer's suit to track the action sequences. Sources: Human silhouette source extracted from Wikimedia Commons; Vicon Vantage™ camera extracted and adapted from the Vicon website available at: <https://www.vicon.com/hardware/cameras/vantage/>; Passive marker extracted and adapted from: <https://www.qualisys.com/hardware/accessories/passive-markers/>. 69

4.1 The optical mocap system setup consisting of nine Vicon Vantage™ cameras (V1 to V9) and a conventional HD video camera (VC). (Human silhouette source: Wikimedia Commons.) 72
4.2 Configuration of fifty-three reflective markers of an optical mocap system for capturing body motion. 73
4.3 Volunteer subject performing the evaluation on a desktop computer with a twenty-two inch monitor. The volunteer interacted with the test application using the keyboard and the mouse, printed instructions detailing the controls were available. 74
4.4 A snapshot of the perceptual evaluation application. A set of movements performed during mocap is presented to the subjects as video and as animations, simultaneously. One character motion was generated by the implemented approach, and the other is the result of the initial retargeting described in Section 3.1. 75
4.5 Question dialog presented to participants for each animation in Brazilian Portuguese. Participants were asked to answer the question "Which animation do you think best represents the movements in the video?" selecting one of the answers: (1) the left one; (2) the right one; (3) they are identical and represent the movement well; or (4) they are identical and do not represent the movement well. 76

4.6 Percentage of votes that (from left to right): preferred the retargeted motion by the implemented methodology (SMR); preferred the retargeted motion by the *Initial Motion Retargeting* only (IMR); found the animations identical and that they do (IW) and do not (INW) represent the movement well. . . 77
4.7 Movement 1 stimulus, on the left, and distribution of votes, on the right. . 79
4.8 Movement 2 stimulus, on the left, and distribution of votes, on the right. . 79
4.9 Movement 3 stimulus, on the left, and distribution of votes, on the right. . 79
4.10 Movement 4 stimulus, on the left, and distribution of votes, on the right. . 80
4.11 Movement 5 stimulus, on the left, and distribution of votes, on the right. . 80
4.12 Movement 6 stimulus, on the left, and distribution of votes, on the right. . 80
4.13 Movement 7 stimulus, on the left, and distribution of votes, on the right. . 81
4.14 Movement 8 stimulus, on the left, and distribution of votes, on the right. . 81
4.15 Movement 9 stimulus, on the left, and distribution of votes, on the right. . 81

A.1 BVH file example. 91
A.2 Skeleton pose in the base pose and during the motion. 92

B.1 Performer's pose during the body surface calibration of the trunk and limbs. 94
B.2 Graphic of the speed of the right hand throughout the first take of the performer's body surface calibration. The couples of red vertical lines with indices on top represent the frame ranges that the speed of the hand approaches zero. 94
B.3 Performer's pose during the body surface calibration of the head. 95
B.4 Graphic of the speed of the right hand throughout the first take of the performer's body surface calibration. The couples of red vertical lines with indices on top represent the frame ranges that the speed of the hand approaches zero. 95
B.5 Performer's pose during the body surface calibration of the back. 95
B.6 Graphic of the speed of the right hand throughout the third take of the performer's body surface calibration. The couples of red vertical lines with indices on top represent the frame ranges that the speed of the hand approaches zero. 96

List of Tables

2.1 Summary of motion retargeting techniques presented in Sections 2.4.1 and 2.4.2. (1) does not use surface information. (2) handles only ground interaction. 35

3.1 The three takes to calibrate the body surface of the mocap performer, the surface points sampled at each take and the type of information extracted from the point: *Mesh* indicates that the point composes a surface mesh, body or head, and *Limb* points define the thickness of the limbs. (*) refers to surface points that the performer calibrates with both hands simultaneously to sample the right and left side of the surface. 53
3.2 Bone segments and joints to compose a kinematic path. Except for the spine and neck, the bones refer to both the left and right sides. The start and end joints are present in *Skeleton Map*, described in Section 3.1.2. . . . 60
3.3 The egocentric coordinates that describe the spatial relationship between the joints of the hands and feet and the surface components. The coordinates are extracted from the pose of the source animation and used to adapt the pose of the target character, given its body proportions and body surface. The displacement vector and importance are stored in the normalized form. 63

4.1 Action sequences captured for the perceptual evaluation. 73
4.2 Votes percentage per movement that: preferred the retargeted motion by the implemented methodology (SMR); preferred the retargeted motion by theInitialMotion Retargetingonly (IMR); found the animations identical and that they do (IW) and do not (INW) represent the movement well. . . 78

B.1 The three takes to calibrate the body surface of the mocap performer, the surface points sampled at each take and the index of the respective frame range in the speed analysis. 96

List of Acronyms

2D Two dimensional

3D Three dimensional

BVH Biovision Hierarchy

CSV Comma-separated values

DOF Degree of Freedom

IK Inverse kinematics

IMR Initial motion retargeting

INW Identical and represent well

IW Identical and do not represent well

LED Light-Emitting Diode

mocap Motion capture

SMR Surface-aware motion retargeting

VFX Visual effects

List of Symbols

$\mathbf{p}$	lowercase letters represent vectors
X	uppercase letters represent matrices
T	translation matrix
R	rotation matrix
M	transformation matrix, the combination of the translation and rotation matrices
M^i	transformation matrix of joint i
M^i_G	global transformation matrix of joint i
M^i_{NG}	new global transformation matrix of joint i
M^i_L	local transformation matrix of joint i
M^i_{NL}	new local transformation matrix of joint i
$M^i(t)$	transformation matrix of joint i in frame t.

Summary

1 Introduction **18**
- 1.1 Computer Animation . . . 18
- 1.2 Problem Definition . . . 20
- 1.3 Objectives and Contributions . . . 21
- 1.4 Organization . . . 22

2 Basic Concepts and Related Works **23**
- 2.1 Historical Perspective . . . 23
- 2.2 Mocap Technology Overview . . . 26
- 2.3 Mocap Pipeline . . . 27
- 2.4 Related Works . . . 30
  - 2.4.1 Motion Retargeting to Different Characters . . . 30
  - 2.4.2 Surface-Aware Motion Retargeting . . . 32
  - 2.4.3 Literature Summary . . . 34
- 2.5 Concluding Remarks . . . 36

3 Preserving Surface Spatial Relationship in Motion Retargeting **37**
- 3.1 Initial Motion Retargeting . . . 38
  - 3.1.1 Skeletal Animation . . . 38
  - 3.1.2 Skeleton Map . . . 41
  - 3.1.3 Bones Alignment . . . 43
  - 3.1.4 Root Translation . . . 45
  - 3.1.5 Retargeted Motions . . . 46
  - 3.1.6 Summary . . . 47
- 3.2 Spatial Relationship Encoding . . . 48
  - 3.2.1 Surface Calibration . . . 49
  - 3.2.2 Computing Egocentric Coordinates . . . 55
  - 3.2.3 Summary . . . 62
- 3.3 Pose Adaptation . . . 63
  - 3.3.1 Computing Target Positions . . . 63
  - 3.3.2 Pose Adjustment with Inverse Kinematics . . . 64
  - 3.3.3 Adjusting Extremity Joints Orientation . . . 65
  - 3.3.4 Example . . . 66
  - 3.3.5 Summary . . . 66
- 3.4 Materials . . . 69
- 3.5 Concluding Remarks . . . 70

4 Perceptual Evaluation **71**
4.1 Test Stimuli 72
4.2 Test Protocol 73
4.3 Results 76
4.3.1 Analysis of the Motion Retargeting per Movement 77
4.4 Concluding Remarks 82

5 Conclusions **83**
5.1 Limitations and Future Work 85

A BVH Example **90**

B Performer Body Surface Calibration **93**

Chapter 1

Introduction

This chapter presents an introduction of the basics concepts related to this work, such as computer animation and motion retargeting (Section 1.1); the problem definition (Section 1.2); the main objectives and contributions (Section 1.3); and the dissertation organization (Section 1.4).

1.1 Computer Animation

Toy Story was released in 1995 as the first full-length film produced entirely through computer animation techniques (HENNE *et al.*, 1996). Besides Pixar's production, several applications employ digital animations to entertain, convey information, educate, among others. Some digital media exploit virtual human models, or avatars, to make human-computer interaction more natural and accessible. *Talita*, for example, is a signing avatar that communicates in the Brazilian Sign Language and helps the deaf and hard of hearing individuals access information (DE MARTINO *et al.*, 2017). *Lu*, the animated virtual saleswoman of the retail company Magazine Luiza, represents the brand in social media and TV commercials (MAGAZINE LUIZA, 2017). Other application examples include automated terminals, interactive virtual worlds, and video-games.

Humans use not only the voice and facial expression cues to understand intentions, motives, and wills, but also the movements of our body, limbs, and hands as they may carry semantic information on the emotion that one is trying to express. The movements of the limbs and hands are particularly important for sign languages as they play a significant role in conveying meaning. The location, orientation, and movement of the hands are basic parameters that characterize a sign in sign languages. Slightly discrepancies will express different messages or even make the sign unrecognizable. Therefore, an accurate motion representation by the virtual agent often is not only desired but mandatory. However, virtual human animation is a challenging effort due to its intrinsic complexity. Besides, humans are keen observers of human motion and can easily notice unnatural artifacts given the familiarity with human movement.

Keyframing is a technique to animate virtual characters. The animator professional adjusts key poses of the 3D avatar in dispersed frames and an automatic algorithm fills the gaps between poses by interpolating them over time, which creates the impression of the desired motion (KERLOW, 2004, p. 272). Keyframe-based animation demands extensive manual and artistic work. Knowledge of human motion dynamics — and even human anatomy — is required to craft realistic human animations (RATNER, 2012, Chapter 11).

Motion capture (mocap) is another digital animation technique. Mocap systems capture the live performance of an actor and animate a Three dimensional (3D) character using the recorded motion. There are three main types of mocap systems available nowadays: magnetic, mechanical, and optical systems (KITAGAWA; WINDSOR, 2012, pp. 8–12). Optical mocap systems can be further divided into two groups: markerless and marker-based optical systems. The later uses a set of infrared cameras to track the position over time of reflective markers placed on the body surface of the performer. Furthermore, its most significant advantage is motion tracking with higher accuracy while the cost its a typical disadvantage.

Mocap allows a faster animation production and lessens the time-consuming work of keyframing animation. Additionally, the animation sequences provide realistic motion dynamics since they were captured from human performance. Still, some post-processing is required to clean-up mocap data, when the system loses track of the markers, and to adjust poses of the virtual agent if the resulting animation presents awkward artifacts. Such artifacts include self-penetration and ill-conditioned poses, as, for instance, the hands piercing through the body or the elbow bending backward.

When working with mocap, a typical problem occurs when the performer proportions are too different from the virtual character. For example, when the movements of a skinny actress must be transferred to a fat ogress. Applying the animation sequence in a 3D character with different body proportions and body shape may result in an odd-looking, unnatural motion. The captured motion of an actor covering his ears, for example, when applied to a 3D character with longer arms, results in a weird animation, as in Figure 1.1, since the hands of the avatar will penetrate its head.

A simple solution would be to design the virtual agent to match the proportions and shape of the mocap performer, or to capture the motion with a new actor that resembles the character. The latter approach can be costly and infeasible as it demands as many actors as different characters. The former approach is also restrictive as it restrains the proportions of the characters to be animated. Both approaches preclude the reuse of captured motion data from animating different characters. An alternative and potentially more advantageous approach is motion retargeting. Motion retargeting is the process of adapting motion to 3D characters while avoiding artifacts in the animation.

Figure 1.1: Snapshot of two characters performing the captured motion of covering the ears. The 3D model on the right has arms fifteen percent longer than the model on the left.

1.2 Problem Definition

Distinct structures are used to assemble virtual characters; the primary structures are the character's body surface, represented by a polygonal mesh; the rigging, that binds the mesh to a skeleton, which deforms the mesh vertices as it moves; and the textures of the mesh. The skeleton in virtual characters is composed of joints that, when rotated progressively, gives the impression of motion by deforming nearby mesh vertices. The joints are organized hierarchically, often starting from the hips, the root joint. The skeleton topology, i.e., the number of joints and their orientation, may be different depending on the software used to create the virtual character. Thus, directly applying the same orientation and progressive rotations from one skeleton to another often results in undesired poses.

The present work focuses on retargeting motion acquired through mocap technique to virtual agents that do not share the same body proportions and body surface with the performer. Additionally, it is not required that the skeletons from the mocap data and the 3D character have identical topology. The approach aims to generate an animation that conveys the self-body interaction and the intended information of the original motion by preserving the spatial relationship of the hands with the body surface.

Assumptions related to this work:

- The mocap data is already "clean" and problems due to poor marker tracking, like those resulting from occlusion of markers or swapped markers, were already processed and solved. Otherwise, artifacts in the source motion data will not be fixed and may be magnified.

- The source and target skeletons, that is, the output skeleton of the mocap system and the virtual agent skeleton, are biped, human-like shaped.

- The animations for the mocap performer body surface calibration, described in Section 3.2.1, are available. Moreover, the 3D model of the target character should also be available to retrieve its body surface information through manual inspection.

1.3 Objectives and Contributions

This work presents an automated motion retargeting algorithm that allows mocap data and animations to be reused between virtual human characters with different body proportions and shapes. The algorithm adjusts the pose of the target virtual character to preserve the spatial relationship with the body surface as the original motion given a body surface calibration of both characters. The process diminishes distortions and artifacts when applying mocap data to different virtual characters. That allows the reuse of motions captured to animate multiple characters. The automatic motion retargeting also reduces the animator specialist's costly and time-demanding work, since often is not one motion that requires inspection and adjustments but several motions. The objectives of the present work are summarized as follows:

- Preserve the verisimilitude of 3D virtual agents' motions captured through mocap;
- Allow the reuse of captured motions in several characters with different body proportions and sizes;
- Avoid artifacts and ill-conditioned poses in the animations;
- Reduce the inspection and editing time required of the professional animator.

The dissertation also presents an evaluation protocol and the analysis of the results of the perceptual evaluation of animations generated by the implemented method.

The challenges of motion retargeting and intelligibility of animated virtual agents were also partially reported in the following publications:

- TONOLI, R. L.; COSTA, P. D. P.; DE MARTINO, J. M. Signing Avatar in Virtual Reality: An Intelligibility Study. In: 20th Symposium on Virtual and Augmented Reality, 2018, Foz do Iguaçu. Proceedings of the WTD 20th Symposium on Virtual and Augmented Reality, 2018.
- TONOLI, R. L.; COSTA, P. D. P.; DE MARTINO, J. M. Semi-Automatic Application for Motion Retargeting of Libras Signing Avatar. In: Décimo Encontro dos Alunos e Docentes do Departamento de Engenharia de Computação e Automação Industrial, 2017, Campinas. Digital Proceedings. Campinas: FEEC: Unicamp, 2017. Available at: <https://www.fee.unicamp.br/dca/x-eadca-programacao>. Accessed: Jan. 13, 2019.

1.4 Organization

The text is organized as follows:

- Chapter 2 presents the main related works involving motion retargeting and motion editing.
- Chapter 3 describes the motion retargeting methodology and details the steps to calibrate the characters' surface, encode the hand relationship with the surface and adjust the target character pose.
- Chapter 4 describes the perceptual evaluation conducted to validate the approach.
- Chapter 5 presents the conclusion and discusses future works.
- Appendix A describes the Biovision Hierarchy (BVH) file format.
- Appendix B presents the procedure of the body surface calibration during motion capture.

Chapter 2

Basic Concepts and Related Works

In this chapter, we introduce some basic concepts related to the synthesis of 3D animated characters with the aid of motion capture technology, highlighting the main challenges involved in the motion retargeting process, and the main problems that may arise in the final animation. We also present a chronological review of the literature, with the existing approaches to minimize artifacts caused by unconstrained motion retargeting.

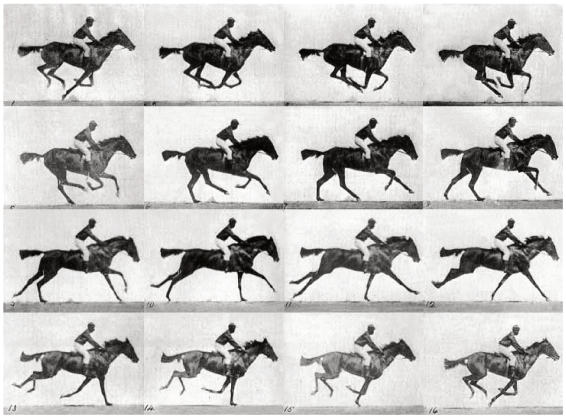
Sections 2.1 and 2.2 describe the motivation behind the use of motion capture (mocap) systems, and provide a brief overview of the existing mocap technologies. The traditional mocap pipeline is detailed in Section 2.3. Section 2.4 presents the related works and discusses some limitations of the proposed approaches. Section 2.5 presents the concluding remarks of the chapter.

2.1 Historical Perspective

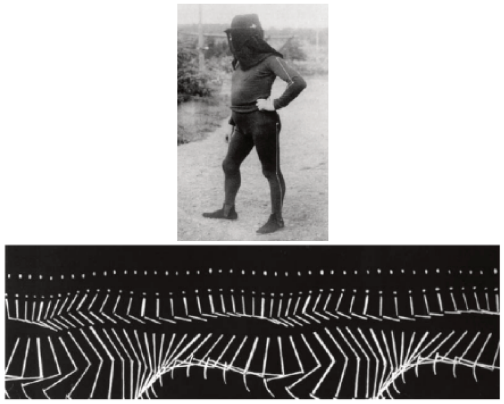
The idea of capturing a movement to reproduce it on a screen is intimately related to the origins of cinema and animation. In 1876, Eadweard Muybridge studied the movement of a racehorse by photographing it with 24 cameras in different positions (Figure 2.1a) (PARENT, 2008, p. 101). In 1882, Etienne-Jules Marey met Muybridge. In the same year, Marey invented a chronophotographic, a fixed-plate camera that allowed him to register sequential images of a movement (Figure 2.1b). Max Fleischer and his brothers patented the *Rotoscope* in 1917. The *Rotoscope* is an apparatus that became very popular in the early animation industry. It enables the animators to draw their characters over a transparent screen in which it is projected the filmed movements of real actors (Figure2.1c). The *Rotoscope* technology was adopted since the first full-length feature film to use cel animation, Disney's 1937 film "Snow White and the Seven Dwarfs". Most of the dancing scenes and all of Prince Charming's character movements were rotoscoped (the Snow White character's movements were *copied* from the Marge Champion model).

Later, computers were exploited to produce digital animations. The development of the computer graphics fields brought computer-generated images from two-dimensional

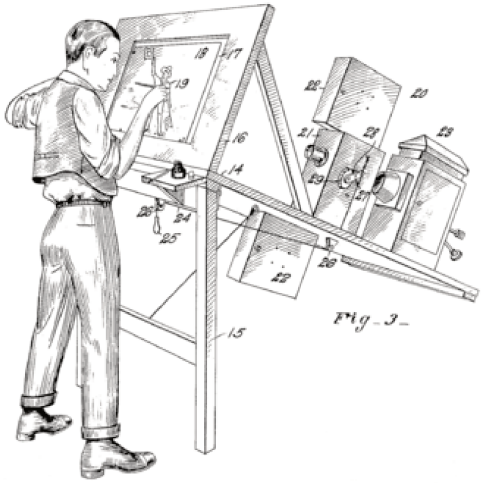
wireframe objects to three-dimensional virtual worlds capable of simulating physics, lighting, and presenting animated virtual characters. Although earlier techniques still have its prestige and niches, computer animation has grown exponentially due to its advantages, such as creating scenes and visual effects otherwise impossible to reproduce in the real world, and due to the vast demand for fast-paced content production.



(a)



(b)



(c)

Figure 2.1: (a) Eadweard Muybridge's galloping horse photographs. (b) The Marey's precursor of a mocap suit and the record of a movement dynamics. (c) *Rotoscope* by Fleischer. Source: (a), (b) Wikimedia Commons. (c) Extracted from (PARENT, 2008, p. 10)

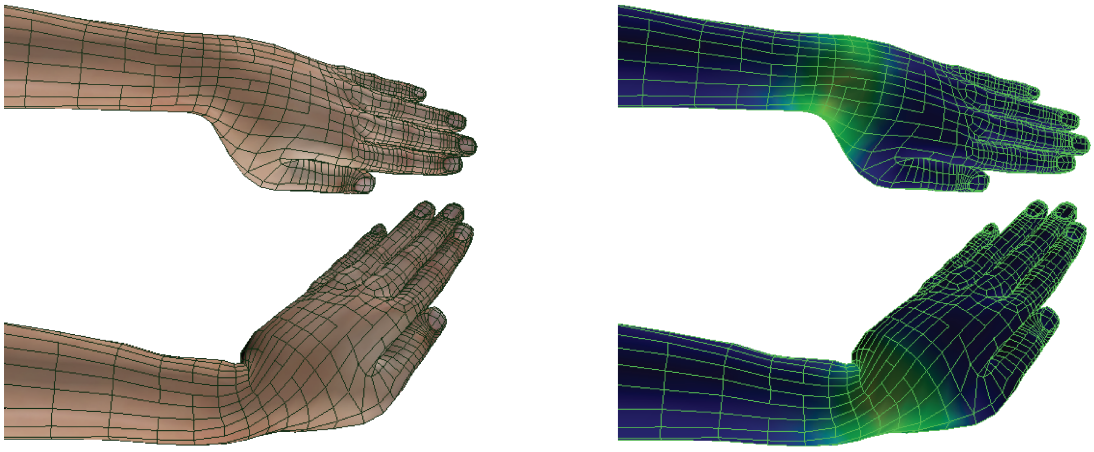
Virtual characters (and objects) are mostly represented by its surface, a hollow polygonal mesh (see Figure 2.2a). 3D modeling software has tools to aid the creation of the characters, its clothes, the environment, and objects of the virtual world. The animation is produced by incrementally transforming the mesh polygons, that is, rotating and translating vertices to move some part of the character. However, as an example, it is not intuitive and difficult for humans to know the transform required for each vertex near the shoulder to raise the character's arm. Hence, often commercial 3D software adopts the skeletal animation technique, in which a skeleton is adapted to the virtual character.

The skeleton is a hierarchical structure composed of joints, yellow circles in Figure 2.2b, and bones that connect a parent joint to its child joint, indicated as yellow lines. The skeleton is mostly used for movement; that is, when a joint is rotated, it moves a bone; then, nearby polygons are deformed accordingly to move the virtual characters' mesh (or surface) (RATNER, 2012, p. 183). Weights based on the proximity to polygons are assigned to bones for deforming a particular region of the mesh (BARAN; POPOVIć, 2007). Figure 2.3 presents a weighting and deformation example for the left-hand bone, where the green-colored region indicates higher weights than blue regions. The process of adapting the skeleton structure to the character's surface and weights assignment is called *rigging*.

(a) (b)

Figure 2.2: (a) Wired polygons of the virtual character surface in black. (b) Skeleton structure of the virtual character in yellow, the circles and the lines represent the joints and bones, respectively.

The *keyframing* technique exploits the (more) intuitive skeletal structure. Animation sequences are created based on its key moments. However, unlike Muybridge's key-pose photographs or key-poses generated by drawings on the *Rotoscope*, the computer graphics animator professional rotates joints to change the character's pose. Once the key-poses are defined, an algorithm computes the character poses to fill the gaps between two key moments. Although providing great control over the animation production, *keyframing* demands much time from the animator professional. Even habitual actions, such as walking, are very hard to reproduce naturally; several body parts are moving simultaneously, some with different velocities or opposite directions, human motion dynamics may vary with the body weight, among others. Modern mocap technologies avoid the challenges of reproducing human motions by capturing those motions from a real human subject.



(a) (b)

Figure 2.3: Straight (top) and rotated (bottom) left hand of a virtual character. (a) Wired polygons of the hand. (b) Weights of the left hand bone, blue and green colors indicate regions with lower and higher weights.

2.2 Mocap Technology Overview

Mocap systems are divided into three main groups:

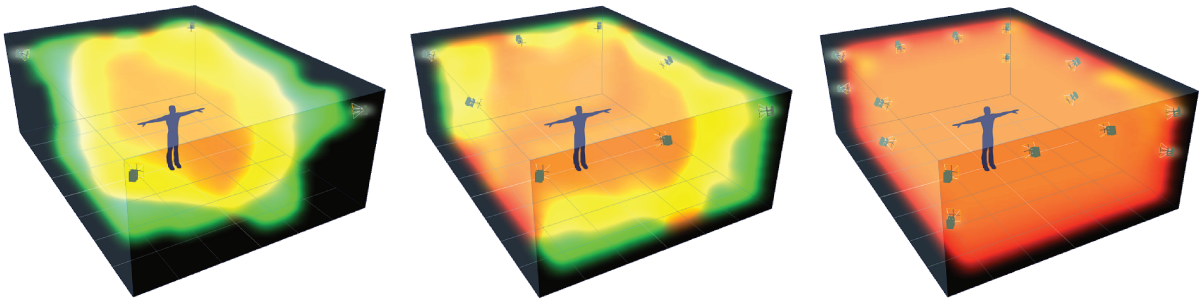
- **Optical systems**: mostly uses reflective (passive) or light-emitting (active) markers attached on the subject's skin or to a mocap suit. Passive markers reflect the infrared lights emitted by the mocap cameras, while active markers emit their light. Since the light intensity is inversely proportional to the square of the distance from the source, active markers systems are less sensitive to light interference from other sources. However, subjects need also to carry batteries to power the markers.

