



## TESI DOCTORAL

Títol	Animation and Interaction of Responsive, Expressive, and Tangible 3D Virtual Characters
Realitzada per	Adso Fernández Baena
en el Centre	La Salle Campus Barcelona
i en el Departament	Grup de Tecnologies Mèdia (GTM)
Dirigida per	Dr. David Miralles Esteban





## Agraïments

La consecució d'aquesta tesi ha estat un camí llarg, algunes vegades molt divertit i d'altres molt dur, on he tingut l'oportunitat de conèixer a diverses persones molt interessants que m'han ajudat i acompanyat, i que de la mateixa manera que aquesta tesi, m'enporto amb mi per sempre.

En primer lloc, m'agradaria agrair al meu director de tesi, en David Miralles, l'oportunitat d'haver pogut realitzar aquest treball. Sense ell, aquesta tesi no hagués estat possible, i sobretot, li agraeixo el fet de d'haver-me ensenyat el món de la recerca. A més a més, durant aquests anys he crescut molt com a professional i persona al seu costat gràcies als seus bons consells i indicacions. Ha estat un plaer treballar amb ell.

A continuació, m'agradaria agrair a en Francesc Alías, director de l'àrea d'HCI (Human Computer Interaction) durant la consecució de la tesi i actualment cap del grup de recerca GTM (Grup de Tecnologies Mèdia) i director d'Enginyeria La Salle, els seus profitosos consells i ànims en moments durs de la tesi. Durant aquest temps, també he tingut el plaer de compartir vivències amb els companys doctorands que van iniciar també aquest camí en temps similars, en Marc Arnella, en Ramon Martín, en Xavier Valero i l'Angels Aragonès, que han estat de gran ajuda.

També vull agrair l'entusiasme, professionalitat i ganes de participar en els nostres projectes d'investigació a l'Eduard Ruesga i la Meritxell Aragonès, tècnics del Medialab, el laboratori de captura de moviment de la Salle.

Seguint, m'agradaria mencionar als meus companys del projecte THOFU (Tecnologías del HOtel del FUTuro, CEN-20101019), que ha abarcat bona part de la meua tesi i del qual he tingut el plaer de ser Team Leader en un dels seus subpaquets de treball. En especial, vull fer menció del meu company Marc Antonijoan per la seva professionalitat i saber fer, i tranquil·litat a l'hora d'afrontar els problemes, que m'ha fet créixer molt. També vul extendre-ho al Raúl Montaña, amb el qual hem treballat colze a colze, i a l'Arturo Roversi, l'Anna Fusté i Judith Amores.

Tambe vull agrair al meu company Roger Boldú la bona feina i la grata companyia, que han donat lloc a la part final de la tesis. També a en Jordi Albó i al projecte PATRICIA (Pain and Anxiety Treatment based on social Robot Interaction with Children to Improve pAtient experience, TIN2012-38416-C03-01,02), que ha finançat part d'aquesta recerca, i a en Marc Expósito per la seva col·laboració. Per altra banda, també agrair al grup de recerca Fluid Interfaces i a la seva directora, Pattie Maes, pel fet d'haver-me deixat passar prop d'un mes al Medialab del MIT (Massachusetts Institute of Technology), un lloc que no deixa a ningú indiferent, i al Xavi Benavides i de nou a la Judith Amores per fer-nos d'anfitrions.

També vull agrair a la Salle, sobretot al departament d'Enginyeria (abans DTM, Depar-

tament de Tecnologies Mèdia) i al grup de recerca GTM (Grup de Tecnologies Mèdia) per haver-me permès realitzar aquest treball i presentar algunes parts en congressos científics internacionals i conèixer llocs com Hong Kong, Cagliari, Las Vegas o Kobe, i tenir l'oportunitat de compartir experiències amb altres investigadors. A l'Emiliano Labrador per la seva ajuda en el disseny gràfic del pòster que encapsala les publicacions d'aquesta tesi. També agrair a tots els companys de la universitat que en algun moment m'han donat un cop de mà.

Per acabar, i el més important, m'agradaria agrair als meus pares el seu suport incondicional i sobretot, l'oportunitat d'haver pogut estudiar una carrera i un màster, que al fi i al cap, m'han permès poder realitzar aquesta tesi. També agrair al meu germà, en Cesc, als meus tiets, Xati i Pepe, cosins i als meus avis les ganes que m'han transmés per poder aconseguir aquest repte. També a tots els meus amics, pel suport incondicional i les festes que m'han deixat perdre.

Gràcies a tots.

## 0.1 Abstract

This thesis is framed within the field of 3D Character Animation. Virtual characters are used in many Human Computer Interaction applications such as video games and serious games. Within these virtual worlds they move and act in similar ways to humans controlled by users through some form of interface or by artificial intelligence. This work addresses the challenges of developing smoother movements and more natural behaviors driving motions in real-time, intuitively, and accurately. The interaction between virtual characters and intelligent objects will also be explored. With these subjects researched the work will contribute to creating more responsive, expressive, and tangible virtual characters.

The navigation within virtual worlds uses locomotion such as walking, running, etc. To achieve maximum realism, actors' movements are captured and used to animate virtual characters. This is the philosophy of motion graphs: a structure that embeds movements where the continuous motion stream is generated from concatenating motion pieces. However, locomotion synthesis, using motion graphs, involves a tradeoff between the number of possible transitions between different kinds of locomotion, and the quality of these, meaning smooth transition between poses. To overcome this drawback, we propose the method of progressive transitions using *Body Part Motion Graphs (BPMGs)*. This method deals with partial movements, and generates specific, synchronized transitions for each body part (group of joints) within a window of time. Therefore, the connectivity within the system is not linked to the similarity between global poses allowing us to find more and better quality transition points while increasing the speed of response and execution of these transitions in contrast to standard motion graphs method.

Secondly, beyond getting faster transitions and smoother movements, virtual characters also interact with each other and with users by speaking. This interaction requires the creation of appropriate gestures according to the voice that they reproduced. Gestures are the nonverbal language that accompanies voiced language. The credibility of virtual characters when speaking is linked to the naturalness of their movements in sync with the voice in speech and intonation. Consequently, we analyzed the relationship between gestures, speech, and the performed gestures according to that speech. We defined intensity indicators for both gestures (*GSI*, Gesture Strength Indicator) and speech (*PSI*, Pitch Strength Indicator). We studied the relationship in time and intensity of these cues in order to establish synchronicity and intensity rules. Later we adapted the mentioned rules to select the appropriate gestures to the speech input (tagged text from speech signal) in the *Gesture Motion Graph (GMG)*. The evaluation of resulting animations shows the importance of relating the intensity of speech and gestures to generate believable animations beyond time synchronization. Subsequently, we present a system that leads automatic generation of gestures and facial animation from a speech signal: *BodySpeech*. This system also includes animation improvements such as: increased use of data input, more flexible time synchronization, and new features like editing style of output animations. In addition, facial animation also takes into account speech intonation.

Finally, we have moved virtual characters from virtual environments to the physical world in order to explore their interaction possibilities with real objects. To this end, we present *AvatARs*, virtual characters that have tangible representation and are integrated into reality through augmented reality apps on mobile devices. Users choose a physical object to manipulate in order to control the animation. They can select and configure the animation, which serves as a support for the virtual character represented. Then, we explored the interaction of *AvatARs* with intelligent physical objects like the Pleo social robot. Pleo is used to assist hospitalized children in therapy or simply for playing. Despite its benefits, there is a lack of emotional relationship and interaction between the children and Pleo which makes children lose interest eventually. This is why we have created a mixed reality scenario where Vleo (*AvatAR* as Pleo, virtual element) and Pleo (real element) interact naturally. This scenario has been tested and the results conclude that *AvatARs* enhances children's motivation to play with Pleo, opening a new horizon in the interaction between virtual characters and robots.



## 0.2 Resum

Aquesta tesi s'emmarca dins del món de l'animació de personatges virtuals tridimensionals. Els personatges virtuals s'utilitzen en moltes aplicacions d'interacció home màquina, com els videojocs o els serious games, on es mouen i actuen de forma similar als humans dins de mons virtuals, i on són controlats pels usuaris per mitjà d'alguna interfície, o d'altra manera per sistemes intel·ligents. Reptes com aconseguir moviments fluides i comportament natural, controlar en temps real el moviment de manera intuïtiva i precisa, i inclús explorar la interacció dels personatges virtuals amb elements físics intel·ligents; són els que es treballen a continuació amb l'objectiu de contribuir en la generació de personatges virtuals responsius, expressius i tangibles.

La navegació dins dels mons virtuals fa ús de locomocions com caminar, córrer, etc. Per tal d'aconseguir el màxim de realisme, es capturen i reutilitzen moviments d'actors per animar els personatges virtuals. Així funcionen els motion graphs, una estructura que encapsula moviments i per mitjà de cerques dins d'aquesta, els concatena creant un flux continu. La síntesi de locomocions usant els motion graphs comporta un compromís entre el número de transicions entre les diferents locomocions, i la qualitat d'aquestes (similitud entre les postures a connectar). Per superar aquest inconvenient, proposem el mètode transicions progressives usant *Body Part Motion Graphs (BPMGs)*. Aquest mètode tracta els moviments de manera parcial, i genera transicions específiques i sincronitzades per cada part del cos (grup d'articulacions) dins d'una finestra temporal. Per tant, la connectivitat del sistema no està lligada a la similitud de postures globals, permetent trobar més punts de transició i de més qualitat, i sobretot incrementant la rapidesa en resposta i execució de les transicions respecte als motion graphs estàndards.

En segon lloc, més enllà d'aconseguir transicions ràpides i moviments fluides, els personatges virtuals també interaccionen entre ells i amb els usuaris parlant, creant la necessitat de generar moviments apropiats a la veu que reproduïxen. Els gestos formen part del llenguatge no verbal que acostuma a acompanyar a la veu. La credibilitat dels personatges virtuals parlants està lligada a la naturalitat dels seus moviments i a la concordança que aquests tenen amb la veu, sobretot amb l'entonació d'aquesta. Així doncs, hem realitzat l'anàlisi de la relació entre els gestos i la veu, i la conseqüent generació de gestos d'acord a la veu. S'han definit indicadors d'intensitat tant per gestos (*GSI, Gesture Strength Indicator*) com per la veu (*PSI, Pitch Strength Indicator*), i s'ha estudiat la relació entre la temporalitat i la intensitat de les dues senyals per establir unes normes de sincronia temporal i d'intensitat. Més endavant es presenta el *Gesture Motion Graph (GMG)*, que selecciona gestos adients a la veu d'entrada (text anotat a partir de la senyal de veu) i les regles esmentades. L'avaluació de les animacions resultants demostra la importància de relacionar la intensitat per generar animacions creïbles, més enllà de la sincronització temporal. Posteriorment, presentem un sistema de generació automàtica de gestos i animació facial a partir d'una senyal de veu: *BodySpeech*. Aquest sistema també inclou millores en l'animació, major reaprofitament de les dades d'entrada i sincronització més flexible, i noves funcionalitats com l'edició de l'estil les animacions de sortida. A més, l'animació facial també té en compte l'entonació de la veu.

Finalment, s'han traslladat els personatges virtuals dels entorns virtuals al món físic per tal d'explorar les possibilitats d'interacció amb objectes reals. Per aquest fi, presentem els *Avatars*, personatges virtuals que tenen representació tangible i que es visualitzen integrats en la realitat a través d'un dispositiu mòbil gràcies a la realitat augmentada. El control de l'animació es duu a terme per mitjà d'un objecte físic que l'usuari manipula, seleccionant i parametritzant les animacions, i que al mateix temps serveix com a suport per a la representació del personatge virtual. Posteriorment, s'ha explorat la interacció dels *Avatars* amb objectes físics intel·ligents com el robot social Pleo. El Pleo s'utilitza per a assistir a nens hospitalitzats en teràpia o simplement per jugar. Tot i els seus beneficis, hi ha una manca de relació emocional i interacció entre els nens i el Pleo que amb el temps fa que els nens perdin l'interès en ell. Així doncs, hem creat un escenari d'interacció mixt on el Vleo (un *Avatar* en forma de Pleo; element virtual) i el Pleo (element real) interactuen de manera natural. Aquest escenari s'ha testejat i els resultats conclouen que els *Avatars* milloren la motivació per jugar amb el Pleo, obrint un nou horitzó en la interacció dels personatges virtuals amb robots.



## 0.3 Resumen

Esta tesis se enmarca dentro del mundo de la animación de personajes virtuales tridimensionales. Los personajes virtuales se utilizan en muchas aplicaciones de interacción hombre máquina, como los videojuegos y los serious games, donde dentro de mundo virtuales se mueven y actúan de manera similar a los humanos, y son controlados por usuarios por mediante de alguna interfaz, o de otro modo, por sistemas inteligentes. Retos como conseguir movimientos fluidos y comportamiento natural, controlar en tiempo real el movimiento de manera intuitiva y precisa, y incluso explorar la interacción de los personajes virtuales con elementos físicos inteligentes; son los que se trabajan a continuación con el objetivo de contribuir en la generación de personajes virtuales responsivos, expresivos y tangibles.

La navegación dentro de los mundos virtuales hace uso de locomociones como andar, correr, etc. Para conseguir el máximo realismo, se capturan y reutilizan movimientos de actores para animar los personajes virtuales. Así funcionan los motion graphs, una estructura que encapsula movimientos y que por mediante búsquedas en ella, los concatena creando un flujo continuo. La síntesis de locomociones usando los motion graphs comporta un compromiso entre el número de transiciones entre las distintas locomociones, y la calidad de estas (similitud entre las posturas a conectar). Para superar este inconveniente, proponemos el método transiciones progresivas usando *Body Part Motion Graphs (BPMGs)*. Este método trata los movimientos de manera parcial, y genera transiciones específicas y sincronizadas para cada parte del cuerpo (grupo de articulaciones) dentro de una ventana temporal. Por lo tanto, la conectividad del sistema no está vinculada a la similitud de posturas globales, permitiendo encontrar más puntos de transición y de más calidad, incrementando la rapidez en respuesta y ejecución de las transiciones respecto a los motion graphs estándares.

En segundo lugar, más allá de conseguir transiciones rápidas y movimientos fluidos, los personajes virtuales también interaccionan entre ellos y con los usuarios hablando, creando la necesidad de generar movimientos apropiados a la voz que reproducen. Los gestos forman parte del lenguaje no verbal que acostumbra a acompañar a la voz. La credibilidad de los personajes virtuales parlantes está vinculada a la naturalidad de sus movimientos y a la concordancia que estos tienen con la voz, sobretodo con la entonación de esta. Así pues, hemos realizado el análisis de la relación entre los gestos y la voz, y la consecuente generación de gestos de acuerdo a la voz. Se han definido indicadores de intensidad tanto para gestos (*GSI, Gesture Strength Indicator*) como para la voz (*PSI, Pitch Strength Indicator*), y se ha estudiado la relación temporal y de intensidad para establecer unas reglas de sincronía temporal y de intensidad. Más adelante se presenta el *Gesture Motion Graph (GMG)*, que selecciona gestos adientes a la voz de entrada (texto etiquetado a partir de la señal de voz) y las normas mencionadas. La evaluación de las animaciones resultantes demuestra la importancia de relacionar la intensidad para generar animaciones creíbles, más allá de la sincronización temporal. Posteriormente, presentamos un sistema de generación automática de gestos y animación facial a partir de una señal de voz: *BodySpeech*. Este sistema también incluye mejoras en la animación, como un mayor aprovechamiento de los datos de entrada y una sincronización más flexible, y nuevas funcionalidades como la edición del estilo de las animaciones de salida. Además, la animación facial también tiene en cuenta la entonación de la voz.

Finalmente, se han trasladado los personajes virtuales de los entornos virtuales al mundo físico para explorar las posibilidades de interacción con objetos reales. Para este fin, presentamos los *AvatARs*, personajes virtuales que tienen representación tangible y que se visualizan integrados en la realidad a través de un dispositivo móvil gracias a la realidad aumentada. El control de la animación se lleva a cabo mediante un objeto físico que el usuario manipula, seleccionando y configurando las animaciones, y que a su vez sirve como soporte para la representación del personaje. Posteriormente, se ha explorado la interacción de los *AvatARs* con objetos físicos inteligentes como el robot Pleo. Pleo se utiliza para asistir a niños en terapia o simplemente para jugar. Todo y sus beneficios, hay una falta de relación emocional y interacción entre los niños y Pleo que con el tiempo hace que los niños pierdan el interés. Así pues, hemos creado un escenario de interacción mixto donde Vleo (*AvatAR* en forma de Pleo; virtual) y Pleo (real) interactúan de manera natural. Este escenario se ha testeado y los resultados concluyen que los *AvatARs* mejoran la motivación para jugar con Pleo, abriendo un nuevo horizonte en la interacción de los personajes virtuales con robots.





# Contents

0.1	Abstract . . . . .	5
0.2	Resum . . . . .	7
0.3	Resumen . . . . .	9
1	Motivation . . . . .	29
1.1	Responsive virtual characters . . . . .	34
1.2	Expressive virtual characters . . . . .	36
1.3	Tangible virtual characters . . . . .	37
1.4	Organization . . . . .	39
2	Notes on Character Animation . . . . .	41
2.1	Motion capture . . . . .	43
2.2	Motion synthesis . . . . .	45
2.3	Motion graphs . . . . .	47
2.3.1	Construction process . . . . .	49
2.3.2	Motion synthesis process . . . . .	51
3	Synthesis and Evaluation of Progressive Transitions in Locomotions using Body Part Motion Graphs . . . . .	53
3.1	Introduction . . . . .	55
3.2	Related work . . . . .	58
3.2.1	Construction process . . . . .	59
3.2.2	Labeling criteria . . . . .	59
3.2.3	Maneuverability . . . . .	59
3.2.4	Parameterization . . . . .	60
3.2.5	Evaluation . . . . .	61
3.3	Body Part Motion Graphs (BPMGs) . . . . .	61
3.3.1	Overview . . . . .	61
3.3.2	Body part specification . . . . .	62
3.3.3	Constructing a Body Part Motion Graph (BPMG) . . . . .	64
3.3.4	Synthesis of progressive transitions . . . . .	66
3.4	Evaluations I: Body Part Motion Graphs (BPMGs) against Standard Motion Graph (SMG) . . . . .	72
3.4.1	Dataset transitions . . . . .	73

3.4.2	Motion to motion transitions . . . . .	75
3.5	Evaluations II: Analyzing body part segmentation in BPMGs . . . . .	77
3.5.1	Connectivity versus split profiles . . . . .	78
3.5.2	Transition costs versus split profiles . . . . .	80
3.5.3	Optimizing motion to motion transitions . . . . .	82
4	Analysis and Synthesis of Body Language driven by Speech Emphasis . . . . .	83
4.1	Introduction . . . . .	85
4.2	Gestures . . . . .	89
4.3	Related work . . . . .	90
4.3.1	Modelling synchrony between gestures and speech . . . . .	90
4.3.2	Synthesis systems of gestures . . . . .	93
4.4	Synthesizing gestures adapted to speech emphasis . . . . .	95
4.4.1	Overview . . . . .	95
4.4.2	Gesture analysis . . . . .	97
4.4.3	Speech-Gesture correlation . . . . .	101
4.4.4	Gesture synthesis using a Gesture Motion Graph (GMG) . . . . .	105
4.4.5	Evaluations . . . . .	111
4.4.6	Results and discussion . . . . .	114
4.5	BodySpeech: Facial and gesture animations . . . . .	118
4.5.1	Setting up . . . . .	119
4.5.2	Facial and body gesture synthesis . . . . .	121
4.5.3	Editing gestures style . . . . .	125
4.5.4	Summary . . . . .	126
5	Tangible Control and Interaction with Robots of Augmented Reality Virtual Characters . . . . .	127
5.1	Introduction . . . . .	129
5.2	Tangible control of augmented reality virtual characters . . . . .	134
5.2.1	AvatAR (Augmented Reality virtual character) . . . . .	136
5.2.2	Interaction system . . . . .	138
5.2.3	Control design . . . . .	139
5.2.4	Implementation . . . . .	141
5.2.5	Discussion . . . . .	144
5.3	Interaction between AvatARs and social robots . . . . .	144
5.3.1	Pleo robot state of the art . . . . .	146
5.3.2	Interaction design . . . . .	148
5.3.3	Implementation . . . . .	153
5.3.4	Experiment design . . . . .	156
5.3.5	Results and discussion . . . . .	158

6	Conclusions, Limitations, and Future Directions	163
6.1	Animation using Body Part Motion Graphs (BPMGs)	165
6.2	Animation and interaction through Gesture Motion Graph (GMG) and BodySpeech	167
6.3	AvatARs animation and interaction with character robots	171
7	Research Outputs	175
7.1	Summary of contributions	177
7.2	Publications and additional material	180
8	Appendix	189
8.1	Speech analysis	191
8.1.1	Intonation in Spanish	191
8.1.2	Speech analysis	192
8.2	Automatic pitch accents detection	194
	Bibliography	197



# Glossary

3D 3-dimensional. 29, 33, 35, 37, 43, 44, 45, 46, 85, 88, 98, 118, 163, 171, 173

AI Artificial Intelligence. 32, 34, 47, 51, 130

AM Autosegmental and Metrical model. 191

APML Affective Presentation Markup Language. 93

AR Augmented Reality. 34, 37, 127, 129, 131, 136, 141, 145, 163, 171, 173, 178, 179

ARA\* Anytime Repairing A\*. 51

AU Action Units. 169

AvatAR Augmented Reality virtual character. 38, 127, 131, 132, 133, 134, 136, 137, 138, 139, 144, 145, 151, 153, 163, 171, 172, 173, 178

BEAT Behavior Expression Animation Toolkit. 93

BF Branching Factor. 168

BP Body Part. 35, 61, 62, 64, 66, 67, 68, 166

BPMGs Body Part Motion Graphs. 35, 53, 57, 58, 61, 64, 65, 66, 70, 72, 73, 74, 75, 78, 80, 87, 141, 163, 165, 166, 167, 168, 177

BVH BioVision Hierarchy. 126

CCR Comparison Category Rating. 113

CGI Computer-Generated Imagery. 29, 32

CMOS Comparative Mean Opinion Score. 113, 114, 116, 169

CRF's Conditional Random Fields. 94

dof degree-of-freedom. 134

DTW Dynamic Time Warping. 68

EV Evidence Variable. 192, 193

EV2 Evidence Variable 2. 192, 193, 194

FMDistance Fast Motion Distance. 99, 100, 121, 123, 168, 177

FMG Feature-based Motion Graph. 59, 165

fps frames per second. 59, 72, 74, 97

GMG Gesture Motion Graph. 36, 83, 87, 95, 105, 106, 107, 108, 111, 113, 118, 119, 122, 125, 163, 167, 168, 169, 170, 177

GP Gesture Phrase. 90, 95, 105, 106, 107, 108, 109, 110, 111, 119, 120, 121, 122, 123, 168, 169

GPLVMs Gaussian Process Latent Variable Models. 94

GPU Graphics Processing Unit. 32

GSI Gesture Strength Indicator. 87, 95, 99, 100, 101, 103, 105, 110, 168, 177

HCI Human Computer Interaction. 32, 85, 86, 134, 156, 163, 167, 171

HMD Head-Mounted Display. 173

HMG Hybrid Motion Graph. 59

HMM Hidden Markov Models. 94

HRI Human Robot Interaction. 38, 127, 133, 134, 145, 163, 171, 178

IK Inverse Kinematics. 65

IMG Interpolated Motion Graph. 59

IoT Internet of Things. 132

IPOCL Intent-based Partial Order Causal Link. 173

ISO International Organization for Standardization. 156

ITU International Telecommunication Union. 113

LM Local Maneuverability. 75, 165, 166

MCC Matthews Correlation Coefficient. 99, 100, 103, 193, 194

MmG Motion-motif Graph. 59

OS Operating System. 126

PAPT Pitch Accent Peak Time. 95, 102, 103, 108, 109, 168, 192

PCA Principle Component Analysis. 49, 134

Pegaso Primal Estimated sub-GrAdient SOLver. 100

PMG Parametric Motion Graph. 60

PSI Pitch Strength Indicator. 87, 95, 103, 105, 108, 110, 112, 168, 193, 194

SAPI Speech API. 118

SCC Strongly Connected Component. 51, 65, 107, 165, 166

SDK Software Development Kit. 169

SMG Standard Motion Graph. 35, 47, 53, 57, 58, 61, 62, 64, 65, 70, 72, 73, 74, 75, 78, 80, 87, 163, 165, 166, 177

SMO Sequential Minimal Optimization. 100

SP Split Profile. 57, 62, 64, 66, 67, 73, 77, 78, 80, 82, 166

SVM Support Vector Machine. 100

TBF Time Between Frames. 73, 74, 165, 166

TCP/IP Transmission Control Protocol / Internet Protocol. 153

ToBI Tones and Break Indices. 192

TRUE Testing platfoRm for mUltimedia Evaluation. 111

TUI Tangible User Interface/s. 34, 37, 127, 130, 132, 135, 137, 141, 163, 171, 178

WcMG Well-connected Motion Graph. 59, 65, 165





# List of Figures

1.1	From left to right, up to down: Buzz Lightyear (a) from "Toy Story" [Pixar, 1995], Carl Fredrickson (b) from "Up!" [Pixar, 2009], Rémy (c) from "Ratatouille" [Pixar, 2007], Po (d) from "Kung Fu Panda" [Dreamworks Studios, 2008], Shrek (e) [Dreamworks Studios, 2001] and Beowulf (f) [Robert Zemeckis, 2007]. . . . .	30
1.2	Graphs of emotional response against similarity to human appearance and movement from [Brenton et al., 2005] . . . . .	31
1.3	A capture from Ghostbusters: Sanctum of Slime [Atari, 2015]. . . . .	32
2.1	An actor dressed in a suit with markers being captured while performing a shoot with a bow. [The Capture Lab, 2015] . . . . .	43
2.2	From right to left, motion capture to virtual character. Actor movement is captured by inferring 3D markers positions through cameras, then these 3D points are mapped to a human model for finally drive a virtual character by retargeting. [Wikipedia Commons, 2015a] . . . . .	44
2.3	Two people are being tracked by a Kinect Xbox One. Note the obtained precision of skeletons although the captured subjects have their hands in touch. Although the obtained results, Kinect's user tracking suffers from occlusion problems because as a unique camera it has only one point of view rather than tracking systems where more cameras are involved. [fxguide, 2015] . . . . .	45
2.4	Videogame character move tree. As graph shows, there are defined paths to transition between motion clips establishing strict relations that could affect to character controllability. . . . .	46
2.5	A standard motion graph [Lee et al., 2002] from two motion capture clips. Vertices are poses from the motion capture data and edges are transitions between similar poses. Black edges are from the original motion capture and gray edges are additional edges found between similar poses. . . . .	47
2.6	This figure summarizes motion graph framework. From a motion capture database, we construct a motion graph by finding transition points, creating transitions between them and finally pruning the graph. After that, we search for motion and generate it in order to obtain the desired motion run by any user or artificial intelligence input. . . . .	48