- **Magnetic systems**: subjects use tracking sensors rather than markers. Each sensor tracks its translation and orientation regarding a magnetic transmitter. As the active markers, the subjects must wear wires and batteries to power the sensors, which can limit their movements.

- **Mechanical systems**: are articulated exoskeletons that track its joint angles using accelerometers. Although they are free of marker occlusions and magnetic interference, the subject's movements are further restricted.

Optical mocap systems based on reflective markers are widely popular. These systems provide the best accuracy and allow a greater range of motion while performing the action sequences. Optical mocap systems require placing cameras around the capture volume to track the markers on the subjects' bodies. The greater amount of cameras avoid markers occlusion and swapping, which lessen the data cleaning and editing time. Figure 2.4 presents a heat map indicating the coverage regions for three mocap setups with different number of camera. Red-colored regions indicate higher accuracy for markers

tracking while green-colored regions the opposite. Given a set of infrared cameras, they must be carefully positioned to maximize the regions covered by at least two or three cameras.



(a) (b) (c)

Figure 2.4: Heat map of capture volume covered by mocap infrared cameras. Regions visible to at least two, three, and four or more cameras are colored in green, yellow, and red, respectively. (a), (b) and (c) volumes contain 4, 8 and 16 cameras. Source: Extracted from <https://www.vicon.com/visualization>. Visualization simulated using Vicon Vantage™ V16 cameras with 18mm lens.

In the present work, an optical mocap system with nine Vicon Vantage™ infrared cameras and retro-reflective markers were used. The hardware system is accompanied by a dedicated software framework Vicon Shōgun™, which interacts with the hardware and implements the pipeline described in the following section.

2.3 Mocap Pipeline

Established the camera positioning and having a rigged virtual character, a typical mocap pipeline for animation purposes, can be summarized as follows:

- **System calibration** depicts for the mocap system the position of the cameras relative to each other, the position of the origin, and the orientation of the volume (axes orientation). Typically, a reference object is used. Our Vicon® system uses a wand with four differently-spaced light-emitting diodes (LEDs) to perform the system calibration.

- **Subject calibration** aids the system in interpreting the set of markers being tracked and identifying each one. It consists of placing the markers on the performer; the markers are placed as the standard marker setup of the mocap software or a customized setup (predefined). The system identifies each marker based on a reference pose, often the T-Pose (see Figure 2.5), and the range of motion, which is a motion sequence that exploits all joints' degrees of freedom (DOFs).

- **Capturing takes** is the desired motions performance; starting and ending a capture take in the T-Pose may later facilitate applying the motion to a virtual character.

- **Marker data to skeleton**, each camera records Two dimensional (2D) images of the reflected infrared light. The mocap software reconstructs the 3D markers position and trajectory. In the marker data, if some markers are swapped or occluded in a given moment, it is possible to indicate its correct position or interpolate the data using adjacent frames. Finally, the mocap software adapts a skeleton to the tracked markers.
- **Motion retargeting** is the process of transferring the motion from one skeleton to another. In particular, the source skeleton is the mocap system output, while the virtual character's skeleton is the target skeleton. A straightforward motion retargeting process is directly copying joint angles from one skeleton to the other.
- **Rendering** using 3D animation software or game engine is the final step to animating virtual characters with the mocap technology.

Figure 2.5: Motion capture performer in the T-Pose with the Vicon Shōgun™ standard 53 marker setup.

After rendering, the animator professional will inspect the animation for artifacts or distortions. When the animation presents glitches such as instantaneous bones flexing or twisting, it may indicate that markers were occluded or swapped; then, marker data should be corrected. However, if there are lots of glitches or it continues throughout the animation length, recapturing the takes may be necessary. It may also indicate that the mocap system is not suited for the captured movement, there are not enough cameras, or the cameras do not see most markers.

However, other distortions that may be found are self-penetration and ill-conditioned poses, e.g., the mocap performer covered his mouth with the hand in the

original motion, but the virtual character's hand is on its forehead. As the example, artifacts in the animation make it less appealing and may hide the original motion semantics. These distortions appear when the performer and the virtual character have different body proportions or body shape. In addition, since there is not a standard way to compose the skeleton structure, their skeletons' topology may be different, which also leads to such artifacts.

Some *keyframing-based* solutions allow the animator professional editing the skeleton motion by rotating the joints (thus, the bones) and correctly posing the character, while others expect the user to define the correct position of a joint and then determines the joint angles to pose the character. The techniques adopted by the first and second solutions groups are called forward and inverse kinematics, respectively (see Figure 2.6). Both solutions require considerable human intervention to adjust the animation.

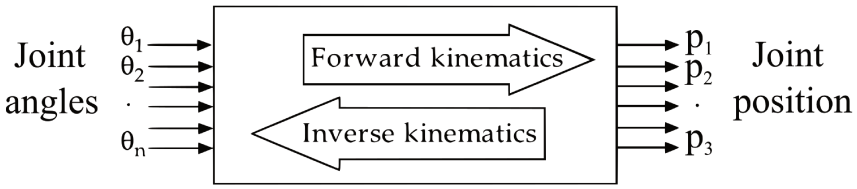
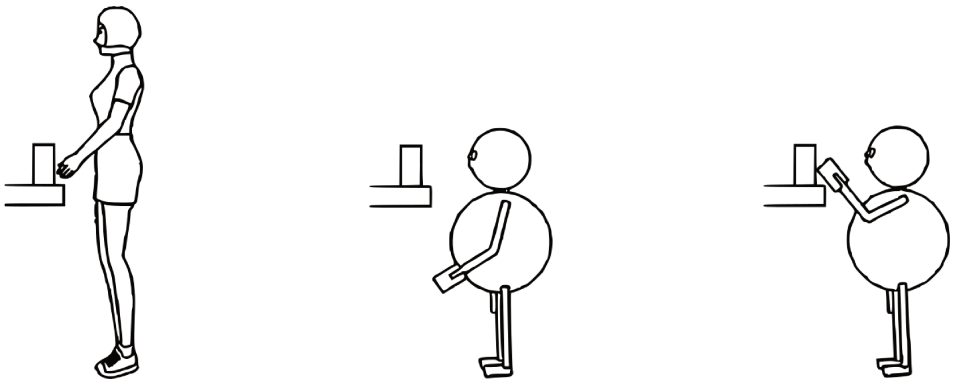


Figure 2.6: Forward and inverse kinematics. Source: Extracted and adapted from (KUCUK; BINGUL, 2006).

Moreover, often, the reuse of captured motions to several characters is desired. In consequence, conventional methods of motion editing to adjust virtual characters in animation sequences present a costly and time-demanding solution. Differently, motion retargeting solutions can avoid some of the unwanted artifacts automatically. Since a variety of situations may be represented by an animated sequence, different motion retargeting strategies may be applied (KITAGAWA; WINDSOR, 2012, pp. 73–74). Each process seeks to preserve (or avoid) key aspects of the motion depending on the situation. As an example, consider retargeting two captured motions of an adult performer to a smaller virtual character: first, the motion to grasp a glass of water on a table, as in Figure 2.7; and second, the movement of arms folding. In the first sequence, a position in the volume is a constraint for the arm, and the child character must reach for the glass even if it means raising the hand. Differently, in the second case, the child's arms must fold across the chest, despite its position relative to the world or its surroundings.

The next section of this chapter presents some researches focused on diminishing the distortions and reducing the required intervention by the animator professional. The present work focuses primarily on motion retargeting approaches since the mentioned distortions emerge when transferring motions between skeletons. However, solutions that depend on user interaction are also relevant to observe common distortions and aspects of the original motion that often are desired to preserve.



(a) (b) (c)

Figure 2.7: (a) The motion of an adult to grasp a glass of water on a table. (b) The motion retargeting to a smaller character preserving the pose of the arm. (c) The motion retargeting to the same smaller character but constraining the position of the hand. Source: Extracted and adapted from (SHIN *et al.*, 2001).

2.4 Related Works

The following sections present the related works to this dissertation. They were divided into two sections: Section 2.4.1 contains the proposed techniques of motion retargeting to virtual characters with different body proportions (i.e., different heights and limbs' size) or different topologies or both. The techniques that account for body surface information during the motion retargeting process are presented in Section 2.4.2. Section 2.4.3 presents a literature summary of the related works.

2.4.1 Motion Retargeting to Different Characters

Gleicher (1998) defined motion retargeting as the problem of adapting motions between characters. He provided a methodology to adapt motion between characters with the same hierarchical structure but with different proportions. With a set of spacetime constraints specified by the user, his method uses Inverse kinematics (IK) to compute the target character's pose that best satisfies the constrains. Such constraints define the position of an end-effector in a given frame, a point in the character that interacts with the environment, such as a foot that touches the floor or a hand holding an object (GLEICHER, 1998). Observe that Gleicher's motion retargeting process adapts the motion to different characters but relies on user input to define which aspect of the original motion to be preserved.

The problem of retargeting motions to characters with different hierarchical structures was later handled by using a correspondence between skeletons manually provided by the user (MONZANI *et al.*, 2000). As Figure 2.8 shows, a one-to-one joint correspondence is detailed. Then, the bones between paired joints in the target skeleton are aligned to the corresponding bone in the source skeleton. Monzani *et al.* (2000) also

allowed the user to define constraints to correct and edit the resulting motion. As Gleicher (1998), IK with the pseudo-inverse method was used to enforce the constraints.

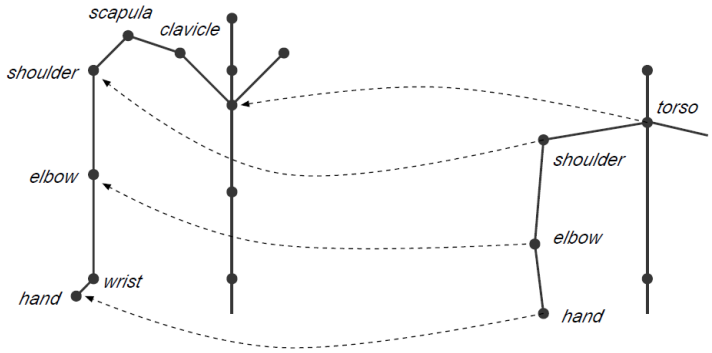


Figure 2.8: Correspondence between two skeletons with distinct topology. Source: Extracted from (MONZANI *et al.*, 2000).

Shin presented an automatic motion retargeting process for *computer puppetry*, which maps the movements of a performer to a virtual agent in real-time (SHIN *et al.*, 2001). Since it is not possible to rely on predefined constraints for online applications, a set of rules was proposed to decide whether to preserve the angles of the joints, the pose, or the position of the joints, the interaction with the environment, of the original motion. This decision is based on the distance of an end-effector to an object. That is, when the end-effector is close to an object, interaction is likely occurring. Then, the end-effector's position is given higher importance than the angles of the joints in the motion retargeting process. Conversely, when an end-effector distances itself from an object, the joint angles are given high priority and thus mimicking the original motion.

Later, Kulpa *et al.* (2005) introduced a normalized representation of the skeleton to depict the motion. The position of the hands, feet, and head are normalized by the length of the respective kinematic chain. The chains are composed of bone segments from the extremity to the root of the chain (the base of the spine, shoulders, or upper legs joints). The motion adaptation is performed in real-time by inverting the normalization process using the dimensions of the target character, which leads to the desired position for each extremity. The target character's pose is then computed analytically for each kinematic chain separately, as in Figure 2.9. Different from the previous works, Kulpa *et al.* (2005) provided a data structure that encodes the motion, based on the skeleton proportions, rather than defining a motion representation that creates a trajectory for the target character to follow. This work was later extended to a real-time motion synthesis framework that adapts existing captured motions given changes in the environment and perturbations, as adding a step on the floor or applying external forces on the character (MULTON *et al.*, 2009).

The problem of motion retargeting between characters with different topology was also addressed for characters with highly distinct structures. Hsieh *et al.* and Lu

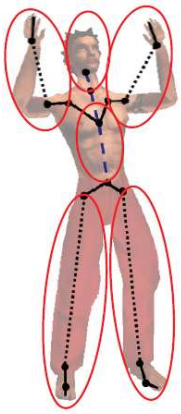


Figure 2.9: Representation of the skeleton by Kulpa *et al.* (2005), the dotted lines represent limbs with variable length. Group of joints, highlighted in red, are adapted separately. Source: Extracted from (KULPA *et al.*, 2005).

et al. applied Monzani's methodology animal-to-human and human-to-animal motion retargeting, respectively (HSIEH *et al.*, 2005; LU *et al.*, 2009). Moreover, Hecker developed a motion retargeting and motion editing application to animate characters created by the user, not limited to bipeds and quadrupeds (HECKER *et al.*, 2008).

However, previously cited methods have no understanding of the body surface, which leads to artifacts when the original character interacts with its own body. Although the cited approaches provide mechanisms to solve the undesired aspects of the motion, they require the manual and artistic intervention of a professional animator, resulting in higher post-production costs and longer lead times.

Recently, Bernardin *et al.* (2017) presented a motion retargeting process that seeks to preserve self-body interaction without surface information. The distances between all joints are computed and normalized in a process similar to Kulpa *et al.* (2005). The body proportions of the target character are used to compute the inverse of the normalized distance. Then, the target joints are positioned to preserve the computed distances. An objective function prioritize distances between closer joints in the skeleton hierarchy (not Euclidean distance); given two joints, a low amount of joints between them indicates high priority. For example, to preserve the distance between the hand and the hips is considered more relevant than the one between the hand and the foot. While this approach can sustain some semantic information of the source motion, self-penetration occurs by changing the body surface of the characters. Hence, there is a clear demand for surface-aware algorithms to handle animation sequences with self-body interactions accurately. The following section describes some of the existing approaches.

2.4.2 Surface-Aware Motion Retargeting

Ho *et al.* (2010) proposed a method for motion retargeting based on a volumetric mesh(HO *et al.*, 2010).The mesh is composed of segments connecting joints of the characters

and vertices of the objects in the environment. The method minimizes the mesh deformation when the character shape changes to preserve the interaction. With similar objectives but focusing on reducing the computational cost, Al-Asqhar *et al.* (2013) proposes expressing a joint position as a weighted sum of the relative distance to sampled points on a surface (AL-ASQHAR *et al.*, 2013). The motion is adjusted by computing the position of the joint when the environment around it changes, handling close interactions. However, as pointed by the authors, the methods may fail with large changes in the characters' shape.

The main problem with the Ho *et al.* (2010) and Al-Asqhar *et al.* (2013) approaches is that the body segments were represented as capsules and boxes, as shown in Figure 2.10, which allows little deformations in the body surface. Additionally, measuring the body surface of the mocap performer remains a problem, since no body surface calibration or estimation was detailed.

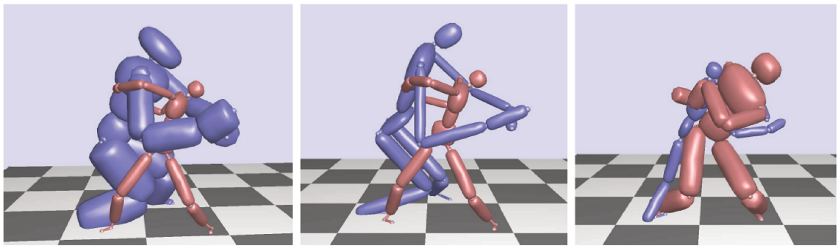


Figure 2.10: Results presented by Al-Asqhar *et al.* (2013) of a motion retargeting to different sizes characters. Source: Extracted and adapted from (AL-ASQHAR *et al.*, 2013)

State-of-the-art works on human shape representation include *MoSh* (LOPER *et al.*, 2014) and *Dyna* (PONS-MOLL *et al.*, 2015). *MoSh*, (Motion and Shape capture) extracts body surface information from the markers of mocap data. Given the mocap markers, it estimates body shape and pose that fits the markers, and it is also capable of deforming the body surface according to the animation. *Dyna* is a mathematical model that synthesizes surface motion and deformations given the mocap data and the body shape (as their template). Both *MoSh* and *Dyna* provide a highly detailed description of the human surface, but they focus on surface motion synthesis and surface motion retargeting rather than skeletal animation. This prevents the use of the generated animation sequences on game engines or commercial animation software since they adopt a skeleton-based approach.

Jin *et al.* (2018) proposes a motion retargeting method that preserves the spatial relationship between characters. The character models were extracted from *Dyna*; thus, they share the same topology. Motion interactions with other characters and with the self-body surface are expressed by collisions between meshes (JIN *et al.*, 2018). When retargeting to characters with distinct body shape or proportions, the process enforces that the collisions occur in the same position as the original motion. That is, the poses of the target characters are adjusted so that their meshes are colliding as the source motion.

A similar approach was presented by computing local interactions with the surface (LIU *et al.*, 2018). Both works use optimization algorithms to attract the current pose to the computed pose of the character.

The methods proposed by Loper *et al.* (2014), Pons-Moll *et al.* (2015), Liu *et al.* (2018), and Jin *et al.* (2018) require characters with a surface mesh compatible with their template. That prohibits the use of commercial software characters since each software adopts a particular standard to compose the mesh and the skeleton of their virtual characters. Hence, these strategies demand enormous adaptations in the mocap pipeline. Research groups or animation studios with an already-defined pipeline may find these adaptations a significant impediment since it means redesigning all characters.

Molla *et al.* (2017), on the other hand, proposed a motion retargeting process based on a surface calibration of the mocap performer and the virtual character (MOLLA *et al.*, 2017). The performer calibration exploits the mocap system: by touching a defined set of points on the trunk, head, and limbs, a virtual representation of the performer's surface is estimated. The set of points is also sampled on the surface of the virtual character. The points of the head and trunk are connected by triangles, creating two meshes, and the limbs are modeled by capsules. Thus, as shown in Figure 2.11, both surfaces can be correlated.

Molla *et al.* (2017) decomposed extremity joints' positions into a weighted sum of the displacement vector from the body surface components, mesh triangles, and capsules. This approach is similar to the one proposed by Al-Asqhar *et al.* (2013), that decomposed the joint positions relative to surfaces from other characters' bodies and the environment. An analogous decomposition is computed for joints' orientation regarding the normals of the body surface components. The sum's weights increase with the proximity and orthogonality to the components. These parameters, defined as egocentric coordinates, compose a data structure that depicts the character pose — and the motion after computing for every frame —. Then, extending the approach proposed by Kulpa *et al.* (2005), normalizing these coordinates by the body proportions of the mocap performer allows the motion retargeting to characters with different body shapes and body proportions. However, the approach does not handle character skeletons with different topologies. Molla *et al.* (2017) compared their methodology with Kulpa *et al.* (2005) and Al-Asqhar *et al.* (2013). The evaluation results, to the detriment of the other two, showed that the volunteer subjects preferred the animations retargeted by their methodology.

2.4.3 Literature Summary

Table 2.1 presents a summary of the related works discussed in Sections 2.4.1 and 2.4.2. Loper *et al.* (2014) and Pons-Moll *et al.* (2015) were not included because they focus on mesh animation for realistic soft tissue deformation effects.

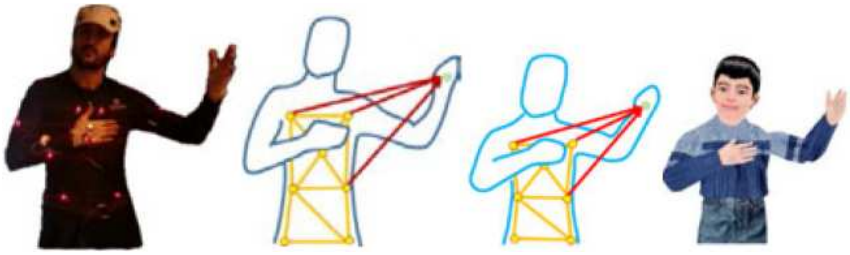


Figure 2.11: Illustration of the methodology proposed by Molla *et al.* (2017). From left to right: mocap performer; left-hand spatial relationship with the surface for the performer; spatial relationship transferred to a child character; final result. The yellow and red lines illustrate the calibrated surfaces and spatial relationships, respectively. Source: Extracted and adapted from (MOLLA *et al.*, 2017)

Reference	Different Topology	Environment Interaction	Self-Body Interaction	Automatic	Real-Time
Gleicher (1998)		X			
Monzani *et al.* (2000)	X	X			
Shin *et al.* (2001)		X		X	X
Hsieh *et al.* (2005)	X				X
Kulpa *et al.* (2005)		X	X[1]		X
Hecker *et al.* (2008)	X				X
Lu *et al.* (2009)	X				X
Multon *et al.* (2009)		X			X
Ho *et al.* (2010)		X	X	X	X
Al-Asqhar *et al.* (2013)		X	X		X
Bernardin *et al.* (2017)			X[1]	X	
Jin *et al.* (2018)		X	X	X	
Liu *et al.* (2018)			X	X	
Molla *et al.* (2017)		X[2]	X	X	X
This work	X	X[2]	X	X	

Table 2.1: Summary of motion retargeting techniques presented in Sections 2.4.1 and 2.4.2. (1) does not use surface information. (2) handles only ground interaction.

In this work, we propose to include the concept of skeleton correspondence (*Skeleton Map*) adopted by Monzani *et al.* (2000), to the egocentric coordinate encoding pipeline proposed by Molla *et al.* (2017). In Chapter 3, we detail how the *Skeleton Map* should be constructed to overcome the limitation of Molla *et al.* (2017). We present a complete implementation of the new proposed pipeline that has no restriction regarding the target character's skeleton topology, as highlighted in the last row of Table 2.1.

2.5 Concluding Remarks

This chapter presented some basic concepts related to the animation of virtual characters. In *keyframing-based* techniques, the artist (or professional animator) has significant influence over the character pose and the motion dynamics. Advances in computer vision and mocap technology allowed to track and transfer real human motion to virtual characters, leading to an animation production more automatic and diminishing human intervention in several steps of the traditional pipeline, including motion editing. However, often reusing the mocap data for multiple characters is a challenge because it causes distortions in the animation when the body proportions and the body shape of the characters and the mocap performer present disparities. An additional challenge is presented when the characters were created and rigged using different commercial animation software since each follows a specific standard. An overview of the mocap technology and its traditional pipeline are also detailed. Next, it was discussed some works that focus on motion retargeting between characters with different topologies or different body proportions and shape. Finally, the approach of this work and how it relates to existing approaches were presented. The implemented methodology is described in the next chapter.

Chapter 3

Preserving Surface Spatial Relationship in Motion Retargeting

This chapter describes the motion retargeting process from mocap data to a 3D virtual human. Figure 3.1 presents an overview of the motion retargeting pipeline.

The first step consists of an *Initial Motion Retargeting* that transfers the mocap data (source animation) to the virtual character skeleton (target skeleton) by aligning and rotating each bone of the target character as the source animation. However, the retargeting may generate artifacts in animations with self-body interaction when the mocap actor and the virtual character have different body proportions or body shape. These artifacts, such as self-penetration and ill-conditioned poses, hinder motion recognition and cause strangeness. The *Initial Motion Retargeting* step is detailed in Section 3.1.

The *Spatial Relationship Encoding* process tackles the distortions. This process expresses the position of the hands and the feet joints of the source skeleton relative to the performer body surface (source surface) and normalized by its body proportions. Later, the virtual character body surface (target surface) and body proportions are used to adjust the target animation resulted from the *Initial Motion Retargeting*. Section 3.2 describes the process to calibrate both body surfaces and to encode the joints' spatial relationship with the surface.

Finally, the *Pose Adaptation* step computes the joints positions that ensure the same spatial relationship with the body surface as the original motion (mocap data), which preserves self-body interaction and avoids distortions. Then, an inverse kinematics algorithm computes the joints angle to adjust the target character in the desired pose (Section 3.3). The motion retargeting results in an animation sequence for the virtual character as a BVH file (Appendix A details the BVH file format data structure).

The last two sections of this chapter presents the experimental setup for recording and processing the mocap data (Section 3.4) and the concluding remarks (Section 3.5).

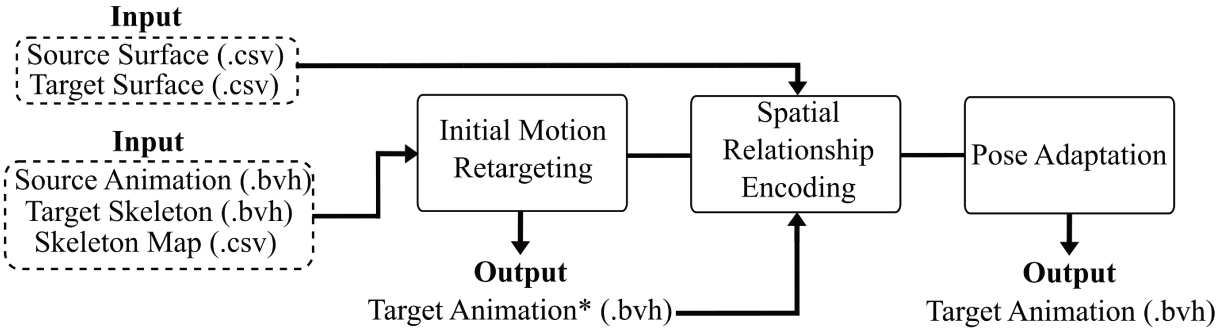


Figure 3.1: Workflow overview of the motion retargeting process that preserves the spatial relationship of joints with the body surface. It receives the mocap data (source animation), the virtual character skeleton (target skeleton), a joints correspondence between both skeletons (skeleton map), and the calibrated surface from both the mocap performer (source surface) and the virtual character (target surface). (*) refers to the retargeted animation sequence disregarding surface relationship.

3.1 Initial Motion Retargeting

This section describes the *Initial Motion Retargeting* process, which receives three inputs: (1) the source animation (BVH file); (2) the target skeleton (BVH file with a single frame); and (3) a correspondence between their joints, the *Skeleton Map* (a text file in which each line contains a correspondence pair between joints in the target skeleton and the template skeleton). The output is the target animation, that is, the target skeleton mimicking the movements of the source skeleton.

The BVH file depicts the hierarchical structure of the skeleton, its topology. The source animation contains the source skeleton topology and the source motion, the joints rotation and translation for every frame. However, since the virtual character is not animated yet, a BVH with a single frame depicts its skeleton structure. The *Initial Motion Retargeting* process computes the target joints rotation and translation to compose the target animation.

The *Initial Motion Retargeting* requires that the initial pose of the source skeleton be the T-Pose. The mocap actor achieves the T-Pose by raising the arms to the sides of the body, placing them parallel to the ground, and keeping both arms and legs straightened (as in Figure 2.5). The target skeleton should also be in the T-Pose.

The output of the *Initial Motion Retargeting* is the target animation disregarding body surface information. At this point, the motion does not exploit yet any "surface-aware" adjustment and may present artifacts and ill-conditioned poses, as shown in Section 3.1.5.

3.1.1 Skeletal Animation

A skeleton is animated by rotating and translating its joints across the frames, conveying the impression of motion. The hierarchical representation of the skeleton allows

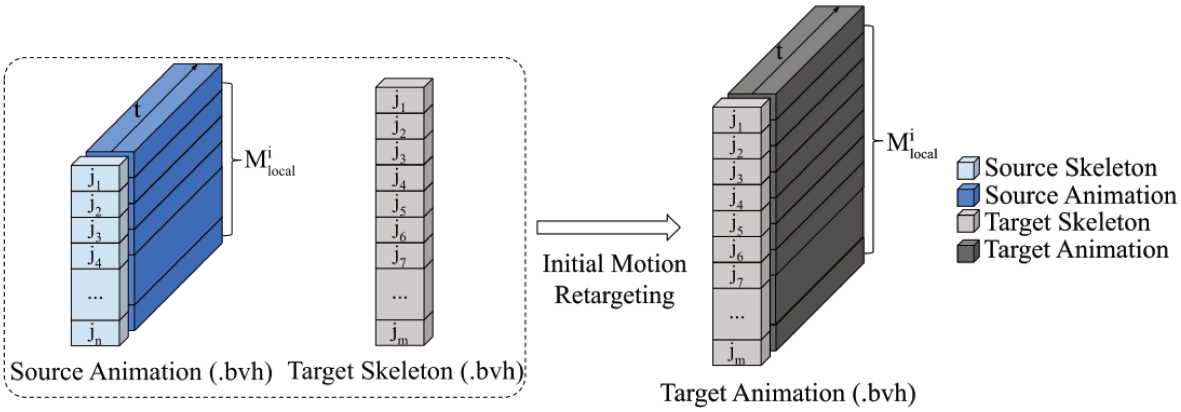


Figure 3.2: The *Initial Motion Retargeting* process transfers the mocap data, the source animation, to the target skeleton.

an all-around description of the motion by only preserving the rotation and translation of a joint regarding its parent joint, i.e., local rotations and translations. This section presents the technical background required to compute the global orientation and position a joint; these parameters are relevant in a motion retargeting process.

The matrix in Equation 3.1[1] represents the displacement of a joint regarding its parent joint, its local translation T_x, T_y and T_z.

$$T = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

The orientation of the joints is given by rotations along the principal axes of its parent joint. For the Euler angles θ_1, θ_2 and θ_3, corresponding to rotations around its parent local z-, x- and y-axis, the rotation matrices are expressed as Equations 3.2, 3.3, and 3.4, respectively.

$$R_z(\theta_1) = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

$$R_x(\theta_2) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta_2) & -\sin(\theta_2) & 0 \\ 0 & \sin(\theta_2) & \cos(\theta_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

[1]Homogeneous coordinates are used to represent translations and rotations as matrices.

$$R_y(\theta_3) = \begin{bmatrix} \cos(\theta_3) & 0 & \sin(\theta_3) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_3) & 0 & \cos(\theta_3) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

Equation 3.5 combines these rotation matrices into a single rotation matrix through matrix multiplication.

$$R = R_z(\theta_1)R_x(\theta_2)R_y(\theta_3) \tag{3.5}$$

Since matrix multiplication is not commutative, the order of the channels specified in the BVH file (Appendix A) dictates the multiplication order. Finally, Equation 3.6 combines both translation and rotation matrices, resulting in the transformation matrix M[2].

$$M = TR \tag{3.6}$$

The hierarchical structure of the skeleton allows the representation of the joint position and orientation, relative to its parent. Therefore, only its local rotation and translation are stored. Since the root joint has no parent, its local values correspond to the global ones, that is, relative to the world origin. To obtain the global transformation matrix M_G^n for a joint n, its local transformation matrix M_L^n is multiplied by its parent local transformation matrix and so on, up to the root joint $n = 0$. Equation 3.7 presents how to compute the global transformation from joint n (MEREDITH; MADDOCK, 2001).

$$M_G^n = \prod_{i=0}^{n} M_L^i \tag{3.7}$$

The global position of joint n is computed by multiplying its global transformation by the vector $[0, 0, 0, 1]^T$ (Equation 3.8)[3].

$$\mathbf{p}_n = M_G^n \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.8}$$

The orientation relative to the origin, i.e., the global orientation, can be recovered from the transformation matrix (SHOEMAKE, 1994). Considering the rotation matrix

[2] A scaling operation could also be included in a transformation matrix, but it is not usual to scale mocap data during the animation. Thus, it is not taken into account in this work.

[3] This work adopts the right-to-left notation for matrix multiplication: given a point $\mathbf{p}_0$ and a transform matrix M, the transformed point $\mathbf{p}_1$ is computed as $\mathbf{p}_1 = M\mathbf{p}_0$.

from Equation 3.5, multiplying the matrices using Equations 3.2, 3.3 and 3.4 results in the matrix given by Equation 3.9 , with $c_i = cos(\theta_i)$ and $s_i = sin(\theta_i)$ for $i = 1, 2, 3$.

$$R = \begin{bmatrix} c_1c_3 - s_1s_2s_3 & -s_1c_2 & c_1s_3 + s_1s_2c_3 & 0 \\ s_1c_3 + c_1s_2s_3 & c_1c_2 & s_1s_3 - c_1s_2c_3 & 0 \\ -c_2s_3 & s_2 & c_2c_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

Given the general rotation matrix in Equantion 3.10, it is possible to retrieve at least one solution for the θ_1, θ_2 and θ_3 values with trigonometric operations using the components of the rotation matrix. First, θ_2 is calculated through Equation 3.11, then, if $\cos(\theta_2) \neq 0$, Equations 3.12 and 3.13 provide the values of θ_3 and θ_1, respectively.

$$R = \begin{bmatrix} r_{00} & r_{01} & r_{02} & 0 \\ r_{10} & r_{11} & r_{12} & 0 \\ r_{20} & r_{21} & r_{22} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

$$\theta_2 = \arcsin(r_{21}) \tag{3.11}$$

$$\theta_3 = \arctan 2 \left(\frac{-r_{20}}{\cos(\theta_2)}, \frac{r_{22}}{\cos(\theta_2)} \right) \tag{3.12}$$

$$\theta_1 = \arctan 2 \left(\frac{-r_{01}}{\cos(\theta_2)}, \frac{r_{11}}{\cos(\theta_2)} \right) \tag{3.13}$$

Slabaugh (1999) details the case when $\cos(\theta_2)$ approaches zero, providing a pseudo-code for the computation of Euler angles from rotation matrices.

The global rotation and position are essential to the motion retargeting process between skeletons that do not share the same topology. The next step is to identify correspondent joints in the skeletons so that a joint in the target skeleton receives the transformations of its correspondent joint in the source skeleton.

3.1.2 Skeleton Map

Figure 3.3 presents two examples of possible representations of a virtual human skeleton. Although both can animate a humanoid-shaped model, transferring motions between the skeletons is not straightforward.

A correspondence between joints of the skeletons is mandatory to identify which joints from the target skeleton should mimic the motion from the source skeleton. It is possible to infer a correspondence using the joints' name automatically, but they may

not be placed at the same point in the respective skeleton, as the shoulders in Figure 3.3. Therefore, a correspondence between skeletons is useful to map joints correctly and to perform the motion retargeting (HSIEH *et al.*, 2005; MONZANI *et al.*, 2000).

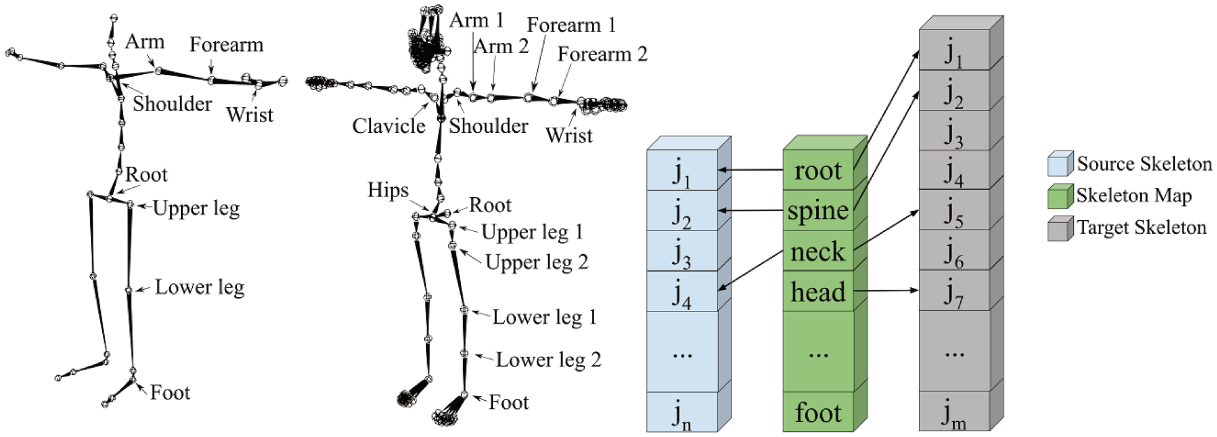


Figure 3.3: On the left part of the image, two skeletons of virtual humans with different topologies. The second skeleton is more complex: the root and the hips joints are not the same, the limbs have intermediate joints, and it has clavicle, feet, hands, and facial joints that are not present in the first skeleton. On the right part of the image, a visual representation of the *Skeleton Map*: a minimal set of named joints (root, spine, neck, etc.) that are required to animate the body of a virtual human and the source and target joints associated to them.

The correspondence of skeletons is specified through the *Skeleton Map*, which is a group of references between joints in the target and source skeletons, respectively. The *Skeleton Map* defines a minimal set of named joints to compose a virtual skeleton. The source and target joints are associated with the virtual skeleton's joints. In Figure 3.3, the virtual skeleton is composed by the joints identified as "root", "spine", "neck", "head", and so on. The *Skeleton Map* illustrated in the figure associates, for example, the source joint j_4 and the target joint j_5 to the generic virtual human "neck" joint.

Joints that are not included in the *Skeleton Map* (such as the source joint j_3 in Figure 3.5) will not be considered in the *Bones Alignment* process, described in Section 3.1.3. If the *Bones Alignment* results in ill-conditioned poses due to incorrect rotations in a particular joint, this joint can be removed from the *Skeleton Map*. The removed joint will then be ignored during the alignment process.

For a template virtual human, the methods described in the next section and the following sections require that both target and source skeletons have at least the following set of joints: one joint representing the hips (base of the spine), two spine joints, a neck joint, a head joint, and right and left joints for the shoulders, elbows, wrists, femurs, knees, and feet, respectively. This set was constructed by intersecting the skeletons' joints from the 3D characters depicted in Figure 3.6 and from the mocap system used in this work.

3.1.3 Bones Alignment

For topologically equivalent skeletons, merely applying the orientations of the source joints at a given frame on the target joints is enough to align the bones, i.e., mimic the source motion. On the contrary, mimicking the motion of a skeleton with distinct topology presents a more challenging problem. The previous section already handles the problem of identifying correspondent joints between the skeletons. However, the orientations of mapped joints may be different even when the skeletons are in the same pose, as the left hands in Figure 3.4.

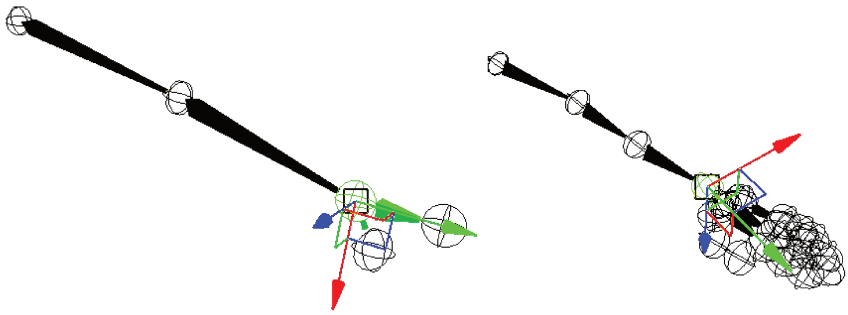


Figure 3.4: The left arm of the source skeleton and the target skeleton on the left and right, respectively. Both palms of the hand are facing the ground with the arms stretched. The hand joints are highlighted in green, and the axes represent their local orientation. Although the skeletons are in the same pose, the local orientation of matching joints is not guaranteed to be equal due to the distinct topology of each skeleton.

The joints orientation are susceptible to change depending on the convention adopted by the 3D software used to create and design the virtual agent. Moreover, a distinct amount of joints prevents the equivalence of joint orientations between skeletons, since a rotated joint on one skeleton may be absent on the other, influencing every joint down the hierarchy. Figure 3.5 also shows the local orientation of mapped joints in colored lines, except for the root joint (red, green, and blue represent the local x, y, and z axes, respectively). The figure illustrates one of the reasons why the *Skeleton Map* and *Bones Alignment* processes are required: copying joint angles from one skeleton to the other will result in ill-conditioned poses.

Bones Alignment is a process focused on aligning the bones of the target skeleton with the bones of the source skeleton, enforcing that the vector of a mapped target joint pointing towards its child joint has the same direction of the correspondent vector from the source skeleton (Equation 3.14). Then, for every frame, the global transformation matrix to rotate a source skeleton joint from the previous frame orientation to the current one is computed and applied to the correspondent target skeleton joint.

The *Bones Alignment* process performs the motion retargeting based on a reference skeleton pose, the T-Pose, as in Figure 3.3. In the first frame, the bones of the target skeleton are aligned with the correspondent bones of the source skeleton. That diminishes

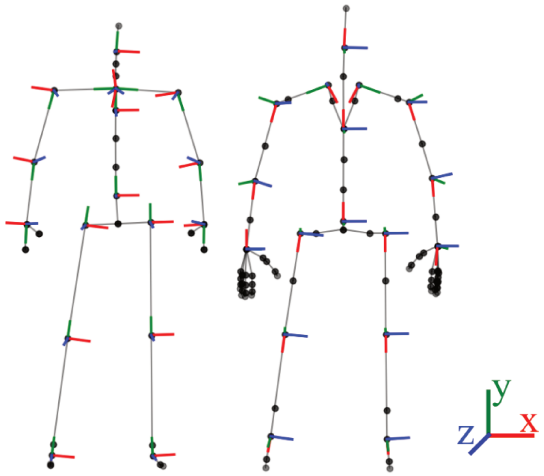


Figure 3.5: The topology of two skeletons. The black dots and lines represent joints and bones, respectively. The colored lines represent the current orientation of the mapped joints, where red, green, and blue represent the local x, y, and z axes, respectively.

the deviation between the postures; otherwise, it would persist throughout the animation. Therefore, although not mandatory, the closer the mocap performer is to the T-Pose, the better the result.

Given a joint n in the source skeleton hierarchy, the bone is the vector pointing to its child joint $n+1$. If the joint has multiple children, the first child usually defines the bone, but it is an error-prone assumption since one can not guarantee that the same rule was applied during the creation of the skeleton. As an example, the upper spine joint may have three children: the left shoulder, the right shoulder, and the neck joint. The alignment of skeletons with different joints as the upper spine's first child will cause ill-conditioned poses. As a solution to this problem, the alignment is performed only on the bones between mapped joints.

The rotation matrix R_A to align the bone vector of the target skeleton onto the bone vector of the source skeleton is computed and applied to the global rotation R^n_G of the target skeleton joint n. This operation results in the matrix R^n_{NG}, the new rotation matrix that represents the aligned orientation of the joint (Equation 3.14).

$$R^n_{NG} = R_A R^n_G \tag{3.14}$$

Then, to recover the new local rotation of the joint R^n_{NL}, as expected by the BVH file format, the new global orientation is multiplied by its parent inverse rotation matrix (Equation 3.17).

$$R^n_{NG} = \left(\prod_{i=0}^{n-1} R^i_L \right) R^n_{NL} \tag{3.15}$$

$$R^n_{NG} = R^{n-1}_G R^n_{NL} \tag{3.16}$$

$$R^n_{NL} = (R^{n-1}_G)^{-1} R^n_{NG} \tag{3.17}$$

This process aligns the bones in the first frame. In the following frames, the rotation R_F of a joint in the source animation from the last frame to the current one replaces the rotation R_A to align bones in Equation 3.14. However, the BVH file does not store the rotation from one frame to the next one. Therefore, for a given joint of the source skeleton, Equation 3.18 computes the rotation matrix required by multiplying $R^m_G(t)$ and $R^m_G(t-1)$, the global rotation matrices to rotate a source skeleton joint in the frames t and $t-1$, respectively.

$$R_F = R^m_G(t)\ (R^m_G(t-1))^{-1} \tag{3.18}$$

Then, in a similar process as for the first frame, Equations 3.19 and 3.20 computes the global and local rotation matrix of the correspondent target joint, respectively.

$$R^n_G(t) = R_F\ R^n_G(t-1) \tag{3.19}$$

$$R^n_L(t) = (R^{n-1}_G(t))^{-1} R^n_G(t) \tag{3.20}$$

Changes in the local transformation matrices affect all the joints below the hierarchy. Therefore, it is imperative that the *Bones Alignment* starts at the root joint, followed by its children, its grandchildren, and so on down the hierarchy, avoiding changes on any of the local rotation matrices R^i_L in Equation 3.15.

3.1.4 Root Translation

So far, this process took care of the orientations, making sure that the joints of the target skeleton rotated as the source skeleton joints. Since our bones do not stretch, the human joints are not modeled as translational joints, and it is not necessary to deal with the translation. However, the root joint, often located in the hips or at the base of the spine, is the only joint with changeable translation to allow the character to move across the 3D space and to represent the character interaction with the ground.

Motion retargeting to characters with different heights results in artifacts such as the feet penetrating the floor or floating above it, and the feet sliding on the floor when walking, also known as “foot skating”. The assumption that both skeletons have the same pose in the first frame allows the use of the ratio of their heights as a normalization factor to handle ground contact (Equation 3.21). The ratio enforces the interaction with the

ground of the source character in the target character and adapts movements across the volume based on the characters' size, e.g., walking and jumping. Therefore, a taller target skeleton will perform a higher jump and a larger step than the shorter source skeleton does.

Since an action may occur in only one plane while not affecting the other, such as walking or jumping, the source motion of the root joint is decomposed into the vertical and horizontal movement that are analyzed separately. The horizontal movement is the projection of the root joint position into the ground.

Given the heights of the root joint of the target and source skeletons in the first frame, h_{tgt} and h_{src}, respectively, the ratio is computed using Equation 3.21.

$$ratio = \frac{h_{tgt}}{h_{src}} \tag{3.21}$$

The horizontal and vertical movements of the root, $\mathbf{g}(t)$ and $\mathbf{h}(t)$, are calculated in Equation 3.22. The translation of the root joint regarding the system origin is equal to these values recombined.

$$\mathbf{g}(t)_{src} = \mathbf{g}(t)_{tgt} ratio \quad and \quad \mathbf{h}(t)_{src} = \mathbf{h}(t)_{tgt} ratio \tag{3.22}$$

3.1.5 Retargeted Motions

Two 3D models were used as target characters to illustrate the steps of the implemented methodology. Figure 3.6 presents two virtual humans with different body proportions and shapes. Additionally, their skeletons have distinct topology from each other and the source skeleton (Figure 3.5).

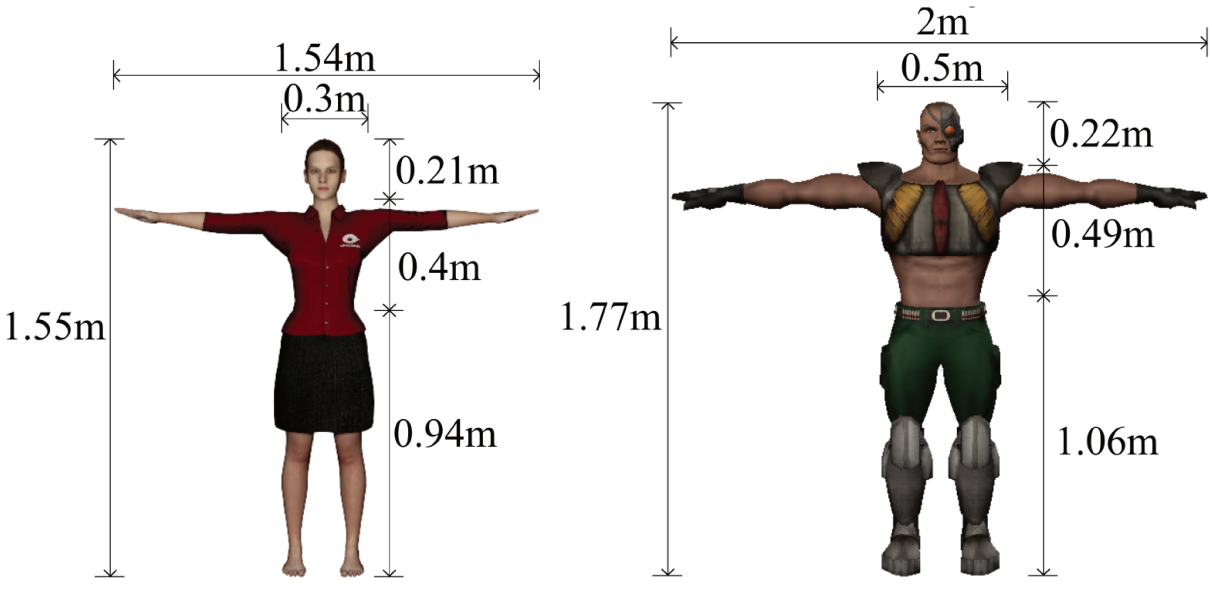


Figure 3.6: Two target characters: Talita, on the left, from project TAS; and Aragor, on the right, available in Autodesk® MotionBuilder®.

Now consider the mocap session in which the performer starts in the T-Pose, lowers his arms, then moves the right hand from the head to the waist, in front of the body, and ends in the T-Pose again. The distance of the right hand from the origin in the original motion and in the retargeted motions are displayed in Figure 3.7. Note that the *Initial Motion Retargeting* preserves the motion dynamics. The position of the hand changes, of course, with the skeleton height.