2.7	Transition blending during correspondence region. Blending is done for all joints $j$ involved in the movement. We want to transition from motion $A$ (blue line) to $B$ (red line). So, we blend signal $O_j^A(t)$ in the region defined by $t_s^A$ and $t_e^A$ , which belongs to origin motion $A$ and joint $j$ , with signal $O_j^B(t)$ in the region defined by $t_s^B$ and $t_e^B$ , from target motion $B$ and the same joint. Note that both correspondence regions are drawn in dashed style. . . . .	50
2.8	Example of strongly connected components of a graph. Note that each node belonging to a group is directly or indirectly connected with each other of the same group. In this case, the final motion graph will be the component formed by nodes 0, 2, 3, 4 and 5. . . . .	51
3.1	Assassins Creed video game screenshot. Altaïr (main character) is walking through a roof. The animation is generated with a move tree. [Gamer Limit, 2015]. . . . .	55
3.2	Distance maps from different body parts. Dark zones contain good transition points rather than light zones, where similarities between postures are low. From top to down, and left to right: Trunk of the body, left upper side, right upper side and lower body. . . . .	57
3.3	BPMGs' transition. Each row represents a body part transition which has an origin frame from the origin motion, a path, and a target frame belonging to a target motion. Motion A (blue) is the origin motion and motion B (red) is the target motion. Each path is drawn with different color regions that describe from which motion clip are belonging frames in each transitions. . . . .	58
3.4	A simple example illustrating the construction steps of Well-connected Motion Graph (WcMG): (a) InterpolationStep, (b) Transition Creation Step and (c) Node Reduction Step. All original nodes are kept. Interpolated nodes and edges outside the best paths set are removed. Here only three discretizations of the interpolation weight are shown as illustration. In this implementation, they used nine, from 0.1 to 0.9 with a 0.1 increment. [Zhao and Safonova, 2009] . . . . .	60
3.5	System overview. From a motion capture database body part motion graphs are constructed after full-body motion is split in several partial motions. Then, user controls motion synthesis that require searching and synchronization process, and motion generation steps. . . . .	62
3.6	Human Kinematic Chains. Joints in red belong to head chain; yellow joints belong to upper right chain; green joints belong to trunk chain; brown joints belong to upper left chain; blue and purple joints belong to lower right chain and lower left chain respectively. . . . .	63

3.7	Walking phases. Walking is a locomotion in which feet alternate swing and stances phases. After both feet are in a stance phase, which means double support, one of them (left or right) starts with a swing phase while the other remains in stance phase. When this swing ends, both are again in double support. Then, the other foot performs a swing phase while now is the other that rests in a stance phase, and so on. . . . .	66
3.8	Body part transition paths. Origin frames are indicated by $i_n$ , target frames by $j_n$ , transition paths are denoted by $p_n$ , being $n$ from 1 to 4 as the number of body parts existing in this candidate transition. . . . .	67
3.9	Synchronization process. At the top, we have the origin frames, paths and target frames of body part transitions. Paths line style denotes how many frames are in each path which is different in all cases. If we compose that transition at that point the gap of frames of each body part does not match with path durations. So, at the bottom, paths are synchronized and fill the gaps exactly by time scaling them. . . . .	68
3.10	Transition scheme. Note that transition response is the elapsed time between the frame when the transition is requested and the beginning of the first body part transition. . . . .	69
3.11	Comparison between a BPMGs' and SMG's transitions. Three instants are captured in images (a), (b) and (c) which illustrate the transition of two virtual characters (skeletons) from walking to running. $t$ and $t_r$ denote the aggregated time and the time where the transition was requested, respectively. The left virtual character is driven by SMG, on the other hand, the right virtual character is driven by BPMGs. The original video can be found at <a href="http://goo.gl/820jDa">goo.gl/820jDa</a> . . . . .	71
3.12	Body parts. Joints in yellow color belong to lower-body; green joints belong to the trunk of the body; red and blue joints belong to right upper-body and left upper-body respectively. . . . .	72
3.13	Body part Split Profiles (SP). From 2 to 5 number of body parts are allowed in SPs. Two different segmentations with 3 and 4 parts are used. . . . .	77
3.14	BPMGs connectivity. This graph shows the connectivity of different split profiles in each similarity threshold value. Each colored bar represents a split profile. . . . .	79
3.15	BPMGs' transition cost. In horizontal axis is defined the similarity threshold and in vertical axis, the transition cost. Each line is related with a body part Split Profile (SP) which color changes along the color scale below indicating the connectivity of this SP from the similarity threshold value. . . . .	81
4.1	Steve Jobs speaking in an Apple keynote [Wikipedia Commons, 2015b]. Steve Jobs was a master nonverbal communicator that usually emphasized nearly every sentence with expansive and illustrative gestures that complemented his words [PROFITguide.com, 2015]. . . . .	85

4.2	Gesture streams synchronized with speech. At the bottom, a gesture stream synchronized with speech only in terms of time by matching apices (from gestures) with pitch accent peaks (from speech). In this case, there is no relation between PSI values and GSI values. At the top, a gesture stream synchronized in terms of time and intensity. In this example picture, we assume that high PSI values are related with high GSI values, and low PSI values with low GSI values. . . . .	88
4.3	Within Autosegmental-Metrical (AM) model [Pierrehumbert, 1980], pitch accents are combined with edge tones, which mark the beginnings and/or ends of prosodic phrases, to determine the intonational contour of a phrase. The need for pitch accents to be distinguished from edge tones can be seen in this contours in which the intonational events, annotated using the ToBI (Tones and Break Indices system) [Silverman et al., 1992b], are an $H^*$ pitch accent followed by an $L-$ phrase accent and a $H\%$ boundary tone. For more details in intonation the reader is referred to Appendix 8.1.1. [Wikipedia, 2015b]	92
4.4	Phases workflow. We part from synchronized speech signal and motion capture data that, in parallel, are analyzed in Speech Analysis 8.1.2 and Gesture Analysis 4.4.2 phases, respectively. Then, we obtain pitch accents times and strengths from speech signal, and strokes apices and strokes strengths from motion capture data. These parameters are include in a Speech-Motion Correlation 4.4.3 process which concludes with synchrony and intensity rules. Then, a gesture motion graph is created in Gesture Synthesis 4.4.4 phase from a labeled gesture database and restricted by the mentioned rules. Finally, we generate gesture animations driven by annotated speech which we have analyzed in Evaluations 4.4.5 phase. . . . .	96
4.5	Set-up of the recording session. An optical motion capture system with 24 infra-red cameras, a clip-on wireless microphone, and a videocamera (not in the image). The actor is wearing a black motion capture suit with reflective markers. . . . .	97
4.6	Example of the entire annotation of a particular video fragment in Anvil. The upper three windows are (from left to right): the command window, the motion capture viewer (in 3D) and the video. The latter two are manually synchronized. The bottom window is the complete video annotation. See the capture in different output formats at same time helps a lot in the annotation task. . . . .	100
4.7	Histogram with the distances in seconds between PAPT and apex times. Positive values indicate that the apex time comes before PAPT. Negative values indicate the opposite. The mean value is 0.1 seconds. Standard deviation is 0.2 seconds. . . . .	102
4.8	Scatter plot of the PSI and GSI values that come from aggressive (circles) and neutral (squares) styles. The linear polynomial straight line represents the correlation between PSI and GSI. The dashed lines represent the margin, experimentally determined, for the intensity rule. . . . .	104

4.9	Joints weights. Spines, neck, rightarm and leftarm, rightforearm and leftforearm have a weight of 1, and the rest of body joints have a weight of 0. . . . .	106
4.10	Edges generation. Edges in GMG connect consecutive GPs in original recordings ( $E_1$ ) as painted with continuous lines, and GPs with a posture similarity under a similarity threshold $st$ ( $E_2$ ) as painted with dashed lines. . . . .	107
4.11	Gesture phrase alignment with anchor points. Dashed lines denote transition motion segments and continuous lines gesture phrases. Apex from gesture phrases are marked with vertical lines. There are two timelines, one for each cue (gesture and speech). Anchor points define the gaps where gestures have to be fitted. . . . .	109
4.12	Snapshot of one step of the test using the online platform TRUE [Planet et al., 2008]. Every step contains two videos which have to be evaluated by the tester. . . . .	112
4.13	Five-point scale CMOS responses of users' preferences when comparing two videos generated with different methods (see Section 4.4.5 for methods descriptions). The diamond marker within the boxplots represents the distributions mean values. . . . .	115
4.14	Preference bars. They show the percentages of users ratings for each comparison. . . . .	115
4.15	System overview. The off-line stage is used to generate a motion graph (at the bottom) and a set of visemes (on top). Then, body and facial animation are obtained from a speech signal in the runtime stage (on the middle). . .	119
4.16	New graph screen. GMG can be parameterized by changing input databases (aggressive, neutral or full, which includes both previous), joint weights for posture similarity computation (dark blue box in the middle), similarity threshold that defines the existence of transitions between GP's (slider on top right). Once the GMG is generated, graph information is displayed at the bottom of the screen to know graph capabilities. . . . .	120
4.17	Emphatic visemes for /ah/ and /aw/ phonemes. . . . .	122
4.18	Gesture alignment. Body gestures are aligned with pitch accents by modifying their length. The goal is to match stroke apexes with pitch accents times ( $t$ ), taking into account anticipation times ( $T_a$ ). We consider two cases: only stroke (on the left) and gestural phrase (on the right). . . . .	123
4.19	Output viseme generation for pitch accents. Pitch accent (PA) strength is used to weight low emphatic, neutral and high emphatic visemes in the morphing process. In this example, higher emphatic visemes have more opened mouths. . . . .	124
4.20	Synthesis screen. On the upper-left corner, there are the buttons to select an audio and generate the animations. At the bottom, there are the configurable pitch accent detection parameters, and sliders for adjusting gestural or facial animation emphasis. . . . .	125

5.1	A screenshot from video game The Sims 3 University Life [EA, 2015]. As could be seen, virtual characters live in a virtual world that simulates a University campus, where they can walk, move on bicycle, and interrelate to each other as students. [JRC Gamecentrum, 2015]	129
5.2	On the left, virtual characters displayed thanks to the detection of card markers [Bimber and Raskar, 2006]. On the right, AR EdiBear application for Samsung Apps [Samsung Electronics Co., Ltd., 2015] which provides a touch-screen interface for controlling the virtual bear.	130
5.3	Screenshot of a two-player MonkeyBridge game. One of the players has already built a bridge for his character, which consists of virtual blocks (models with the dark wooden texture) and physical tiles (bright balsa-wood and stone cubes showing through the virtual objects). The monster character of the user standing in the middle has just hopped over from a virtual tile onto a physical platform. The user is holding the next building block in his hand. [Barakonyi et al., 2005]	131
5.4	AvatAR interaction system which is formed by a physical cube and a tablet device. A virtual character is displayed on the screen inside the cube.	132
5.5	Vleo and Pleo mixed environment is formed by a Pleo, a physical cube and a tablet device. Vleo appears on the tablet screen when the cube is detected.	134
5.6	Users can pull directly on the marionette strings with a second hand to gain a greater range of independent arm movement beyond the paddle control. [Mazalek and Nitsche, 2007]	135
5.7	Cube for controlling an AvatAR. Note that faces denote the type of locomotion being the observed face reproduced on the virtual character.	136
5.8	An augmented reality virtual character (AvatAR) is displayed inside the cube. Due to the camera is observing the idle face, AvatAR is in rest pose.	137
5.9	Interaction system. The system is formed by a tablet and a cube that user manipulates behind the tablet (inside the interaction area appointed in red slashed lines) enabling to see the virtual character displayed over the camera visualization of the cube.	138
5.10	Control scenarios.	140
5.11	WalkRun blend tree state which at same time is formed by Walk and Run blends tree states. Walk blend tree state is formed by WalkLeftShort, WalkLeftMedium, WalkLeftWide, Walk, WalkRightWide, WalkRightMedium and WalkRightShort. Run blend tree state is formed by RunLeftMedium, RunLeftWide, Run, RunRightWide and RunRightMedium.	142
5.12	Graphic design of the target cube. Note that icon images denote idle (left), walking (top and down) and running (right) motion clips.	143
5.13	Pleo "eating". When you buy Pleo, it comes with an electronic dna green leaf. If you call your Pleo and show it the leave, it will come and get close to the leaf and sniff it (actually at this moment it is watching the leaf with a small camera on his nose) then finally bite the leaf and a crunch sound will be heard.	146

5.14	Interaction flow system. User manipulates a physical cube that drives a virtual character. This virtual character acts on Pleo robot and user perceives the robot's reactions. . . . .	149
5.15	Vleo-Pleo interaction. Vleo makes Pleo happy when it is throwing kisses, so Pleo's emotional states shifts from angry to neutral and happy emotional state. Contrarily, Vleo makes Pleo angry when it is shouting so Pleo's emotional state shifts from happy to neutral and angry. . . . .	150
5.16	Cube gestures for controlling Vleo animation. . . . .	152
5.17	Communication system. Pleo controller communicates with the server through bluetooth technology, on the other hand, Vleo controller communicates with the server through wifi using TCP/IP protocol. . . . .	153
5.18	Top, the state machine of Vleo actions; bottom, the blend tree used to animate Vleo during shouting action. This last blend tree is formed by a head animation ('pleo_emprenyat_head' in the image), an animation that contain head and legs movement ('pleo_emprenyat_head_legs') and finally an animation that brings together the last two plus a tail movement ('pleo_emprenyat_full').	154
5.19	Physical cube built to drive Vleo. Their faces are illustrated with Pleo images that represents the actions that Vleo can perform. . . . .	155
5.20	The Pleo's connectivity has been enhanced adding a bluetooth board and antenna through the Pleo UART (Universal Asynchronous Receiver/Transmitter) system. The two points to access the UART are circled in red. . . . .	156
5.21	The facilitator explaining to a subject how to control Vleo. . . . .	157
5.22	Happy feedback perception and angry feedback perception. . . . .	159
5.23	Time taken to get Pleo happy and time taken to get Pleo angry. . . . .	160
5.24	Play preference . . . . .	161
6.1	On the left, a virtual mesh fitted on to a face thanks to the detection of key face points[behance, 2015]. On the right, a colored face of the virtual mesh whose shape denotes relax in the facial expression of the subject. . . . .	170
6.2	A depiction of a Microsoft HoloLens user navigating Windows Holographic, with an application window on the left, and the Holographic Start menu on the right. [Wikipedia, 2015a] . . . . .	174
7.1	Graphical abstract. . . . .	183
8.1	Example of the entire annotation of a particular audio fragment. The FD annotation tier contains the tags from the perceptual test. Sentence translation: "For the umpteenth time I am at the Medialab." . . . . .	193
8.2	Pitch accents detection. On top, there is the speech signal. Below, intensity (green) and pitch (blue) curves are displayed. In addition, PA strength and time are shown. At the bottom, the intonation is represented (using the ToBI system [Silverman et al., 1992a]) with the affected vowel. The image was created thanks to the Praat software [Boersma and Weenink, 2011]. . . . .	195





# List of Tables

3.1	Joints belonging to each Human Kinematic Chains. . . . .	64
3.2	Transition duration and response in BPMGs and SMG. All results are in frames (60 fps). Color yellow denotes that transitions can be achieved between two motion clips; color green for three motions and color magenta for four motion clips. . . . .	74
3.3	Motion to motion transition duration and response. Above, SMG results; Down, BPMGs' results. Motion clips $C_n$ placed on rows are origin motions, and motion clips $C_n$ placed on columns are target motions. Values in the top left are durations and values in the lower right corner are responses, both in frames. Motion clips are normal walking $C_1$ , running $C_2$ , long step walking $C_3$ and slow walking $C_4$ recorded at 60 fps. . . . .	76
3.4	Motion to motion optimal split profiles. In this matrix cells show the optimal split profile that connects pairs of origin motion clip (rows) and target motion clip (columns). . . . .	82
4.1	MCC results for all stroke classifications with 3 parameters: FMDistance, velocity range and maximum velocity. . . . .	101
4.2	Graph results varying similarity threshold from 0.3 to 1.0. Original size denotes the number of nodes initially belonging to GMG. Original BF defines the branching factor of graphs after deleting edges that overcome the specified similarity threshold. BF is computed as the number of edges divided by the number of nodes. SCC size describes the number of nodes belonging to the largest strongly connected component. Finally, SCC BF denotes the branching factor of the resulted SCC. . . . .	108
4.3	Possible answers for each step of the subjective test and its corresponding quality values. . . . .	114
4.4	List of all types of comparisons between animations. . . . .	114
4.5	15 phoneme categories. Each category maps to a single viseme. Symbols are codified with MRPA (Machine Readable Phonemic Alphabet). . . . .	121
5.1	Assessment questionnaire children perception . . . . .	158
8.1	MCC results for all variables. . . . .	194

8.2 MCC results for all the pitch accent classifications. . . . . 194

# Chapter 1

## Motivation

High quality virtual characters have starred in big Computer Animation or Computer-Generated Imagery (CGI) animation productions. From the premiere of "Toy Story" [Pixar, 1995], companies like Pixar [Pixar, 2015] and Dreamworks Studios [Dreamworks Animation Llc., 2015] have launched a myriad of films where 3-dimensional (3D) nonrealistic cartoon humans [Pixar, 2008][Pixar, 2009][Pixar, 2012], anthropomorphic animals [Pixar, 2003][Pixar, 2007][Dreamworks Studios, 2008][Dreamworks Studios, 2011] and fantastic creatures [Dreamworks Studios, 2001][Dreamworks Studios, 2004][Dreamworks Studios, 2007] are involved. These CGI virtual characters look stylized and polished (see Figure 1.1a-e) and they are perceived by viewers as live entities. In fact, these are incredibly convincing cartoon characters. Audiences can tell that these characters look and feel real because they are modeled precisely and are clearly animated. The audience empathizes with the virtual characters engaging them throughout the story. Nevertheless, they do not look or move entirely realistic nor human-like.

Other CGI movies like "Polar Express" [Robert Zemeckis, 2004] or "Beowulf" [Robert Zemeckis, 2007] recreate humans (see Figure 1.1f). In these cases the film critic exposes the lack of vitality of these virtual humans [Kenneth Turan, 2007b][Kenneth Turan, 2007a], noting that users tend to feel repulsion when virtual character's attempt to resemble more closely to human-like behavior and aesthetic as Uncanny Valley hypothesis states [Mori et al., 2012]. Motion artifacts that produce unnatural human movement can be one of the many reasons for this reaction. There is a clear desynchronization between specific parts of the body, for example: body movement with head gestures or body posture with facial expressions, as well as un-syncing between the emotions and the speech reproduced.



(a)



(b)



(c)



(d)



(e)



(f)

Figure 1.1: From left to right, up to down: Buzz Lightyear (a) from "Toy Story" [Pixar, 1995], Carl Fredricksen (b) from "Up!" [Pixar, 2009], Rémy (c) from "Ratatouille" [Pixar, 2007], Po (d) from "Kung Fu Panda" [Dreamworks Studios, 2008], Shrek (e) [Dreamworks Studios, 2001] and Beowulf (f) [Robert Zemeckis, 2007].

As a result of Brenton's work [Brenton et al., 2005] a set of heuristics for creators of animated characters may be produced to avoid the Uncanny hypothesis. They suggest that it is important to balance the realism of the various elements (graphics, animation, interaction) and to pay particular attention to certain aspects such as the appearance of the eyes [Geller, 2008]. In Figure 1.2 we can see two graphs indicating the emotional response of users against similarity to human appearance and movement. Specifically, the graphs show that the reaction of users to the movement of virtual characters is similar in shape to their reaction to the appearance of virtual characters, resulting a valley in both cases. These graphs, Figure 1.2, evidence that how virtual characters are animated affects more dominantly user's reaction than the appearance. Therefore, movement is significant for the perception of realism and sense of familiarity of the virtual characters.

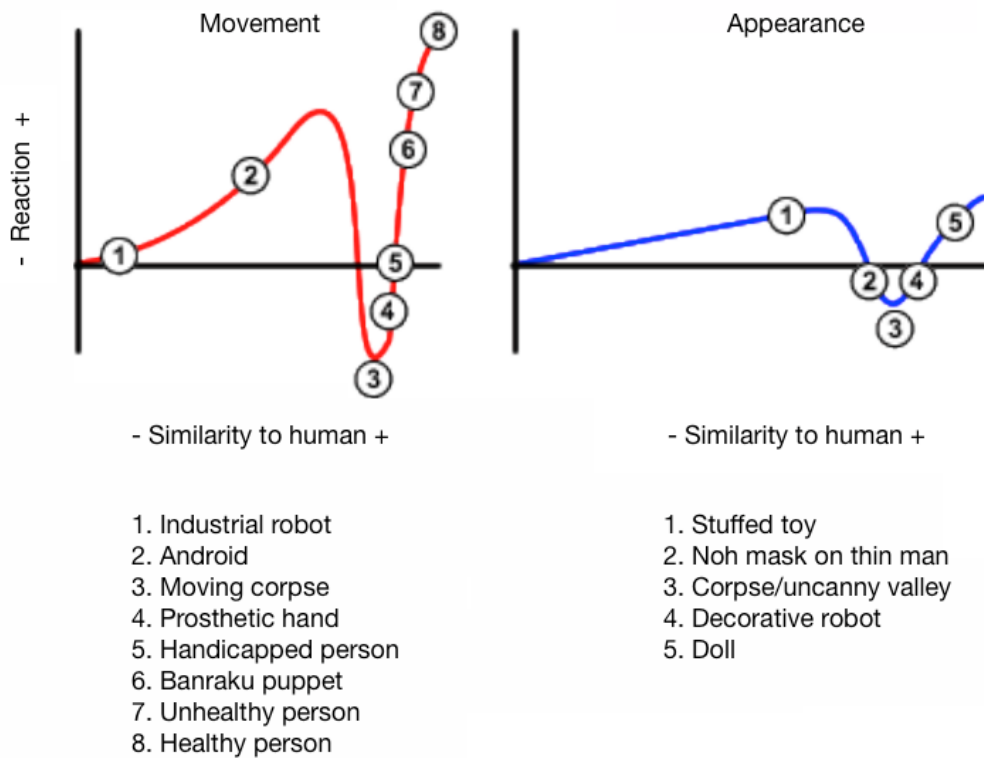


Figure 1.2: Graphs of emotional response against similarity to human appearance and movement from [Brenton et al., 2005]

In CGI movies, virtual characters' animation is carefully selected and cooked by expert animators. This occurs regardless how animations are created, either by keyframing, like it was done in the movies: "Toy Story" [Pixar, 1995], "The Incredibles" [Pixar, 2008], "Up!" [Pixar, 2009], "Nemo" [Pixar, 2003], "Kung Fu Panda" [Dreamworks Studios, 2008][Dreamworks Studios, 2011] or "Shrek" [Dreamworks Studios, 2001][Dreamworks Studios, 2004][Dreamworks Studios, 2007]; or captured by motion capture technology as in "Polar Express" [Robert Zemeckis, 2004] or "Beowulf" [Robert Zemeckis, 2007]. Animators invest a lot of time and resources to get the final animations that are used on the big screen. This fact is evidence of the difficulty in generating realistic and "natural" virtual characters. As in all audiovisual productions, viewers have a passive role because they only observe, therefore, no action can affect to the performance of the CGI films. On the contrary, in Human Computer Interaction (HCI) applications (e.g a video game), there is a real-time interaction between users, virtual characters, and indirectly with the virtual environment that are automatically and continuously synthesized. In these cases, the final animation depends on the user that controls the virtual character or any Artificial Intelligence (AI) system. Here the animation systems must decide which body posture is the most appropriate to show in each moment under the continuous demands of the user. The virtual environment can also affect the virtual characters' movement, for example when a virtual element collides with a virtual character it provokes the character to fall. This can explain why attempting realistic and "natural" animations worsen in interactive applications.



Figure 1.3: A capture from Ghostbusters: Sanctum of Slime [Atari, 2015].

Although later advances on Graphics Processing Unit (GPU) algorithms [Nguyen, 2007] allows to produce graphically realistic human-like virtual characters (see Figure 1.3 ), the illusion of being perceived as real disappears when virtual characters act, because their behavior and movements do not match their graphical realism. How virtual character animations and behaviors are being perceived by users play an important role in HCI applications, such as virtual worlds and video games. Virtual characters, part of the context, are the main mechanism used in an application. We also have also to take into account that users are capable of

perceiving subtle changes in motion [Johansson, 1973][Johansson, 1976] and can distinguish emotions [Atkinson et al., 2004][Blake and Shiffrar, 2007] [Atkinson et al., 2007], making the creation of "natural" and interactive virtual characters more challenging. Overall, there are several challenges to address in developing interactive virtual characters that Magnenat et al. [Magnenat-Thalmann and Thalmann, 2005] classifies as: a good representation of faces and bodies, a flexible motion control, a high-level behavior, emotional behavior, a realistic appearance, interacting with the virtual world, and interacting with the real world.

A good representation of faces and bodies is the primary step in dealing with animation data. This provides a starting point to generate appropriate animations that create "natural" interactive virtual characters. Then, automatic motion synthesis [Pejsa and Pandzic, 2010][Van Welbergen et al., 2010][Geijtenbeek and Pronost, 2012] is intended to provide flexible motion control over virtual characters, which means that virtual characters can freely perform the desired motion at the desired time. This allows the virtual characters' movements to properly synchronize with virtual world events and/or users input commands, maintaining the coherence and sense of control over the time. On the other hand, high-level behavior is focused on implementing decision algorithms for autonomous virtual characters [Maes, 1995][Cassell et al., 2000], usually named agents, that commonly interact directly with users programmed for an specific task. In addition, emotional behavior tries to convey expressivity through movement [Vinayagamoorthy et al., 2006]. This makes virtual characters more credible. So, providing high-level behavior and emotional behavior to virtual characters enables their use in interactive applications where complex interactions with other virtual characters, the virtual world and/or users are needed. For example, we consider a complex interaction a dialogue between a user and a virtual character that acts as an assistant whose purpose is to help the user accomplish a task.

Furthermore, realistic appearance [Kalra et al., 1998] including aspects such as the simulation of the hair [Ward et al., 2007], muscles [Teran et al., 2005] or cloth [Bridson et al., 2002] is considered basic in the creation of natural virtual characters. Within the interaction, we need to consider that virtual characters are part of an environment, real or virtual, they coexist with other elements. So, how virtual characters interrelate with their counterparts should be in a natural way [Abbott et al., 1999][Olfati-Saber, 2006][Thalmann, 2007], and, especially, how they precisely manipulate virtual/real objects. Yet, integrating virtual characters in the real world is a real challenge. These topics are in trend on ongoing research. Despite all the open discussions mentioned above, in this work we focus on flexible motion control, emotional behavior, and interacting with the real world. We consider these challenges more interesting, as the scientific community is far from optimum results or they are in an early stage. The approaches we have adopted for these challenges are explained below in depth.

The goal of this thesis is to create realistic and natural interactive 3D virtual characters ready to seamlessly interact with the virtual world and/or the real world, by being *responsive*, *expressive* and *tangible*. To achieve this goal, we first foster on improving the response of

locomotion synthesis methods for navigation purposes. Then, we enter on emotional behavior synthesis by studying and generating appropriated body language for speaking situations. Finally, we focus on how to move virtual characters in the real world by proposing an interaction system, aiming to explore new interaction opportunities. Further on, we work on topics related to how we take virtual characters from virtuality to reality causing the role of the user to become increasingly relevant. We bring virtual characters even closer to users in three levels:

- *Increasing character controllers response*: The main purpose of character controllers is to interpret and produce the desired animation according to any user (or AI) input. If response is high, user feels that the virtual character is moving properly to his indications, and as a consequence, immersion and engagement increase. Achieving smooth and natural motion streams while getting a high response from the character controller is a challenge in character animation which we approach by improving data-driven motion synthesis methods.
- *Adding emotions to virtual characters*: Expressive animations are those that incorporate emotions to virtual characters. Emotional attitude increase credibility in virtual characters due to it is a very characteristic aspect of humans. Thus, emotional behavior needs to be plug into virtual characters to increase empathy with users and create an emotional bond between them. In this work, we address emotional behavior by analyzing the relation between body gestures and speech, and constructing a data-driven gesture controller able to generate gestural behavior linked to the emotion of any input speech.
- *Moving virtual characters to real world*: Bringing virtual characters from virtuality to reality requires several steps. We mix Augmented Reality (AR) and Tangible User Interface/s (TUI) to enable this transfer that opens varied alternatives to interact with virtual characters. Thus, we create a tangible character controller that allows a flexible motion control over an augmented reality virtual character. Then, we propose several interaction scenarios which include different ways for controlling virtual character motion, and interaction between virtual characters and social robots through interactive storytelling.

## 1.1 Responsive virtual characters

Virtual worlds and video games are very rich and complex environments where a large number of animated elements coexist. These elements need to move purposefully to react convincingly to inputs, and being able to change their activities on demand. In case of virtual characters, aside from the need of motions to be clean and appear human, players must control characters smoothly, and not see jumps or jerks resulting from sudden movements



on the control. A character controller must be able to manage unanticipated (usually in realtime) demands which might be originated either by a player or by game AI. Therefore, having a flexible control with high response on the movement while preserving the naturalness of movement is necessary to steer the virtual characters in interactive applications.

In order to generate human motion there is a trade-off between control and realism. To ensure realism, movements are recorded in motion capture labs before being reproduced. This guarantees to preserve the subtleties of human motion into virtual characters [Pullen and Bregler, 2002]. From there, many data-driven techniques have been emerged to automatically generate human motion [Pejsa and Pandzic, 2010][Van Welbergen et al., 2010] allowing flexible control in realtime by creating powerful character controllers. Although there have been great results in terms of realism of the movements achieved, there is still room for improvement in the response of the character controllers.

One of the most known is motion graphs [Kovar et al., 2002][Lee et al., 2002][Arikan and Forsyth, 2002]. Standard Motion Graph (SMG) is a technique that embeds motion data from motion capture in a graph in order to connect similar motion units (e.g. motion frames, segments or clips) under a distance metric [van Basten and Egges, 2009] for transitioning between them. This approach enables the mobility of a virtual character in a virtual world. The response and transition execution depends on how the connections are established and which technique is used for transitioning. Not finding smooth transitions or finding too long time transitions weakens the maneuverability of virtual characters when its time to synthesize new motions. So, there is an evident lack of motion control which is constrained with the input animations and metric used to connect them. Thus, in the research of improving flexible control in SMG, we present a method that contributes with a more flexible structure that enable to find more transitions between a limited set of animations.

Progressive transitions using Body Part Motion Graphs (BPMGs) tackle the mentioned issues by changing the criteria of comparison and connection of motion frames, therefore the way to generate transitions between motion clips. Usually, in SMG motion frames are connected under a full-body similarity distance metric. By contrast, we break full-body motion in partial movements belonging to Body Part (BP), creating a set of motion graphs, each one belonging to a BP. Separating full-body motions allows to find more similarities between the input motion capture data. This means that each BPMGs searches for a transition that must accomplish the target motion, and BP transitions do not require to begin and finish exactly at the same time. Then, we reconstruct full-body motion by composing partial motion transitions progressively inside a window of time, and if required, we modify their lengths to synchronize them. Both operations are not noticeable for the user upon generating smooth and plausible motions which increases the opportunities to transition between the input motion data. Therefore, BPMGs method improves response in data-driven motion synthesis, specially in cases when the input motions clips that conforms the character controller are very different e.g. running against a slow walking.

## 1.2 Expressive virtual characters

A believable 3D virtual character should be provided with emotions in order to be as "natural", human-like, and to be perceived as the designer intended [Blake and Shiffar, 2007]. Human behavior is greatly influenced by emotions, intentions, attitudes and moods that vary depending on the context of action. However, interactive virtual characters tend to be a bit wooden with respect to their expected behavior [Vinayagamoorthy et al., 2006] since their realistic appearance suggest a realistic behavior. Endowing emotional behavior to virtual characters is difficult to achieve. In real life people interact with each other providing variety of behaviors according to their personalities and situations, in case of virtual characters, their expressiveness is limited. If virtual characters are not expressive enough it is possible that the credibility will disappear when the user interact with them.

In virtual worlds, expressive animation is more evident and necessary in certain situations. The use of autonomous (or non-autonomous) speaking virtual characters is wide. In some situations virtual characters speak to users in video games for example, when a character explains how to address a challenge. Other types of application can be such as for marketing purposes, or for customer attention and tools for communication. Transferring emotions to virtual characters in these contexts boosts and enhances a relationship with users. From character animation point of view, it is a very challenging task due to the complexity of body language which entails a lot of types of gestures. Once again, reusing recorded data is a good solution for quality results. But it is hard to develop a flexible gesture controller able to generate appropriate gestures to any speech due to the amount of meaning they carry. However, automatic generation of appropriate body language of virtual characters has been widely researched [Levine et al., 2009][Levine et al., 2010] [Kipp et al., 2007][Neff et al., 2008].

The plausibility of speaking virtual characters is linked to the appropriateness between speech and gestures [McNeill, 1985][Loehr, 2004][Leonard and Cummins, 2011]. So, we propose a system able to automatically generate body gestures from an input audio speech taking into account the emphasis relationship of both cues. We name it Gesture Motion Graph (GMG) as it is based on motion graphs technique. GMG selects and concatenates gestures from a graph according to an input speech, which defines the strength and the timing of the gestures to select. We relate them thanks to a set of rules extracted from a correlation study between dynamic parameters of motion and prosodic parameters of speech, these latter define their emphasis. We use both aggressive and neutral gestures and speeches in order to cover a wide spectrum of performances with different levels of emphasis. By selecting a gesture dataset from extreme emotions, aggressive and neutral, in terms of activation, it provides us a wide range of gestures, some of them, not exactly expressing the expected emotion since it is very difficult to maintain the level of activation throughout time. Thus, as a result, we are able to generate appropriated gesture animation from any input speech, no matter the associated emotion to it because we base the selection of gestures on the emphasis of the speech. However, the style performance of the generated gesture motions

from the GMG with the correlation rules is constrained to the input corpus of gestures and speeches, and so, to the actor that has been captured, serving as a baseline.

Expressive animation goes beyond the generation of realistic behavior for virtual characters. It can be used for customizing the behavior of virtual characters, pushing the creation of motion styles associated to the personalities of virtual characters. As described, we use rules for driving gestural animation from speech, however by modifying these rules it is possible to get variations in gesturing performance. These could slightly change the intention of the speech providing more or less credibility, but in any case, it allows the animator to have a flexible motion control over emotional behavior of virtual characters. To address this, we present BodySpeech, an automatic gesture animation system driven by an audio speech input, which includes facial and gesture animation synthesis that enables to create custom performances ready to be used in virtual world or video games, or any other interactive application.

### 1.3 Tangible virtual characters

Virtual environments are the habitat of virtual characters, but in recent times, virtuality is getting closer to reality [Poslad, 2011]. In the near future we expect that technology will be integrated into our surroundings so that we are not directly aware of its existence, but it can be used by users without much or none attention [Weiser, 1993]. Such situation describes ubiquitous computing which is a utopian that has been trying to achieve for years. Many researchers have focused on ubiquity to enhance the dialogue between users and computers (embedded computers), and the design of interfaces that make interaction with digital information more intuitive, simple, and natural. To overcome the barrier between the virtual and physical worlds visualization modes have emerged such as augmented reality and various other modes for interacting with digital information such as tangible user interfaces.

Augmented reality (AR) is when virtual objects overlaps with the real world. A camera on a device capture the physical reality as the screen displays the virtual objects onto that real world surrounding, then analyzes the components to seamlessly integrate the virtual content with the physical. AR enables the inclusion of digital content in the real world, and so, it opens opportunities for designing interactions between virtual characters and the physical world. Currently, virtual characters are used in AR applications for commercial and entertainment purposes. Virtual characters appear through the detection of some static markers, or not [Klein and Murray, 2007], but their behaviors are usually limited to reproduce some predefined animations in place, not allowing the option to move the virtual characters around the real world nor enabling control over the type of motion they perform. Although reality (marker) and virtuality (virtual character) are mixed, there is no seamless integration. A marker has a technical function which hinders the interaction experience of the user, it is simply a means to an end of positioning a virtual element in the real world. Due to this, we propose to use 3D markers, specifically, cubes with textured sides (regarded as current markers). In this manner we enhance the importance of the marker by giving it meaning

in the real world. Cubes are physical containers of digital information, in this case, virtual characters. From a narrative point of view this makes sense, a virtual character came alive when a device is pointing at the cube, on contrast, the virtual character lies hidden inside the cube (there is no virtual window). Furthermore, the marker is a physical object that allows us to design a tangible user interface (TUI) [Ishii and Ullmer, 1997] for controlling a virtual character. TUI allows to design interactions using everyday objects, breaking the traditional approach of interacting with computers and giving a strong sense of immediacy to the user. In our case, we base the interaction on manipulating the cube and gesturing with it. Thus, we also could address flexible motion control over augmented reality virtual characters with the aim of improving their interaction opportunities with the real world.

In this context, we present Augmented Reality virtual character (AvatAR). AvatARs are virtual characters visualized through mobile devices thanks to augmented reality and also controlled by tangible cubes. The manipulation of these tangible cubes allow to fine control the positioning and behavior of virtual characters. We relate cube faces to motion clips and so, the user could select the motion clip by orienting the cube to the camera. This converts the cube as a continuous character controller, where each orientation have a unique output motion clip resulting from the blending of reference motion clips. As regards to virtual characters positioning, it is directly mapped from the real position of the cube seen by the mobile device. This leads user to drive precisely virtual characters in real world, enabling them to have a similar maneuverability capacities as in video games or virtual worlds. AvatARs also have the features to be smart elements (tangibility and intelligence) allowing the interaction with other real objects. Smart or Intelligent environments are composed by physical and smart objects, that contain some digital information. These smart objects communicate with each other in order to execute collaborative tasks or simply for monitoring purposes. To put a virtual character in these environments is not a trivial task from the experience design point of view. In order to explore virtual character interaction with the physical world, smart objects should be as smart as virtual characters for the interaction to be just as meaningful and rich as in virtual worlds. In this context, character robots could be a good option. Thus, by joining AvatARs and character robots it is possible to create a meaningful mixed world where these types of entities can coexist.

Pleo [Inno Labs, 2014] is a social robot with the shape of a dinosaur. Among its common uses as a toy, it is used for reducing anxiety and stress to hospitalized children [Angulo et al., 2012][Larriba et al., 2015]. Despite its potential benefits, the experience of the children with Pleo is short and limited due to Pleo's little depth in expressive behaviors, in other words, it was difficult to understand the meaning of its movements. To improve the Pleo's experience to further enable long-term and richer interaction experiences are challenges in Human Robot Interaction (HRI). That is where AvatAR come in. Using an AvatAR ensures there will be a fluid and smooth interaction between the children and the virtual characters, since they can control the characters expecting a more precise and immediate response. So, we introduce Vleo, an AvatAR counterpart of Pleo, in the Pleo experience by putting them together in a mixed environment. With the aim to enable a direct relation between Pleo and Vleo, we link

the emotional behavior of Pleo to the emotional behavior of Vleo, which is controlled by the children through the cube, and also, it drives the performance of Pleo. To that effect, Pleo reacts to Vleo actions, and this occurs immediately and by exaggerating output animations and sounds. We use emotional states we consider that could be clearly identified such as happy and angry by interpreting character emotional behavior. This way we help children better interpret the Vleo-Pleo relationship understanding what is happening, and be able to enjoy playing in this scenario. This scenario clearly allows for myriad of applications to be designed in order to extend the time of the Pleo experience. More importantly, from this proposal the interaction between virtual and physical characters could be enhanced by taking advantage of puppetry, and the interactive storytelling advances and benefits.

## 1.4 Organization

This document is organized in 7 chapters: after the introduction, we begin with a background that covers the full manuscript and continuing with three chapters, each one assessing one of the challenges outlined above. Then, overall conclusions and future work. Finally the last chapter, describes the contributions and summarized publications.

The structure of the document is as follows:

- *Chapter 1: Introduction.* In this chapter we introduce the thesis by addressing the motivation and the research statement.
- *Chapter 2: Notes on Character Animation.* In the second chapter we are going to introduce motion capture, motion synthesis concepts, and techniques related with this work.
- *Chapter 3: Synthesis and Evaluation of Progressive Transitions using Body Part Motion Graphs.* In this third chapter we are going to propose a locomotion synthesis method that enables us to create a flexible and responsive character controller.
- *Chapter 4: Analysis and Synthesis of Body Language driven by Speech Emphasis.* In the fourth chapter we will face the generation of appropriated body language to speech. To that effect, we present a study of the correlation between gestures and speech, then, we propose a motion synthesis method that automatically generates gestural behavior of virtual characters from an arbitrary input speech.
- *Chapter 5: Tangible Control and Interaction with Robots of Augmented Reality Virtual Characters.* Here we present a concept that enables virtual characters to rest over the real world which are controlled by tangible objects and visualized through a hand-held device by an augmented reality. Then, we explore how to interactively control motion and the interaction of augmented reality virtual characters with character robots.
- *Chapter 6: Conclusions, Limitations, and Future Directions* In this chapter we expose the conclusions, limitations and future directions of this thesis.

- *Chapter 7: Research Outputs.* In this final chapter we highlight the contributions of this thesis into the field of character animation. Additionally, there is a collection of publications that have emerged from this research.

## Chapter 2

# Notes on Character Animation

### Chapter Abstract

Previously to get into this thesis, it is important to lay some foundations. In this chapter we are going to present some important concepts that will be used throughout the whole document. We first introduce motion capture concepts. Following, we relate motion synthesis processes from manual or semi-manual methods to automatic methods. Then, we review motion graphs technique, one of the most used in automatic data-motion synthesis domain. If the reader is familiarized on these topics, the author suggests to continue reading from Chapter 3.





## 2.1 Motion capture

Usually, 3D virtual characters are driven by motion capture data. Motion capture is the process of registering movements performed by a person, who is called actor, by some mechanisms or devices. Motion capture systems can be categorized in two main groups: markers motion capture [Bodenheimer et al., 1997] or markerless motion capture [Moeslund and Granum, 2001][Moeslund et al., 2006]. Capturing motion with markers implies using some kind of sensors or devices that the actor must wear to help cameras recognizing motion or to send data to a manager system for further treatment, as see in Figure 2.1. This fact can produce some distortion in motion because actor could be uncomfortable or simply constrained for this devices. In the other hand, markerless motion capture avoid these problems because is based on computer vision algorithms that deal with images from capturing cameras. Although, this type of motion capture usually has less precision than markers motion capture.



Figure 2.1: An actor dressed in a suit with markers being captured while performing a shoot with a bow. [The Capture Lab, 2015]

The most common motion capture system is optical motion capture which consists on capturing 3D point positions belonging to markers by using a set of strategically placed infrared cameras in a room [Bodenheimer et al., 1997]. Although the robustness and precision of these systems, a cleanup process usually is required. Oclusions provoke errors in time correspondences which have to be manually repaired. Once we have the motion capture data cleaned, it is transferred to a human model. This is known as skeletonization [Silaghi et al., 1998]. The human skeleton model is a kinematic tree of joints which its hierarchical structure imitates a human body [University of California, San Diego. CSE 169: Computer Animation, 2015]. The human anatomy is simplified in order to easily deal with data. Finally, motion capture data serves to drive some 3D model (see Figure 2.2). The

virtual character is represented with a mesh, and we must determine how the vertices of the polygons that form the mesh change when the skeleton moves. The process of building a mapping from skeleton to polygon vertices is referred to as skinning [James and Twigg, 2005].

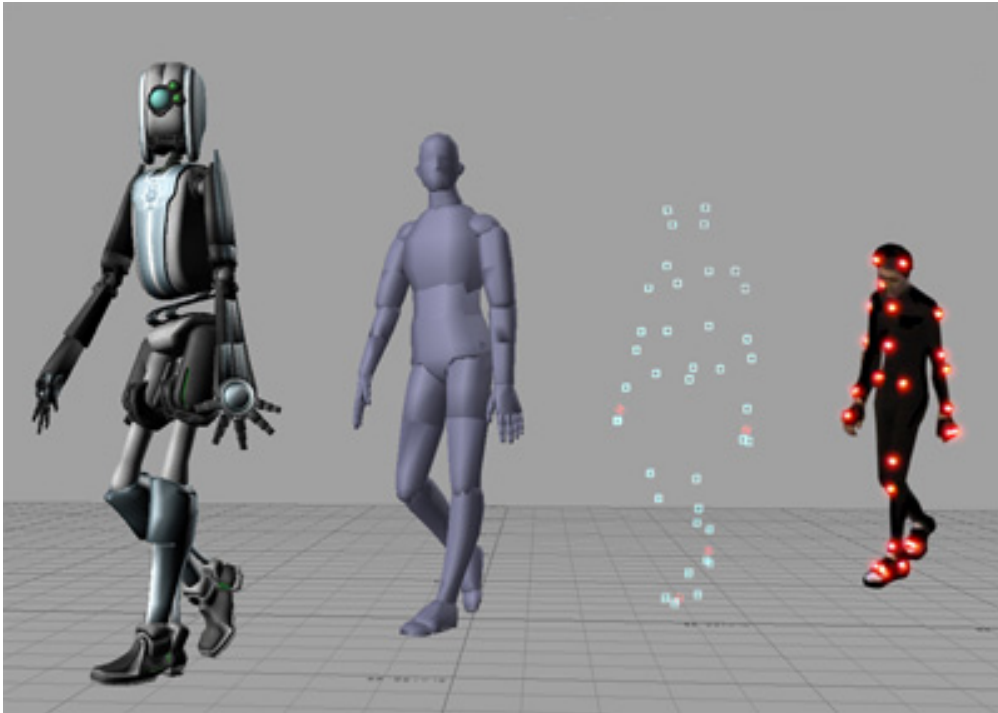


Figure 2.2: From right to left, motion capture to virtual character. Actor movement is captured by inferring 3D markers positions through cameras, then these 3D points are mapped to a human model for finally drive a virtual character by retargeting. [Wikipedia Commons, 2015a]

Markerless motion capture is based on how computer vision algorithms interprets captured images from cameras. There is a lot of literature about this type of algorithms. Moeslund has pursued an extensive review in [Moeslund and Granum, 2001] [Moeslund et al., 2006]. In addition to the algorithms is very important the type of the cameras and the set up. Markerless motion capture could be done using a camera, using stereographic cameras, 3D cameras or multiple cameras. Using only a unique camera avoid synchronization problems faced by systems with multiple cameras, although these last increase precision in tracking results. Kinect [Microsoft, 2015b] is the most know 3D camera for markerless motion capture. Kinect is distributed jointly with video game console Kinect One, thus, it is mainly used for playing with games, where an accurate tracking is not necessary to engage players in them. In Figure 2.3 can be seen how Kinect One estimates skeletons from two users.

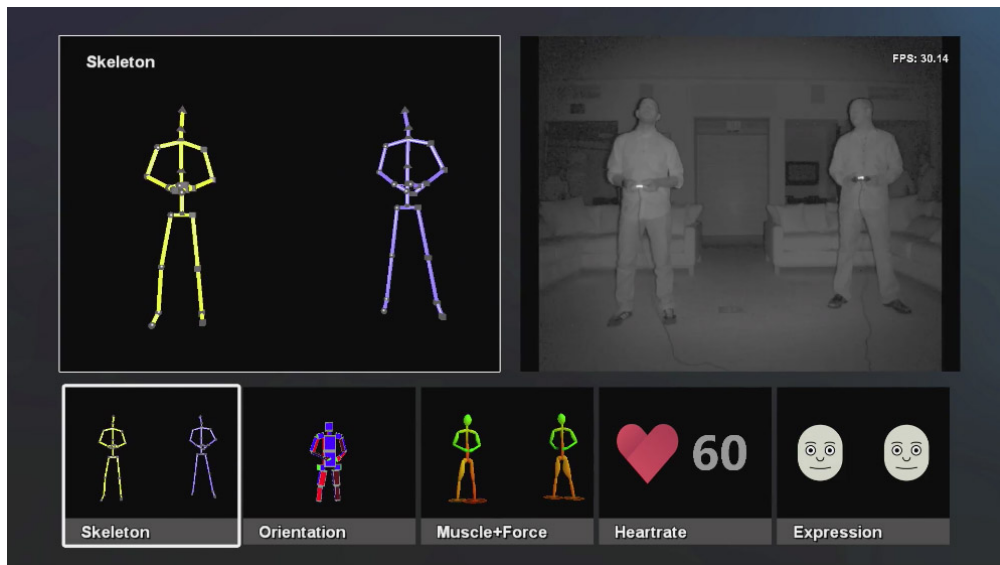


Figure 2.3: Two people are being tracked by a Kinect Xbox One. Note the obtained precision of skeletons although the captured subjects have their hands in touch. Although the obtained results, Kinect's user tracking suffers from occlusion problems because as a unique camera it has only one point of view rather than tracking systems where more cameras are involved. [fxguide, 2015]

## 2.2 Motion synthesis

Although motion capture is a popular way to generate realistic motions for interactive applications, its workflow is very tedious and specific for each type of motion and application. So, the idea of reuse motion clips [Bruderlin and Williams, 1995] [Witkin and Popovic, 1995] [Rose et al., 1998] [Rose et al., 1996] from motion libraries [Carnegie Mellon University Graphics Lab, 2004] [Müller et al., 2007] take importance in projects with low budgets or limited time, or simply to modify motion capture data.

Through the use of motion capture clips it is necessary creating transitions between them in order to generate a continuous motion stream. Concatenate motion segments using blending is one solution for creating smooth transitions between them. However, generating high quality transitions using blending is still difficult and involves significant manual labor. An animator often needs to go backs and forth to find a good transition point and good parameters for blending to obtain a pleasing transition. Also, an appropriate transition point is needed to achieve transitions without visual discontinuities. To that effect, animators use animation authoring softwares like 3D Studio Max [Autodesk, 2015a], MotionBuilder [Autodesk, 2015c] or Maya [Autodesk, 2015b]. These semi-manual techniques are used to create/adapt animations and to create transitions ready to drive virtual characters, the whole set being used to provide a character controller.

Characters in 3D video games are able to perform a range of movements. In order to encode these range of movements in a structure called move tree, also named motion tree. A move tree is hierarchical structure of nodes which must always terminate with animations at its extremities. Thanks to this organization, there is always one possible path between the root node (idle pose) and an animation node. Unfortunately, there is little literature explaining how were constructed these move trees [Menache, 1999][Tanco and Hilton, 2000]. Move trees are hand-made created by expert animators that spend a lot of hours on this task. They select the appropriated motion clips and generate by hand all transitions needed. Nowadays, Mecanim [Unity Technologies, 2014a] system belonging to Unity Game Engine [Unity Technologies, 2014b] permits to generate move trees very fast, thanks to an intuitive graphical user interface.

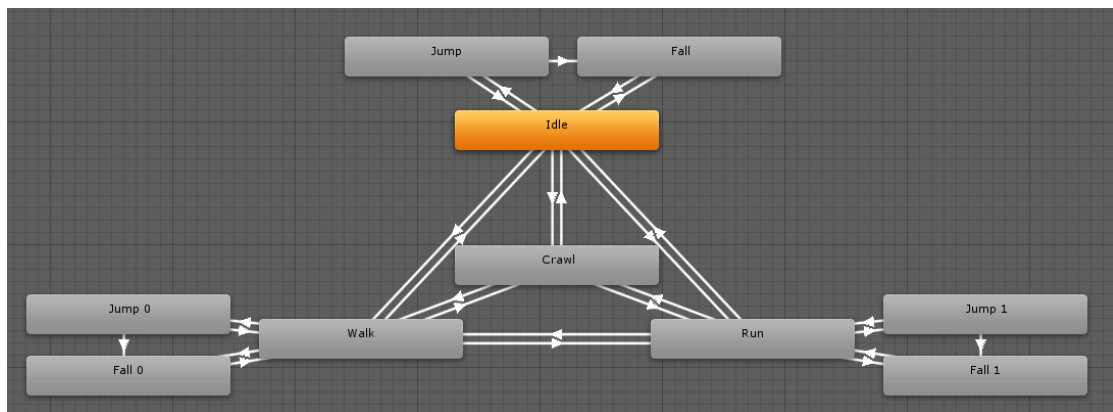


Figure 2.4: Videogame character move tree. As graph shows, there are defined paths to transition between motion clips establishing strict relations that could affect to character controllability.

Providing a set of possible actions to a virtual character is the aim of motion graphs [Kovar et al., 2002] and move trees. Blend trees are state machines where all transitions have been created before. As we can see in Figure 2.4 there are specific paths to achieve one motion clip from another, by jumping through intermediate motion clips in some cases. So, the topology of move trees defines which direct transitions between motion clips exist, however, usually there are not direct transitions between all motion clips due to the extensive time needed to create them. Moreover, all transitions are established in a precomputed frame of each clip. This means that if a character is walking and it has to start running, it may exist a delay on user request. Therefore, a challenge is to automatically create character controllers, with the same quality of move trees, able to transition between motion clips without time or motion order limitation, this is goal of motion graphs [Kovar et al., 2002]. This approach enables novice users to easily synthesize human motions for creating quality applications.

Motion graphs take part of discrete methods for automatic motion synthesis. In [Van Welbergen et al., 2010] there is a depth review in discrete methods. Moreover, a lot of research have been done in automatic motion synthesis beyond discrete methods. An extensive literature of automatic motion synthesis can be found in [Pejsa and Pandzic, 2010][Guo et al., 2015]. Automatic motion synthesis methods can be mainly grouped in three approaches: discrete methods, continuous methods [Geijtenbeek and Pronost, 2012] and hybrid methods [Zordan et al., 2005][Wang et al., 2014]. Discrete methods use motion data (usually from motion capture) to preserve raw quality rather than continuous, that use mathematical algorithms and physical simulations to drive characters. Obviously, discrete and continuous approaches have many strengths but some drawbacks, so hybrid methods combine them. Next, we only review standard motion graphs (SMG) method. Even though SMG method belongs to motion synthesis, we dedicated the next section in order to go into more details.

### 2.3 Motion graphs

Motion graphs emerged in 2002 [Kovar et al., 2002] [Lee et al., 2002] [Arikan and Forsyth, 2002] as a very promised technique for automatic synthesis of human motion. The idea is automatically assemble a graph from a motion capture database and using this graph to synthesize motions that meet specific goals from users or some AI system. To that effect, motion graphs reorganize motion data by finding correspondences between frames of former motion clips and as a consequence, concatenating them by transitions. In this manner, the quality of motion capture is preserved. Figure 2.5 shows a motion graph. As noted, three research papers marked the first original "motion graph" in character animation community. Although all these three papers have the same idea, they differed in some technical aspects [Rahim et al., 2009] such as on the distance metrics, transition scheme, method for getting sparsity in the graph and search scheme.