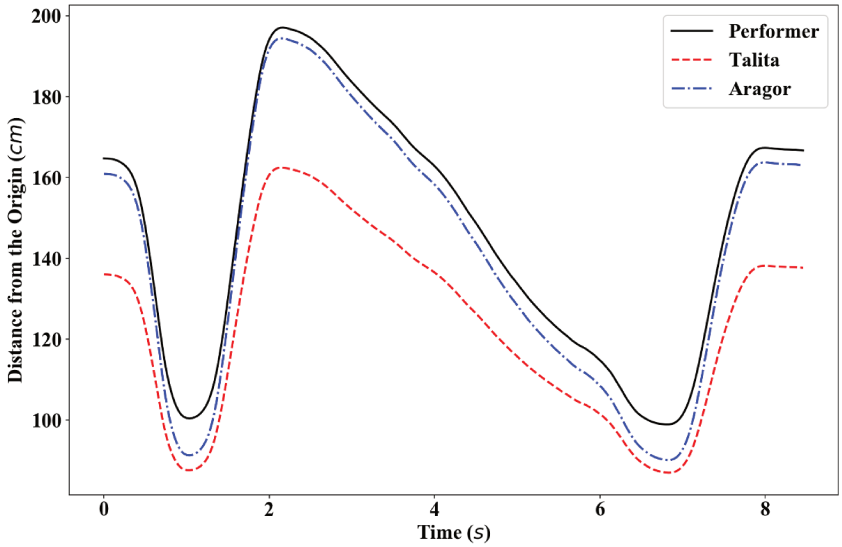


Figure 3.7: Right-hand joint distance from the origin during an action sequence, in centimeters, of the performer, Talita, and Aragor skeletons represented by the black, red, and blue lines, respectively. The motion was retargeted to the characters Talia and Aragor through the *Initial Motion Retargeting* process.

However, distortions are found in some of the animations. Figure 3.8 presents the pose of three animations and the artifacts found. The first pose (Figure 3.8a), crossed arms around the waist, resulted in self-penetration between the forearms. The second one is the movement of taking an oath (Figure 3.8b). It lacks contact with the right hand and the chest. The third and fourth snapshots, Figures 3.8c and 3.8d, represent the same animation of holding the hand in front of the mouth. The hand is too low and too far from the mouth, although the source animation had no contact.

3.1.6 Summary

Section 3.1.1 describes the use of local rotation and translation to animate the skeleton. The *Skeleton Map*, described in Section 3.1.2, details the correspondence between skeletons. The correspondence is mandatory to transfer motion between equivalent joints. Since skeletons can have distinct topology, neither the joint location in the hierarchy nor its name is reliable to infer the correspondence between skeletons.

Lastly, in Section 3.1.3, the *Bones Alignment* computes the motion of the target skeleton. In the first frame, the bones between mapped joints in the target skeleton are

Figure 3.8: Pose animations with motion retargeting artifacts: (a) self-penetration of the arms; (b) lack of contact of the right hand with the chest; (c) and (d), front and side view of the same pose, wrong hand placement.

aligned with the correspondent bones of the source skeleton. For the rest of the frames, the transformation of joints in the source skeleton from the previous frame to the current is applied to the target joints.

This process results on the target skeleton performing the source movement, but ill-conditioned poses and artifacts, such as self-penetration and "foot skating", may be present due to differences in the body proportions or body surface between the target character and the mocap performer.

3.2 Spatial Relationship Encoding

This section describes how the motion retargeting becomes aware of the characters' surface. The process includes a surface calibration step, described in Section 3.2.1. The calibration consists of sampling points on the body surface of the mocap performer and the 3D model. Then, the position of these points over the animation are estimated; two surface meshes for the torso and the head are created, a capsule is used to represent each limb, and it is computed the egocentric coordinates of the joints regarding each triangle composing the meshes and each capsule limb. The coordinates encode the position of a joint based on

its spatial relationship with the surface elements, such as distance and orthogonality. The coordinates are normalized with the body proportions of the source skeleton and adapted to the target skeleton. The position of the joint in the target skeleton is then recovered through the denormalization of the egocentric coordinates using the body proportions and surface meshes of the target skeleton.

As is Figure 3.9, in this work, the surface relationship between the source and target character is preserved for the joints right and left hands and feet.

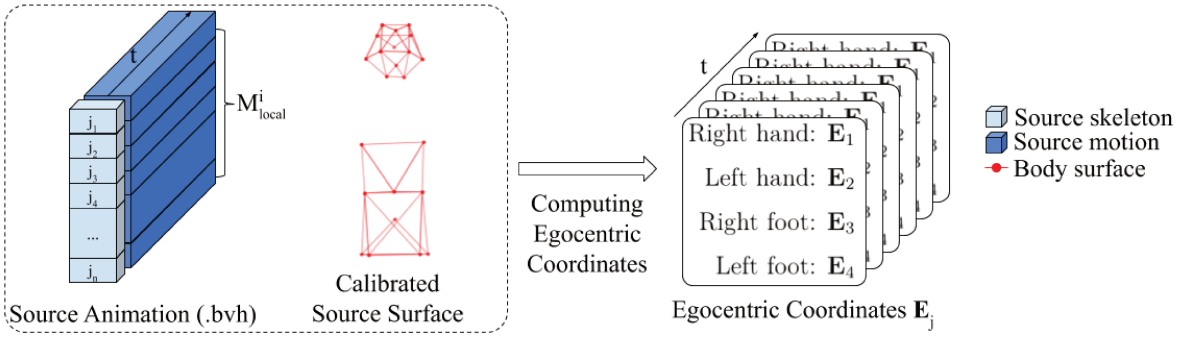


Figure 3.9: The relationship between the joints and the body surface of the mocap performer is encoded in the egocentric coordinates $\mathbf{E}_j$. The source animation provides the position of the joints as well as the body proportions, the bone lengths. The performer body surface is calibrated during mocap. An egocentric coordinate details the relationship between a single joint and the surface components in one frame.

At the beginning of a mocap session, the performer touches a set of points on the surface of his body with his hands. The surface is estimated through the position of his hands during the contact. Then, the algorithm creates two meshes with the points for the torso and head, and it estimates the thickness of the limbs using twice the distance of the hand joint to the correspondent bone limb. The same points and limb' thickness are collected on the 3D model character. Then, the motion retargeting algorithm enforces proportional distances between the hands and surface meshes and limbs as the source animation.

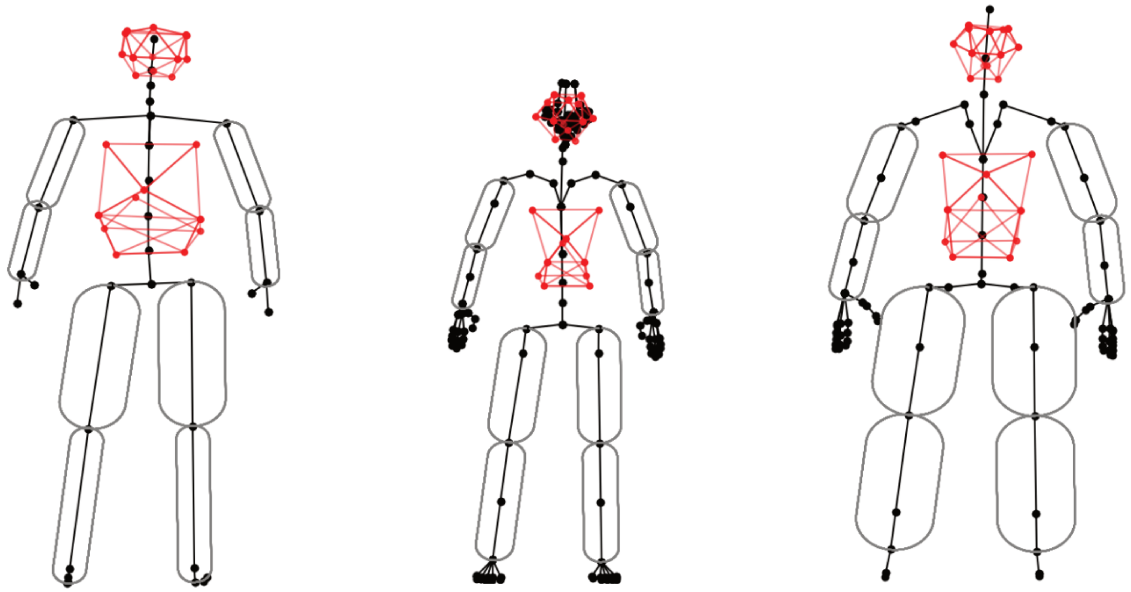
The surface could also be estimated based on the position of the markers, but it narrows the application of these methods to optical mocap systems that use reflective markers. Therefore, the present approach could be used with any mocap system available if the system's output is a skeletal animation, as described in Section 3.1.1, since the input is an animation file and do not depend on mocap system's specifications.

3.2.1 Surface Calibration

Molla *et al.* (2017) defined a set of points to characterize the body surface of the 3D model and the performer (MOLLA *et al.*, 2017), the yellow dots in Figure 3.11.

The points are connected through triangles to create a mesh that represents the body surface. The points on the body surface of the character and the performer need to be collected on the same relative positions since the triangle mesh differences between the characters must represent differences in the characters' surface. Thus sampling surface points from different places introduces noise to the analysis. The thickness of the limbs is also registered to model limbs as capsules.

Figure 3.10 presents the goal of the surface calibration. The figure presents the front view of the calibrated surfaces for the mocap performer (Figure 3.10a) and both target characters Talita and Aragor (Figures 3.10b and 3.10c, respectively), presented in the previous section. The surface components include limb capsules and surface meshes. The surface calibration process acquires the information required to detail the body shape of the performer and virtual characters.



(a) (b) (c)

Figure 3.10: Skeletons, meshes components, and limbs capsules composing the body surface of the (a) mocap performer and the target characters (b) Talita and (c) Aragor. The red dots indicate surface points calibrated. The gray capsules represent each modeled limb.

The 3D artist or animator obtains the position of the surface points and the thickness of each limb through manual inspection in a 3D modeling software to calibrate the surface of the virtual character. Since the unit of measurement of the third-party software and the animation file may be different, the position of the root joint is also registered for comparison with the target skeleton's T-Pose in the BVH file. It is critical that the character is in the T-Pose during this inspection. The position of the surface points and thickness of the limbs are saved in a Comma-separated values (CSV) file and can be reused whenever this model is the target character (Figure 3.12). Motion Retargeting to new characters requires performing this process to specify its body surface.

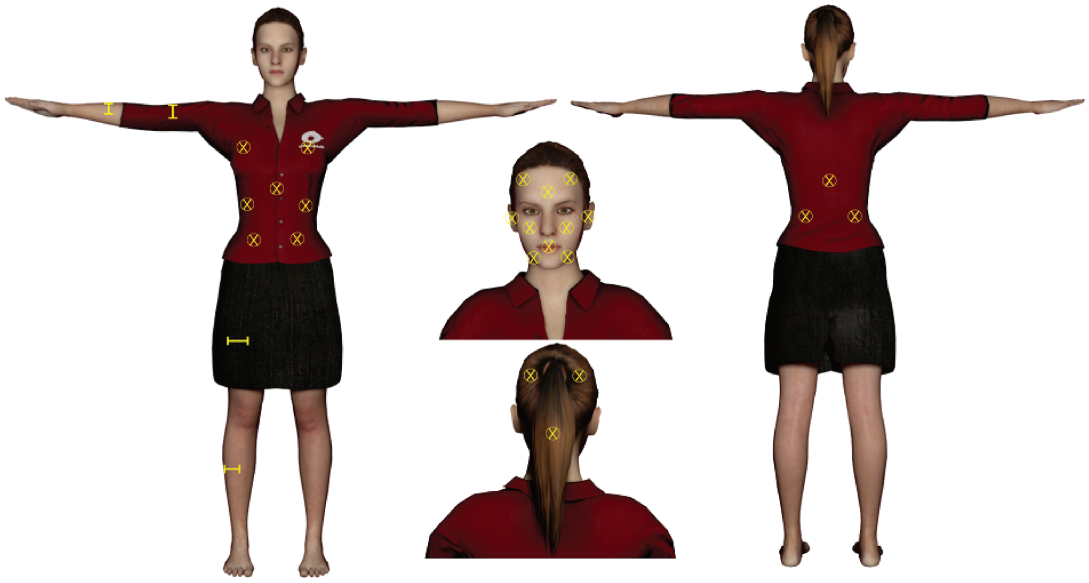


Figure 3.11: A 3D character and the sampled points and limb radius that the yellow markers and lines indicate. The points are the edges of the triangles composing two meshes, head and body mesh, that represents the surface of the virtual human. Capsules represent the limbs. Capsule radius and length are equal to half the thickness and the bone segment length.

The surface calibration of the mocap performer occurs at the beginning of the mocap procedure. The performer must touch the surface of the body with his hands in the same position as the set of points defined. The movements of the surface calibration are captured as mocap clips, saved as BVH files, and the position of the surface points are retrieved automatically (Appendix B). The surface calibration is required only once per mocap performer.

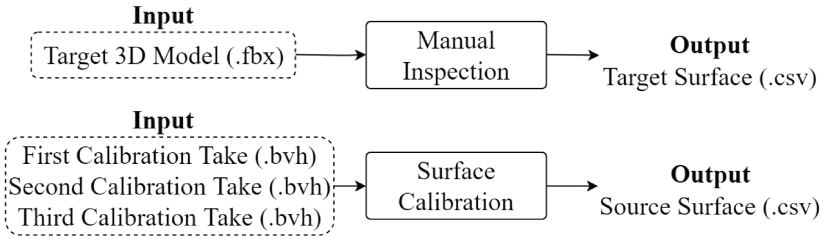


Figure 3.12: The process to obtain the target and surface CSV files, that specifies a static representation of the body surface for the performer and 3D character. The surface of the target character is acquired manually while the body surface of the mocap performer is extracted from three mocap takes.

The surface calibration of the performer is partitioned into three mocap takes, which aids the performer to properly touch the points without feeling tired or mistaking the position of the points. Table 3.1 details the sampled points at each take, in order, and the surface information retrieved from each point. The first take includes points of the front torso mesh and the limbs, the second take contains points of the head, and the last

take is composed of points in the back. The second take poses are shown as an example in Figure 3.13.

Figure 3.13: Head surface calibration poses (second BVH take). Surface points extracted from each pose, from left to right: forehead, ear, and chin on the first line; cheek, mouth, and forehead middle on the second line; and nape and nape middle at the bottom.

With prerecorded videos of a person executing the calibration movements, the mocap performer is instructed to mimic the actions of the videos. That ensures that the performer calibrates the surface points in the same order as the videos and allows an automatic acquisition of the points' position. If concluded correctly in the first try, this surface calibration process takes less than two minutes.

In the post-processing stage, an algorithm searches in the three calibration takes for the frames in which the performer is touching a surface point. Since the hands of the performer are steady while touching the surface point, and assuming that the performer calibrated the points in the correct order (as Table 3.1), the algorithm finds zero-speed hand moments and set the position of the hand joint as the position of the respective surface point. The body surface points are then attached to nearby joints, as detailed in the next section, and the local transformation of each point regarding the attached joint is saved in CSV file. This file depicts the performer's calibrated body surface, and it is reused when retargeting motions from that performer.

The arms, forearms, thighs, and shins are modeled as capsules with radius R. Hence, $2R$ represents the thickness of the limb. The distance of the hand joint to the projected point in the limb bone segment is used to estimate the capsule radius. Given the capsule in Figure 3.14 centered at the origin, a point $\mathbf{p} = (x, y, z)$ in its surface satisfies Equation 3.23. The radius R of the capsule is half the limb thickness, and its length $2L + 2R$ is the distance between the limbs' joints.

Take	Surface Point	Surface Information
First	Chest*	Mesh
	Abdomen*	Mesh
	Hip*	Mesh
	Thigh*	Limb
	Shin*	Limb
	Abdomen up	Mesh
	Right arm	Limb
	Right forearm	Limb
	Left arm	Limb
	Left forearm	Limb
Second	Forehead*	Mesh
	Ear*	Mesh
	Chin*	Mesh
	Cheek*	Mesh
	Mouth	Mesh
	Forehead middle	Mesh
	Nape*	Mesh
	Nape middle	Mesh
Third	Loin*	Mesh
	Loin up	Mesh

Table 3.1: The three takes to calibrate the body surface of the mocap performer, the surface points sampled at each take and the type of information extracted from the point: *Mesh* indicates that the point composes a surface mesh, body or head, and *Limb* points define the thickness of the limbs. (*) refers to surface points that the performer calibrates with both hands simultaneously to sample the right and left side of the surface.

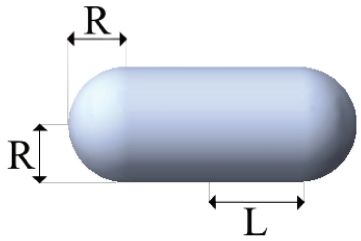


Figure 3.14: Capsule with length $2L$ and radius R.

$$x^2 + y^2 + \frac{1}{4}(|z - L| + |z + L| - 2L)^2 - R^2 = 0 \tag{3.23}$$

Surface Motion Estimation

Collecting the static position of points at the surface of the mocap performer is not enough. Each position is recovered in a particular frame and pose, but the performer may walk around, jump, rotate the head, or the torso. Since the method from Section 3.2.1 does not keep track of the surface points throughout the animation, their positions will not represent the surface movement and deformation. To estimate the deformation

and translation of the surface, each calibration mesh point is attached to a joint, as in Figure 3.15. Thus, each point moves as the attached join.

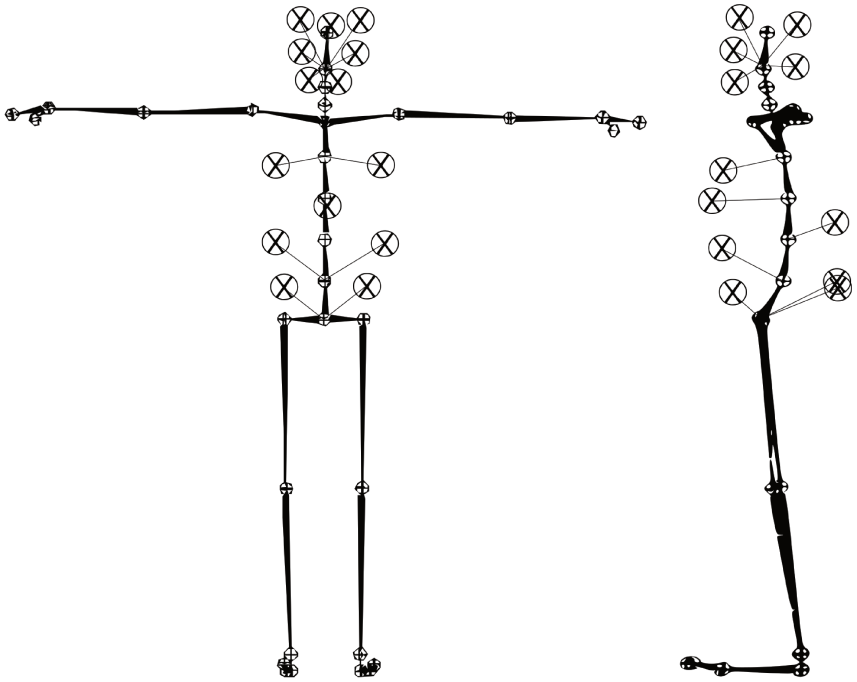


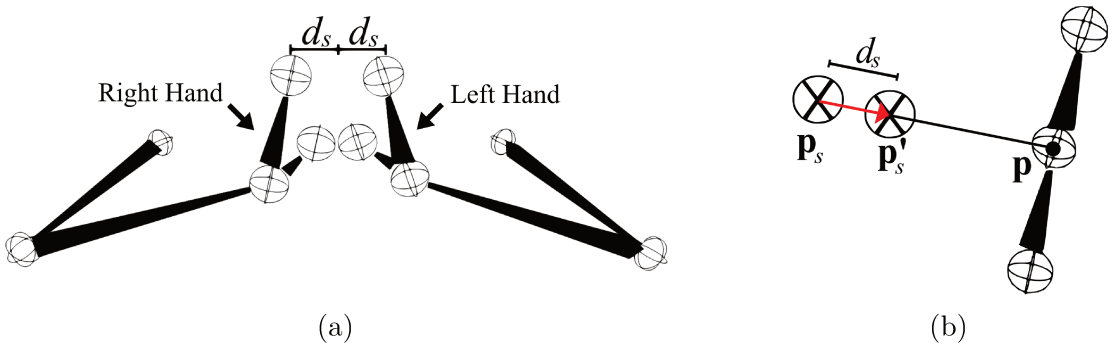
Figure 3.15: The front and side view of a skeleton on the left and right, respectively. The circumferences with crosses inside represent some of the surface points, and the lines show their attached joint.

Attaching surface points to a nearby joint also allows the computation of a more accurate position of the surface points. That is, using the hand joint position as the position of a body surface point will also include the hand surface (the distance of the joint to the actual skin). To avoid this issue, the performer is instructed to place both hands together, as in Figure 3.16a. The distance from the joint to the skin d_s is estimated as half the distance of both joints[4]. Then, as in Figure 3.16b, given the line segment from a surface point $\mathbf{p}_s$ and the attached joint $\mathbf{p}$, the surface point position is updated attracted towards $\mathbf{p}$ by a d_s amount (Equation 3.24).

$$\mathbf{p}'_s = (1-k)\mathbf{p}_s - k\mathbf{p}, \text{ with } k = \frac{d_s - \|\mathbf{p}_s - \mathbf{p}\|}{\|\mathbf{p}_s - \mathbf{p}\|} \tag{3.24}$$

To estimate the surface motion and deformation, the rotations of the attached joints are exploited. During the calibration of a surface point, given the translation matrix T_G^p, representing its position $\mathbf{p}'_s$, the algorithm computes the local transform matrix M_L^p of the point regarding the global transform matrix M_G^n of the attached joint n (Equation

[4]Note that the distance d_s is computed from the joints used to sample the surface points. As depicted in Figure 3.16a, the right and left middle hand joints were used in this work.



(a) (b)

Figure 3.16: (a) Pose with the hands pressed together to compute the position of the hand surface. The distance between the hand joints, the middle hand joints, in this case, is given by $2\mathbf{d}_s$. (b) The recalculation of the surface point position $\mathbf{p}_s$ given the position $\mathbf{p}$ of the attached joint.

3.25).

$$M_L^p = (M_G^n)^{-1} T_G^p \tag{3.25}$$

That is comparable to adding a new child joint on joint n. Therefore, Equation 3.8 gives the position of the surface points during the animation. All rotations and translations of joints above the hierarchy are concatenated and applied on the surface point.

The capsule represents the limbs: the arms, forearms, and upper and lower legs. Its center is fixed in the center of the bone segment of the respective limb. Hence the motion estimation along the animation is not necessary.

Surface Mesh

As the final stage of the calibration, the meshes are assembled to obtain an actual surface from the surface points. The surface points of the head and body create the head and body meshes, respectively. The meshes are groups of triangles with edges as surface points. The triangles to compose the meshes are chosen in a way to maximize the body surface area covered by the mesh. Also, triangles should not penetrate the actual body surface.

The meshes for the target and source characters are equivalent, i.e., the surface points used to compose each triangle in both meshes are the same.

3.2.2 Computing Egocentric Coordinates

The egocentric coordinates encode the spatial relationship between the position of the joints and the surface components, limb capsules and meshes, in the source character. Later, the parameters of the target character are used to decode the coordinates. This results in the position of a target joint that has the same spatial relationship between its

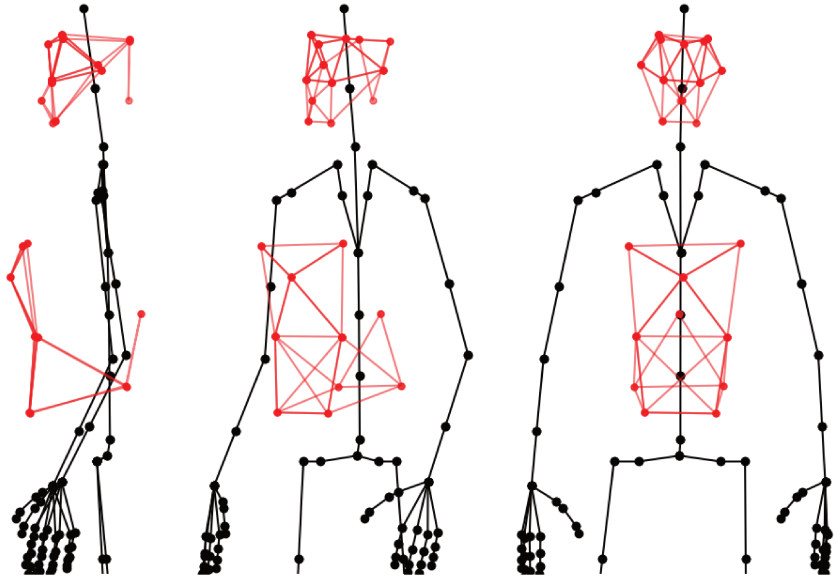


Figure 3.17: Head and body surface meshes at different angles. The edges of the triangles, the red dots, represent the surface points from the surface calibration. Black dots and lines represent the joints and bones, respectively.

character surface as the source joint. The coordinates are computed for the extremity joints: the hands and the feet.