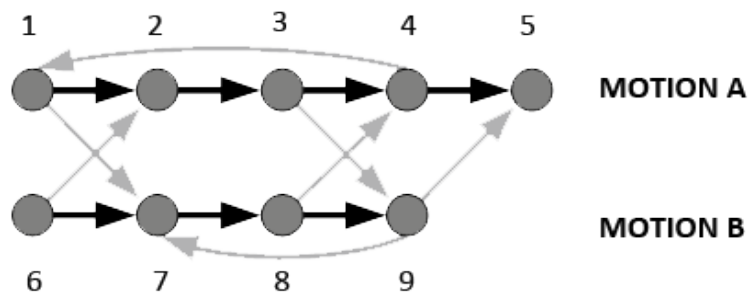


Figure 2.5: A standard motion graph [Lee et al., 2002] from two motion capture clips. Vertices are poses from the motion capture data and edges are transitions between similar poses. Black edges are from the original motion capture and gray edges are additional edges found between similar poses.

Although motion graphs have been implemented in different ways, there is a common framework (see Figure 2.6), and by this we refer to standard motion graph (SMG). This framework is mainly composed by two phases: motion graph construction and producing motion from a graph walk. The former phase also is composed by several stages: finding candidate transition points, create new transitions and pruning the graph. Motion graphs starts with a database of several motion streams. Firstly, it is necessary to split these streams in frames or small clips of several frames. Finding candidate transitions consists in detect similar poses between the pieces. The way we determine that two poses are similar is denoted by a distance metric. After that, new motions between transition points must be generated. Then, some post-processing to motion graphs have to be applied to ensure motion quality and controllability. Once motion graph is constructed, its time to proceed to the next phase. Second phase consists on searching for a motion and generate it. Both phases are explained in depth in Section 2.3.1 and Section 2.3.2, respectively.

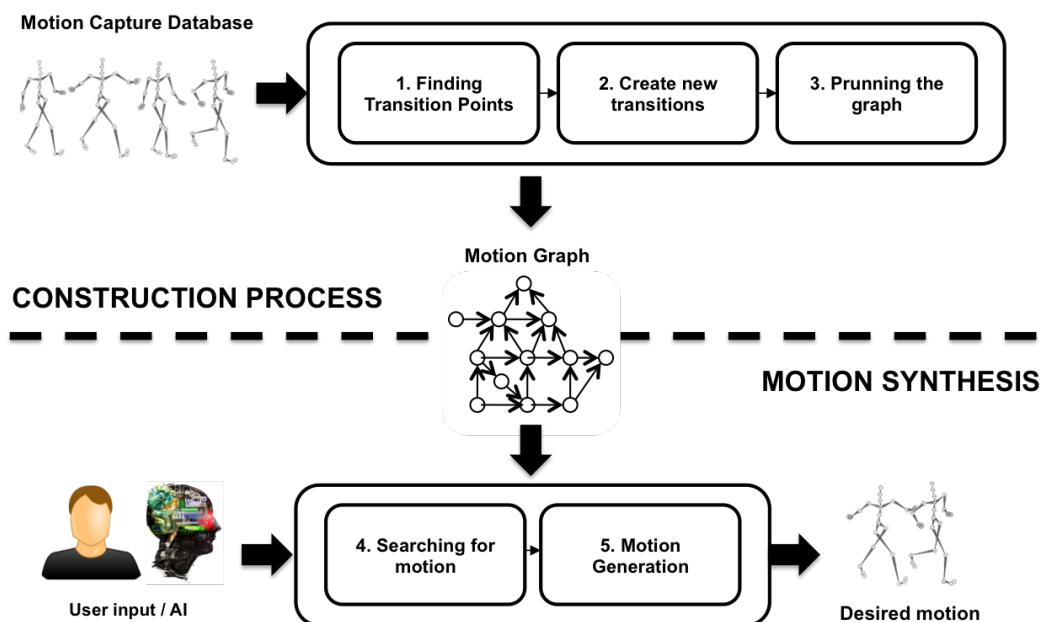


Figure 2.6: This figure summarizes motion graph framework. From a motion capture database, we construct a motion graph by finding transition points, creating transitions between them and finally pruning the graph. After that, we search for motion and generate it in order to obtain the desired motion run by any user or artificial intelligence input.

### 2.3.1 Construction process

#### Step 1. Finding candidate transition points

Distance metrics are used to detect good transition points. This process is based on comparing frames, or windows of frames, using a distance metric. A key factor in this comparison is the way that data is presented. Usually, motion capture data [Meredith and Maddock, 2001] specify root position and joint rotations of a hierarchical skeleton on each frame of the motion. Although, it could be represented by joint positions or in a particular way due to some processing step.

A discussion of the most common distance metrics (joint angles [Lee et al., 2002] [Arikan and Forsyth, 2002], point clouds [Kovar et al., 2002] and Principle Component Analysis (PCA) has been done in [van Basten and Egges, 2009]. In this section, we will only review joint angles distance metric. Joint angles distance metric is based on calculating the difference between joint angles values and optionally, a number of derivatives such as velocity and acceleration. Lee et al. [Lee et al., 2002] propose the following difference between frame  $a$  and  $b$ .

$$d(a, b) + vd(\dot{a}, \dot{b}) \quad (2.1)$$

where  $d(a, b)$  describes the differences of joint angles and global position, and the second term represents the differences of the joint velocities. Term  $v$  defines the influence of velocity. They used

$$d(a, b) = \|p_a - p_b\|^2 + \sum_{k \in \mathbb{J}} w_k \|\log(q_{b,k}^{-1} \cdot q_{a,k})\|^2 \quad (2.2)$$

The first term describes the squared difference between the global position of frame  $a$  and  $b$  whose positions are indicated by  $p_a$  and  $p_b$ . The second term describes the weighted sum of the orientations differences.  $q_{b,k}$  and  $q_{a,k}$  denote the unit quaternions describing the orientation of a joint  $k$  on frames  $a$  and  $b$ . Note that because of the Euclidean summing operation, the orientations need to be written in suitable format, in this case the exponential map [Grassia, 1998]. Arikan et al. [Arikan and Forsyth, 2002] uses a similar approach than [Lee et al., 2002] but with some differences. They compare animations by evaluating the joint position, joint velocity, torso velocity and torso acceleration. It is especially important to focus on  $w_k$  term. This term denotes the importance of joint  $k$  in the computation of joint angles distance metric, it is commonly known as joint weight. Joint weights are used to discard skeleton joints that are not relevant in postures comparison such as hands or fingers. At the same time, joint weights serve to emphasize some joints, for example, arms and rightupleg for searching transition points in locomotions.

### Step 2. Create new transitions

Once we have the candidate transition points calculated is time to determine which ones are chosen. A threshold value  $st$  (see Formula 2.3) is selected by the user for achieving smooth transitions, assuming that lower values connects similar poses and higher values different poses. This threshold is generally known as similarity threshold.

$$T(a, b) = \begin{cases} 0 & \text{if } d(a, b) \geq st \\ 1 & \text{if } d(a, b) < st \end{cases} \quad (2.3)$$

Then, only local maxima candidate transition points are usually selected. After this selection, transitions are created. Transition method is very important for getting plausible and realistic motions. Motion blending is used for generating transitions, although it is also used for motion editing. Motion blending is achieved by taking an arbitrary number of input motions and, based on particular requirements of the blending function, determining the output motion [Bryson, 2005][Feng et al., 2012]. In the case of transitions, we blend two correspondence regions from origin motion to the target motion like in Figure 2.7.

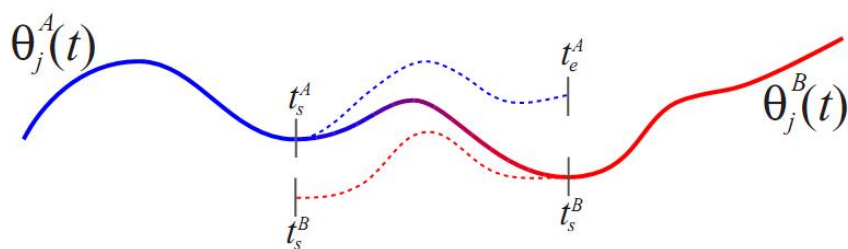


Figure 2.7: Transition blending during correspondence region. Blending is done for all joints  $j$  involved in the movement. We want to transition from motion  $A$  (blue line) to  $B$  (red line). So, we blend signal  $O_j^A(t)$  in the region defined by  $t_s^A$  and  $t_e^A$ , which belongs to origin motion  $A$  and joint  $j$ , with signal  $O_j^B(t)$  in the region defined by  $t_s^B$  and  $t_s^B$ , from target motion  $B$  and the same joint. Note that both correspondence regions are drawn in dashed style.

### Step 3. Pruning the graph

When using a motion graph for generating unlimited motion streams it is important to ensure that all nodes are connected. In the preceding steps of construction process, we are only finding and creating transitions without taking into account if some transition reach a node with some problems. There are three kind of problems [Arikan and Ikemoto, 2006] :



- Dead ends: A dead end is a node in the graph that no have outgoing edges. So, it is impossible to generate a continuous stream when a dead end is visited.
- Sinks: A sink is a zone in the graph that only permit to reach a subset of the nodes. If this happens, then it will be not possible to jump between different motions despite an unlimited motion stream could be generated.
- Problems with the labeling: One of the virtues of motion graphs is that is able to generate transitions between motions without motion order restriction rather than a move tree. But sometimes, it is important to take into account the type of motion, or the state of motion (i.e. flight phase of walking motion) to restrict connectivity. If it is not done, the construction process produces this issue.

So, in order to generate realistic and faithful motions to animations requirements it is necessary to eliminate the nodes which contain the mentioned problems. To do it, strongly connected components (Strongly Connected Component (SCC)) of motion graphs are computed [Gibbons, 1985]. SCC are subgraphs that ensures a path from each node to every other node as it is shown in Figure 2.8. So, final motion graph become the largest SCC of the initial motion graph.

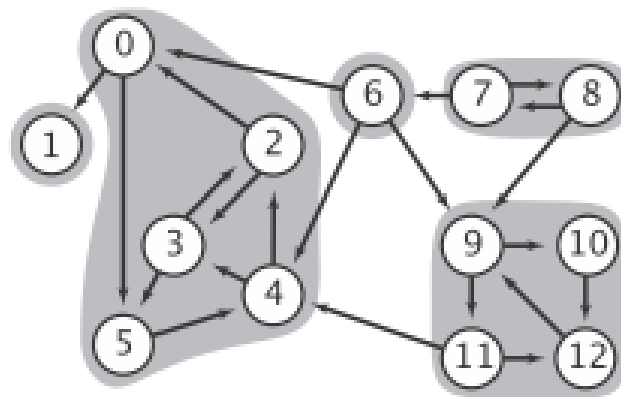


Figure 2.8: Example of strongly connected components of a graph. Note that each node belonging to a group is directly or indirectly connected with each other of the same group. In this case, the final motion graph will be the component formed by nodes 0, 2, 3, 4 and 5.

### 2.3.2 Motion synthesis process

#### Step 4. Searching for motion

Searching for motion consists on generating graph walks. A graph walk is a succession of nodes, each consecutive node connected by an edge. In order to find a graph walk, usually,

two nodes are determined, the origin and the target. The origin node is related with the current status of the animation, and the target node with user or some AI-task requirement. Then, a search criteria is applied to find a path between these two nodes. Cost functions serve as search criteria. So, minimizing cost functions and fulfilling animation requirements is the goal. There exist a variety of search algorithms [Gibbons, 1985] such as depth-first search, breadth-first search, Dijkstra's algorithm, Floyd-Warshall, A\*, Anytime Repairing A\* (ARA\*), etcetera.

#### Step 5. Motion generation

If we have a complete path from one node of the graph to an other, it is straightforward to generate the corresponding motion of the character along this path. The path of the graph walk consists of a sequence of transitions. But in our motion graph these transitions correspond to motion clips. This means that to generate the animation we have to join these clips together. The last thing to do is to translate the motion clips to the right position and orientation with an appropriate translation. These translations are multiplied from edge to edge when passing the path.

Until now we have pointed some concepts and algorithms which are used in this thesis. The whole work revolves around virtual characters, and the generation of its movements, behaviors and interaction. In Chapter 3 and Chapter 4, we use motion graphs technique (explained in Section 2.3) to generate virtual characters motion. On the other hand, in Chapter 5 we use move trees which are discussed in Section 2.2. In all chapters, the motion data used is from motion capture (explained in Section 2.1), even in Chapter 4 we have captured them.

## Chapter 3

# Synthesis and Evaluation of Progressive Transitions in Locomotions using Body Part Motion Graphs

### Chapter Abstract

Virtual worlds are digital environments where, among other things, virtual characters move around. In case of virtual characters, their movements should to be natural and smooth in order to be credible in view of the user. Get this along with the possibility to immediately change the type of locomotion is the first challenge we face. So, in this chapter we present an automatic locomotion synthesis method called progressive transitions using Body Part Motion Graphs (BPMGs). This method enables virtual characters to quickly transition between different types of locomotions, acting as a character controller. BPMGs is based on motion graphs technique which in turn is a data-driven motion synthesis method. Its improvement is focused on character controllability and maneuverability allowing faster transitioning between different locomotion. Briefly, it consists on creating several motion graphs each one belonging to a body part (a set of joints), for then generating transitions between locomotions by composing body part transitions. As a result, BPMGs are up to three times faster than SMG in terms of response time. So, we relate BPMGs construction and locomotion synthesis processes. Then, we perform two experimental analysis to proof BPMGs performance. The first one is focused on evaluating BPMGs capabilities against SMG in terms of connectivity of the input motion dataset, transition response, and transition duration of the resulting output animations. In these tests we only used a body part split profile (set of body parts). The second is focused on how different body part segmentations affect on the metrics presented in the first experimental analysis. This chapter is based on the published papers "Progressive transitions using body part motion graphs", SIGGRAPH Asia 2011 Posters, ACM [Fernández-Baena and Miralles, 2011] and "Fast response and quick progressive transitions using body part motion graphs", Eurographics 2012 Short Papers, The Eurographics Association [Fernández-Baena and Miralles, 2012].



### 3.1 Introduction

Locomotions permit the virtual characters can move freely in virtual worlds. Virtual characters have the capacity of perform different type of locomotions, like walking, running or jogging. Usually, these locomotions are extracted from motion capture clips in order to preserve realism of real movements from actors [Bruderlin and Williams, 1995][Witkin and Popovic, 1995][Rose et al., 1998][Rose et al., 1996], therefore, virtual characters are naturally animated for the sake of user experience.

In video games industry, motion clips are concatenated by transitions which are included in a move tree. A move tree [Menache, 1999][Tanco and Hilton, 2000] is a tree where each node is a motion clip and paths are transitions (see a detailed description in Section 2.2). Jumping from one node of the move tree to another, characters perform continuous motions while user input is satisfied (see Figure 3.1). Move trees produce high quality results, but transitions are immovable and it requires human labour to set up.



Figure 3.1: Assassins Creed video game screenshot. Altaïr (main character) is walking through a roof. The animation is generated with a move tree. [Gamer Limit, 2015].

Alternatively, automatic motion synthesis techniques such as motion graphs overcomes the limitations of move trees while maintaining high quality. Motion graphs [Kovar et al., 2002][Lee et al., 2002][Arikan and Forsyth, 2002] is a powerful method that permits generating unlimited motion streams which we have deeply reviewed in Section 2.3. Motion graphs take profit of automatically computing similar poses between motions capture data in order

to create transitions between it. A similarity threshold is used to discriminate between similar and non similar poses, as this threshold is determined empirically by forcing transitions to be smooth. So, concatenating motion clips in similar poses produces smooth motion streams. Moreover, the automatization of this process permits to generate more than one smooth transition between pairs of motion clips.

Despite the benefits of motion graphs, they suffer from other issues [Arikan and Ikemoto, 2006]. Finding similar poses is mandatory. Motion graphs are limited to input motion data, for that reason, the number of possible transitions depends on the similarity of poses of the input motion data. On this basis, smooth transitions between locomotions which contain few similar poses could not exist in a motion graph, otherwise, it is highly likely that transitioning between some locomotions could be achieved by using intermediate ones. Not finding smooth transitions or finding too long ones weakens the maneuverability of virtual characters when its time to synthesize new motions. So, improving character maneuverability is the aim of our method, especially where the input locomotions are significantly different.

Although, a lot of improvements in motion graphs have been presented in the past years [Kovar and Gleicher, 2004][Shin and Oh, 2006][Heck and Gleicher, 2007][McCann and Pollard, 2007][Safonova and Hodgins, 2007][Beaudoin et al., 2008] [Wang and Bodenheimer, 2008] [Zhao and Safonova, 2009][Ren et al., 2010] [Hu et al., 2011][Min and Chai, 2012][Mahmudi and Kallmann, 2013] (see Section 3.2 for detailed information), not vary the way of transitioning between postures which is done with full-body at same time, acting as a block. The cause of this is that transition points are also pursued taking into account the full-body. Two frames having the same posture, except for some joints, could overcome the similarity threshold even though the majority of the full-body is similar enough. This fact provokes some limitations in character controllability, it reduces the possibility of transitions between different kind of motions [Zhao and Safonova, 2009]. In contrast, we explore to transition unlinking one to other part of the body, as this meant that, a priori, more transitions can be found. Following, we discuss it.

In order to look for transitions, similarities between postures have to be computed. These similarities indicate which candidate transition points may exist in the input motion data. Distance maps is a common way to see posture similarities. In Figure 3.2 there are some examples of distance maps. Each pixel denotes the similarity between two postures extracted from a similarity matrix between all pairs of frames. Values are normalized between 0 and 1 to match them in a gray scale. Figure 3.2 shows four different distance maps, belonging each one to a group of joints, so similarity only takes into account joints from each group. As we can see, good transition points (dark zones) are located in close positions of each map, but not exactly in the same place. These differences show that if we select the same connections for all joints these will be not optimal. A full-body distance map is result of a combination of body part distance maps, so joining different body parts could hide partial motions behavior and force some parts to a bad transition. Therefore, we propose to divide the body into body parts for transitioning between motions.

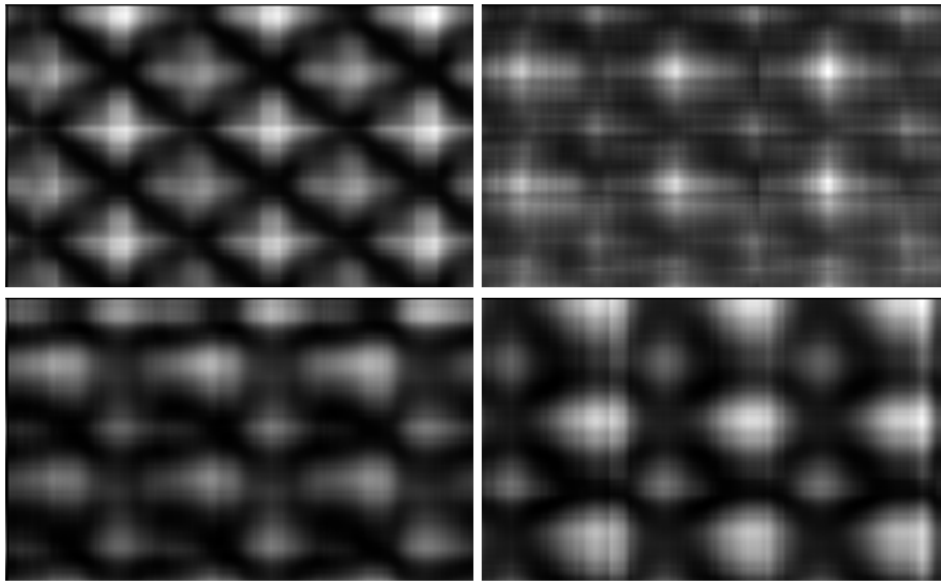


Figure 3.2: Distance maps from different body parts. Dark zones contain good transition points rather than light zones, where similarities between postures are low. From top to down, and left to right: Trunk of the body, left upper side, right upper side and lower body.

After realizing that full-body transitions are not optimal for all body parts, we propose Progressive Transitions using Body Part Motion Graphs (BPMGs). BPMGs consists on create a motion graph for every defined body part, each one composed by a set of joints. Then, body part transitions are generated independently of one another. After generating them, they must be synchronized. In Figure 3.3 there is an example transition computed with BPMGs using four body parts. We can see that the transition paths between different body parts contains frames from different motion clips in different times. So, split body leads each part to transition without constraints imposed for other body parts.

We demonstrate that dataset connectivity increases with our method, but also it is important to note that embedding partial motions in individual structures increase richness and variability of body part transitions. This is thanks of to each body part is free to perform its transition. Although [Lee et al., 2010] argued that discrete methods such as motion graph are slower in response (by a factor of two) than continuous one, we will show that BPMGs are up to three times faster than Standard Motion Graphs (SMG) in terms of response time. Moreover, BPMGs overcomes SMG transition duration. More details are exposed in Section 3.4.

Although the benefits we obtain splitting body, not all segmentation possibilities affect in the same way to our method. We have used different body part segmentations which are defined as Split Profile (SP). So, connectivity and transition cost computations have

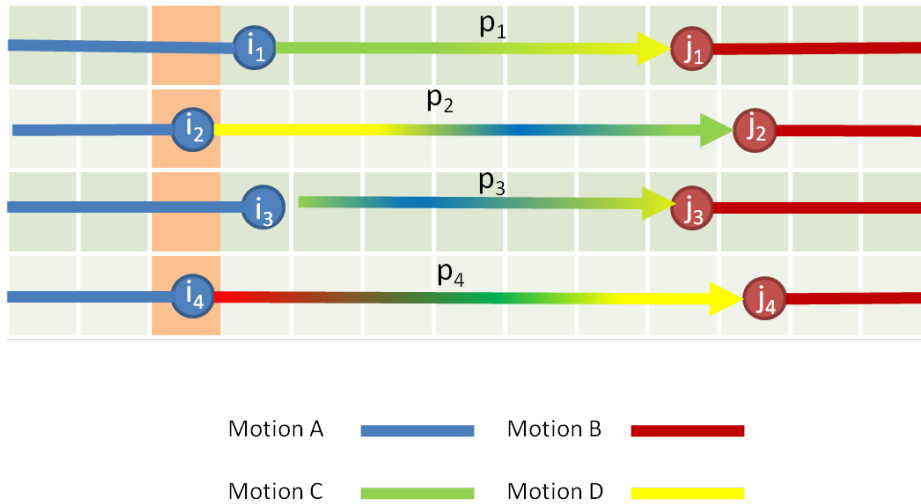


Figure 3.3: BPMGs' transition. Each row represents a body part transition which has an origin frame from the origin motion, a path, and a target frame belonging to a target motion. Motion A (blue) is the origin motion and motion B (red) is the target motion. Each path is drawn with different color regions that describe from which motion clip are belonging frames in each transitions.

been addressed in order to identify which SP improve the performance with our dataset in Section 3.5. Results show that some body parts as upper right body and upper left body are better in different joint groups, because joining them difficult to find good transitions in it. Besides finding the best split profile to use with our dataset we have studied motion to motion performance to check if global results fits with local ones. As we expect, motion to motion transitions also have an appropriate split profile that brings out the best of our method.

In conclusion, we present a variation of how to transition in SMG which implies the creation of a new structure and different data processing. As we have said, body segmentation is the starting point to improve aspects such as controllability and maneuverability with a limited set of input motions. Even beyond this, several optimizations can be carried out in order to improve further transition performance and motion quality that we will discuss in 6.1.

## 3.2 Related work

Previously to explain BPMGs method, we summarize motion graphs state of the art in order to discuss BPMGs contribution in this field. Recall that in Section 2.3 there is an explanation of standard motion graphs. A variety of graph-based motion synthesis systems have been developed after motion graphs appeared. Each variation is focused on improve one or more



aspects of the motion graph framework. The following is a brief explanation of the most relevant works, from our point of view, catalogued by construction process, labeling criteria, maneuverability, parameterization and evaluation.

### 3.2.1 Construction process

Motion graphs construction process is linked with the computation of distance metric between all frames of the input motion dataset. This causes a time-consuming phase because of the large number of frames. Considering a dataset of 60 frames per second (fps), the number of similarities to compute is 3540 ( $60 * 60 - 60$ ). In order to improve this issue Hu et al. [Hu et al., 2011] improve construction process by sampling uniformly distance maps to compute similarity between frames. Another approach is presented in [Mahmudi and Kallmann, 2013]. Mahmudi et al. [Mahmudi and Kallmann, 2013] presented Feature-based Motion Graph (FMG) which consists on choosing transition points based on relevant features which reduces graph construction time and leads to improve search performances. In this manner, they also avoid to compute all frames of the dataset.

### 3.2.2 Labeling criteria

Using the naive motion graph version it is possible to connect individual motion sequences. However, understanding the underlying structure of these motion sequences is difficult for the user and do not help in motion synthesis. In order to solve this lack, Beaudoin et al. [Beaudoin et al., 2008] present Motion-motif Graph (MmG). Its improvement is based on clustering similar motions in order to create an understandable structure to the contents. Similar sequences are grouped allowing a motif creation. Their representation is mainly designed for analyzing and visualizing the contents and connectivity rather than motion synthesis and control targeted, which is the case of Motion Graphs ++ [Min and Chai, 2012]. Motion Graphs ++ provide a highly structured, contact aware and semantic embedding for motion analysis and synthesis. The most relevant contribution in semantics is that they permit semantic motion editing based on semantic motion analysis such as modify a motion by adding/or deleting "walking" verb.

### 3.2.3 Maneuverability

The fact that motion graphs use a limited set of motions reduces the maneuverability of characters driven by them. Maneuverability is established by the number of transitions in the graph. One approach is to split motion data in order to get small fragments to concatenate. This is the case of [McCann and Pollard, 2007]. On the other hand, Safonova and Hodgins [Safonova and Hodgins, 2007] solve maneuverability problem in a different manner. They present Interpolated Motion Graph (IMG) in order to find more transitions. This graph is created from the product of two identical motion graph. In a similar way than [Safonova and Hodgins, 2007], Zhao and Safonova [Zhao and Safonova, 2009] propose Well-connected Motion Graphs (see Figure 3.4). Well-connected Motion Graph (WcMG) exploit the potential

of interpolation to increase connections in a graph and to achieve smoother transitions. More recently, Ren et al. [Ren et al., 2010] introduce an optimization-based graph combining continuous constrained optimization with graph-based motion synthesis. In this manner, they obtain a high maneuverability because they ensure transitions between a predefined set of nodes. Improve the connectivity and maneuverability of motion graphs is also our goal. Later, Xing et al. propose Hybrid Motion Graph (HMG) [Xing et al., 2014] for creating character animations, which enhances the graph-based structural control by motion field representations for efficient motion synthesis of diverse and interactive character animations.