Reference Point and Displacement Vector

The position of a extremity joint j is decomposed into the sum of the reference point $\mathbf{x}$ and the displacement vector $\mathbf{v}$ from the *ith* surface component (Equation 3.26). In the mesh components, the reference point is located in the triangle centroid. For the limb capsules, the reference point is the intersection between the capsule surface and the line that connects the joint and the center of the capsule. As in Figure 3.18, displacement vector $\mathbf{v}$ is the vector from the reference point to the joint.

$$\mathbf{p}_j = \mathbf{x}_i + \mathbf{v}_i \tag{3.26}$$

The reference points of limb capsules are stored as normalized cylindrical coordinates if it lies on the cylinder surface, and as spherical coordinates if it is on the surface of the half spheres. Given a reference point on the cylinder surface $\mathbf{x} = (R, \theta, z)$, where R is the capsule radius (half the limb thickness), and $\mathbf{x} = (R, \theta, \psi)$ on the half sphere surface, their normalized form are expressed as Equation 3.27, respectively, with L as half the capsule length.

$$\hat{\mathbf{x}} = (1, \theta, \frac{z}{L}) \text{ and } \hat{\mathbf{x}} = (1, \theta, \psi) \tag{3.27}$$

Foot contact with the ground is handled by considering the projection of the foot

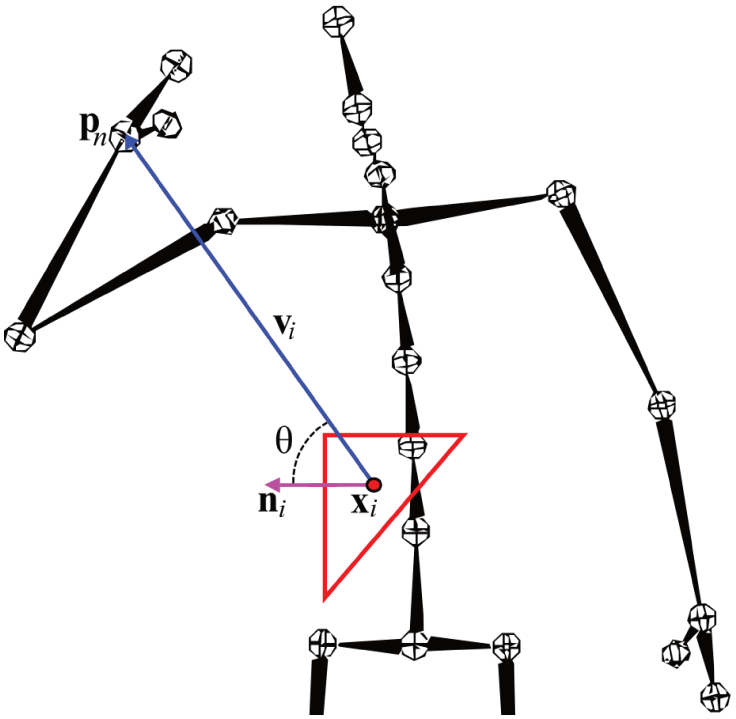


Figure 3.18: The decomposition of the position a joint n, the right hand joint in the figure, relative to the *ith* surface component. The position of the joint $\mathbf{p}_n$ is the sum of the reference point $\mathbf{x}_i$, the red dot, and the displacement vector $\mathbf{v}_i$, the blue arrow. The pink arrow represents the surface normal of the mesh component, and θ is the angle between the normal and the displacement vector.

position into the ground, $\mathbf{g}$, computed in Equation 3.22, as a reference point.

The reference point and displacement vector from all the surface components, i.e., limb capsules and mesh components, are combined in a weighted sum (Equation 3.28. The importance factor λ expresses the influence of each component on this combination.

$$\mathbf{p}_j = \sum_{i=1}^{m} \hat{\lambda}_i(\mathbf{x}_i + \mathbf{v}_i) \tag{3.28}$$

Importance Factor

The importance factor λ of a surface component is a metric that encodes the proximity and orthogonality between the component and the position of the joint. The goal of the importance factor is to allow that the surface components nearer and perpendicular to the joint have a higher contribution to indicate the joint's position.

Furthermore, small distances between joints and surface components may indicate that an interaction is occurring, for example, covering the eyes with the hand. In this case, the position of the hand should be more dependent on the mesh components on the head than on the torso. Additionally, while covering an ear with the hand, the importance of components on the side of the head should be higher than those on the face or the back of the head.

The proximity metric is obtained by taking the inverse of the distance between the joint and the surface component, represented by the displacement vector magnitude. Therefore, closer components will have stronger influences. As proposed in the previous section, the hand surface is deducted from the distance for accuracy (Equation 3.29).

$$\lambda_p = \frac{1}{\|\mathbf{v}\| - d_s} \tag{3.29}$$

The orthogonality is expressed as a function of θ, the angle between the displacement vector $\mathbf{v}$ and the surface component normal $\mathbf{n}$. The function decreases as the angle increases. However, different than Molla *et al.* (2017), it is proposed that surface components still contributes when the joint is behind them. Since some body parts have no calibration points, such as the sides of the trunk, the other surface components can still provide useful information.

$$\lambda_\perp = \frac{\cos\theta + 1}{2} \tag{3.30}$$

Both metrics are combined to compose the importance factor, as shown in Equation 3.31. Then, the importance factor λ_i of each surface component is normalized so that their sum is equal to one (Equation 3.32).

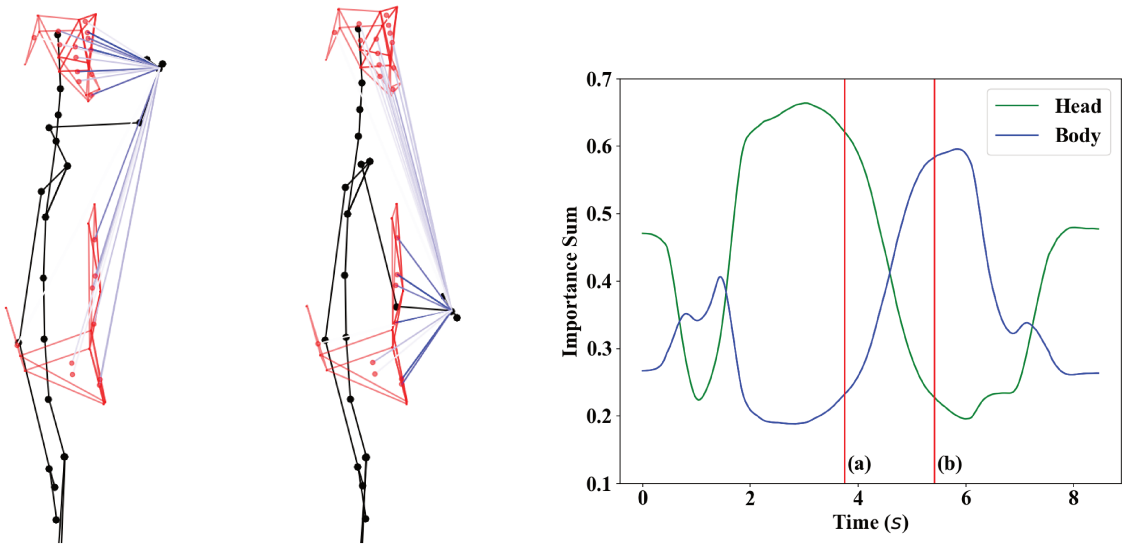
$$\lambda = \lambda_p \lambda_\perp \tag{3.31}$$

$$\hat{\lambda}_i = \frac{\lambda_i}{\sum_{i=1}^{m} \lambda_i} \tag{3.32}$$

Therefore, the importance factor reduces the weight of surface components not orthogonal to the displacement vector or far from the joint. However, Equation 3.32 still allows components to contribute when all are under both circumstances.

Consider, again, the motion described in the previous section, in which the performer moves the right hand from the head to the waist. The egocentric coordinates encode the spatial relationship of the right hand to the body surface. Figure 3.19 depicts the reference point, as red dots, and the displacement vectors, as blue lines, when the right hand of the performer is in front of his face and body. Note that the intensity of the blue color decreases proportionally to the respective importance factor. Both proximity and orthogonality metrics are balancing the weights of each component given the joint position. Therefore, when the hand is on the face (Figure 3.19a), surface components on the head have higher importance (darker blue) since the hand is near and perpendicular to the components' normal. The analogous is valid for Figure 3.19b. Conversely, the displacement vectors of surface components on the sides and the back of the performer are barely visible. Figure 3.19c shows the importance sum of surface components from the body and head in different colors. The vertical lines indicate the moments that both

snapshots were taken, notice that their sums invert from one moment to the other.



(a) (b) (c)

Figure 3.19: The skeleton pose during an animation that the right hand passes in front of the face (a) and the waist (b), and the sum of the importance factor during the animation (c). In (a) and (b), the displacement vectors are represented by blue lines, components with higher importance have darker intensities of blue. Also, the reference points are shown as red dots. In (c), the sum of the importance factor for the head and body components are presented in green and blue, respectively. The vertical red lines correspond to the moment of the poses in (a) and (b).

The position of a target skeleton extremity joint is obtained through Equation 3.28, applying the reference points from the target surface and the displacement vectors from the source skeleton. The distance from the joint to the surface is the same as the source character, but its global position changes with the target surface. Differences in the body shape, such as limb thickness, head, and chest size, are appraised due to the use of reference points from the target character. Still, differences in the body proportions and bone length among the skeletons were not taken into account, which results in ill-conditioned poses or in a position impossible to reach.

Kinematic Path Normalization

The kinematic path normalization adjusts the joints' position according to the length of the bones in the skeleton. The kinematic path is the route through joints from a reference point to another joint in the hierarchy. Figure 3.20 represents two kinematic paths for the right hand joint: on the left, the path from the reference point of a body mesh component; and on the right, the path from the reference point of a limb capsule.

The goal of the kinematic path normalization is to adjust the position of the joint regarding the length of the bone segments. Thus, due to different topologies in the source and target skeletons, the joints constituting the kinematic path are some of those

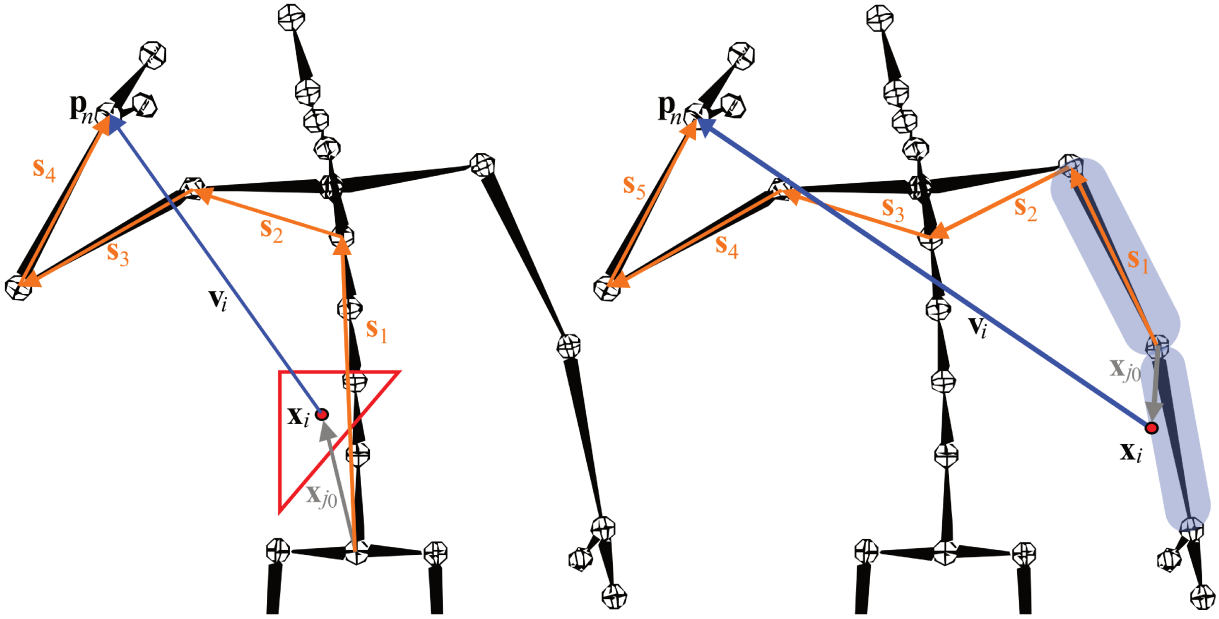


Figure 3.20: The kinematic path, in orange, from the hand joint to two surface components. On the left, a triangle of the body mesh, where the reference point $\mathbf{x}_i$ is the centroid. On the right, a limb capsule of the left forearm, with $\mathbf{x}_i$ as the intersection of the line from the right hand to the center of the bone with the capsule surface. The vector $-\mathbf{x}_{j0}$ starts on the root joint of the kinematic path and ends on the reference point.

related in the *Skeleton Map* (Section 3.1.2). A single segment for the spine is proposed due to the different numbers of joints composing the spine found in skeletons from different third-party sources. Discrepancies on the positioning of the shoulder and clavicle joints, when depicted, are also avoided by considering a single path from the top of the spine to the arm joint[5]. The bone segments that could be present in a kinematic path are listed in Table 3.2.

Bone Segment	**Start Joint**	**End Joint**
Spine	Hips	Upper Spine
Neck	Head	Upper Spine
Clavicle	Upper Spine	Arm
Arm	Arm	Forearm
Forearm	Forearm	Wrist
Hip	Hips	Upper Leg
Femur	Upper Leg	Lower Leg (Knee)
Tibia	Lower Leg (Knee)	Foot

Table 3.2: Bone segments and joints to compose a kinematic path. Except for the spine and neck, the bones refer to both the left and right sides. The start and end joints are present in *Skeleton Map*, described in Section 3.1.2.

The kinematic path of surface components on the body starts in the hips joint, in

[5]Note that a joint is named after the bone it controls, and not after its exact location on the body. Take the left skeleton on Figure 3.3 as an example, the joint "Arm" seems to be positioned on the shoulder, and the "Shoulder" joint is located near the chest or clavicle — that it is not depicted.

the head joint for head components, and, for limb capsules, it starts in the parent joint of the bone segment. The path ends on the respective extremity joint, the hands or the feet joints. Given the cosine of the angle α_k between the bone segment k and the displacement vector $\mathbf{v}$ (Equation 3.33), the displacement vector can be expressed in terms of the n bone segments of the kinematic path as Equation 3.34 and its normalization is presented in Equation 3.35.

$$\cos(\alpha_k) = \frac{\mathbf{v}}{\|\mathbf{v}\|} \cdot \frac{\mathbf{s}_k}{\|\mathbf{s}_k\|} \tag{3.33}$$

$$\mathbf{v} = -\mathbf{x}_{j0} + \sum_{k=1}^{n} \mathbf{s}_k \tag{3.34}$$

$$\hat{\mathbf{v}} = \frac{\mathbf{v}}{\tau}\ ,\ \tau = \sum_{k=1}^{n} \|\mathbf{s}_k\| \ |\cos(\alpha_k)| \tag{3.35}$$

Equation 3.35 normalizes the displacement vector with the contribution of each bone segment to the vector. This supports the adaptation of the position of the joint in skeletons with different sizes. As an example, the source pose of an adult with arms stretched results in a position impossible to reach to a child as the target character. Inversely, the adult as the target character will not have his arms stretched, since the relative distance of the hands to the body in the child is smaller than in the adult.

The set of cosines in the kinematic chain of a surface component is stored as the normalization coefficient C in Equation 3.36.

$$C = \{|\cos(\alpha_1)|, |\cos(\alpha_2)|, \ldots, |\cos(\alpha_n)|\} \tag{3.36}$$

Joints Orientation

The last information composing the egocentric coordinates is the orientation of the extremity joints regarding the surface components. The orientation of the hands and feet can carry semantic information; therefore, the motion retargeting should reorient their positions reflecting the differences in the shape of the body surface.

Due to different topologies, the local coordinates of the hands are not reliable to represent the surface orientation. If present, the middle hand or the middle finger joint is used to define the vector $\mathbf{s}_O$, the vector that points from the hand joint to the middle hand or middle finger joint. Considering that the first frame represents the T-Pose (palms facing the ground), this vector is rotated by ninety degrees on the global z-axis and used to represent the surface orientation, that is, the vector pointing out of the palms. The same is done for the feet. Then, for each surface component, the angle β_i is obtained to

align the hands and feet with the surface normal (Equation 3.37).

$$\beta_i = \arccos\left(\frac{\mathbf{s}_O}{\|\mathbf{s}_O\|} \cdot \frac{\mathbf{n}_i}{\|\mathbf{n}_i\|}\right) \tag{3.37}$$

3.2.3 Summary

So far, the egocentric coordinates depicted in Table 3.3 were computed in the source animation for the joints: right and left hand, and right and left foot. Each coordinate preserves the information listed:

- **Normalized reference point** ($\hat{\mathbf{x}}$): the intersection with the capsule surface of the line starting at the joint position and ending at the center of the capsule. It is stored for capsule components only since the reference points of mesh components are given by the centroids of the mesh triangles.
- **Normalized displacement vector** ($\hat{\mathbf{v}}$): the displacement of the joint to the surface component normalized by τ, the normalization factor from Equation 3.35. It gives the direction of the joint from the component. When multiplied by the normalization factor of the target skeleton, it results in the displacement of the joint to the surface component, given its kinematic path. The displacement shrinks or expands if the bone segments of the target skeleton are smaller or larger than those of the source skeleton, respectively.
- **Normalized importance** ($\hat{\lambda}$): the surface component weight to compose the position of the joint. Surface components near and more perpendicular to the joint — to the displacement vector — have a stronger influence on the computation of the joint position.
- **Normalization coefficients** (C): the set of values that describes the contribution of bone segments to compute the normalization factor τ. The contribution is given by the absolute value of the cosine of the angle between the bone segment and the displacement vector.
- **Relative joints orientation** (β): the angle between the surface normal of the hands or feet and the surface component normal. The orientation of the hands and foot near a surface component should follow the changes in the shape of that component from the source to the target character.

The egocentric coordinates of the joint j regarding the m surface components, mesh triangles and capsules, are stored as shown in Equation 3.38.

$$\mathbf{E}_j = \{\mathbf{e}_{j,1}, \mathbf{e}_{j,2}, \ldots, \mathbf{e}_{j,m}\}, \text{ where } \mathbf{e}_{j,i} = \left(\hat{\mathbf{x}}_i, \hat{\mathbf{v}}_i, \hat{\lambda}_i, C_i, \beta_i\right) \tag{3.38}$$

Coordinate	Name	Equation
$\hat{\mathbf{x}}$	Normalized reference point	3.27
$\hat{\mathbf{v}}$	Normalized displacement vector	3.35
$\hat{\lambda}$	Normalized importance	3.32
C	Normalization coefficients	3.36
β	Relative joints orientation	3.37

Table 3.3: The egocentric coordinates that describe the spatial relationship between the joints of the hands and feet and the surface components. The coordinates are extracted from the pose of the source animation and used to adapt the pose of the target character, given its body proportions and body surface. The displacement vector and importance are stored in the normalized form.

In this work, the spatial relationship between the joint and the body surface components is preserved for the joints: (1) Right hand; (2) Left hand; (3) Right foot; and (4) Left foot. Therefore, the set of egocentric coordinates computed depicted in Equation 3.39.

$$\begin{aligned} \text{Right hand: } \mathbf{E}_1 &= \{\mathbf{e}_{1,1}, \mathbf{e}_{1,2}, \ldots, \mathbf{e}_{1,m}\} \\ \text{Left hand: } \mathbf{E}_2 &= \{\mathbf{e}_{2,1}, \mathbf{e}_{2,2}, \ldots, \mathbf{e}_{2,m}\} \\ \text{Right foot: } \mathbf{E}_3 &= \{\mathbf{e}_{3,1}, \mathbf{e}_{3,2}, \ldots, \mathbf{e}_{3,m}\} \\ \text{Left foot: } \mathbf{E}_4 &= \{\mathbf{e}_{4,1}, \mathbf{e}_{4,2}, \ldots, \mathbf{e}_{4,m}\} \end{aligned} \tag{3.39}$$

3.3 Pose Adaptation

The *Pose Adaptation* receives encoded source animation, given by the egocentric coordinates computed in Section 3.2. A new pose is computed using the body proportions of the target character to revert the coordinates. This pose preserves the same spatial relationship with the surface as the original motion. Then, inverse kinematics is used to attract the target character into the pose.

The spatial relationship information between the hands and feet joint and the body surface from the source character, the mocap performer, is encoded by the egocentric coordinates. Therefore, the operations in this section are performed on the skeleton, surface, and joints of the target character, the 3D model.

3.3.1 Computing Target Positions

The target position $\mathbf{t}_j$ of joint j is computed by solving Equation 3.28 for the m surface components. The egocentric coordinates provide the importance factor, the reference point is obtained from the surface components, and the displacement vector, in its normalized form, is also present in the egocentric coordinates. The steps to recover the

target positions are listed:

1. Measure the lengths $\|\mathbf{s}\|$ of the bone segments. The bones in the kinematic path are defined on Table 3.2 along with their mapped joints.

2. Combine them with the normalization coefficients C to obtain the normalization factor τ (Equation 3.35).

3. Calculate the displacement vectors multiplying τ by the normalized displacement vectors $\hat{\mathbf{v}}$ as
$$\mathbf{v} = \tau\hat{\mathbf{v}} \tag{3.40}$$

4. Compute the reference points $\mathbf{x}$ for mesh triangles, their centroid, and denormalize $\hat{\mathbf{x}}$ for limb capsules. Given the normalized reference point $\hat{\mathbf{x}} = (1, \theta, z)$ on the cylinder surface and $\hat{\mathbf{x}} = (1, \theta, \psi)$ on the half spheres surface, the reference point is given by
$$\mathbf{x} = (R, \theta, zL) \text{ and } \mathbf{x} = (R, \theta, \psi), \tag{3.41}$$
respectively, where L is half the capsule cylinder length and R the capsule radius (half the limb thickness).

5. Determine the joint position $\mathbf{t}_j$ adding the reference points and the displacement vectors:
$$\mathbf{t}_j = \sum_{i=1}^{m} \hat{\lambda}_i(\mathbf{x}_i + \mathbf{v}_i) \tag{3.42}$$

These steps result in the target position of the joint j. Thus, performing it in the egocentric coordinates $\mathbf{E}_1$, $\mathbf{E}_2$, $\mathbf{E}_3$, $\mathbf{E}_4$, results in the target position for the right and left hand and foot joints.

3.3.2 Pose Adjustment with Inverse Kinematics

Now that the target positions for the desired joints are computed, the next step is to adjust the skeleton pose. Inverse kinematics is used to calculate the orientations of joints in the skeleton that results in the desired pose.

The inverse kinematics is applied independently in the four limbs: right and left arms and legs. The kinematic path[6] comprehend the joints from the shoulder to the hand (for the arms), and the joints from the femur (Upper Leg joint) to the foot (for the legs).

[6]The path of joints in a skeleton hierarchy is usually called "kinematic chain" by references in the research field of kinematics. For consistency with the previous sections, this work adopts the term "kinematic path".

Since no overlap of joints occurs in the kinematic paths, the limbs do not compete with each other. That is, the joints in a IK are only trying to reach one target position. When that is not the case, undesired artifacts in the skeleton pose may appear, especially when the target positions are out of reach. As an example, if the kinematic path of the arms comprehended the spine, the IK could keep rotating the spine in order to each arm reach its target position.

The inverse kinematics with Jacobian transpose proposed by Buss (2004) is used to bring the joints that had the egocentric coordinates computed closer to their respective target position (BUSS, 2004). First, the displacement between them is computed with $\mathbf{d} = \mathbf{t} - \mathbf{p}$. Then, the inverse kinematics problem is solved to find values to update the joint rotation angles by a small amount. These steps are repeated until $\mathbf{d}$ is sufficiently small. In order to embrace skeletons with diverse topologies, no rotation constraints are assumed for the joints; therefore, joints in the inverse kinematics algorithm have 3DOFs.

3.3.3 Adjusting Extremity Joints Orientation

The final step of the *Pose Adaptation* is to adjust the joints orientation regarding the body surface components. A surface component or the overall body shape may change from the source to the target character. The orientation of the joint that may be interacting with or touching the surface components, the extremity joints, need to follow those changes.

As an example, imagine a mocap animation of a slim man sliding his hand from the neck to the belly. The motion retargeting of this animation to a pregnant character should account for the bigger breasts and belly. The methods described in Sections 3.3.1 and 3.3.2 deal with the position of the hands; therefore, the resulting animation will avoid self penetration by better positioning the hand. However, the impression of sliding the hands across the surface is achieved by the palm following the variations of the body surface.