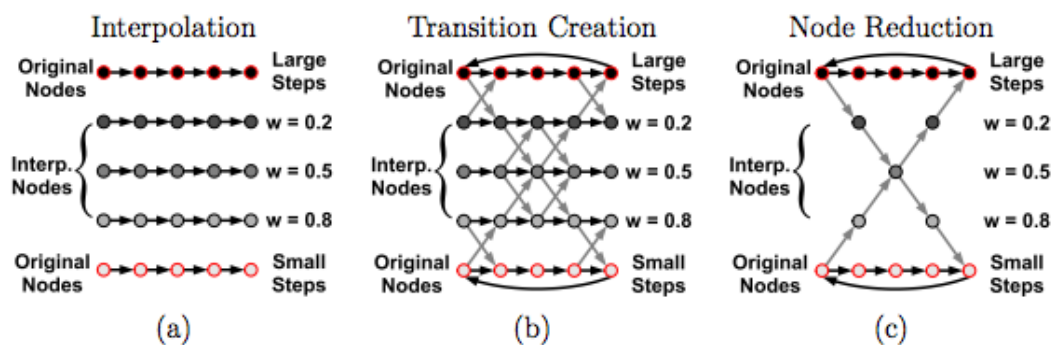


Figure 3.4: A simple example illustrating the construction steps of Well-connected Motion Graph (WcMG): (a) Interpolation Step, (b) Transition Creation Step and (c) Node Reduction Step. All original nodes are kept. Interpolated nodes and edges outside the best paths set are removed. Here only three discretizations of the interpolation weight are shown as illustration. In this implementation, they used nine, from 0.1 to 0.9 with a 0.1 increment. [Zhao and Safonova, 2009]

### 3.2.4 Parameterization

As we have already said, motion graphs reuse input motion data. This limits output animations to the motion dataset used to construct the motion graph. So, to obtain more flexibility in motion synthesis using motion graph is another research line. Kovar and Gleicher [Kovar and Gleicher, 2004] improves scalability in motion graph searches by providing automated methods for identifying logically similar motions in a data set and using them to build a continuous and intuitively parameterized space of motions. Parameterize motions is also the goal of [Shin and Oh, 2006][Heck and Gleicher, 2007]. Shin et al.[Shin and Oh, 2006] achieve new motions by blending similar motion segments that are edges in the graph, on the other hand, Heck and Gleicher[Heck and Gleicher, 2007] use linear blending for transitioning between motion clips using a structure called Parametric Motion Graph (PMG).

### 3.2.5 Evaluation

Motion graphs share the same goal of animating characters, but comparison between them is difficult due to their differences. Reitsma and Pollard [Reitsma and Pollard, 2007] propose some methods for comparing performances of graph-based methods. This comparison is conducted based on a set of tasks and environment capabilities. Motion graph performance is also studied in [Wang and Bodenheimer, 2008]. Wang and Bodenheimer made an exhaustive study of transition length in linear transitions which is related with transition response. They focused on determining which frame length to use in order to achieve pleasant animations. Also, [Zhao and Safonova, 2009] evaluates the performance of well-connected motion graph presenting a set of useful measures.

BPMGs method introduce some variations in construction process and evaluation respect to SMG, but its contribution is mainly focused on maneuverability. The way that BPMGs increases SMG maneuverability is encoding motions in several motion graphs. For that purpose, virtual characters' body is segmented in several body parts in order to create a motion graph for each one. This enables to find more posture similarities. This is the key. Similarities can be found due to only joints that belongs to a body part affects to distance metric. Thus, some difficulties appear for generating output motions. Body part transition points do not exactly coincide within them in terms of time, neither body part transitions lengths. So, we deal with this issue searching transitions for each part of the body in a time window. Then, body part transitions have to be synchronized to generate coherent full-body transitions. All the details are described later.

## 3.3 Body Part Motion Graphs (BPMGs)

In this section, we explain the Body Part Motion Graphs (BPMGs) method. Firstly, we take an overview of the method in order to clarify the steps that makes it up. Then, we relate construction process that leads to generate BPMGs from a set of input motion clips. Finally, it is described how we get the final motion that fulfill the input requirements.

### 3.3.1 Overview

As in standard motion graphs, our method consist in two phases (Figure 3.5). The first phase is construction process, where BPMGs are created from a set of motion clips. So, for each defined body part, a motion graph will be constructed following the known construction steps of detecting candidate transitions, transitions creation and pruning. However, the difference between SMG construction process is that in BPMGs, each BP motion graph uses partial motions, as a consequence, a previous step (Motion Split) for defining the body parts have been added.

Then, in order to generate transitions between motion clips we have to search for motion in body part motion graphs and generate the final motion. Motions in our system are

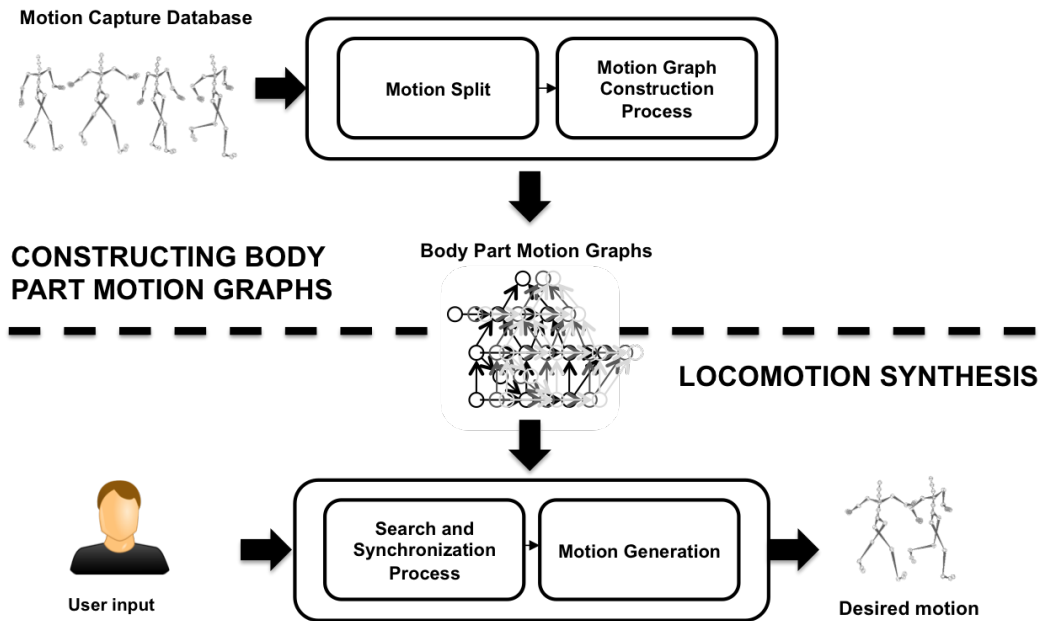


Figure 3.5: System overview. From a motion capture database body part motion graphs are constructed after full-body motion is split in several partial motions. Then, user controls motion synthesis that require searching and synchronization process, and motion generation steps.

distributed in several graph structures belonging to BP. Due that, to synthesize transitions we have to individually look for BP transitions in each graph and then compose them taking into account temporal constraints. The aim of our transition method is to allow body parts transition without full-body unity, but finally all transitions have to get at a full-body frame. In this way, the unique phase of a locomotion stream that not uses original full-body frames is transition, because a mixture of partial frames is used.

### 3.3.2 Body part specification

The objective of this step is to define which partial motions are used to create body part motion graphs. In case of full-body motions, we can define a motion like  $M(t) = P_{root}\bar{R}_{joints}$ , where  $P_{root}$  defines the global position and  $\bar{R}_{joints}$  the joint orientations along the time. Usually, root is appointed as hips joint, and the joints are the rest. In a similar way, we can define a partial motion as like  $M_p(t) = P_{root_n}\bar{R}_{joints_n}$ , where  $n$  indicates the BP. So, previously to split full-body motion into partial motions it is mandatory to decide which joints are included in each one and which are its roots.

Selecting a motion split could be done by several criteria. We name the specification of Body Parts (BP) as body part split profile (SP). SMG could be considered as a unique body

part, and pursuing body parts in an extreme, it could be generated as many body parts as joints in the skeleton. In our case, we based the selection of body parts by grouping kinematic chains of human skeleton. Kinematic chains refer to a set of rigid bodies (in our case bones) connected by links (in our case joints). Like a usual chain, the bones are constrained by their connections to other bones.

As Vukobratovic states in his book "Introduction to Robotics" [Vukobratovic, 2012], human skeleton can be considered as a complex kinematic chain. The kinematic chains connected to the support are basic chains, while the chains connected to them, but not by means of the support, are satellite chains (satellites). The procedure of separating one complex chain into a number of simple ones is hereby practically defined. Thus, the notion of an independent kinematic may be defined as a kinematic chain is said to be independent if its motion is independent of the satellite chains, with respect to the support, e.g. if its last members are connected to the support. This means that a basic chain is independent. In that way in each complex linked mechanism, such as human body, only one autonomous chain is obtained, remaining chains being satellites to the autonomous one.

So, in a human skeleton it is possible to define the illustrated independent kinematic chains of Figure 3.6. Thus, taking the model of a human, it is possible to isolate the chains of 'legs', 'body', 'arms' and 'neck'; here the chain consisting of 'legs' is independent since the chains of 'body' and 'arms' impose no kinematic constraint on its, and viceversa. Trunk chain (illustrated in green in Figure 3.6) is the basic chain while the others are satellite ones. Therefore, we define the human kinematic chains shown in Table 3.1.

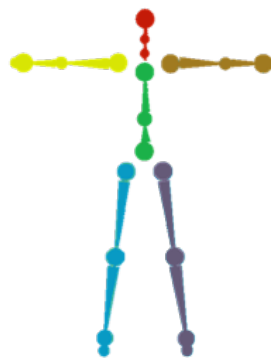


Figure 3.6: Human Kinematic Chains. Joints in red belong to head chain; yellow joints belong to upper right chain; green joints belong to trunk chain; brown joints belong to upper left chain; blue and purple joints belong to lower right chain and lower left chain respectively.

So, we define a body part split profile as a combination of human kinematic chains. For example, a body part could be formed by  $HC_1$  and  $HC_2$ . In this case, the resulting body part split profile will be  $BP_{12}$ . In this manner, we know the former human chains that belongs

Id	Kinematic Chain	Root	Joints
$HC_1$	Head chain	Spine3	Neck and Head
$HC_2$	Trunk chain	Hips	Spine1, Spine2 and Spine3
$HC_3$	Upper right chain	Spine3	RightForeArm and RightArm
$HC_4$	Upper left chain	Spine3	LeftForeArm and LeftArm
$HC_5$	Lower right chain	Hips	RightUpLeg, RightLeg and RightFoot
$HC_6$	Lower left chain	Hips	LeftUpLeg, LeftLeg and LeftFoot

Table 3.1: Joints belonging to each Human Kinematic Chains.

to a body part just taking into account the nomenclature adopted. However, we consider to always join  $HC_5$  and  $HC_6$  in order to avoid instabilities in motion synthesis, taking into account that we are dealing with locomotions. A split profile formed by all human chains (in view of the  $HC_5$  and  $HC_6$  integration) is what we used in the experimental analysis of 3.4. Therefore, we use a set of the possible combinations in the experimental analysis of 3.5.

### 3.3.3 Constructing a Body Part Motion Graph (BPMG)

As we have mentioned, we split full-body in distinct BP that make up a SP, and construct a motion graph for each part. Therefore, a BPMG from BP  $n \in SP$  consists of a set of nodes  $N_n$ , a set of edges  $E_n$  and edge weights  $W_n$ .  $N_n$  represents body part frames. Elements of  $E_n$  are edges connecting body part frames which are consecutive in the original motion capture clips or similar, so graph edges are directional.  $W_n$  is a set of edge weights that describe connections in order to search for transitions. So, we compute distance metrics between all body part frames for each BP defined by the SP. We use joint angles distance metric [Lee et al., 2002] defined in Equation 2.2.

After distance metrics computation we have to create transitions between frames that are below a similarity threshold, as in SMG construction process. As in [Arikan and Forsyth, 2002], we create a transition ( $E$ ) between frames ( $N$ ) where that distance lies below a similarity threshold ( $st_n$ ), and so, it is inserted as a computed edge. So, a threshold is set for each body part motion graph. In this manner, we provide a more intuitive way for a non-expert user to decide which parts of the motion are important as joint weights in joint angle distance metric.

Thus, in order to use a BPMGs for locomotion synthesis we set edge weights (not to be confused with joint weights  $w_k$ ) between BP frame  $i_n$  and BP frame  $j_n$  from  $n$  as

$$W_n(E_n(i_n, j_n)) = \begin{cases} 0 & \text{if } d(i_n, j_n) \geq st_n \\ d(i_n, j_n) & \text{if } d(i_n, j_n) < st_n \end{cases} \quad (3.1)$$

where  $d(i_n, j_n)$  is the distance metric between BP frame  $i_n$  and BP frame  $j_n$  from  $n$ ,

and  $st_n$  is the similarity threshold for BPMGs from BP  $n$ .

Then, it is necessary to prune graphs if we do not want unexpected results in motion synthesis as explained in 2.3.1. First, we use Tarjan's algorithm [Tarjan, 1972] to find the largest Strongly Connected Component (SCC) of each graph and we refuse nodes that are not in. So, we avoid dead ends and ensure connection between all nodes in our graphs.

Our experimental analysis shows that a large number of smooth transitions can be generated thanks to BPMGs with the help of body segmentation. BPMGs nodes provide connections between poses that are similar, but not similar enough to create smooth transitions dealing with full poses as in SMG, such as poses from locomotions with different step lengths. As WcMG [Zhao and Safonova, 2009], BPMGs also provide transitions between different motion clips even if such transitions are not explicitly captured. For example, a transition from running to long step walking while moving the arms considerably is possible, because the arms movements from running and the arms movements that lead to long step walking can be connected by traversing distinct partial frames, even when no such motion capture data exists. On the other hand, SMG needs to use walk motion as a bridge for transitioning between running and long step walking. Therefore, it is possible to choose a very low similarity threshold for a variety of motions to create a graph with very good connectivity. Section 3.4 gives a detailed analysis of the connectivity and threshold choices for BPMGs and SMG in order to compare them.

Special case: body part  $BP_{56}$

We have restricted connections in body part motion graphs where are implied the couple  $HC_5-HC_6$ . We only allow edges between the same phase of locomotions as in [Lee et al., 2002]. Walking and running motions have swing and stance phases (see Figure 3.7) , so we remove all edges where different phase frames are connected. We detect these phases thanks to computing feet contact with the floor. We use a double filter which evaluates foot height and velocity. We do this process in BPMGs creation avoiding any Inverse Kinematics (IK)-based or other post-processing step for foot skating correction. This is done before computing the largest SCC. This strategy provides a lot of benefit in terms of preserving motion quality, but reduces the maneuverability of motion graphs. In the case of SMG, it is important to cut this transitions, although maybe the upper transition is good enough. In these cases is where BPMGs take advantage by preserving these transitions, moreover when the correlation between upper and lower body is different between the origin motion and the target motion.

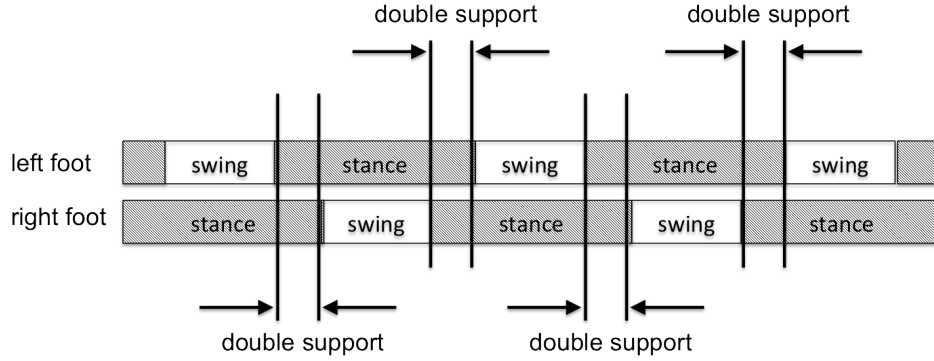


Figure 3.7: Walking phases. Walking is a locomotion in which feet alternate swing and stances phases. After both feet are in a stance phase, which means double support, one of them (left or right) starts with a swing phase while the other remains in stance phase. When this swing ends, both are again in double support. Then, the other foot performs a swing phase while now is the other that rests in a stance phase, and so on.

### 3.3.4 Synthesis of progressive transitions

#### Search

Usually, synthesis criteria is based on finding the minimum path that satisfies search requirements, for example, an origin node ( $i$ ) and a set of possible target nodes ( $j \in J$ ). This could be used to select a good transition from a locomotion to another, where the target frame could be any frame of the target motion clip. So, in this context, we define a target motion clip as our unique requirement for a transition. We use Dijkstra's algorithm [Gibbons, 1985] for searching the shortest path between origin frame  $i$  and all frames  $j$  belonging to  $J$  in each BP of the SP. We define our cost function for a BPMG transition as

$$C_n(i_n, j_n) = W_n(E_n(i_n, j_n)) \quad (3.2)$$

where  $W_n(E_n(i_n, j_n))$  is the weight of the edge between BP frame  $i_n$  and BP frame  $j_n$  from  $n$  defined in Equation 3.1.

Motions are distributed in different motion graphs, so, we must find a transition for each BPMGs. As a naive solution, we can select the best combination of minimum paths taking into account all body part motion graphs. But in this case, we have to evaluate a lot of cases consuming a lot of processing time. Moreover, we have to take into account that joining body part transitions could produce some instabilities in resulting motion. For example, an arm can move very fast while the rest of the body is moving slowly. For taking up these problems we suggest to adopt some optimizations explained below.

To avoid a big cost of computation we have set priorities to BPMGs that defines the search order. A body part motion graph with a big priority means that we have to minimize



its transition cost, so we first search in that BPMGs and use the origin and target output frames as input frames to search the best paths of the other BPMGs. Furthermore, this provides more benefits. Expressed in another way, one body part drives the synthesis of the others, containing the basic information of the movement which takes part. This has further implications. Prior BP transition marks transition timing due to its duration is not modified in the synchronization process (more details in Section 3.3.4). In case of locomotions, not modifying the timing of  $BP_{56}$  animations ensures to display physical possible postures. So, we assign  $BP_{56}$  as a prior body part. Otherwise, resulting transitions may include some motion artifacts.

The latter found body part transitions are not mandatory to match exactly with the origin and target frame defined by the first search, contrarily, they have to be within a time window we called progressiveness window. Prior BP transition origin frame (from origin motion) and end frame (from target motion) also mark the starting of progressiveness window (see Figure 3.8, on top), which permits to connect frames after origin frame and end frame from this transition. These variations at the beginnings and the endings of body part transitions produce progressiveness to generate the complete transitions. A window, more precisely two pair windows, defines potential transition progressiveness. This fact further increases connectivity between motions but increases the risk of obtaining unnatural transitions. So, defining an appropriate value of this progressiveness will be important.

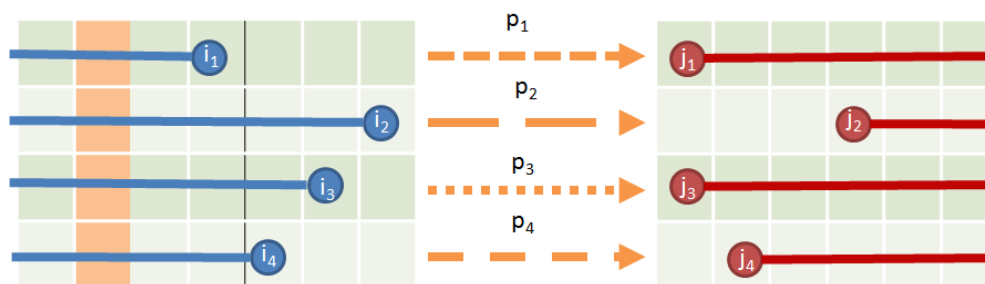


Figure 3.8: Body part transition paths. Origin frames are indicated by  $i_n$ , target frames by  $j_n$ , transition paths are denoted by  $p_n$ , being  $n$  from 1 to 4 as the number of body parts existing in this candidate transition.

From the previous point, we get several combinations of possible body part transitions to be synchronized. However, we refuse combinations where body part transitions have very different lengths, because remaining as a candidate transition could produce jerky movements (some body parts will move much faster than others). For all BP transitions except of prior one, we compute the euclidean differences between their length ( $d_l$ ). Experimentally, we decided to remove combinations when some body part transition  $d_l$  exceeds 20%. By doing this, we want to preserve as much as possible the timing of the former animations. This decision has been taken due to the authors observed that transitions with this characteristic look a little jerky because some artifacts appeared in the first tests. Finally, we pick the

combination of body part transitions with the minimum accumulated cost, which means, the sum of the cost from formula 3.2 from each BP of the SP.

At this stage, we obtain the body part transition paths needed to achieve the full-body transition. As it is illustrated in Figure 3.8, the full-body transition is formed by each body part  $n \in BP$  transition which is defined with an origin frame  $i_n$ , a target frame  $j_n$ , an a path of frames  $p_n$  that conforms the transition itself.

### Synchronization

With all transitions selected (a path for each body part motion graph) we need to synchronized them in order to enable the full-body transition. Selected paths in transitions usually do not have the same frame duration since it is not a restriction in the search (only transitions which include great differences in duration are discarded). So, if it is necessary we change the frame rate of transitions paths in order to match their durations. This process is illustrated in Figure 3.9.

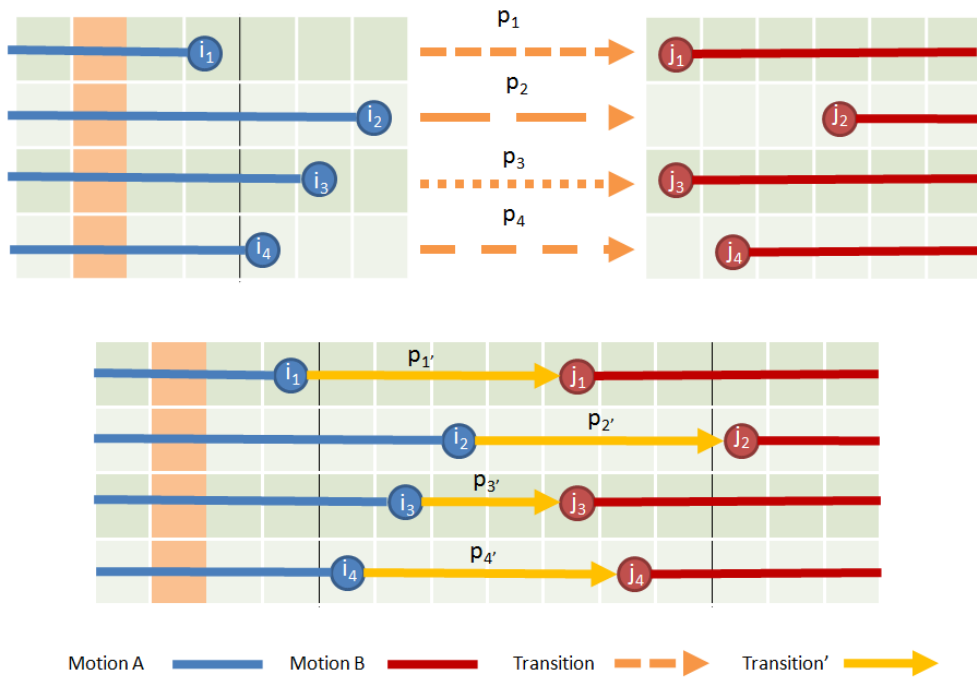


Figure 3.9: Synchronization process. At the top, we have the origin frames, paths and target frames of body part transitions. Paths line style denotes how many frames are in each path which is different in all cases. If we compose that transition at that point the gap of frames of each body part does not match with path durations. So, at the bottom, paths are synchronized and fill the gaps exactly by time scaling them.

We compute the factor needed to scale path durations and we apply linear interpolation to match the target length from the original one. As mentioned, this could produce some motion artifacts, but the restriction applied before in the previous section minimizes this fact. Thus, instead of using more complex methods for synchronizing transitions such as Dynamic Time Warping (DTW) [Müller, 2007], we address the mentioned straightforward solution which is enough for our purpose, and thus we can test motion synthesis performance of this method thinking on in a near future converting it into a realtime character controller.

### Motion generation

After searching process and transition path scaling are done we get a transition scheme like in Figure 3.10. For each body part motion graph  $p$ , we have an origin frame  $i_p$ , a transition path  $p_p$ , an end frame  $j_p$ . Transition path length is defined by  $l_p$ . So, each transition is composed by origin frames vector  $\bar{I}$ , transition paths duration vector  $\bar{L}$  and end frames vector  $\bar{J}$ . We will use this nomenclature for formula 3.3 in the next Section 3.4.1 belonging to evaluations.

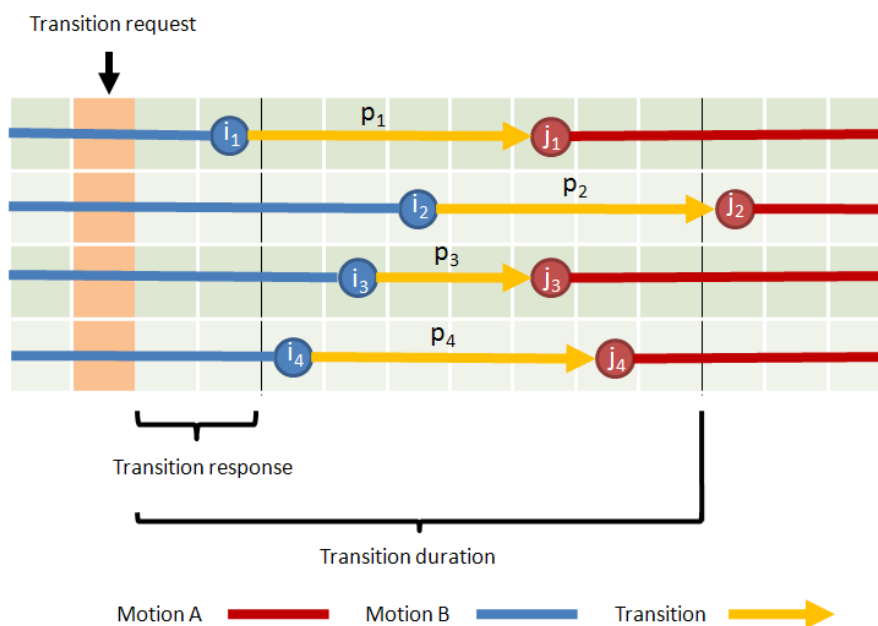
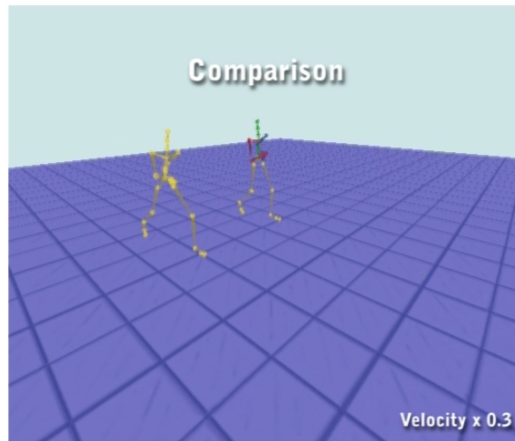


Figure 3.10: Transition scheme. Note that transition response is the elapsed time between the frame when the transition is requested and the beginning of the first body part transition.

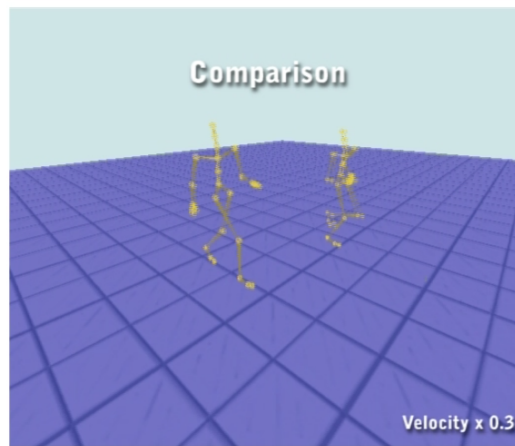
The final animation is composed by stitching synchronized body part transitions. It is known that simply jumping from one frame to another produce small discontinuities in the motion. One solution is modifying the motion sequence after the transition to match the sequence before the transition [Witkin and Popovic, 1995], another is to use linear blending

[Lee et al., 2002]. In our case, we just use a low-pass filter to smooth transitions due to we achieve high connectivity with low similarity thresholds. Additionally, we compute the global position from global velocities embedded in lower body motion graph.

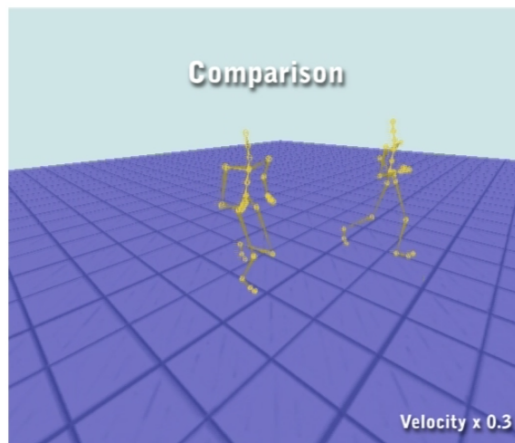
In Figure 3.11 there are some screenshots from a video that display two virtual characters (rendered as skeletons) transitioning from a walking to a running motion, one driven by a SMG (on the left) and the other driven by BPMGs (on the right). As the time  $t$  goes on, BPMGs is able to transition from walking to running long before SMG. In this example, transition was requested at time  $t_r$ , and BPMGs method starts to transition just after 0.01 seconds (see (a) from Figure 3.11), contrarily, SMG has not achieve the transition until after 1.00 seconds.



(a)  $t = t_r + 0.01s$



(b)  $t = t_r + 0.50s$



(c)  $t = t_r + 1.00s$

Figure 3.11: Comparison between a BPMGs' and SMG's transitions. Three instants are captured in images (a), (b) and (c) which illustrate the transition of two virtual characters (skeletons) from walking to running.  $t$  and  $t_r$  denote the aggregated time and the time where the transition was requested, respectively. The left virtual character is driven by SMG, on the other hand, the right virtual character is driven by BPMGs. The original video can be found at [goo.gl/820jDa](http://goo.gl/820jDa).

### 3.4 Evaluations I: Body Part Motion Graphs (BPMGs) against Standard Motion Graph (SMG)

In order to evaluate the performance of progressive transitions using BPMGs against SMG we have used a dataset  $C$  with different motion clips from CMU Motion Capture Database [Carnegie Mellon University Graphics Lab, 2004]. Dataset contains four different motions: Normal walking  $C_1$ , running  $C_2$ , long step walking  $C_3$  and slow walking  $C_4$ . All motions in the dataset are locomotions and we want to study their transitions. We use these locomotions because they have different speeds and body postures and these obstruct transitions between them [Zhao and Safonova, 2009]. Motions have a frame rate of 60 fps.

We have created standard motion graphs and body part motion graphs with different similarity threshold values using this dataset. Transitions from both methods have been globally analyzed (Dataset Transitions) and particularizing the performance between motions (Motion to Motion Transitions). We have compared transition duration and transition response from both points of view.

Then, we show the rest of the parameters we have used to create the graphs. In case of body part motion graphs, we relate body parts, the associated weights belonging to body parts and joints, and also the search priorities and the duration of the progressiveness windows. This is explained below.

- *Split Profile*. In Figure 3.12 are illustrated the body parts used in these experiments.

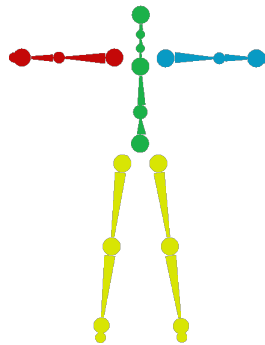


Figure 3.12: Body parts. Joints in yellow color belong to lower-body; green joints belong to the trunk of the body; red and blue joints belong to right upper-body and left upper-body respectively.

- *Joint weights*. Setting joint weights in motion graphs technique can alter system performance as demonstrated in [Wang and Bodenheimer, 2008]. We set a weight of 0.3

for lowerback, leftforearm and rightforearm joints; a weight of 0.5 for neck joint; a weight of 0.7 for leftarm and rightarm joints; a weight of 1 for hips, leftleg and rightleg joints; and a weight of 1.2 for leftupleg and rightupleg joints. We considered that these joints have to be weighted rather than the others, and also we assume that some of them (e.g. leg and up leg) are more relevant in locomotion movement than others (e.g. lowerback ), and so, they have a bigger weight. Although, some optimization could be addressed, like in [Wang and Bodenheimer, 2008], to improve the performance thanks to these weights.

- *Body Part threshold.* As we have said before partial motions are embedded in individual motion graphs and so a similarity threshold for each one is needed. Underside of selecting a threshold for each BPMGs, we use a global threshold and we distribute it between the parties. This distribution leads to create a body part weight which defines the importance of it like joint weights do in each graph. In order to compare BPMGs with SMG, body part weights are set proportionally according to the sum of joint weights belonging to the current body part. In this manner, we not alter results by selecting better similarity thresholds for BPMGs method and joint weights affect in the same way to both methods.
- *Search parameters.* To search transitions in our method we have to set BPMGs priorities and a time window which defines the progressiveness of transitions. To select priorities of BPMGs, we have determined that lower body part have priority rather than the rest. Lower body part is common in all SP and it is the base of locomotions. On the other hand, the duration of progressiveness window is set to 10 frames (0.16 seconds). This value was set without any reference of the current state of the art, as up to our knowledge, there is nothing similar to our approach enough to rely on. Thus, we understand that setting a high value for the progressiveness window could produce some strange full-body transitions with fast and slow movements of some of the body parts, and on the other hand, maybe connectivity and responsive metrics could achieve better results. However, in order to find an optimal value it will be necessary to study the naturalness of the output motions while varying this value.

### 3.4.1 Dataset transitions

As a first measure of the interactivity of BPMGs we have computed Time Between Frames (TBF) [Zhao and Safonova, 2009]. TBF consists in computing the average transition time between all pairs in the dataset. This measure indicates how the dataset is connected from a global point of view, assuming that all nodes belonging to it have the possibility to achieve any of them by traversing the graph. In SMG, transition time is computed straightforward by counting how many frames elapsed from transition request until the target motion is running. In case of BPMGs, transition time is computed doing

$$T_{duration}(f_{req}, C_t) = (i_p - f_{req}) + l_p + (max(\bar{J}) - j_p) \quad (3.3)$$

where  $f_{req}$  is the frame which user has requested the transition,  $C_t$  is the target motion clip,  $p$  indicates any body part index and  $max(\bar{J})$  is the later end frame of body part transition paths. Note that this formula can be operated using any possible value of  $p$ .

TBF shows how quickly we can achieve the target motion. On the other hand, we also want to evaluate when transition is started after user has requested one. Transition response (see Fig. 3.10) is the amount of frames between the requested transition frame and the beginning of this transition. To compute this value in SMG we have to subtract the starting transition frame from the requested transition frame. As we have mentioned, in BPMGs not all joints transition at same time but they do progressively. So, if one body part starts to transition means that the avatar starts to react. Then, BPMGs transition response is computed by  $min(\bar{I}) - f_{req}$ , where  $min(\bar{I})$  is the first starting frame of body part transition paths and  $f_{req}$  is the requested frame.

We have calculated TBF and response varying the similarity threshold from 0.06 to 0.165. The results of these tests, expressed in frames from 60 fps motion clips, are shown in Table 3.2. Comparing both methods, we notice that BPMGs transition duration is lower than SMG transition duration when both have the same threshold and even when the threshold is larger for BPMGs. This means that BPMGs transitions are quicker than SMG and also smoother. Besides this, it should be highlighted that BPMGs achieve connection to all clips in the dataset with a lower threshold than SMG. In Table 3.2, cells have different background color depending on how many motion clips we can transition with the corresponding similarity threshold. Cells in color yellow means connectivity with two motion clips; color green for three motions and color magenta for four motion clips. As seen, SMG achieves connectivity with all motion clips at 0.165 and BPMGs at 0.105.

Threshold	SMG		BPMG's	
	TBF	Response	TBF	Response
0.06	19.99	14.24	17.24	4.34
0.075	19.65	13.61	17.28	4.42
0.09	18.95	13.30	17.27	4.41
0.105	18.43	12.92	17.30	4.40
0.12	18.18	12.74	17.30	4.37
0.165	18.20	12.53	17.30	4.37

Table 3.2: Transition duration and response in BPMGs and SMG. All results are in frames (60 fps). Color yellow denotes that transitions can be achieved between two motion clips; color green for three motions and color magenta for four motion clips.



In the case of transition response, the difference between BPMGs and SMG is more evident. The set of response values belonging to SMG is between 12.53 frames and 14.24 frames. However, BPMGs provides results between 4.34 and 4.40 frames. So, we have an improvement of more than 3 times in terms of response time.

### 3.4.2 Motion to motion transitions

Measuring the average transition time and transition response of whole dataset gives an idea of how both methods works. Although, we do not have any information about motion clips specifically. Reitsma and Pollard [Reitsma and Pollard, 2007] proposed Local Maneuverability (LM) metric to evaluate how long it takes the character to perform any other action of the dataset. This measure was created for working with action motions since it takes into account the beginning and the end of dataset motion clips. In our case, we are evaluating LM of locomotions so we have used an adaptation of the known formula for computing this measure. In human locomotions it does not matter when the motion is started or ended because all time is performing the locomotion. So, we look for how fast we can transition from a locomotion to any instance of another locomotion. A motion to motion LM is computed as

$$LM(C_K, C_L) = \frac{1}{\|K\|} \sum \min(T(k, L)) \quad (3.4)$$

where  $C_K$  is the origin motion clip,  $C_L$  is the target motion clip,  $\|K\|$  is the amount of frames of clip  $C_K$ , and  $\min(T(k, L))$  is the duration of the minimum transition from instance  $k$  of motion clip  $C_K$  to any instance  $L$  of motion clip  $C_L$ . We have also computed transition response between motion clips. In order to do it Formula 3.4 has been used again but swapping transition duration term for transition response. In this manner, we can evaluate response and transition duration between motion clips.

In Table 3.3 there are results for LM and response times between motions. We have used different similarity thresholds for both methods. The thresholds chosen are the minimum values that allow transitions between all clips of the dataset. Similarity threshold values are 0.165 and 0.105 for SMG and BPMGs respectively. BPMGs transition responses are better than SMG results in all cases, in case of transition times, only in two cases ( $C_1$  to  $C_3$  and  $C_4$  to  $C_3$ ) BPMGs performs worst than SMG, being BPMGs results better in most cases (10 out of 12).

If we look at the highest values of SMG results, we notice that  $C_2$  as target motion is the one further away. Specifically, if we want to transition from  $C_4$  to  $C_2$  using SMG, it took 25.00 frames of duration. However, if we use BPMGs the values are 9.40. So, BPMGs is more than two times better in transition duration in the slowest transition between motion clips. And if we pay attention to the response time in the same case, BPMGs response time is close to zero while SMG is 18.23 frames. In fact, there are many zero values in response between motion clips. Specifically there are 7 zero values of 12 possible transitions between

motions. So, in these cases when a transition is requested the virtual character starts to transition to the target motion immediatly without delay.

	$C_1$	$C_2$	$C_3$	$C_4$
$C_1$	0	10.71 / 6.35	3.68 / 0.43	3.40 / 0.63
$C_2$	4.26 / 0.84	0	5.26 / 0.84	7.26 / 0.84
$C_3$	5.71 / 2.52	15.86 / 11.50	0	5.23 / 2.91
$C_4$	3.73 / 0.01	25.00 / 18.23	2.86 / 0.00	0

	$C_1$	$C_2$	$C_3$	$C_4$
$C_1$	0	5.91 / 0.00	3.90 / 0.00	2.61 / 0.02
$C_2$	2.41 / 0.00	0	4.97 / 0.00	3.29 / 0.00
$C_3$	5.11 / 0.06	9.54 / 1.53	0	3.95 / 0.15
$C_4$	3.40 / 0.00	9.40 / 0.01	3.10 / 0.00	0

Table 3.3: Motion to motion transition duration and response. Above, SMG results; Down, BPMGs' results. Motion clips  $C_n$  placed on rows are origin motions, and motion clips  $C_n$  placed on columns are target motions. Values in the top left are durations and values in the lower right corner are responses, both in frames. Motion clips are normal walking  $C_1$ , running  $C_2$ , long step walking  $C_3$  and slow walking  $C_4$  recorded at 60 fps.

### 3.5 Evaluations II: Analyzing body part segmentation in BPMGs

In this experimental analysis we want to study how body segmentation affects to body part motion graphs method. To do it, we have used the same dataset and setup as in the previous Section.

*Body parts.* In Figure 3.13 are shown different body part split profiles (SP). These profiles covered from two body parts until five body parts. In all cases, lower body is set as a body part because splitting in two parts provokes instabilities. Note that some profiles contain the same number of body parts but they are different in order to analyze which partition is better.

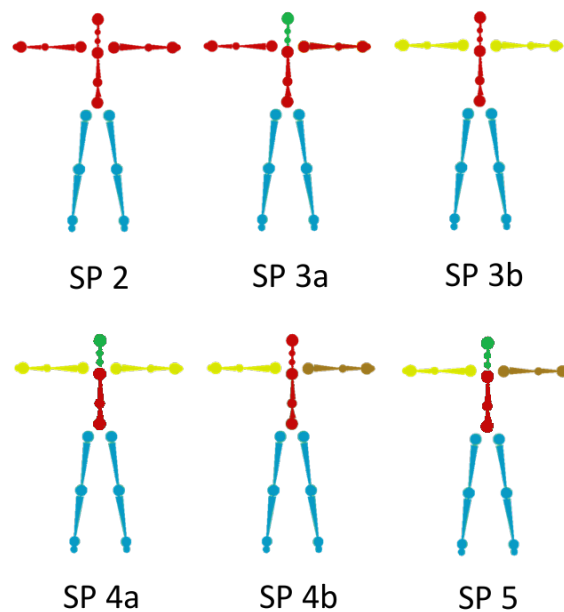


Figure 3.13: Body part Split Profiles (SP). From 2 to 5 number of body parts are allowed in SPs. Two different segmentations with 3 and 4 parts are used.

### 3.5.1 Connectivity versus split profiles

Connectivity in locomotion system defines how frames are connected. In SMG, connectivity is computed by dividing strongly connected frames (frames connected under the similarity threshold restriction) by total frames of the dataset [Zhao and Safonova, 2009]. In our case, we have different graphs that together contain full motions and the connectivity of each graph is not the connectivity of BPMGs. The way to compute the connectivity of our method is to compute the percentage of intersected frames from all graphs (depending on SP).

Connectivity results are shown in Figure 3.14. As we can see, SMG are not the best option at least if a low similarity threshold is required. So, we can validate the hypothesis that some joints configurations are responsible for not connecting full-body frames. In the first value of thresholds it is remarkable that SP4b achieves almost the maximum connectivity of our dataset. For the same threshold SP4a and SP5 have also noticeable results. Two iterations later, SP2 reaches SP4b connectivity doing useless some partitions in the upper body. Maximum connectivity is achieved by this order: SP4b, SP5, SP3a, SP4a, SMG, SP2, SP3b.

These results allow us to identify which body parts segmentation are not useful to improve connectivity. From this we can structure how to split our avatars.

- The results from SP4b and SP5 show that considering arm motions as two independent parts allow more connectivity. SP4a and SP3b confirm this conclusion.
- To consider head motion as an independent body part is almost always a good option. Just an exception: SP4b.

We have said that connectivity of individual graphs are not representative of dataset connectivity. Although, body part motion graphs connectivity improves richness of body part transitions.

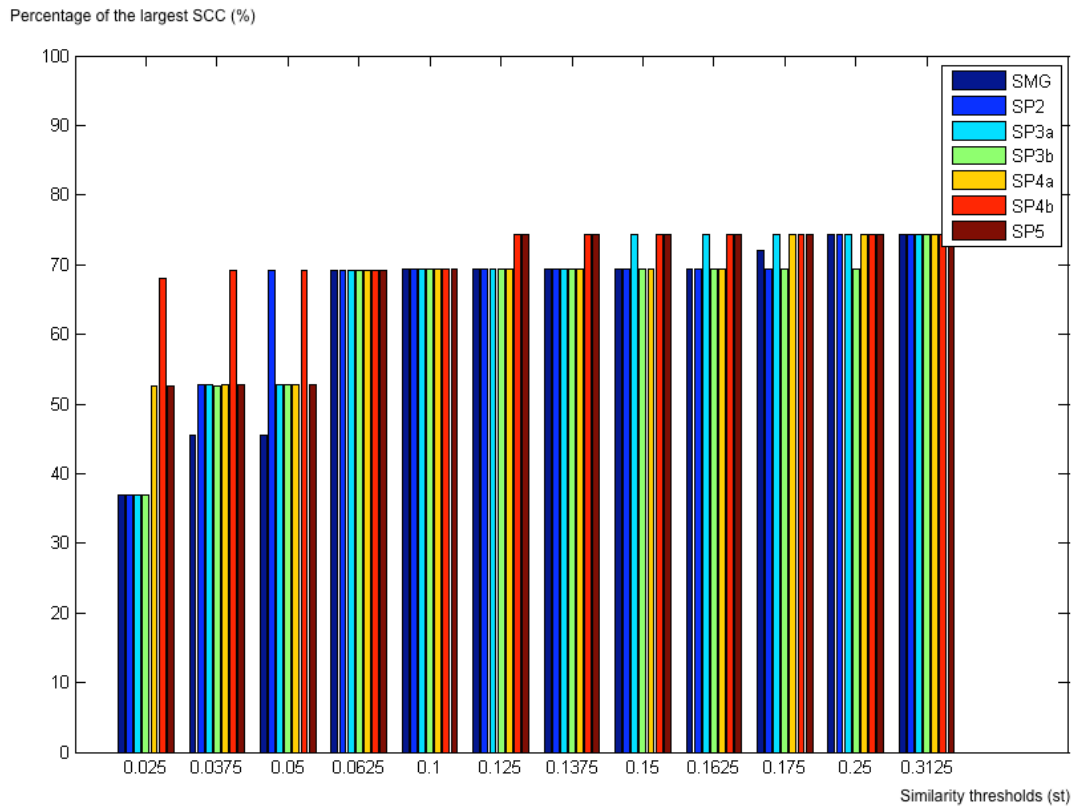


Figure 3.14: BPMGs connectivity. This graph shows the connectivity of different split profiles in each similarity threshold value. Each colored bar represents a split profile.

### 3.5.2 Transition costs versus split profiles

Smoothness between transitions are related with selected path costs (see Formula 3.2). Transition cost involves the sum of metric distances between frames, from user input until the target motion is running completely. To compute BPMGs transition cost, we have to take into account that we select paths that have been modified in the synchronization phase (see Section 3.3.4). Thus, we have recomputed metric distances between full-body frames of the generated motions. In case of SMG, we have directly used the cost (see Formula 3.2) of the selected path. To study these costs we have computed the average of transition costs between all pairs in dataset, as done in [Zhao and Safonova, 2009] for interactivity purposes. We show results in Figure 3.15.

Logic suggests that if we divide the body in body parts, the total cost of the transitions have to be smaller than full-body since each partial transitions is not conditioned by the rest. Results confirm this idea as is shown in Figure 3.15. In this graphic we show the relation between cost and similarity threshold and also connectivity for each split. Almost each signal follows a similar pattern which begins growing quickly and suddenly decreases until grow a little to finally stabilize. That occurs because there are few edges below the similarity threshold and so few connections exists in graphs until some key connections are opened and paths became shorter and with less cost.

Slopes changes mean different connectivity which is shown in colors. So, the lowest transition cost with the best connectivity is achieved by SP5, although SP4b is closer to it. The rest of SP results show that separate in body parts optimize transition cost. Splitting body arm motions decreases cost two times, and splitting in two sides permits improve more than three times smoothness.

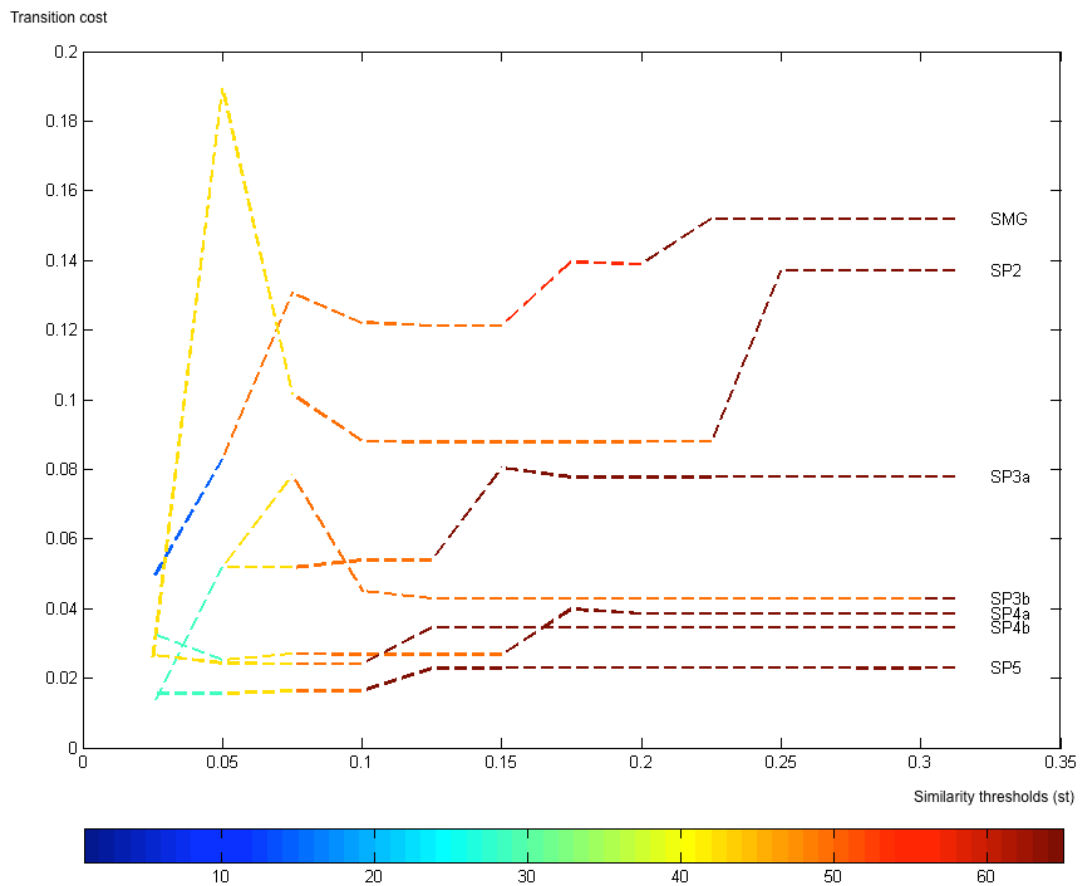


Figure 3.15: BPMGs' transition cost. In horizontal axis is defined the similarity threshold and in vertical axis, the transition cost. Each line is related with a body part Split Profile (SP) which color changes along the color scale below indicating the connectivity of this SP from the similarity threshold value.

### 3.5.3 Optimizing motion to motion transitions

In our dataset there are four different motion clips. If the goal is to transition between motion clips then it will be necessary to create new motion graphs. In this case, it will be possible to find new optimal split profiles solutions different to the obtained in the former dataset.

In this way, we have analyzed connectivity and transition cost between pairs of motion clips and we have found the break point representing the maximum connectivity, the minimum transition cost and body part number. To do it, we can not use the same motion graph generated from the original dataset so we have constructed a new body part motion graphs. The reason is because pruning graphs using whole dataset could delete some strongly connected component which is not the largest, but satisfy transitions between two motion clips. Apart from that, we follow the full process explained before.

	$C_1$	$C_2$	$C_3$	$C_4$
$C_1$	SP4b	SP4b	SP4b	SP4b
$C_2$	SP3a	SP3a	SP3a	SP3a
$C_3$	SP4b	SP3a	SP2	SP2
$C_4$	SP4b	SP3a	SP2	SP2

Table 3.4: Motion to motion optimal split profiles. In this matrix cells show the optimal split profile that connects pairs of origin motion clip (rows) and target motion clip (columns).

After all computations, we get an optimal split profile for each motion to motion transition. In Table 3.4 results are shown. As we can see, some SP are repeated but we also obtain some variety. SP3a and SP4b are the best split profiles because are optimal in 6 of motion to motion transition each one. Then, SP2 is selected 4 times. So, we have proved that SP4b is not the best split profile in all motion to motion transitions rather than in dataset transitions.

It is really interesting that depending on origin and target motion is better to use one or other split profile. This could add an additional step to the common way of generating motion graphs by using an algorithm able to analyze and compute the optimal motion segmentation for each two frames, or motion clips.



## Chapter 4

# Analysis and Synthesis of Body Language driven by Speech Emphasis

### Chapter Abstract

Let's move on the second challenge of this thesis: synthesis of expressive motion. Once we have gone indeed motion synthesis, in our case focused on generating fast transitions between locomotions, we want to give expressivity to virtual characters and increase the interaction between the user by evoking user responses through virtual characters' movements. Thus, in this chapter we present a synthesis method for animating speaking virtual characters which is based on a Gesture Motion Graph (GMG). GMG is a graph structure that leads to synthesize appropriate gestures according to speech in terms of time and emphasis. To that effect, we previously have studied the relation between both cues to extract a set of rules able to drive the GMG. So, we firstly explain the corpus we have recorded and then the conducted correlation analysis. Below, GMG construction and synthesis steps are expounded. Finally, a user study that tests the plausibility of the generated animations is shown, then, results are discussed. After that, we present BodySpeech, an automated system to generate gesture and facial animations driven by an audio speech. The fact of adding facial animation leads to use virtual characters in end-user applications. This time body gestures are synthesized using an improved version of the GMG. Concurrently, facial animation is generated by lip synching, adding emphatic hints according to intonation strength. Furthermore, we have implemented a tool for animators. This tool enables us to modify the detection of pitch accents and the intonation strength influence on output animations, allowing animators to specify the style of gestural performances. This chapter is based on the published papers "Gesture synthesis adapted to speech emphasis", *Speech Communication Journal (Special Issue on Gesture and Speech in Interaction)* [Fernández-Baena et al., 2014] and "BodySpeech: A configurable gesture and facial animation system for speaking avatars", *Computer Graphics and Virtual Reality 2013 (CGVR-2013)* [Fernández-Baena et al., 2013].