This adjustment is performed only if the skeleton has a joint depicting the middle of the hand, such as the middle hand joint or middle finger joint. The orientation vector of the palm $\mathbf{s}_O$ is computed in the first frame by rotating the vector from the hand joint to the middle hand joint in ninety degrees on the global z-axis. Then, for the rest of the frames, the same local transforms M_L of the hand is applied on the surface orientation vector $\mathbf{s}_O$.

For every surface component i, the target orientation vector $\mathbf{o}_i$ is computed rotating the component surface normal $\mathbf{n}_i$ by its relative joint orientation β_i in the axis pointing in the direction $\mathbf{s}_O \times \mathbf{n}_i$ (Equation 3.43).

$$\mathbf{o}_i = R(\beta_i)\mathbf{n}_i \tag{3.43}$$

Combining the vectors from the components and weighting them with the normalized importance, also stored in the egocentric coordinates, results in the target orientation vector $\mathbf{o}$ from Equation 3.44.

$$\mathbf{o} = \sum_{i=1}^{m} \hat{\lambda}_i \mathbf{o}_i \tag{3.44}$$

Finally, the rotation matrix R_A to align the (current) orientation vector $\mathbf{s}_O$ on the target orientation vector is computed and applied to the joint. The updated local rotation of the joint, as the BVH requests, is recovered in the process described in Section 3.1.1.

3.3.4 Example

Figure 3.21 presents the pose of the Aragor character before and after the motion retargeting process. The displacement vectors' length and the placement of the reference points were adjusted regarding its body proportions and surface shape.

The red cross in Figure 3.21 represents the target position of the right hand, computed through denormalizing the egocentric coordinates, and the green dot is the current joint position. The trajectory of the right hand was adjusted by the inverse kinematics algorithm, as shown in Figure 3.22.

Notice that the moment when an action happens, that is, the hand starts to accelerate or decelerate, remains unaltered. However, since the position of the hand is adjusted, the speed of the movement may change.

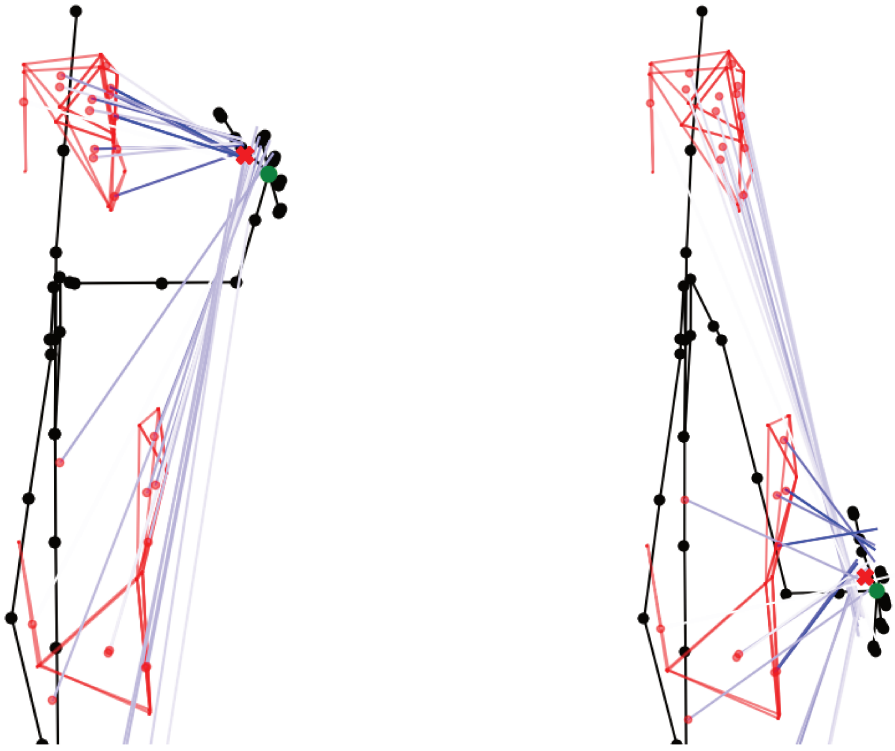
Applying the *Spatial Relationship Encoding* and *Pose Adaptation* processes on the motions shown in Figure 3.8, the distortions presented are diminished. Therefore, the preservation of the surface spatial relationship contributed to better reproduction of the original motion. The resulting animations are displayed in Figure 3.23.

3.3.5 Summary

In this Section, it was detailed pose adjustment of the retargeted animation from Section 3.1. The calibrated surface of the target character and the egocentric coordinates, both described in Section 3.2, are used in the *Pose Adaptation* process.

First, given the egocentric coordinate $\mathbf{E}$ of a joint, the parameters reference point $\hat{\mathbf{x}}$, displacement vector $\hat{\mathbf{v}}$, and importance $\hat{\lambda}$ are denormalized using the body proportions and calibrated body surface of the virtual character to compute the joint target position $\mathbf{t}$.

Then, the inverse kinematics gives the local rotation angles of joints in the skeleton for it to reach the target positions. The angles are updated in small steps to control the skeleton pose better. Also, each limb is adjusted separately and independently.



(a) (b)

Figure 3.21: Aragor's skeleton pose during an animation that the right hand passes in front of the face (a) and the waist (b). The red cross and green dot represent the target and current position of the right hand, respectively.

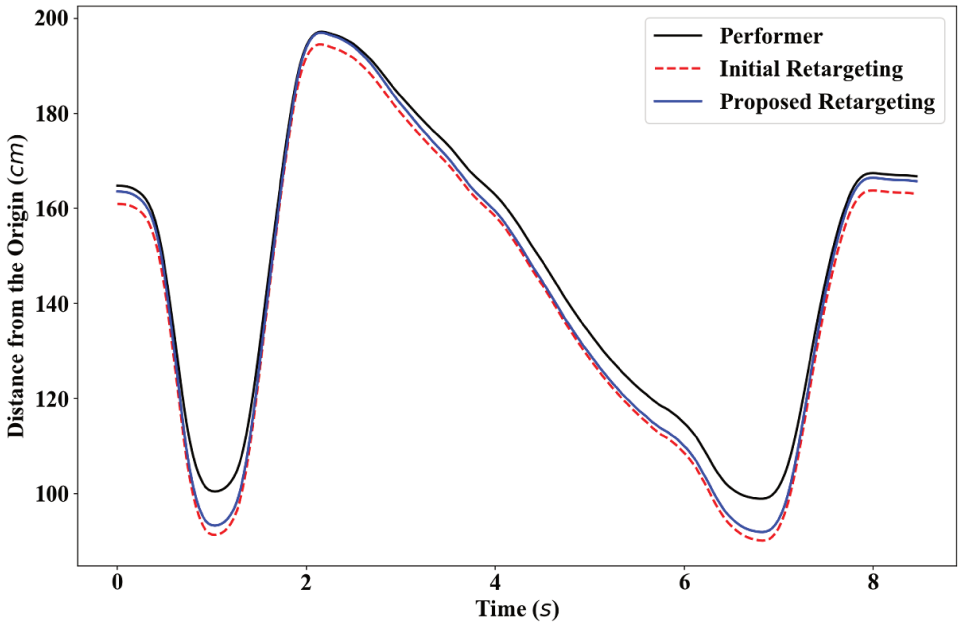


Figure 3.22: Right-hand joint distance from the origin during an action sequence of the performer (black) and the virtual character Aragor after the *Initial Motion Retargeting* (red) and after the motion retargeting process proposed (blue).

Figure 3.23: Pose animations without (left) and with (right) the motion retargeting preserving spatial relationship: (a) and (b) the self-penetration of the arms was diminished; (c) and (d) the right hand was attracted to the chest; (e) and (f) the mouth was covered by the hand; (g) and (h) the hand contact with the mouth was preserved.

Therefore, a competition between them is avoided, and the position and orientation of the spine and hips remain unaltered.

Finally, the orientation of the hands and the feet **o** are adjusted based on the relative joints orientation parameter β to reflect any changes on the target character surface in comparison to the source one.

3.4 Materials

This section details the experimental setup, third-party software, and implementation involved in the process of the present work. The mocap system is equipped with nine Vicon Vantage™ infrared cameras, and a conventional digital video camera, as shown in Figure 3.24. The accompanying Vicon Shōgun™ software was used to track the performer's movements and clean the mocap data, such as occluded and swapped markers. Passive markers were fixed on the performer's body surface as the Vicon Shōgun™ standard markers placement.

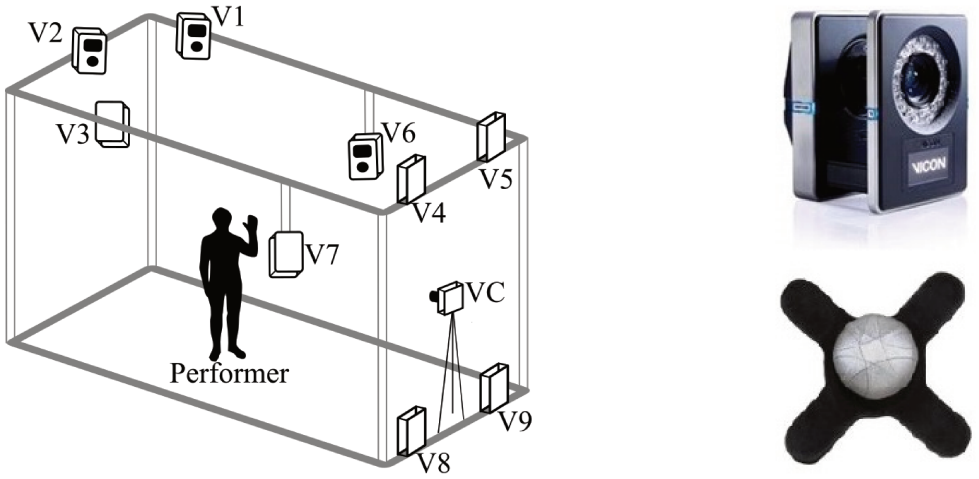


Figure 3.24: On the left, the optical mocap system setup consisting of nine Vicon Vantage™ cameras (V1 to V9 and on the top right), and a conventional HD video camera (VC). Passive markers (on the bottom right) were placed on the performer's suit to track the action sequences. Sources: Human silhouette source extracted from Wikimedia Commons; Vicon Vantage™ camera extracted and adapted from the Vicon website available at: <https://www.vicon.com/hardware/cameras/vantage/>; Passive marker extracted and adapted from: <https://www.qualisys.com/hardware/accessories/passive-markers/>.

Aragor and Talita, the 3D virtual characters depicted in Figure 3.6, were obtained from Autodesk® MotionBuilder® animation software and from De Martino *et al.* (2017), respectively.

The motion retargeting process was implemented in Python, and it is executed offline in the post-processing stage via script. The evaluation system depicted in Chapter 4 was implemented using Unity® game engine.

3.5 Concluding Remarks

This chapter described the implemented methodology for preserving the spatial relationship between joints and the body surface in motion retargeting. The movements of an actor recorded by mocap technology may be distorted when retargeted to a virtual model that does not share the same body proportions and body surface shape. Distortions in the animation may confuse or annoy the spectator. The approach minimizes this problem. The motion retargeting process is initiated via a python script and occurs offline, in the mocap post-processing stage.

First, a correspondence between skeletons is acquired through the *Skeleton Map*, enabling to correlate joints in both skeletons. Then, the *Initial Motion Retargeting* is applied to create the target animation. This process consists of aligning both skeletons' bones and applying the same transforms of the source joints on the correspondent target joints. The actor and virtual model body surfaces are calibrated. The surface calibration of the performer and virtual character must be done only once for each performer and each virtual character, respectively. The spatial relationship of source hand joints with the actor body surface is computed, normalized, and stored as egocentric coordinates. The same relationship is enforced on the target animation to adjust its pose. The adjustment accounts for the target skeleton body proportions and shape.

The resulting motion can be used to animate virtual characters while preserving the spatial relationship with the surface as the original mocap motion. The process is capable of automatically transfer the movements of mocap technology to virtual agents even when the performer and the digital character have different body proportions and shapes. This approach diminishes the distortions found in the conventional motion retargeting process and preserves the self-body interaction of the mocap performer. The quality of the animations generated via the motion retargeting process is assessed by a perceptual evaluation described in the next chapter.

Chapter 4

Perceptual Evaluation

This chapter presents the results of a perceptual evaluation conducted to assess the resulting animations from the implemented motion retargeting methodology.

The animation sequences generated by the presented approach should maintain key aspects of the original motion, such as self-contact and body surface interaction. Objective measurements comparing original versus retargeted trajectories could provide information on how the motion retargeting process changed the trajectories of joints to adjust the target character's pose. However, analyzing joint motion or parameters like joint speed and angular velocity fails to assess subjective aspects and to answer if the resulting animations look natural.

Subjective tests assess the specific perception of a person regarding the stimuli presented. They are convenient to evaluate the level of realism of the resulting animation. Subjective tests can also assess the acceptability of embodied virtual agents for human-computer interaction applications.

Except for Molla *et al.* (2017), none of the works discussed in Chapter 2 performed any evaluation with human subjects. Molla *et al.* (2017) conducted a perceptual evaluation in which 21 subjects were asked to compare the animations generated by their methodology to the ones produced by the existing alternative methodologies from Kulpa *et al.* (2005) and Al-Asqhar *et al.* (2013).

We conducted a perceptual evaluation that focused on assessing the contribution of the implemented motion retargeting methodology to a greater animation videorealism. Since the target avatars may be designed to be purposely non-humanoid or to present unrealistic appearance, in the present work, the animation videorealism is defined as a measure of how close the resulting avatar movements are to the original performance. In particular, our methodology aims to contribute to the proper animation of the primarily conveyed information (motion semantics), the perceived smoothness and the coherence of movements, and the avatar self-awareness.

The evaluation was conducted showing to volunteer participants: (1) the recorded video of the performer during the mocap session; (2) the retargeted animations, before the

Spatial Relationship Encoding and *Pose Adaptation* pipeline steps described in Chapter 3 (see Figure 3.1); and (3) the retargeted animation resulting from the complete pipeline.

The participants were asked to evaluate the retargeted animations in comparison with the "gold-standard", provided by the recorded video of the performer.

This chapter is organized as follows: in Section 4.1, we present the mocap system setup in which the stimuli were recorded and a brief description of the movements evaluated; Section 4.2 details the evaluation protocol adopted; Section 4.3 presents the overall results and a discussion for each stimulus; and Section 4.4 presents the concluding remarks.

4.1 Test Stimuli

The perceptual study consisted of a set of selected movements evaluated by the volunteer subjects. Nine movements were recorded for the study as video and mocap data, simultaneously. As shown in Figure 4.1, the movements were acquired using nine Vicon Vantage™ infrared cameras (V1 to V9) and a conventional digital video camera (VC).

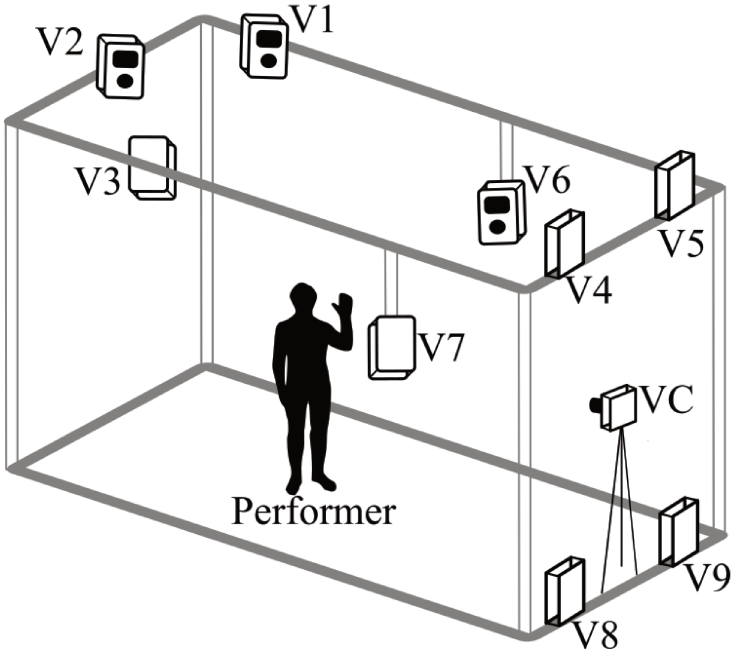


Figure 4.1: The optical mocap system setup consisting of nine Vicon Vantage™ cameras (V1 to V9) and a conventional HD video camera (VC). (Human silhouette source: Wikimedia Commons.)

Figure 4.2 presents the configuration of the markers on the performers' body surface. No facial or finger animation was captured. The skeleton animation, the joint rotation angles and the translation values, were recovered through Vicon Shōgun™ and exported as a BVH file. Talita, the signing avatar from the TALES project, was adopted as the target character for the study (DE MARTINO *et al.*, 2017).

Table 4.1 lists a description of the movements used in the study. The movements

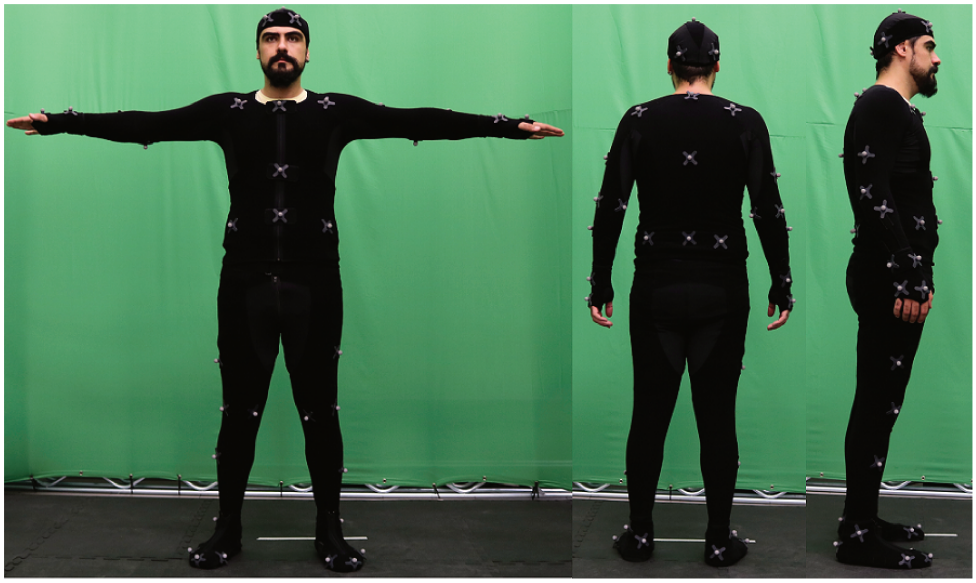


Figure 4.2: Configuration of fifty-three reflective markers of an optical mocap system for capturing body motion.

were chosen based on typical distortions caused by the retargeting process, and some of them were inspired by movements that are common to sign languages.

Identification	**Movement**
1	Touching the chin with the back of the hand
2	Covering the mouth with the hand
3	Scratching the head with both hands
4	Touching the point of the fingers in front of the body
5	Touching the forehead with the palm
6	Saluting
7	Protecting the eyes from the sun
8	Covering the eyes
9	Placing palm on the side of the head

Table 4.1: Action sequences captured for the perceptual evaluation.

Two motion retargeting processes were applied on all movements: the *Initial Motion Retargeting* (Section 3.1) and the complete motion retargeting process (Sections 3.2 and 3.3). The later preserves the surface spatial relationship of the hands with the surface from the original movement.

4.2 Test Protocol

Twenty-five subjects were involved in the evaluation to assess the realism of the resulting animations (17 men, 8 women). Their ages ranged from 20 to 59 years, and they had no previous contact with the research subject.

The evaluation was conducted by an application running on a desktop computer

with a conventional twenty-two-inch monitor (see Figure 4.3). The participants were in a dedicated room to avoid distractions and to ensure privacy, under the supervision of a researcher. They were briefly introduced to the study protocol and the instructions to control the test application. The participants were also asked to sign a consent form.

Figure 4.3: Volunteer subject performing the evaluation on a desktop computer with a twenty-two inch monitor. The volunteer interacted with the test application using the keyboard and the mouse, printed instructions detailing the controls were available.

The test application was developed in Unity® game engine, a 3D development platform. Subjects were presented to two animated 3D characters along with the respective video for each movement (Figure 4.4). The virtual agents, one on each side of the screen, were animated by different motion retargeting processes. The test application randomly defined their position on the screen (left or right).

The characters' movements were presented in a 3D virtual world, as opposed to Molla *et al.* (2017), that presented the animation sequences as recorded video clips. In the 3D virtual world, the participants were able to interact with the environment through limited control over the virtual world, the camera positioning, and the virtual characters. The interaction with the virtual world enabled the participants to perform a careful inspection of the animation and to detect subtle details. The user control included the following actions:

- move camera up and down;
- zoom in and out (move the camera forward or backward);
- change camera positioning to one of the options:
    1. default position, as in Figure 4.4, with the video and both characters present;
    2. center camera in front of the left character;
    3. center camera in front of the right character;

Figure 4.4: A snapshot of the perceptual evaluation application. A set of movements performed during mocap is presented to the subjects as video and as animations, simultaneously. One character motion was generated by the implemented approach, and the other is the result of the initial retargeting described in Section 3.1.

4. center camera in front of the video, removing both characters;
5. remove the video and approximate both characters.

- rotate the characters clockwise and anti-clockwise around their axes;
- replay the video and animations simultaneously;
- and open the question dialog to evaluate the movement, as in Figure 4.5.

Subjects were allowed to replay the video and animations simultaneously as much as they want. No time limit was enforced. After analyzing each movement, the subjects were asked which animation they think best represents the movements in the video. They could choose one of the answers:

- The left one.
- The right one.
- They are identical and represent the movement well.
- They are identical and do not represent the movement well.

One movement was evaluated at a time. By answering the question, the session moved to the next movement. The presentation order of movements was randomized for each subject. After analyzing all movements, the session ended automatically.

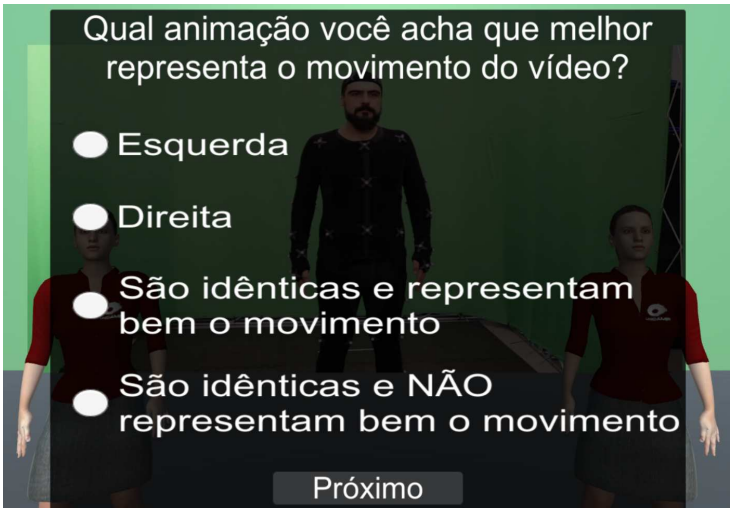


Figure 4.5: Question dialog presented to participants for each animation in Brazilian Portuguese. Participants were asked to answer the question "Which animation do you think best represents the movements in the video?" selecting one of the answers: (1) the left one; (2) the right one; (3) they are identical and represent the movement well; or (4) they are identical and do not represent the movement well.

4.3 Results

Subjects were asked the question: "Which animation do you think best represents the movements in the video?". This section presents the perceptual evaluation results, adopting the following convention:

- SMR corresponds to the number of votes received by the animation that was generated by the complete pipeline (Surface-aware Motion Retargeting);
- IMR identifies the votes received by the animation generated from the simple *Initial Motion Retargeting*;
- IW represents the votes given to the option that states that both animations represent the performer's movement identically well;
- INW represents the votes given to the option that states that both animations do not represent the performer's movement properly.

Figure 4.6 presents the overall percentage of votes, combining the answers of all subjects to all stimuli. The barplot shows that 72.9% of subjects found that the movements retargeted by the implemented methodology represented the performer's movements in the video more faithfully than those retargeted only through the *Initial Motion Retargeting* process (8.4%). This indicates that maintaining the spatial relationship with the surface contributed to greater videorealism of the target animation. Additionally, 15.1% of the overall votes indicated that both animations are equal and represent the video movements well, while 3.6% found the opposite.

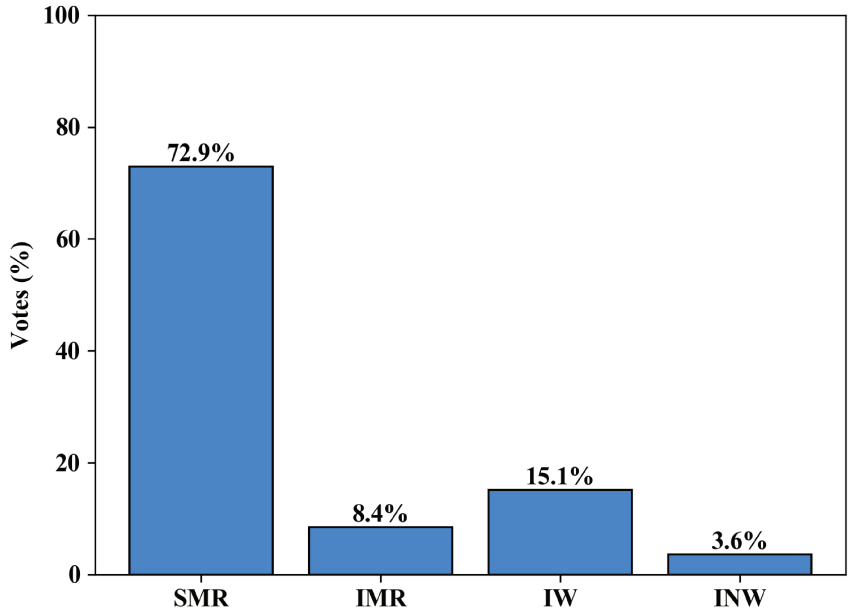


Figure 4.6: Percentage of votes that (from left to right): preferred the retargeted motion by the implemented methodology (SMR); preferred the retargeted motion by the *Initial Motion Retargeting* only (IMR); found the animations identical and that they do (IW) and do not (INW) represent the movement well.

4.3.1 Analysis of the Motion Retargeting per Movement

In this section, we discuss the results for each stimulus. The stimuli included animations with distortions in the *Initial Motion Retargeting*, such as penetration (3, 4, and 5), lack of contact (1 and 6), and hand misplacement regarding the surface (all except movement 4).

Figures 4.7 to 4.15 summarize the individual results. In each figure, it is displayed a snapshot of the screen presented to subjects (left) and the distribution of votes the stimulus received (right). The snapshot represents the movement in a key-pose that represents the moment of most significant interaction between the hands and the body surface. The key-pose interaction is also highlighted for each virtual agent. The highlights are shown in a view angle that favors an artifact visualization.

Table 4.2 summarizes the subjects' votes percentage for each movement presented. The individual analysis of each stimulus shows that SMR votes were higher than IMR for all stimuli. That indicates that the preservation of spatial relationship between the surface contributes to a higher videorealism. The movements 1, 2, 5, 7 and 8 received SMR votes above the overall mean, 72.9%, as shown in Figure 4.6, these stimuli are presented in Figures 4.7, 4.8, 4.11, 4.13, and 4.14, respectively. The stimuli in which the SMR votes were below the overall mean included the movements number 3, 4, 6 and 9, shown in

	Votes (%)			
Movement	**SMR**	**IMR**	**IW**	**INW**
1	80	4	16	0
2	100	0	0	0
3	44	28	20	8
4	28	16	40	16
5	92	8	0	0
6	60	8	28	4
7	92	0	4	4
8	96	4	0	0
9	64	8	28	0
Overall	72.9	8.4	15.1	3.6

Table 4.2: Votes percentage per movement that: preferred the retargeted motion by the implemented methodology (SMR); preferred the retargeted motion by theInitialMotion Retargetingonly (IMR); found the animations identical and that they do (IW) and do not (INW) represent the movement well.

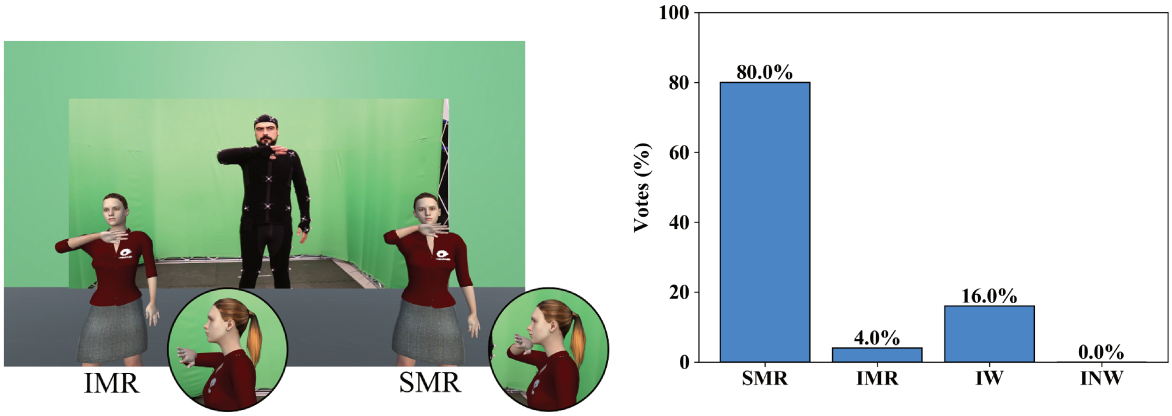
Figures 4.9, 4.10, 4.12, and 4.15, an individual discussion for each one of these movements is presented.

- **Movement 3**, in Figure 4.9, presents the movement of scratching the head, similar to washing the hair. The *Initial Motion Retargeting* resulted in the hands' movement a little forward of the head, and not by its side. Furthermore, the point of the index finger penetrates the head. The surface relationship motion retargeting adjusted the hands' position to the side of the head. However, finger penetration was not handled, which could have made participants unsure of what animations were more faithful to the video.

- **Movement 4**, in Figure 4.10, the hands are close to each other with a contact in the tip of the fingers. The placement of both hands regarding the body surface is good, but there is penetration between fingers. The SMR animation had the hands moved away from each other, but the penetration was still present. Additionally, the orientation of the hands was slightly inclined in both animations. This may be the reason for the higher number of INW votes, indicating that both animations were inconsistent.

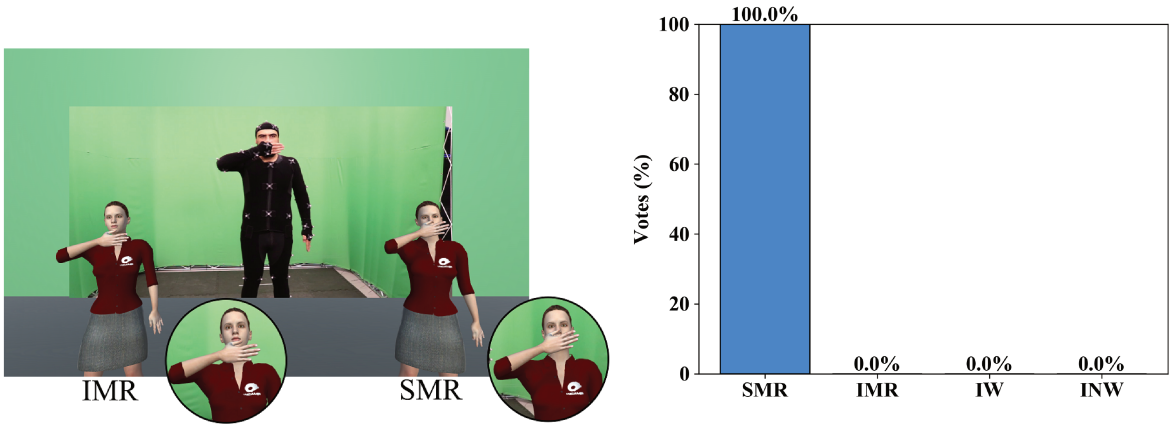
- **Movement 6**, in Figure 4.12, the hand of the IMR animation is a little low, and it lacks contact with the forehead. The complete motion retargeting process brought the hand closer and adjusted its position, but it was a subtle correction.

- **Movement 9**, in Figure 4.15, is analogous to movement 3. The hands of the IMR animation should be placed on the side of the head, above the ear, as the SMR animation.



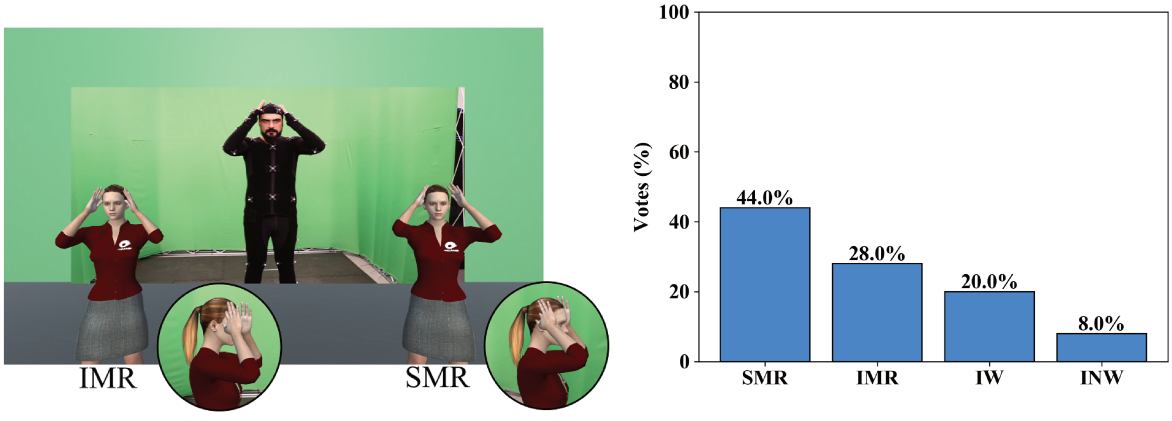
(a) (b)

Figure 4.7: Movement 1 stimulus, on the left, and distribution of votes, on the right.



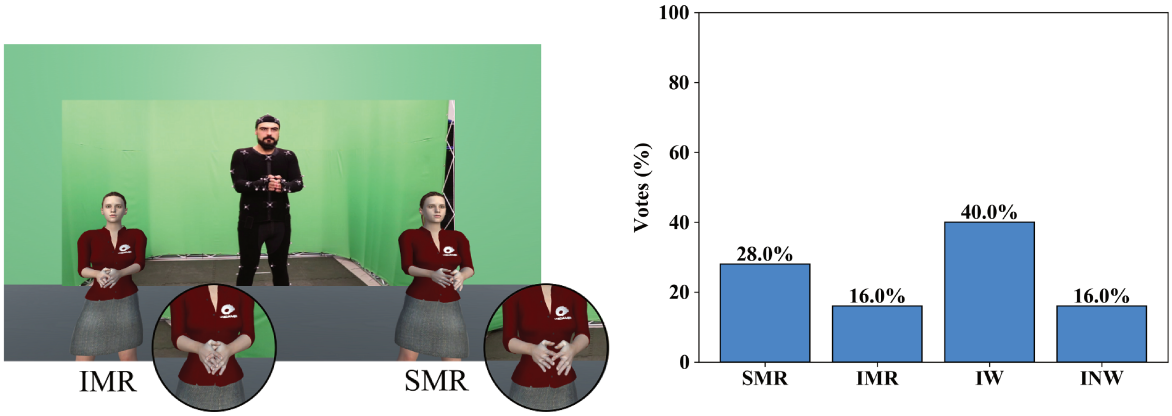
(a) (b)

Figure 4.8: Movement 2 stimulus, on the left, and distribution of votes, on the right.



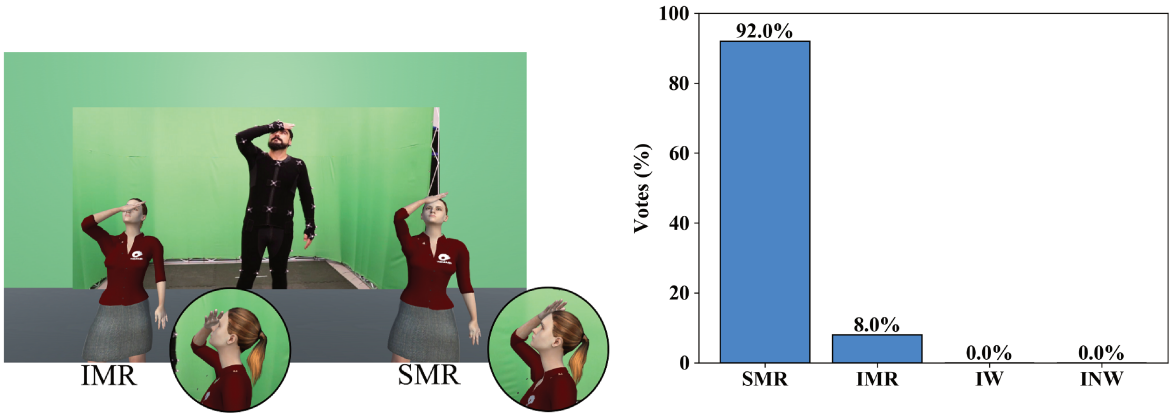
(a) (b)

Figure 4.9: Movement 3 stimulus, on the left, and distribution of votes, on the right.



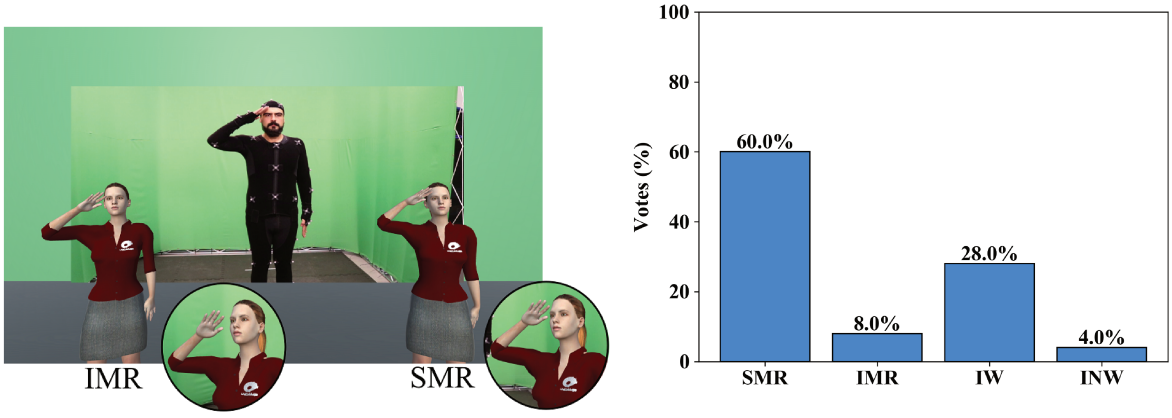
(a) (b)

Figure 4.10: Movement 4 stimulus, on the left, and distribution of votes, on the right.



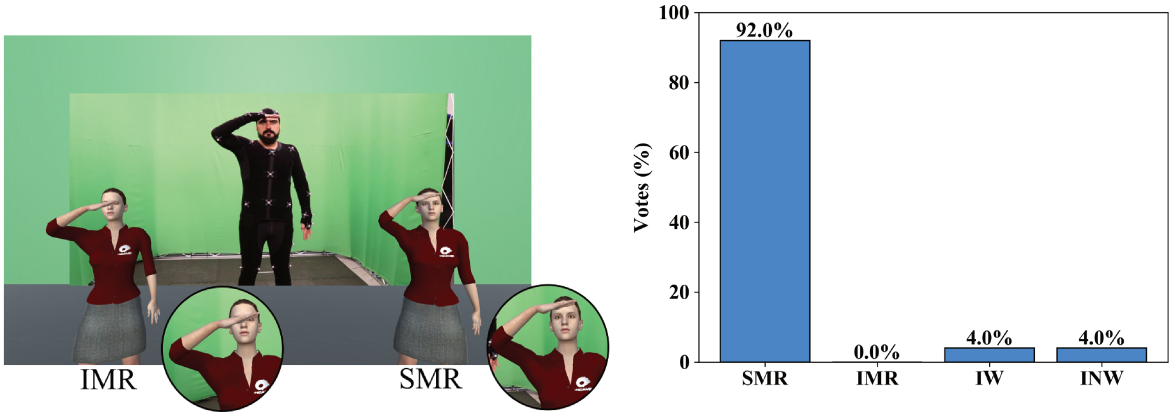
(a) (b)

Figure 4.11: Movement 5 stimulus, on the left, and distribution of votes, on the right.



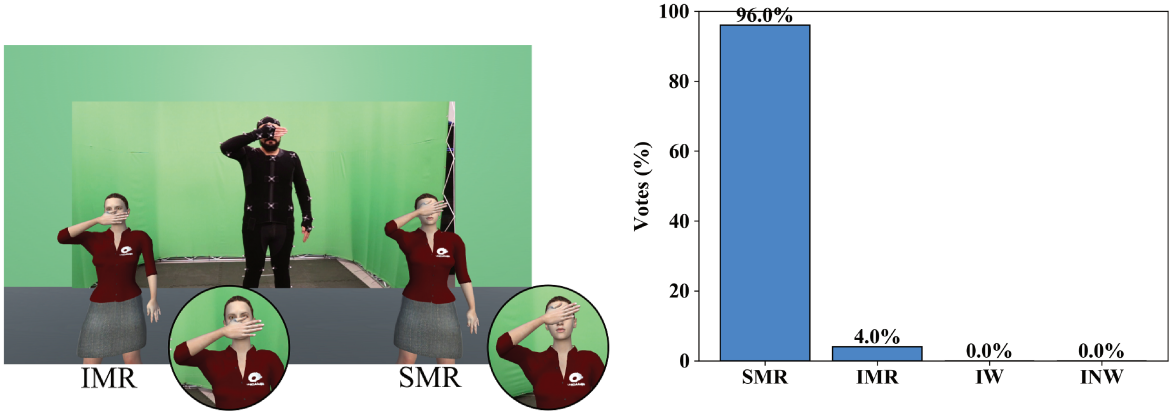
(a) (b)

Figure 4.12: Movement 6 stimulus, on the left, and distribution of votes, on the right.



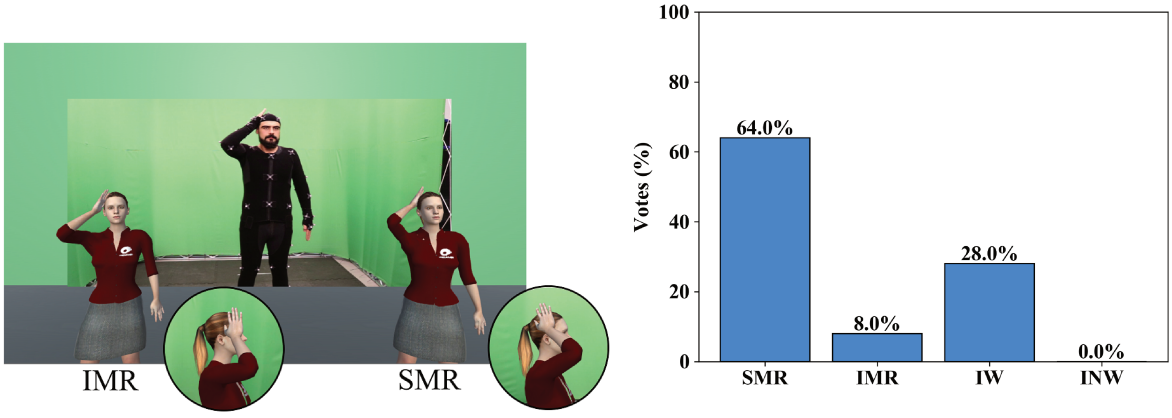
(a) (b)

Figure 4.13: Movement 7 stimulus, on the left, and distribution of votes, on the right.



(a) (b)

Figure 4.14: Movement 8 stimulus, on the left, and distribution of votes, on the right.



(a) (b)

Figure 4.15: Movement 9 stimulus, on the left, and distribution of votes, on the right.

The original motion is displayed in two dimensions as recorded videos, i.e., only the front view of the performer was recorded. This attribute undermines the subjects' spatial understanding and may fail to expose features of the movements otherwise revealed by observing through other directions. Therefore, some of these features could have been unnoticed by the subjects.

The stimuli 3 and 4 presented the more balanced vote distribution (indicating an overall uncertainty). Also, movements 3 and 4 were the ones that still had distortions. The artifacts present on both movements were finger penetration, which was not accounted for in the implemented methodology. Besides, we emphasize that the adopted mocap technique had no finger capturing and, consequently, the source skeleton had no depiction of finger positioning. As future work, the methodology could be extended to handle finger positioning analogously as it handles hand positioning.

4.4 Concluding Remarks

This chapter described the adopted evaluation protocol of the motion retargeting process. Volunteer subjects judged the faithfulness of two animated characters on representing the movements of a real human being. One of the characters had its motion retargeted by the surface-aware approach, which enforces the same hand spatial relationship with the body surface as the original motion. Subjects evaluated nine action sequences captured through mocap technique, to animate the virtual agents, and recorded through a digital camera, presented to subjects as a reference. Subjects voted which animation they found that most resembles the original movement, presented as recorded video. They could indicate that they found the animations similar and resembling or not the movement in the video.

From the result analyses, retargeted animations that still showed artifacts, even though they were diminished, had a more uniform distribution of votes. Thus, the subjects' attention is being drawn from the adjustments to the still-present artifacts.

Given the animation sequences where these even distributions were found, future improvements should focus on further adjusting hand orientation, strongly avoiding self-penetration, and extending the methodology to handle finger joints. The later does not imply capturing finger motion, but to inspect the generated animation for finger collision with the surface components.

The results show that the animated character adjusted by the implemented methodology always scored better than the animation generated from simple initial retargeting. This implies that preserving the relationship of joints with surface components benefit motion retargeting algorithms with the intention to ensure self-contact and interaction with the body surface. It is clear that the motion retargeting process contributes to a higher similarity of the animation to the original motion.

Chapter 5

Conclusions

With the increase in targeted broadcasting TV, the availability of low cost internet access, the penetration of mobile devices, the growing popularity of streaming video, and the new demand for immersive experiences such as Augmented Reality and Virtual Reality, the entire Animation, Visual Effects (VFX) and Games industry has become one of the fastest-growing segments in the global media and entertainment market. According to the *Global Animation, VFX & Games Industry Report* (2019), the total value of the global animation industry was US$ 259 billion in 2018 and is projected to reach US$ 270 billion by 2020 (MARKETS, 2019).

In this context, mocap is widely applied as a key technology to deliver high-quality animation with reduced animator hours dedicated to *keyframing* and other traditional animation techniques.

However, as described in Chapter 2, the process of transferring the movements of a human performer to a virtual character is not flawless. In particular, the present work approached three existing challenging problems:

- **Automation:** fully automated motion retargeting is particularly desirable for dynamic content generation that requires a fast production cycle (while an animation movie may take several months or even years to be produced, TV commercials or online material demand a much faster production time);

- **Mocap Reuse or Character-Agnostic Motion Retargeting:** a classical problem in motion retargeting is that final animation quality depends on the similarity between body dimensions of the actor and the target avatar. This limitation typically prevents the retargeting of mocap material to different characters, but also hinders the creation of certain types of content (for example, if we would like to transfer the signing gestures of an adult sign language teacher to an animated child character).

- **Semantic Preserving Motion Retargeting**: humans are biological systems specialized in analyzing movements and attributing meaning to them. For this reason, as

important as to develop an automatic character-agnostic motion retargeting process, it is to preserve the gestural meaning of the original movement. This aspect is particularly important for the generation of signing avatars.

Considering the motion retargeting pipeline described in Chapter 3, we implemented a *Spatial Relationship Encoding* process that is capable of preserving the spatial relationship between body parts and the body surface observed in the source action sequence, performed by the mocap actor (Section 3.2). Given a correspondence map between captured and virtual character's skeletons and the surface metrics obtained from a 2-minute guided calibration process that is executed only once, the implemented algorithm is fully automated, and it does not require any human intervention. We also implemented *Pose Adaptation* algorithm that is not dependent on the target skeleton topology. In other words, the approach enables the reuse of mocap material to any virtual character with known skeleton topology.

In order to assess the efficacy of the motion retargeting pipeline in preserving the gestural meaning of the original movement, we conducted a perceptual study where volunteer participants where asked to compare the recorded video of an actor during a mocap session to two retargeted animations, with and without the *Spatial Relationship Encoding* process in the animation generation pipeline. The results of our evaluation indicated that the process in the motion retargeting pipeline was capable of better representing the original movement characteristics, by preserving the interaction of the hands with the body surface (Chapter 4).

In summary, some of the main characteristics of the resulting motion retargeting pipeline are:

- **Topology-independent automatic motion retargeting**: the implemented methodology can transfer mocap data to virtual human characters created using different available commercial 3D modeling and animation software.
- **Surface interaction faithful representation**: gestures and actions often include self-body interactions. The present work preserves such interaction even when the virtual characters have different proportions and surface shapes than the mocap performer.
- **Allows the reuse of the animation to several characters**: since the motion is normalized and expressed by the performer body proportions and surface shape, respectively, it can be retargeted to characters that even existed by the time of the mocap recordings. The only requirement is a surface calibration of the virtual character that takes a few minutes.
- **Allows the adjustment to previously captured motions**: if the actors are available to a new mocap session, their surface calibration process can be performed

and then used to transfer previously captured motions to virtual characters created before or after the first mocap session.