## 4.1 Introduction

Nowadays, there is a myriad of HCI applications using 3D virtual characters to interact with users in virtual environments. In order to enhance their realism, avatars try to emulate human beings as much as possible from thinking (artificial intelligence) to moving (character animation [Pejsa and Pandzic, 2010][Van Welbergen et al., 2010]), or speaking (speech synthesis [Bulut et al., 2007][Qin et al., 2010][Roekhaut et al., 2010]). For synthesizing good quality speaking virtual characters, it is essential to synchronize virtual characters speech with body language to resemble as much as possible humans. Body language is a cornerstone in communication (see Figure 4.1), because it accompanies and emphasizes speech content and also provides additional information to the audience [Quintilian. and Butler, 1920].

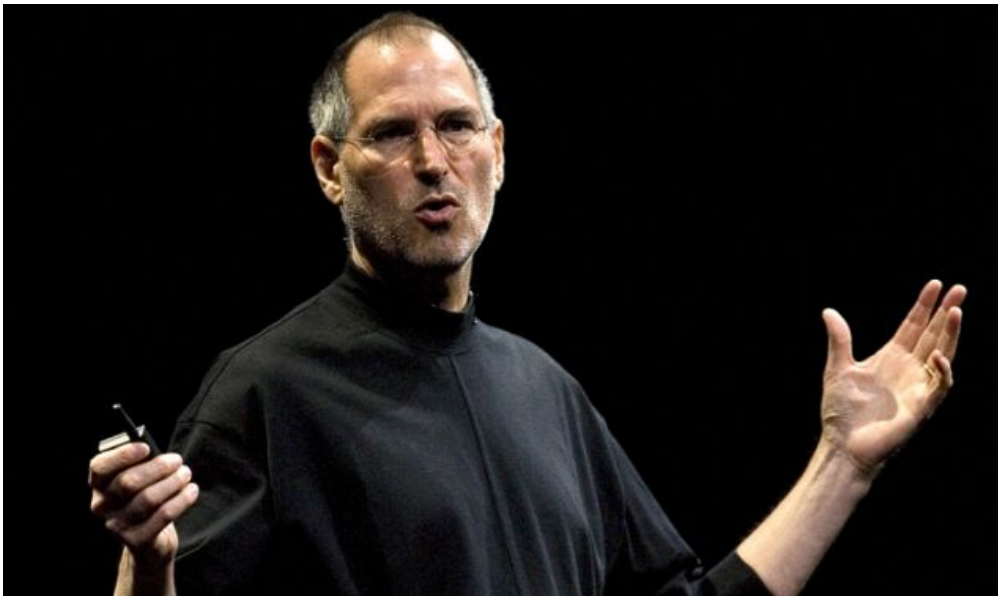


Figure 4.1: Steve Jobs speaking in an Apple keynote [Wikipedia Commons, 2015b]. Steve Jobs was a master nonverbal communicator that usually emphasized nearly every sentence with expansive and illustrative gestures that complemented his words [PROFITguide.com, 2015].

The generation of appropriated body language to a specific speech stream is a complex task. It is known that speech and gestures are related [McNeill, 1985][Loehr, 2004][Leonard and Cummins, 2011]. However, it is difficult to extract a set of rules capable of covering the broad variety of gestures taxonomy [McNeill, 1992] (iconic, metaphoric, deictic and beats) and then using that information to drive a gesture synthesis system. Therefore, for commercial purposes, where plausibility and realism are the most important, body language animations are pre-cooked or motion capture data is recorded concurrently with the speech

to be launched later without modification. On one hand, generating adhoc animations for each piece of speech provokes a dramatic increase of production costs but it features high quality. On the other hand, automatic [Levine et al., 2009][Levine et al., 2010] or semi-automatic [Kipp et al., 2007][Neff et al., 2008] systems are able to synthesize appropriate gestures by analyzing the speech signal, although the quality of the results is not as good as in hand-made animations. Moreover, a wide review of data-driven algorithms to synthesize hand and finger can be found in [Wheatland et al., 2015], which addressed methods used on areas such as manipulation in addition to communication area, what we are dealing with in this part of the work.

Therefore, most of the related work (see Section 4.3) tend to focus on the relationship between gestures and speech, or in the generation of gestures. In our case, we address both objectives to achieve better results and not to alter these with external dependencies. Regarding the relationship between gestures and speech, we rely on [Loehr, 2004], which looks at whether there are temporary correspondences between gestures and voice, particularly between the apex and pitch accents (more details in Section 4.4.2 and Section 8.1.2). In addition, we compare intensity values, which we previously categorize using two indicators that are linked to speech emphasis. Therefore, the generation of gestures is linked to the emphasis of the speech. Despite other systems such as [Levine et al., 2010] also uses prosodic parameters to drive synthesis, they do not previously categorize or analyze or relate gesture and speech signals, and therefore, there is no indeed knowledge of how motion is created from speech. Regarding the quality of the animation, we can say that our results are at least as much good as in [Levine et al., 2010], because even we compare ourselves with them in order to verify the importance of the relationship between gestures and speech (see Section 4.4.5), we can extrapolate the good results saying that our output gesture animations are believable beyond the intent of the speaker. Furthermore, the structure we use to generate these movements facilitates editing. Changing the relationship with the speech of the gestures can generate different intentions. It is important to note that this is done without modifying the gestures database used for the synthesis.

So, we address a study of the relationship between gestures and speech in order to improve the plausibility of gesture animations. Our objective is to generate appropriate gestures to a speech input with varying emphasis, so as to increase the credibility of speaking virtual characters. Thus, we have analyzed an all-inclusive corpus for gestures and speech (both recorded concurrently) so as to extract synchrony and intensity rules. Our corpus contains a wide variety of gestures and speech utterances with different strength levels. The synchrony rule defines temporal correspondences between speech and gesture, and the intensity rule relates speech and gesture strength levels. Then, we use these rules to drive our gesture synthesizer. We successfully applied our algorithm to synthesize speech with varying emphatic levels. Hence, our system is very suitable for applications where emphatic speaking virtual characters are required such as in video games, virtual worlds, communication tools or any type of HCI, where the realism of the character animation increases the product quality. At same time, we consider that intensity and synchrony rules serve as a valuable theoretical

base of how gestures are related to speech, which could be extended with more rules or furthermore, could be used to drive other gesture synthesizers.

We propose a framework that ties together tasks of data preparation (audio and motion) and run-time processes that use that data for generating gesture animations. To begin with, we define two variables containing the intensity (or strength) of each pitch accent and gesture, which are named as Pitch Strength Indicator (PSI) and Gesture Strength Indicator (GSI), respectively. These variables have been calculated using prosody parameters from speech signal and kinematic parameters from gestures contained in the corpus. These computations require a previous segmentation of the signals, which has been conducted manually in this work to minimize inaccuracies. To validate the ability of intensity variables on determining the strength of each cue, proper perceptual tests have been conducted. Following, we search for correlations between GSI and PSI resulting in an intensity rule. In addition, we have also extracted a synchrony rule based on temporal distances between pitch accents peaks and gestures apexes present in the corpus.

In this chapter, we also use motion graphs [Arikan and Forsyth, 2002][Kovar et al., 2002][Lee et al., 2002] (see Section 2.3 for depth information about) to generate an unlimited gesture stream as in [Stone et al., 2004]. Motion graphs have been used for several purposes such as gesture synthesis [Stone et al., 2004], dance synthesis [Xu et al., 2011], animal locomotion synthesis [Wampler et al., 2013], or to generate avoidance motions [Oshita and Masaoka, 2011]. In our implementation, we use a similar graph construction process as [Xu et al., 2011], and for selecting appropriated motion segments (gestures), we use a combined distance metric which coincide with [Oshita and Masaoka, 2011], although we use it for gesture synthesis purpose instead of avoidance motion.

So, we introduce Gesture Motion Graph (GMG) which takes prosody features from speech as input. GMG is formed by a set of gestural phrases belonging to our corpus. To drive gesture selection it takes into account the strength similarity between speech and gestures (intensity rule). Additionally, we use the synchrony rule to temporally match motion sequences with speech. To that effect, we obtain a gesture stream based on the temporal and strength relationship between speech and gestures as it is shown in Figure 4.2.

In order to explore the user perception we have conducted a user study to test the plausibility of GMG output animations generated using the mentioned rules. We measure the relative quality between original animations and synthesized animations, which are generated both with and without intensity and synchrony rules. Results from this test show that using the intensity rule together with the synchrony rule enhance the credibility of emphatic speaking animations. As mentioned, GMG is based on SMG, as BPMGs method that has been explained in the previous Chapter. However, GMG and BPMGs differ in several technical aspects such as the way of transitioning and search process. A further discussion is addressed in Section 6.2.

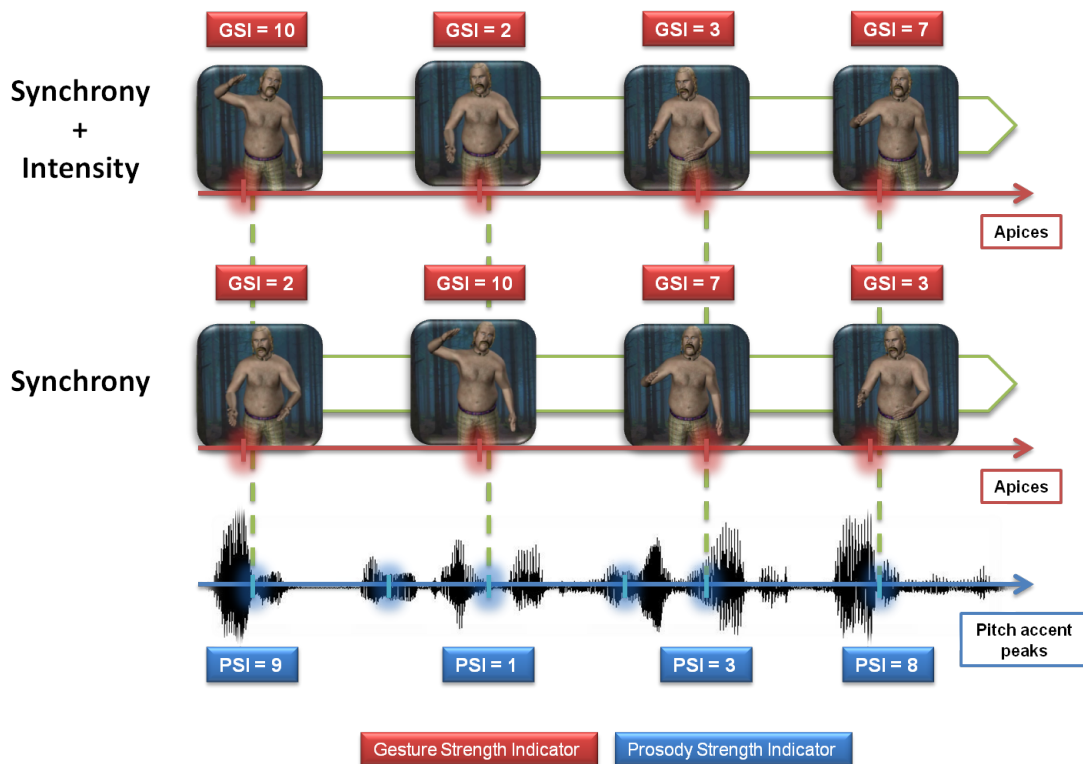


Figure 4.2: Gesture streams synchronized with speech. At the bottom, a gesture stream synchronized with speech only in terms of time by matching apices (from gestures) with pitch accent peaks (from speech). In this case, there is no relation between PSI values and GSI values. At the top, a gesture stream synchronized in terms of time and intensity. In this example picture, we assume that high PSI values are related with high GSI values, and low PSI values with low GSI values.

Finally, we have developed an authoring tool for generating custom gesture animations from an input speech. In order to do it, we have automatized the full process of gesture synthesis. First, we consider the automatic detection of pitch accents avoiding a manual annotation phase. This new way of detecting pitch accents is accompanied by a new pitch accent strength computation (more details in Appendix 8.2). Because of this, we have defined a new gesture-speech strength relation used in gesture selection (see Section 4.5.2). In terms of animation, we have enriched the GMG by reusing the input data to create more gestures (see Section 4.5.1). In addition, in order to avoid stroke modification (the most meaningful part of gestures) we have improved gesture temporal alignment with speech (see Section 4.5.2). Moreover, to improve output motion quality we use optimal blend length [Wang and Bodenheimer, 2008] to create blended transitions between gestures.

Furthermore, in order to use the virtual character in a virtual world it makes necessary

to implement facial animation, specially when virtual characters speak and gesture face to the user. So, in the same manner like in gesture synthesis, we have based output facial animation on speech input. Lip movements [Cvejic et al., 2010] and jaw positioning [de Jong et al., 1993] are related to stressed and unstressed syllables. Based on this, we propose a relation between speech signal and visemes (face shape) in terms of emphasis, in addition of matching phonemes with visemes by lip synching. To that effect, we have added facial animation in BodySpeech. Then, BodySpeech is implemented as a Unity3D plugin. This tool permits to specify the input gesture database and to parameterize the generated gesture performances by modifying their relation with the input speech.

After having traveled from Chapter 3, where a locomotion synthesis method is proposed, we affront this chapter with the objective to provide emphatic gestural behavior to virtual characters. We part from speech emphasis in order to generate appropriate gestures and facial expressions, as a consequence, users perceive the emotions of the virtual characters thus creating a greater emotional bond between the user and the character (discussed in Section 4.4.6). So, the user can be more immersed and engaged in virtual worlds or video games. Beyond virtual worlds and video games, virtual characters that are able to naturally speak and act are also useful in communication or marketing tools where they are used as a autonomous virtual attendees to serve potential customers, solve incidences, provide information, etcetera. Therefore, moving around virtual environments (Chapter 3) and interact with the user (Chapter 4) are common facets in virtual character behavior which we have addressed up to this chapter.

## 4.2 Gestures

In order to better understand the following sections, it is important to review some gesture foundations and taxonomy. A gesture is a movement or a succession of movements performed by the human body, primarily but not always, with the hands and arms. Gestures differ from other body movements because they entail communicative goals. They are part of the body language.

Gestures were classified by McNeill [McNeill, 1992] in a four class taxonomy: iconic, metaphoric, deictic and beats. Iconic gestures represent concrete entities or actions; metaphoric gestures refer to abstract concepts; deictic gestures are used to indicate locations in space; beats are gestures that do not contain semantic content. Beats usually involve simple movements such as up and down or back and forward motions, and are the most used gestures constituting about half of all gestures [McNeill, 1992]. Our work is based on the generation of gestures based solely in prosody, ignoring the semantic content of the accompanying speech signal. Beats are the only type of gestures whose form does not depend on content, therefore becoming the only appropriate gestures for our study. Furthermore, the semiotic value of a beat is similar to emphasis [McNeill, 1992], and it is known that prosody correlates well with emphasis [Terken, 1991]. This suggests possible links between beats and prosody.

These links are analyzed in Section 4.4.3.

In order to study gestures in detail, Kendon [Kendon, 1980] proposed a segmentation scheme. He defined the terms "gesture unit", "gesture phrase" (Gesture Phrase (GP)) and "gesture phase". A gesture unit starts with a rest pose, contains one or several consecutive gestures, and ends with another rest pose. A GP is what it is generally called a gesture. A GP is composed by several gesture phases, some of them being mandatory and some optional. These gesture phases are:

- Preparation. Body parts move from a rest position to the initial position of the gesture. This movement can move towards the opposite direction of the main direction of the gesture. This phase is optional.
- Stroke. It is the phase that contains the 'expression of the gesture', whatever it may be. This phase involves greater effort than any other phases. A GP must contain a stroke.
- Retraction. Body parts are moved to the rest position. This phase can not be present if the speaker concatenates a stroke phase with another stroke.

Kita et al. [Kita et al., 1998] identified another phase:

- Pre-stroke or post-stroke holds. These are temporary cessations of movement that occur immediately before or after a gesture stroke. Holds are optional.

And later on, Kipp [Kipp, 2004] suggested another one:

- Recoil phase. Occurs after a forceful stroke when the hand lashes back from the end position of the stroke. It could be seen as an specific type of retraction phase.

The stroke is the most meaningful part of a gesture, and it is the only mandatory phase [McNeill, 1992]. Other phases are placed before or after the stroke. There exists a special stroke typology called "stroke hold" by McNeill [McNeill, 2008] or "independent hold" by Kita et al. [Kita et al., 1998]. These are strokes where hands remain static with no motion. In this work, as a first approach we do not differentiate between motion or motionless strokes.

## 4.3 Related work

### 4.3.1 Modelling synchrony between gestures and speech

Little attention was given to gesture and speech synchronization until the twentieth century. Condon [Condon, 1976] defined a hierarchy of gestures, stating that smallest and



fastest movements are synchronized with small speech units, such as phonemes and syllables, and slower movements are synchronized with larger speech units, such as phrases. More surprisingly, Condon observed that listeners move in synchrony with the speaker's speech (Interactional Synchrony).

Kendon [Kendon, 1980] defined the following synchronization rule between gesture and speech: The extension phase of a beat gesture would coincide with, or slightly precede, the onset of a stressed syllable. McNeill [McNeill, 1992] suggested a similar rule: The stroke of a gesture phrase is always completed either before or at the same time as the tonic syllable of the concurrent tone unit. Nobe [Nobe, 1996] observed that this statement not only holds for tonic syllables, but also for peaks of intensity. Subsequent studies proposed other anchor points of synchrony between gestures and speech. For instance, Valbonesi et al. [Valbonesi, 2002] defined the speech focal points, which are computed using five prosodic features of speech signal; and gesture focal points, which are computed as local maxima or minima of hand traces. They observed that speech focal points occurred either near a gesture focal point or within the duration of a stroke.

Dwight Bolinger [Bolinger, 1986] observed a synchrony rule between speech and gesture: pitch and body parts rise and fall together, to reflect increased or decreased tension. However, in a latter work, Loehr did not find evidence of this [Loehr, 2004]. However, he observed that apexes of gestures (points of maximum extension) tend to co-occur with pitch accents (see Figure 4.3). Renwick et al. [Renwick et al., 2004] also used pitch accents as speech anchor points, but their corresponding gesture anchors were hits. Hits are abrupt stops or pauses in movement, which break the flow of gestures. These are only present in discrete gestures, and correspond to apexes of these type of gestures. Leonard and Cummins [Leonard and Cummins, 2011] studied the time alignments between several anchor points, and concluded that the apex of the beat gesture shows less temporal variability with respect to speech than with any other point within the gesture.

Antonijoan [Antonijoan, 2012] performed a study of the impact on the perceived quality of animations when using a temporal synchrony rule that aligns pitch accents and apexes of beat gestures. Although an increase of perceived quality in synchronized animations versus not synchronized ones was observed, it was evident that the temporal synchrony rule alone was not sufficient to ensure a proper synchrony for all cases. The perception of animation quality dropped when gesture and speech emphasis levels did not match each other. Levine et al. [Levine et al., 2009] [Levine et al., 2010] proposed a gesture animation system that, besides aligning syllables and gestures temporally, employed the correlation between prosody and kinematics of motion to select appropriate gestures. Levine system uses a probabilistic model from which it is not possible to infer general synchrony rules directly. This work proposes a new synchrony rule that correlates strength levels of speech signal with strength levels of gestures, used to match emphasis between the two modalities.

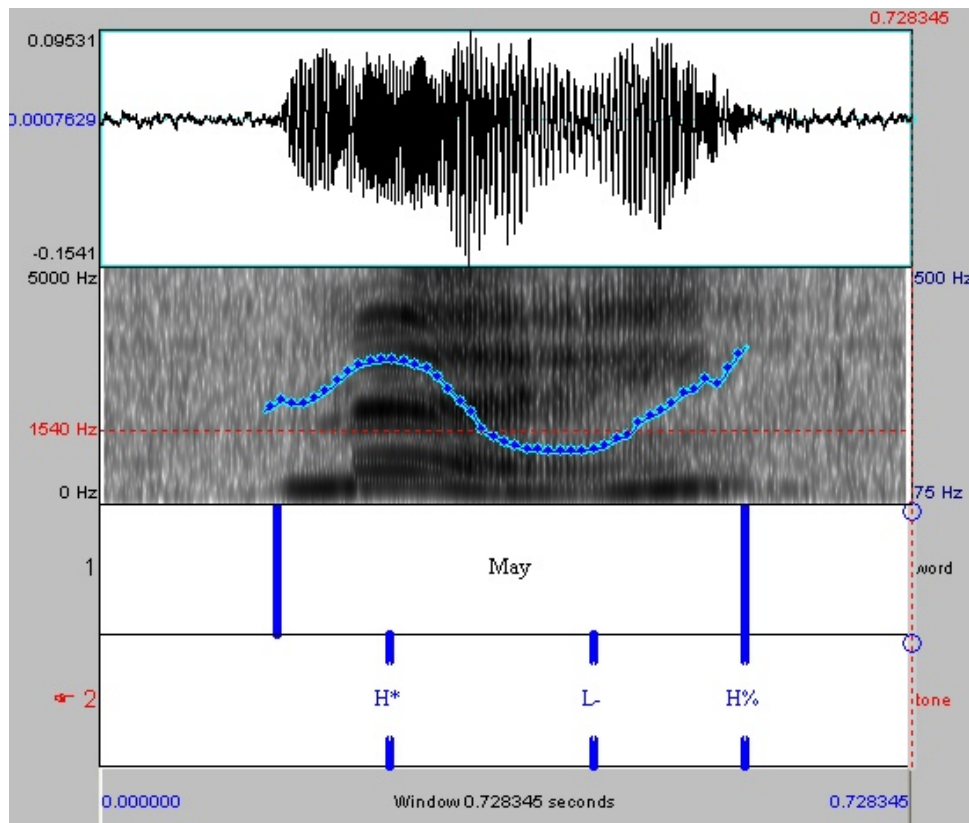


Figure 4.3: Within Autosegmental-Metrical (AM) model [Pierrehumbert, 1980], pitch accents are combined with edge tones, which mark the beginnings and/or ends of prosodic phrases, to determine the intonational contour of a phrase. The need for pitch accents to be distinguished from edge tones can be seen in this contours in which the intonational events, annotated using the ToBI (Tones and Break Indices system) [Silverman et al., 1992b], are an  $H^*$  pitch accent followed by an  $L-$  phrase accent and a  $H\%$  boundary tone. For more details in intonation the reader is referred to Appendix 8.1.1. [Wikipedia, 2015b]

### 4.3.2 Synthesis systems of gestures

Usually, gesture synthesis consists in generating motion according to an input. The input typology varies from system to system. Some of the most commonly used are: text, text augmented with tagging information, communicative goals, or speech signal. Apart from input data, gesture synthesis systems differ in the way motion is generated, from procedural systems to example-based systems, which usually reorganize motion capture clips from a database. So, we analyze gesture synthesis from these two point of views: input data typology and motion synthesis.

#### Synthesis systems of gestures by input data

Cassell et al. [Cassell et al., 2001] created a system called Behavior Expression Animation Toolkit (BEAT) that generates gestures using only text as input. The system analyzes the language structure of the text, and links it to gestures with a set of pre-defined rules. It generates animation commands that can be used as input in an animation system. The spark system [Vilhjálmsón, 2003] extends the BEAT system by generating gestures that relate to actions occurring in a face-to-face conversation in a virtual environment.

Pelachaud [Pelachaud, 2005] presented a system that generates expressive synthetic speech and gesturing from an Affective Presentation Markup Language (APML) file. APML is a markup language that defines a set of expressivity tags, used to augment text information. Neff et al. [Neff et al., 2008] also take a tagged text as input. In addition, their gesture generation algorithm makes use of a set of parameters captured from a specific speaker, which allow generation of gesture animation that imitates the speaker's style.

Some systems use communicative intentions or environmental constraints to generate both textual messages and gestures. Cassell et al. [Cassell et al., 1994] proposed a dialogue planner that generates a conversation between two agents based on pre-defined beliefs and goals. The system is capable of generating speech and animation to perform the conversation. Sluis and Krahmer [van der Sluis and Krahmer, 2007] defined an algorithm that generates a combined action of gesture and speech given a situation in which it was necessary to localize an object in the space. The selection of appropriate gesture and verbal message is determined by a cost function that takes in account the effort associated to perform gestures or the effort to produce a linguistic description. Kopp et al. [Kopp et al., 2008] proposed a psycholinguistic model that, given a communicative goal as input, creates interrelated iconic gestures and speech.

Other systems, such as those of Chiu et al. [Chiu and Marsella, 2011][Chiu and Marsella, 2014] or Levine et al. [Levine et al., 2009] [Levine et al., 2010], are capable of generating animations based only on speech signal. In this way the transcription of the message or the communicative intention are not considered, and it is possible to generate gestures simultaneously to speech production. These systems use the prosodic parameters present in the

acoustic signal, which are known to correlate well with emphasis [Terken, 1991]. This allows to adapt gesture production to the variations of emphasis of speech. However, prosody cues alone are not sufficient for generating any type of gestures [Levine et al., 2009] as prosody does not carry semantic content. So, its use is limited to produce only beat gestures which is our case. Moreover, the systems presented above rely on statistical models with unobserved states for modeling the relation between prosody and gestures, which leaves this correlation hidden in these models. We propose deepening the knowledge on how speech emphasis interacts with gesture production, previously defining a way to measure speech emphasis and gesture intensity, and extracting rules that govern the correlations between these. Furthermore, we present an animation system that uses these rules to generate plausible animation with appropriate gestures according to speech emphasis.

Moreover, Marsella et al. [Marsella et al., 2013] propose a method that utilizes semantics in addition to prosody in order to generate virtual character performances that are more appropriate than methods that use only prosody, like ours. Along similar lines, Sadoughi et al. [Sadoughi et al., 2014] and Lhommet and Marsella [Lhommet and Marsella, 2014] focus the generation of gesture performances by considering the semantic meaning of the discourse, the latter work by using metaphoric gestures.

#### Synthesis systems of gestures by motion generation

Some animation systems generate gesture motion procedurally [Cassell et al., 2001] [Pelachaud, 2005] [Neff et al., 2008]. A set of rules define trajectory, speed and orientation of body parts to create continuous animation. This allows for flexibility, creating all sorts of gestures by modifying parameters of the synthesizer. However, synthesizers driven by motion capture data may produce more realistic animations than rule-based (procedural) systems because they take advantage of the fact that original captured motion clips naturally contain the subtleties of human motion, which are difficult to reproduce in rule-based systems. Luo et al. [Luo et al., 2009] used mocap sequences to generate body animation. This animation is combined with arm motion created with a procedural algorithm. Stone et al. [Stone et al., 2004] use a database of mocap data with different gesture styles for generating full-body movement. They organize sound and motion segments in a motion graph [Kovar et al., 2002][Lee et al., 2002][Arikan and Forsyth, 2002] used in synthesis. A cost function is used to select transitions between nodes of the graph, based on communicative and animation smoothness requirements, to create new gesture combinations. Our approach is similar to Stone's because we also use motion graphs to connect motion units. However, our system is capable of adapting gestures to any speech input, and Stone's algorithm is limited to a pre-defined grammar. Moreover, our cost function incorporates the proposed speech to gesture synchronization rule based on strength levels.