- **Integrates with standard software**: the retargeted motion is the standard skeletal animation used by available commercial 3D modeling and animation software.
- **Demand few changes in the mocap pipeline**: the extra steps required by the presented methodology is the *Skeleton Mapping* (performed once whenever a new skeleton topology is used), the performer surface calibration (once for each performer), and the virtual character surface calibration (once per character). The motion retargeting process is initiated via a Python script and takes place offline in the post-processing stage.

Finally, the action sequences resulting from the motion retargeting pipeline and stored as skeletal animations can be used as inputs (pre-retargeted content) to algorithms that handle interactions with the environment and with other virtual characters. Moreover, facial animation data may be combined with the skeleton, towards realistic avatars.

5.1 Limitations and Future Work

The results of our perceptual evaluation indicated that artifacts such as self-penetration and extremities orientation still are challenges that should be further tackled.

We highlight that penetration avoidance is not an active process in the implemented methodology but rather a by-product of preserving the spatial relationship with the body surface. Considering the hands, for example, there is no verification of penetration in the body surface mesh or a limb capsule; consequently, the hand is not pushed away if it does penetrate the body surface component. What (somewhat) guarantees that penetration will not occur is that, when the hand approaches a surface component, that component's importance increases, preventing the hand from approaching any further, even if other components are pushing it towards the body surface. However, two problems arise. First, the importance of other components may still be relevant and outdo the closer component, forcing the hand through the surface. Second, the fingers' spatial relationship with the surface were not computed; therefore, no reference to their distance from the body surface is provided. As future work, both problems can be solved by checking for penetrations from one frame to the next and moving the joint to the projected point on the surface component. Note that this approach may firmly change the trajectory of the joint, and some filtering or smoothing algorithm would be required.

In order to adjust the orientation of the hands and the feet, it is necessary to adjust the body surface components' normal vectors. Our results showed that when a body part is near the surface component, the current approach is reliable to adjust its

orientation. However, when the body part distances itself from the surface, it would be necessary to preserve the global orientation of the source motion. As an example, when executing a gesture in front or on the side of the body (not on the body surface), the orientation of the hands should not be influenced by the body surface components.

The extension of the methodology to handle finger motion data is also a possibility. Natural gestures often include finger movement; therefore, convincing digital animations must represent them. Expressing finger positioning by its spatial relationship with the body surface, along with a more detailed body surface representation, may achieve good results even with subtle motions such as touching the point of the nose with the finger. Moreover, finger resolution in mocap data can also improve the body surface calibration accuracy.

Further perceptual evaluation to assess the generated animations is recommended. In particular, an evaluation protocol focused on the conveyed information rather than faithfulness to the original motion may further verify the benefits of the implemented methodology on preserving the motions' semantics. As an example, the protocol we adopted in previous work to evaluate the comprehension of animated sign gestures by deaf subjects, could be adapted (TONOLI *et al.*, 2018).

References

AL-ASQHAR, R. A.; KOMURA, T.; CHOI, M. G. Relationship descriptors for interactive motion adaptation. In: ACM. *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. [S.l.], 2013. p. 45–53.

BARAN, I.; POPOVIć, J. Automatic rigging and animation of 3d characters. *ACM Trans. Graph.*, ACM, New York, NY, USA, v. 26, n. 3, jul. 2007. ISSN 0730-0301.

BERNARDIN, A.; HOYET, L.; MUCHERINO, A.; GONÇALVES, D.; MULTON, F. Normalized euclidean distance matrices for human motion retargeting. In: ACM. *Proceedings of the Tenth International Conference on Motion in Games*. [S.l.], 2017. p. 15.

BUSS, S. R. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. *IEEE Journal of Robotics and Automation*, v. 17, n. 1-19, p. 16, 2004.

DE MARTINO, J. M.; SILVA, I. R.; BOLOGNINI, C. Z.; COSTA, P. D. P.; KUMADA, K. M. O.; CORADINE, L. C.; BRITO, P. H. da S.; AMARAL, W. M. do; BENETTI, Â. B.; POETA, E. T. *et al.* Signing avatars: making education more inclusive. *Universal Access in the Information Society*, Springer, v. 16, n. 3, p. 793–808, 2017.

GLEICHER, M. Retargetting motion to new characters. In: ACM. *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. [S.l.], 1998. p. 33–42.

HECKER, C.; RAABE, B.; ENSLOW, R. W.; DEWEESE, J.; MAYNARD, J.; PROOIJEN, K. van. Real-time motion retargeting to highly varied user-created morphologies. In: ACM. *ACM Transactions on Graphics (TOG)*. [S.l.], 2008. v. 27, n. 3, p. 27.

HENNE, M.; HICKEL, H.; JOHNSON, E.; KONISHI, S. The making of toy story [computer animation]. In: IEEE. *COMPCON'96. Technologies for the Information Superhighway Digest of Papers*. [S.l.], 1996. p. 463–468.

HO, E. S.; KOMURA, T.; TAI, C.-L. Spatial relationship preserving character motion adaptation. In: ACM. *ACM Transactions on Graphics (TOG)*. [S.l.], 2010. v. 29, n. 4, p. 33.

HSIEH, M.-K.; CHEN, B.-Y.; OUHYOUNG, M. Motion retargeting and transition in different articulated figures. In: IEEE. *Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*. [S.l.], 2005. p. 6–pp.

JIN, T.; KIM, M.; LEE, S.-H. Aura mesh: motion retargeting to preserve the spatial relationships between skinned characters. In: WILEY ONLINE LIBRARY. *Computer Graphics Forum*. [S.l.], 2018. v. 37, n. 2, p. 311–320.

KERLOW, I. V. *The art of 3D: computer animation and effects*. 3. ed. Hoboken, NJ, USA: John Wiley & Sons, 2004. 451 p. ISBN 0-471-43036-6.

KITAGAWA, M.; WINDSOR, B. *MoCap for artists: workflow and techniques for motion capture*. [S.l.]: CRC Press, 2012.

KUCUK, S.; BINGUL, Z. Robot kinematics: Forward and inverse kinematics. In: *Industrial Robotics: Theory, Modelling and Control*. [S.l.]: IntechOpen, 2006. p. 117–36.

KULPA, R.; MULTON, F.; ARNALDI, B. Morphology-independent representation of motions for interactive human-like animation. In: WILEY ONLINE LIBRARY. *Computer Graphics Forum*. [S.l.], 2005. v. 24, n. 3, p. 343–351.

LIU, Z.; MUCHERINO, A.; HOYET, L.; MULTON, F. Surface based motion retargeting by preserving spatial relationship. In: ACM. *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*. [S.l.], 2018. p. 7.

LOPER, M.; MAHMOOD, N.; BLACK, M. J. Mosh: Motion and shape capture from sparse markers. *ACM Transactions on Graphics (TOG)*, ACM, v. 33, n. 6, p. 220, 2014.

LU, W.; LIU, Y.; SUN, J.; SUN, L. A motion retargeting method for topologically different characters. In: IEEE. *2009 Sixth International Conference on Computer Graphics, Imaging and Visualization*. [S.l.], 2009. p. 96–100.

MAGAZINE LUIZA. *Conteúdo Especial para Pesquisas Acadêmicas: Informações e Imagens sobre a Estrutura da Empresa, Ações Internas e Campanhas Realizadas no Decorrer de cada Ano*. [S.l.], 2017. Accessed: June 30, 2019. Available from Internet: <https://ri.magazineluiza.com.br/informacao_estudantes.aspx>.

MARKETS, l. R. a. *Global Animation, VFX & Games Industry: Strategies, Trends & Opportunities, 2019*. 2019. Available from Internet: <https://www.researchandmarkets.com/reports/4721808/global-animation-vfx-and-games-industry>.

MEREDITH, M.; MADDOCK, S. Motion capture file formats explained. *Production*, p. 1–36, 01 2001.

MOLLA, E.; DEBARBA, H. G.; BOULIC, R. Egocentric mapping of body surface constraints. *IEEE transactions on visualization and computer graphics*, IEEE, v. 24, n. 7, p. 2089–2102, 2017. ISSN 1077-2626.

MONZANI, J.-S.; BAERLOCHER, P.; BOULIC, R.; THALMANN, D. Using an intermediate skeleton and inverse kinematics for motion retargeting. *Computer Graphics Forum*, v. 19, n. 3, p. 11–19, 2000.

MULTON, F.; KULPA, R.; HOYET, L.; KOMURA, T. Interactive animation of virtual humans based on motion capture data. *Computer Animation and Virtual Worlds*, Wiley Online Library, v. 20, n. 5-6, p. 491–500, 2009.

PARENT, R. *Computer animation: algorithms and techniques*. 2. ed. Burlington, MA, USA: Morgan Kaufmann Publishers, 2008. 593 p. ISBN 978-0-12-532000-9.

PONS-MOLL, G.; ROMERO, J.; MAHMOOD, N.; BLACK, M. J. Dyna: A model of dynamic human shape in motion. *ACM Transactions on Graphics (TOG)*, ACM, v. 34, n. 4, p. 120, 2015.

RATNER, P. *3-D human modeling and animation*. 2. ed. Hoboken, NJ, USA: John Wiley & Sons, 2012. 316 p. ISBN 0-471-21548-1.

SHIN, H. J.; LEE, J.; SHIN, S. Y.; GLEICHER, M. Computer puppetry: An importance-based approach. *ACM Transactions on Graphics (TOG)*, ACM, v. 20, n. 2, p. 67–94, 2001.

SHOEMAKE, K. Euler angle conversion. *Graphics gems IV*, Academic Press Professional, p. 222–229, 1994.

SLABAUGH, G. G. Computing euler angles from a rotation matrix. *Technical Report*, 1999.

TONOLI, R. L.; COSTA, P. D. P.; MARTINO, J. M. D. Signing avatar in virtual reality: An intelligibility study. In: SVR, 20., 2018, Foz do Iguaçu. *Proceedings of the WTD 20th Symposium on Virtual and Augmented Reality*. Foz do Iguaçu, 2018.

Appendix A

BVH Example

The BVH format is used to store the animation data of a single skeleton, it provides the information to assemble the skeleton and animate each bone in a hierarchical fashion. A BVH file contains the structure definition and the base pose of a skeleton in the header, and the motion data in the body. Meredith e Maddock (2001) provides a detailed description of the BVH file structure, how to parse it and draw the animation (MEREDITH; MADDOCK, 2001).

The keywords **HIERARCHY** and **MOTION** indicate the beginning of the header and the data section of the BVH file, the body, respectively. The BVH header contains three information regarding a joint:

- its place in the hierarchy, obtained through its position in the file;
- the distance from its parent joint, called offset;
- and the number of channels, as well as their type and order.

The channels are the DOFs of a joint to translate or rotate regarding its parent joint. Although not mandatory, usually a joint will have three channels for rotation, one for each axis, and three channels for translation, again, one for each axis. However, to animate virtual humans, usually a joint will have only the three channels for rotation, since the bones do not stretch, the translation would be equal for every frame and thus can be ignored.

The body contains two lines specifying the number of frames and the frame time of the animation, lastly, for each frame, one line holding the values of each channel for each joint, at that frame, in the order presented in the header. Thus, the given file has ten lines with eighteen values, being six values for each one of the three joints, as shown in Figure A.1. The animation's frame rate is calculated through the frame time on the second line of the file as in Equation A.1. Figure A.2 presents the structure of the skeleton defined by the BVH file in the base pose, at frame one, five and ten, from left to right.

```
HIERARCHY
ROOT Joint1
{
   OFFSET 0.0 0.0 0.0
   CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
   JOINT Joint2
   {
      OFFSET 10.0 0.0 0.0
      CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
      JOINT Joint3
      {
         OFFSET 10.0 0.0 0.0
         CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
         End Site
         {
            OFFSET 10.0 0.0 0.0
         }
      }
   }
}
MOTION
Frames: 10
Frame Time: 0.00833333
10.0 0.0 0.0 0.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0  → Frame 1
10.0 0.0 0.0 10.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0 → Frame 2
10.0 0.0 0.0 20.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0
10.0 0.0 0.0 30.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0
10.0 0.0 0.0 40.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0      .
10.0 0.0 0.0 50.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0      .
10.0 0.0 0.0 60.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0      .
10.0 0.0 0.0 70.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0
10.0 0.0 0.0 80.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0
10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 90.0 0.0 0.0 10.0 0.0 0.0 0.0 0.0 0.0 → Frame 10
 ⏟⏟⏟  ⏟⏟⏟  ⏟⏟⏟  ⏟⏟⏟  ⏟⏟⏟  ⏟⏟⏟
Translation Rotation Translation Rotation Translation Rotation
  Joint1    Joint1    Joint2    Joint2    Joint3    Joint3
```

Figure A.1: BVH file example.

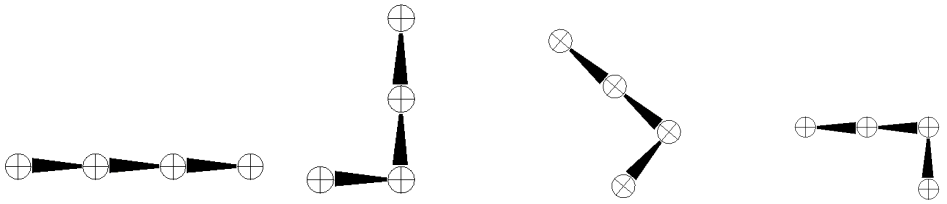


Figure A.2: Skeleton pose in the base pose and during the motion.

Although straightforward and readable to humans, the BVH file has a couple of drawbacks. First, there is no information regarding the unit of the translation and offset values, so it can be meters, millimeters or any arbitrary unit (rotational values are expressed in degrees); and second, no explicit rotational values of the joints are provided for the base pose, therefore, the joints orientation in the base pose are unknown.

$$frame\ rate = \frac{1}{frame\ time} \tag{A.1}$$

Appendix B

Performer Body Surface Calibration

In the post-processing stage, the algorithm searches in the three calibration takes for the frames in which the performer is touching a surface point. The first take is used to calibrate the trunk and the limbs; in the second take the head is calibrated; and the back in the third take. Since the hands of the performer are steady while touching the surface point, the algorithm finds zero-speed hand moments and, assuming that the performer calibrated his surface points in the correct order (as Table 3.1), the position of the hand joint represents the position of the surface point. Given the position of a hand joint $\mathbf{p}(t)$ at frame t and the position in the last frame $\mathbf{p}(t-1)$, the speed $s(t)$ at the interval between $t-1$ and t is simply:

$$s(t) = \|\frac{\mathbf{p}(t) - \mathbf{p}(t-1)}{t-(t-1)}\| = \|\mathbf{p}(t) - \mathbf{p}(t-1)\| \tag{B.1}$$

The following figures present performer's pose during the calibration of a body surface point for the trunk and limbs (Figure B.1), the head (Figure B.3), and the back (Figure B.5). The images were extracted from the reference video presented to the mocap performers during the body surface calibration procedure. Moreover it is presented the automatic analyses of the right hand speed and the frame ranges where it approaches zero, indicating that the hands are steady while calibrating a surface point. Figures B.2, B.4, and B.6 present the analyzes for the first (trunk and limbs), second (head), and third (back) takes, respectively. Since the performer calibrates mimicking the movements of prerecorded videos, the sequence of the poses executed is known.

Table B.1 correlates the pose and the frame ranges during each calibration take. The frame ranges not depicted in the table correspond to the T-Poses at the end and the beginning of each take, and the transition poses that the mocap performer lowers the arms before posing for the next body surface point. Observe that the analyses presented correspond to the right hand and the body surface points calibrated by the right hand (right chest, right hip, right thigh, etc.); the analysis for the left hand is performed in an analogous manner.

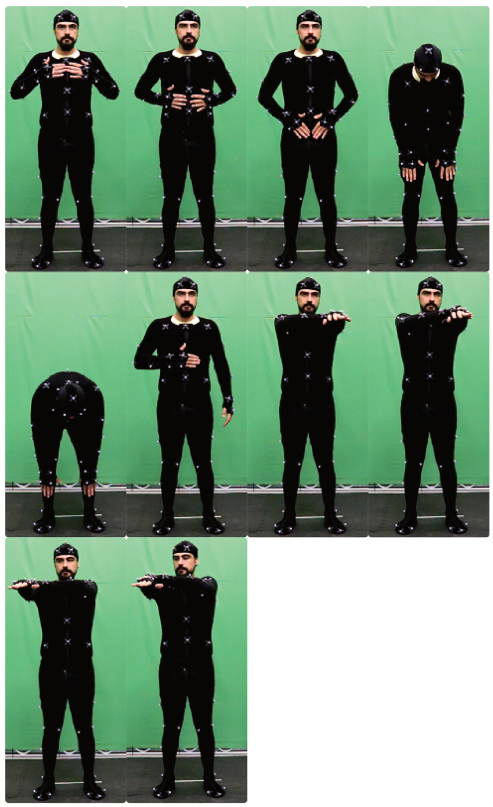


Figure B.1: Performer's pose during the body surface calibration of the trunk and limbs.

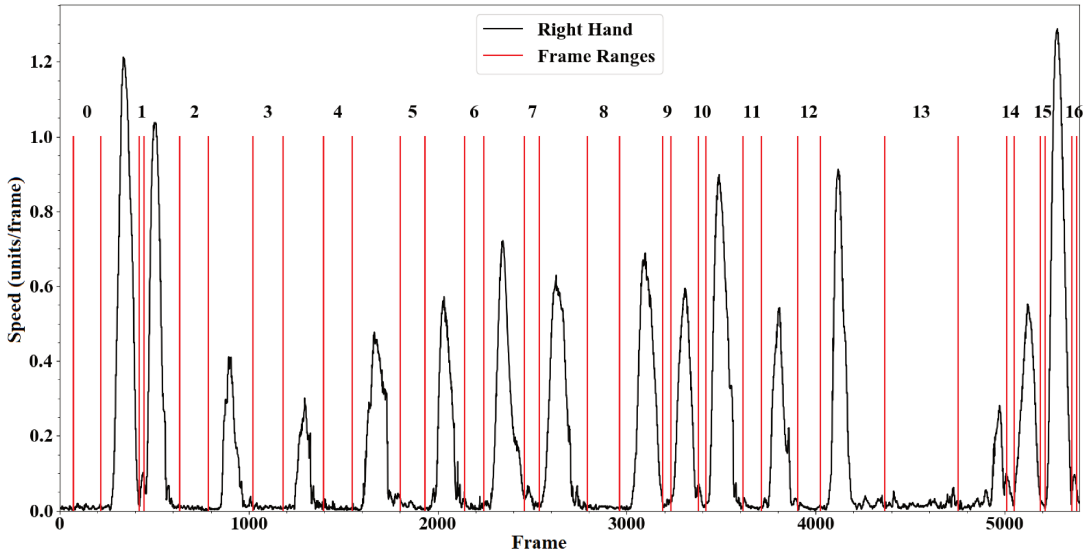


Figure B.2: Graphic of the speed of the right hand throughout the first take of the performer's body surface calibration. The couples of red vertical lines with indices on top represent the frame ranges that the speed of the hand approaches zero.

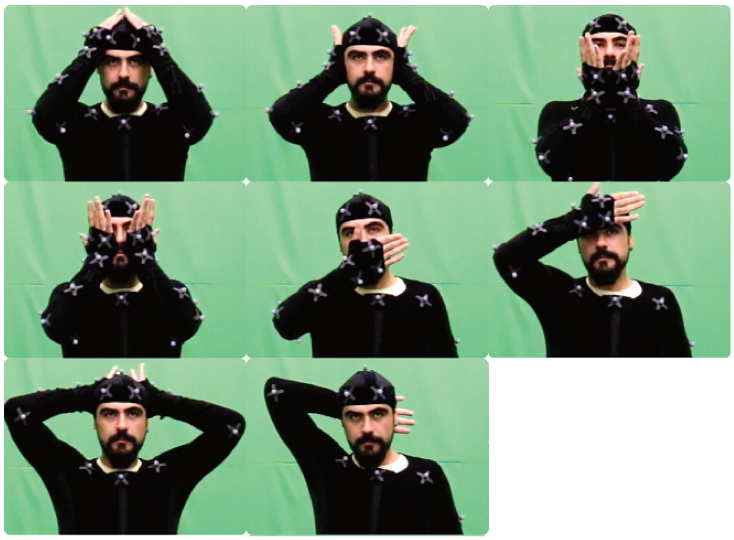


Figure B.3: Performer's pose during the body surface calibration of the head.

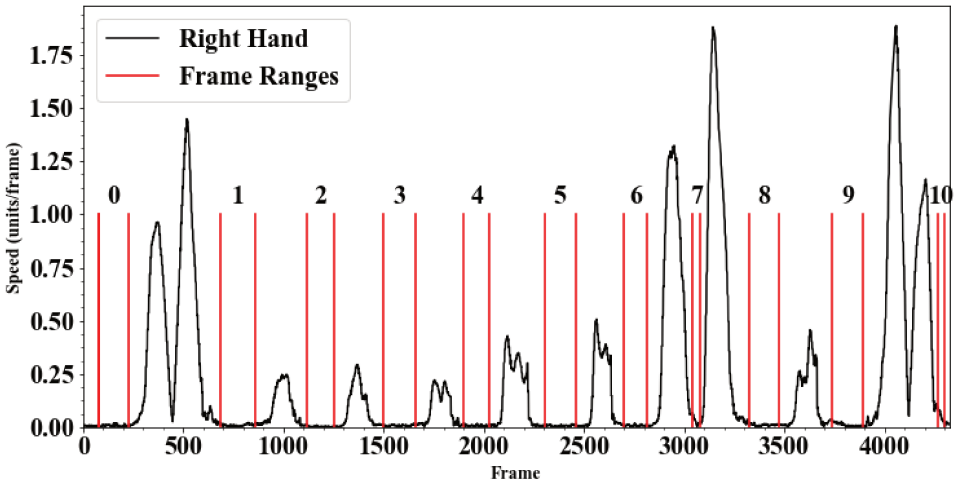


Figure B.4: Graphic of the speed of the right hand throughout the first take of the performer's body surface calibration. The couples of red vertical lines with indices on top represent the frame ranges that the speed of the hand approaches zero.

Figure B.5: Performer's pose during the body surface calibration of the back.

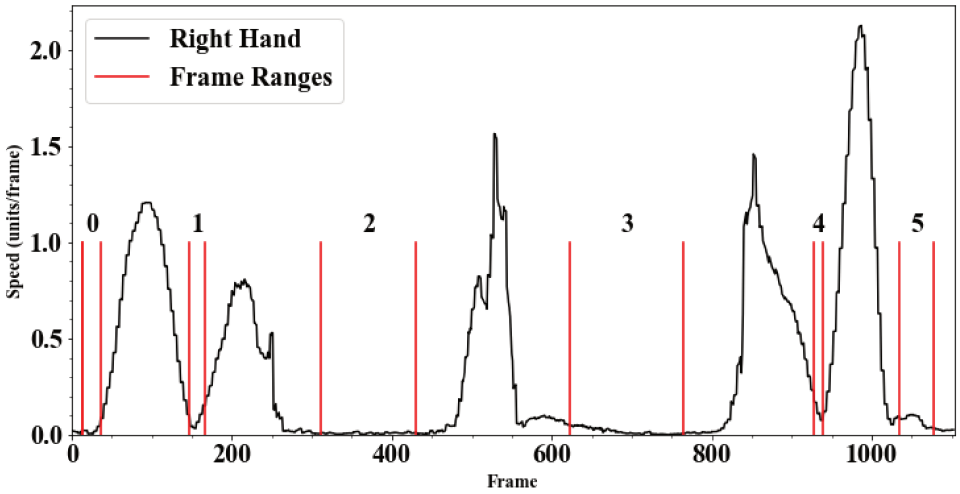


Figure B.6: Graphic of the speed of the right hand throughout the third take of the performer's body surface calibration. The couples of red vertical lines with indices on top represent the frame ranges that the speed of the hand approaches zero.

Take	Surface Point	Frame Range Index
First	Chest	2
	Abdomen	3
	Hip	4
	Thigh	5
	Shin	6
	Abdomen up	8
	Left arm	10
	Left forearm	11
Second	Forehead	1
	Ear	2
	Chin	3
	Cheek	4
	Mouth	5
	Forehead middle	6
	Nape	7
	Nape middle	9
Third	Loin	2
	Loin up	3

Table B.1: The three takes to calibrate the body surface of the mocap performer, the surface points sampled at each take and the index of the respective frame range in the speed analysis.

The position of hand during calibration dictates the position of the body surface point. The local transformation matrix of each body surface point to a nearby joint is stored in a CSV file. Since the limbs are modeled as capsules, their radius are stored in the file instead of its transformation matrix. Each line of the file correspond to a body surface point; the elements of the transformations matrix are separated by a comma.