In [Levine et al., 2009] [Levine et al., 2010], Levine et al. use Hidden Markov Models (HMM) [Rabiner and Juang, 1986] and Conditional Random Fields (CRF's) [Lafferty et al., 2001] rather than graphs to concatenate animation units. Their system uses gesture phases

as units, while our system uses gesture phrases. This results in a loss of flexibility (a larger database is required), but it may increase the perceived plausibility of the animations, because less modifications over the recorded motions are to be performed. More recently, Chiu and Marsella in [Chiu and Marsella, 2014] also use CRF's for the mapping from speech annotation while the motion synthesis uses Gaussian Process Latent Variable Models (GPLVMs) [Lawrence, 2005] to learn a low dimensional space.

## 4.4 Synthesizing gestures adapted to speech emphasis

### 4.4.1 Overview

In this section, we provide a brief overview of which phases we have accomplished to generate gestures according to an input speech. These phases are illustrated in Figure 4.4. To begin, we have captured speech and gesture representations (details in Section 4.4.2) to obtain enough data in order to study the relationship and the dependency between both cues. So, we get an all-inclusive corpus which we foremost analyzed separately. Both cues are analyzed in a similar way with the objective of classify pitch accents and strokes according to their strength (see Section 8.1.2 and Section 4.4.2).

As outputs, we get gesture strength indicator (GSI), apex time and length from gestures; and pitch strength indicator (PSI) and pitch accent peak time (Pitch Accent Peak Time (PAPT)) from speech. GSI is the value we obtained by computing the strength of a stroke gesture (details in Section 4.4.2), as mentioned, the most significance phase of a gestural phrase (GP). This indicator defines the intensity of the gesture movement. Apex time is the point of maximum extension and length is the duration of all the phases that form the GP. On the other hand, PSI indicates the intensity of the speech in an stressed syllable. Analogously to apex time, PAPT is the point which is considered as the anchor point [Leonard and Cummins, 2011] [Loehr, 2004]. More details on speech intonation in Spanish (Section 8.1.1) and the speech analysis (Section 8.1.2) performed in order to annotate and classify pitch accents can be founded in Appendix 8.1. After classifying both cues, we proceed to study their relationship by correlating strength values and time occurrences. So, from speech-gesture correlation phase (see Section 4.4.3) we obtain intensity and synchrony rules.

Our synthesis system (see Section 4.4.4) consists of two phases: graph construction and gesture production. Graph construction is an off-line process where GPs are organized in a graph structure which we have called gesture motion graph (GMG). GMG is a structured graph where GPs are catalogued according to their stroke intensity, length and apex position, and connected based on postural and velocity similarity. Once GMG is created we use it for generating gestures. Output motion is constrained by an input speech signal and synchrony and intensity rules. As mentioned before, pitch accents are manually selected from a speech cue, and these pitch accents can be accompanied or not by gestures. So, it is mandatory to decide which of them will coexist with a gesture in order to synthesize body language.

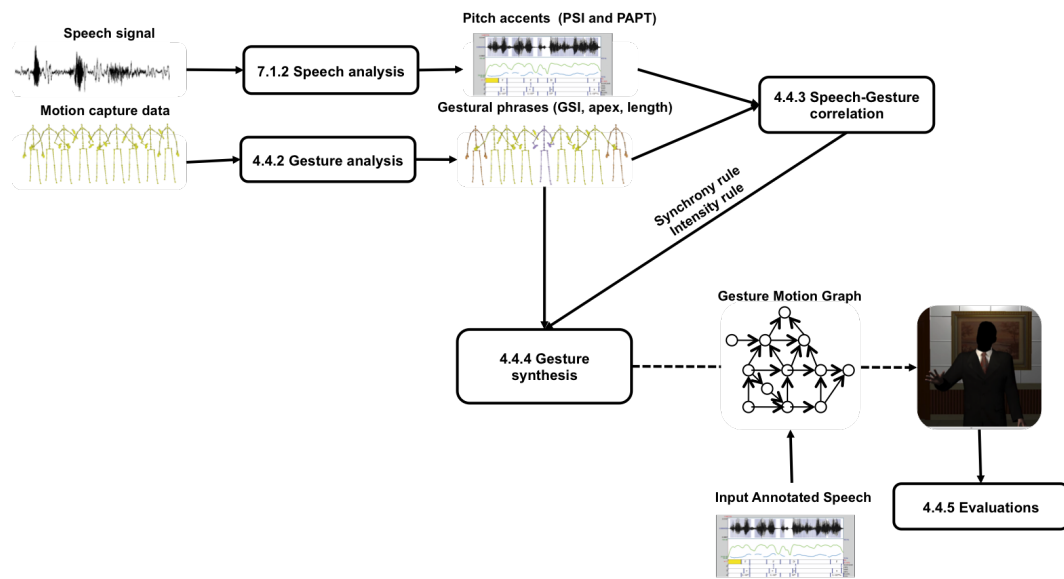


Figure 4.4: Phases workflow. We part from synchronized speech signal and motion capture data that, in parallel, are analyzed in Speech Analysis 8.1.2 and Gesture Analysis 4.4.2 phases, respectively. Then, we obtain pitch accents times and strengths from speech signal, and strokes apices and strokes strengths from motion capture data. These parameters are include in a Speech-Motion Correlation 4.4.3 process which concludes with synchrony and intensity rules. Then, a gesture motion graph is created in Gesture Synthesis 4.4.4 phase from a labeled gesture database and restricted by the mentioned rules. Finally, we generate gesture animations driven by annotated speech which we have analyzed in Evaluations 4.4.5 phase.

Given the fact that our main purpose is to verify the importance of relating emphasis of both cues (speech and gestures) to later synthesize realistic and credible animations, we have also manually select the pitch accents which will be synchronized with gestures. Expert annotators based their selection on speech intonation (without taking into account video recordings). Then, the system sequentially selects the GP which is the most suited for each pitch accent. In order to produce a plausible motion stream and smooth transitions between GPs we use motion blending. At this point, we test the plausibility of gesture animations by conducting a user test (Section 4.4.5) whose participants were asked to compare between synthesized and original gesture performances.

## 4.4.2 Gesture analysis

### Gesture and speech capture

In order to perform the following experiments it is necessary to capture synchronized gesture and speech data. A corpus composed of motion capture data and speech was recorded to obtain animations and their corresponding video recordings. The corpus consists of 6 clips that last slightly more than one minute each, in which an amateur actor with motion capture recording experience was asked to perform an improvised monologue with a concrete speaking style and performing only beat gestures. The selected styles were: aggressive and neutral. The chosen styles cover a wide spectrum of the activation dimension [Russell, 1980]. We recorded these 2 styles in order to obtain a rich gestures database, as some specific gestures are correlated to certain emotions [Kipp and Martin, 2009] and gestures differences between emotions can be explained by the dimension of activation [Wallbott, 1998].

For each style, three clips were recorded. The recording session took place at the MediaLab [La Salle Campus Barcelona - Universitat Ramon Llull., 2012] of La Salle Campus Barcelona - URL. This laboratory has 24 Vicon MX3 cameras that allow full-body motion capture with a frame rate of 120 fps. The recording set-up is illustrated in Figure 4.5.



Figure 4.5: Set-up of the recording session. An optical motion capture system with 24 infrared cameras, a clip-on wireless microphone, and a videocamera (not in the image). The actor is wearing a black motion capture suit with reflective markers.

The video channel was captured with a fixed-position video camera, whereas the speech signal with a clip-on microphone attached to the flap on the actor's suit. Both channels

were saved in a single video file. Hereafter, the video is used for the following purposes: to extract the prosody parameters, temporally aligning motion data with audio data, and to analyze the shape and timing of gestures.

The data files obtained by both equipments start at slightly different times as they are manually activated. Later, these data channels are aligned in time so as to reproduce the original sequence on the original animation mode. Anchor points in both video and motion data were created thanks to a t-pose in order to make this alignment possible.

## Video annotation

The tool selected for the annotation of the videos was Anvil [Kipp, 2001] as it was developed for video analysis of gestures and it had already been used in related works [Kipp et al., 2007]. Moreover, Anvil allows to load 3D motion capture data, which has been used in the annotation process. The annotation was divided into three groups: left hand, right hand and both hands. There are four annotation levels per hand:

- Phase: In this annotation tier all gesture phases explained in Section 4.2 are annotated.
- Stroke: This tier contains the tags assigned in the perceptual test described in Section 4.4.2. There are two options: weak or strong stroke.
- Apex: Indicates the apex time of the stroke (the point of maximum extension) that will be compared with its speech analogous (see Section 8.1.2). Usually, this time coincides with the end of the stroke for discrete strokes, although a little recoil may occur. In continuous strokes, the apex time might not be at the end of the stroke but around the middle. The apex time has to coincide with the point of maximum extension of the stroke [Leonard and Cummins, 2011]. Hence, the annotation differentiates between discrete and continuous strokes.
- Velocity: The velocity curve obtained from mocap data is loaded into this tier, so this process did not entail an annotation task. It is helpful in the annotation process, specially during the location of apex times. In discrete strokes, the point of maximum extension must coincide with a minimum of the curve. As mentioned before, continuous strokes may have the point of maximum extension around the middle. This is also reflected in the velocity curve as a minimum. For discrete gestures, we have also taken into account a technique that consists of analyzing consecutive frames to spot the apex time by detecting a change in the blurriness of the hand [Shattuck-Hufnagel et al., 2007]. Basically, the image of the hand sharpens in the point of maximum extension. However, we find faster, easier and more reliable to use the velocity curve.

As our synthesis system considers one gesture as a combination of both hands, the “phase” annotation and the “apex” tier of the left and right hands are merged in the group of “both hands”. If the annotations of each hand match, the corresponding annotation is



introduced in the “both hands tier”. However, sometimes the annotations of the left and right hand tiers are not equal. When this occurs, a decision has to be made. In general, in cases of obtaining different phases for each hand, those of greater relevance (oriented to the gesture synthesis) have preference, i.e., in decreasing order of relevance: stroke - preparation - hold. In addition, there are two more situations regarding only the stroke phases where a decision is also needed:

1. The actor sometimes performs a strong stroke with one hand that tends to produce an almost involuntary stroke of the other hand that is significantly less important. In these cases, we take the most significative stroke.
2. Two strokes from different hands may come from the same gesture, but they start and/or end at slightly different times. We have opted for selecting all the range covered by both strokes.

Following the definition of apex that we have adopted (point of maximum extension), each stroke can only have one apex time. The complete annotation process of a particular video fragment is depicted in Figure 4.6.

#### Classification of stroke strength

The main goal of classifying strokes according to their strength is to evaluate if stronger accents are associated with stronger gestures, and vice versa. To that effect, we have represented the strength level of each stroke as a numerical variable, which is denoted as Gesture Strength Indicator (GSI). This variable results from the computation of the Fast Motion Distance (FMDistance) [Onuma et al., 2008]. FMDistance is a distance function that approximates the kinetic energy of the rigid body (or set of bodies) attached to a joint, and it is used for classifying motion capture sequences. It is independent on the sequence length, and only depends on the number of joint angles involved. The main idea is to calculate the total kinetic energy of body joints considering, among other parameters, the moment of inertia ( $m$ ) of each joint. Onuma et al. states that the value of  $m$  heuristically models the moment of inertia: hip joints get high values, shoulders get a bit smaller, knees are next, elbows are next, etc. In our case, we only take into account the upper-body movement, to that effect, we select both forearms ( $m = 2$ ), arms ( $m = 3$ ) and shoulders ( $m = 3$ ). Moreover, we select the logarithmic computation of FMDistance reported in [Onuma et al., 2008], following the authors recommendations.

In order to validate this variable, a perceptual test was conducted. Specifically, two experts on video annotation carried out the first stage of the evaluation by tagging a total of 792 stroke phases, which were annotated along all the videos, as weak or strong strokes according to their perception. After this process was finished, a third expert annotator determined the final tag in case of disagreement.

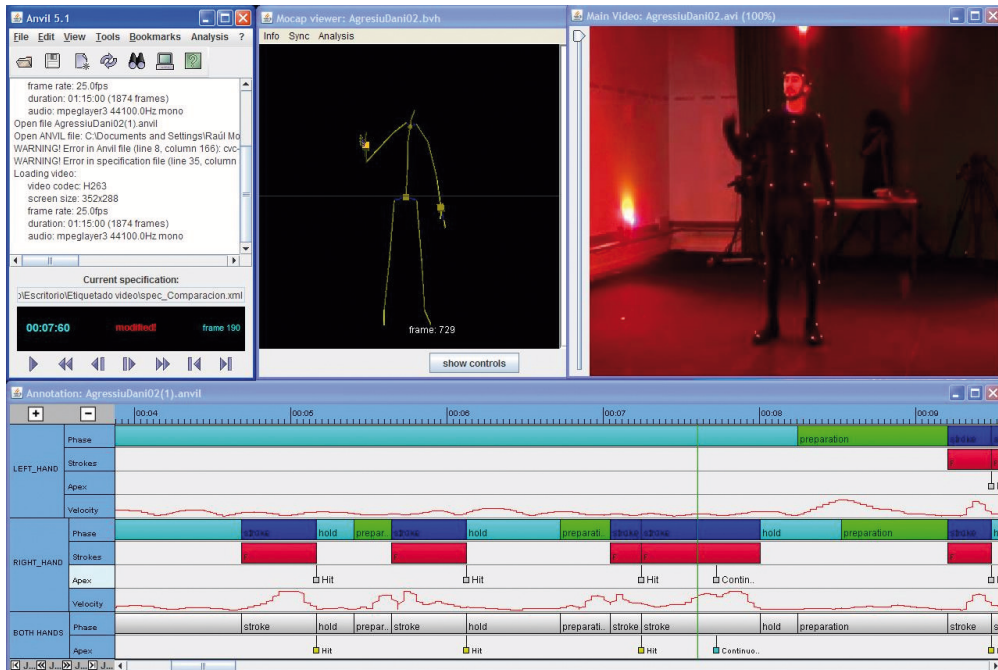


Figure 4.6: Example of the entire annotation of a particular video fragment in Anvil. The upper three windows are (from left to right): the command window, the motion capture viewer (in 3D) and the video. The latter two are manually synchronized. The bottom window is the complete video annotation. See the capture in different output formats at same time helps a lot in the annotation task.

In order to quantify how well the FMDistance discriminated between weak and strong strokes, we selected the Matthews Correlation Coefficient (MCC) [Matthews, 1975], as it measures the quality of binary classifications and, unlike the F-measure [van Rijsbergen, 1974], it also takes into account "true negatives" of the resulting confusion matrix. MCC returns a value between -1 and +1, being -1 total disagreement between prediction and observation, +1 a perfect prediction and 0 no better than random classification. The threshold value to distinguish between weak and strong strokes was experimentally determined for each method in order to maximize the MCC. The MCC coefficient obtained for the FMDistance is 0.563. This coefficient reaches 0.5 when 75% of cases are correctly predicted, so we can say that this GSI surpasses that percentage.

As well as considering the FMDistance, we have checked some other available parameters from the mocap data in order to validate the FMDistance. The parameters were: trace (covered distance of the hand), duration, maximum velocity, accumulated kinetic energy, maximum acceleration, maximum deceleration and velocity range. More than 100 classification configurations mixing these parameters were generated using logistic regression in Weka [Hall et al., 2009] to choose the best combination of parameters that could lead to

Classifier	MCC
Fisher discriminant	0.562
SPegasos	0.554
Logistic regression	0.551
SMO	0.525

Table 4.1: MCC results for all stroke classifications with 3 parameters: FMDistance, velocity range and maximum velocity.

the most appropriate measure of gesture strength. We observed the computed F-Measure by Weka so as to evaluate which parameters performed better. F-Measure measures the classification accuracy and it considers both the precision and the recall and, in our case, results as the harmonic mean of both. FMDistance, velocity range and maximum velocity obtained the highest scores so, eventually, we decided to classify using only these parameters. We used different linear classification methods as Fisher's Linear Discriminant Analysis and some methods implemented in Weka, which are: Logistic Regression and Support Vector Machine (Support Vector Machine (SVM)) using Sequential Minimal Optimization (Sequential Minimal Optimization (SMO)) [Platt, 1999] and the stochastic variant of the Primal Estimated sub-GrAdient SOLver for SVM (SPrimal Estimated sub-GrAdient SOLver (Pegaso)s) [Shalev-Shwartz et al., 2007]. In Weka, these methods return the coefficients used for the linear combination, so we could implement the classification algorithms using them and, eventually, compute the MCC. Table 4.1 shows all the classifications of the combination of FMDistance, velocity range and maximum velocity according to weak/strong tags of strokes. It can be observed that FMDistance is the best GSI candidate as no method surpasses its MCC value (0.563). Although the differences are not significant, it is to note that our intention was only to validate FMDistance as a measure for quantifying gesture strength. Hence, FMDistance is used in the following correlation analysis.

#### 4.4.3 Speech-Gesture correlation

After the gesture analysis, a total of 792 strokes were identified in the corpus (sum of right and left hand strokes). However, in the 'both hands' group (see Section 4.4.2) there were 484 strokes. As the number of pitch accents resulted in 882, the number of apex times is almost half of this. Therefore, we had to identify the most appropriate pitch accent for each apex time. Moreover, we had to establish a GSI value for the cases where a stroke in "both hands" tier came from left and right hand strokes simultaneously. We decided to choose the highest GSI value for these cases as it is usually the parameter which carries more information about the speaker gesture intention. However, in some cases, we sum both GSIs when the gesture seems to be a both hands coordinated stroke, carrying more emphasis than a stroke with only one hand.

## Correlation analysis

In order to perform the correlation analysis it was necessary to undertake a new annotation phase. For this annotation stage, we imported the "words" and "pitch accents" tiers from the Praat text grids (see Figure 8.1) to Anvil (over the previous annotation of Section 4.4.2). Then, two more tiers were created, one to indicate if the pitch accents had an associated stroke (YES/NO), and another one to indicate if the stroke in the "both hands" tier came from left hand (L), right hand (R) or both hands (B). As a result, we were able to link each gesture with its corresponding pitch accent.

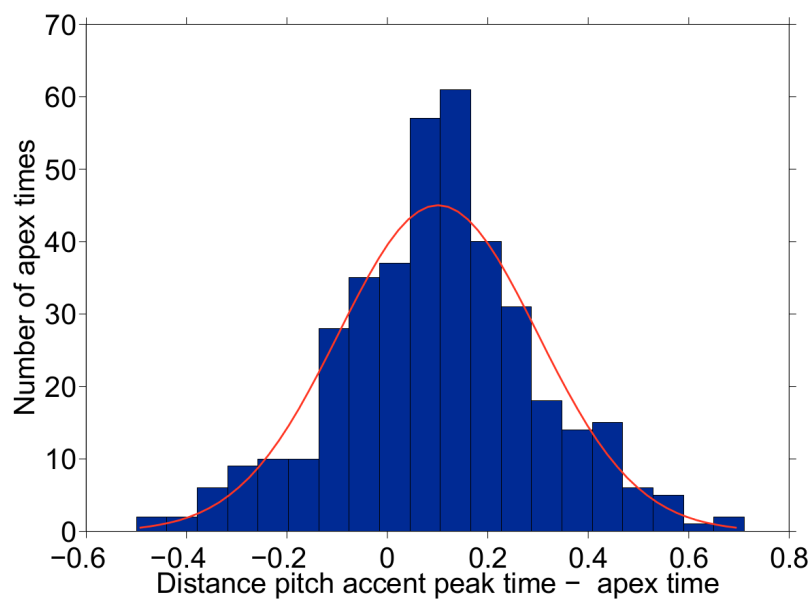


Figure 4.7: Histogram with the distances in seconds between PAPT's and apex times. Positive values indicate that the apex time comes before PAPT. Negative values indicate the opposite. The mean value is 0.1 seconds. Standard deviation is 0.2 seconds.

The first analysis considers distances between pitch accent peak times (PAPT's) and apex times. PAPT indicates the time where the pitch achieves its maximum value inside the syllable as usually the peak is taken as a point of reference [Leonard and Cummins, 2011] [Loehr, 2004]. As it can be observed in Figure 4.7, the distribution of distances is gaussian-like and it can be concluded that, in most cases (71.72%), the gesture precedes the pitch accent. This is in agreement with Kendon [Kendon, 1980] and McNeill [McNeill, 1992] observations, although our approach differs as we consider concrete speech signal and gesture anchor points. Moreover, there are contradictory results in the related literature, as in [Leonard and Cummins, 2011], where the point of maximum extension of the gesture tended to occur after a pitch accent peak with a mean of (approximately) -0.1 seconds, whereas our mean is +0.1 seconds. However, in [Loehr, 2004], a similar study revealed a

mean of 0 seconds and a standard deviation of 0.27 seconds, which are closer to our results.

For the application of the synchrony rule in the synthesis phase we ignored values greater than the 75th percentile and smaller than the 25th percentile. Therefore, distances between PAPT's and apexes must fall between -0.03 and 0.22 seconds. There are two reasons for this selection. On one hand, we wanted to discard possible outliers. We could have chosen more extreme percentiles (e.g., 90th and 10th) to deal with this but, on the other hand, we wanted to ensure that the pitch accent-gesture association was well established in the final synthesis. As the mean distance between pitch accent peaks is 0.5 seconds, values greater than 0.22 seconds could have brought the gesture too close to a neighbor pitch accent. This could lead to perceiving the gesture associated to a wrong pitch accent, which would result in a less natural animation.

The following analysis evaluates the relationship between pitch accent and stroke strengths. In order to compute the correlation between GSI and PSI, we generated a scatter plot (see Figure 4.8) and computed the Pearson's correlation coefficient. This coefficient measures the correlation of two variables (PSI and GSI in our case), so the linear dependency can be observed. Results can be between -1 and 1. Being 1 a perfect positive correlation, -1 a perfect negative correlation and 0 means that there is no linear relationship between the variables. This coefficient resulted in a value of 0.525, which is not a high correlation but it can be interpreted as a tendency. We also applied a natural logarithm transformation to the PSI variable so as to have similar values in both axes.

At first sight, some outliers are noticeable in Figure 4.8. These outliers come from two different situations: the strength indicators do not reflect precisely what it is perceived or simply there is no relationship between the gesture and the prosodic characteristics of speech signal. The PSI and GSI have been selected as they resulted in the best candidates to quantify speech and gesture strength in terms of MCC, respectively. However, PSI can be less precise in situations where the duration of the syllabic nucleus is stretched by a hesitation of the speaker (greater duration leads to a greater PSI), or when a short duration in a pitch accent perceived as strong also penalizes the pitch range parameter (as it cannot increase significantly in a short duration range), for example. This proves that perceived and computation differences exist. In the same way, GSI can also be misleading if a slow gesture has a long trace, for example, as it will most likely be perceived as a weak gesture but the kinetic energy may grow considerably. We conclude that there are situations where the prosodic strength and the gesture strength do not have a clean match. It is worth pointing out that in Figure 4.8 the distinction between aggressive and neutral does not imply that all gestures from the aggressive style are "aggressive gestures" and the same happens for the neutral style. This distinction is only based on the data obtained from each style but, for example, some gestures in the neutral style may be considered as more "aggressive gestures" than some gestures in the aggressive style and vice versa. In spite of this, as can be observed in Figure 4.8, most of the aggressive data is spread along the top-right zone of the scatter plot while the opposite occurs for the neutral data.

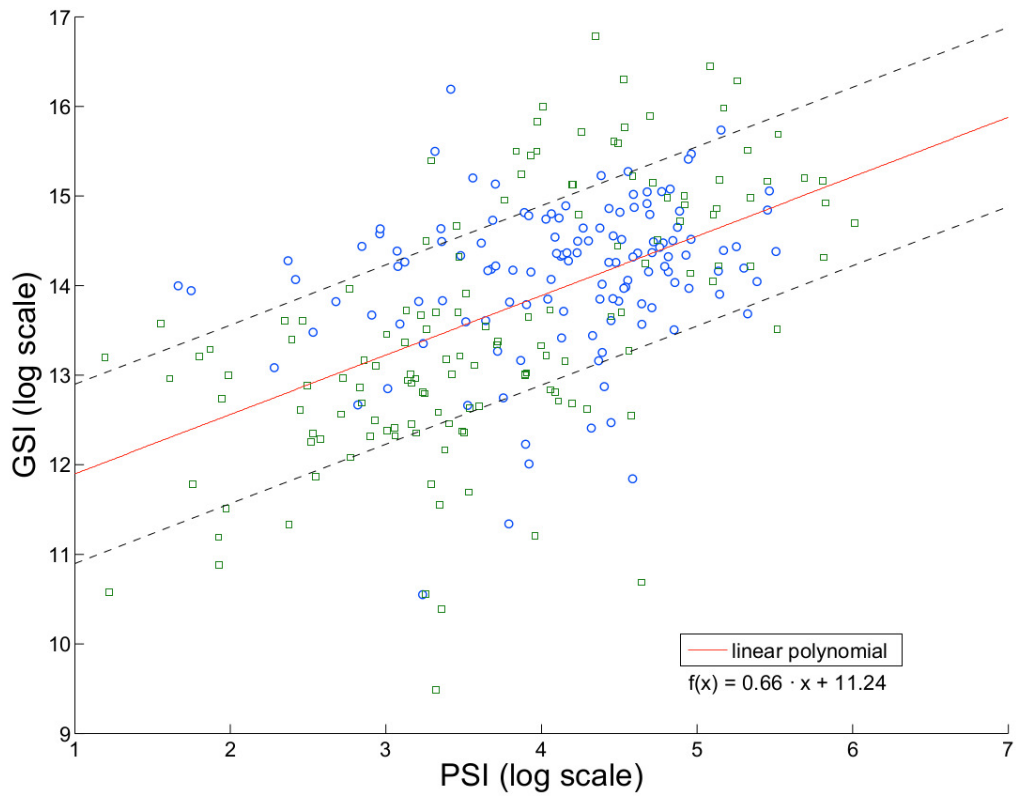


Figure 4.8: Scatter plot of the PSI and GSI values that come from aggressive (circles) and neutral (squares) styles. The linear polynomial straight line represents the correlation between PSI and GSI. The dashed lines represent the margin, experimentally determined, for the intensity rule.

## Correlation rules

In this section, the synchrony and intensity rules extracted from the data are detailed. These rules are obtained using all the corpus data and can be applied to both aggressive and neutral styles. These styles were selected in order to cover different levels of emphasis, but our intention was not to restrict the rules to these styles only. Other speaking styles could be suitable to work with these rules. However, this would require additional testing.

The synchrony rule is derived from the correlation analysis. We obtained a synchrony window of -0.03 to 0.22 seconds, being positive values an anticipation of the stroke apex relative to the pitch accent peak, whereas negative values indicate the opposite. This window defines our synchrony rule.

For the intensity rule, we derive two interpretations based on the results (see Figure 4.7 and Figure 4.8). If we take the linear polynomial line (see Figure 4.8), we can say that, given a PSI value, the GSI has to be near this straight line. However, according to the scatter plot, we define a margin of GSI-PSI correspondences. Experimentally, we have set this margin to 1 above and below the linear polynomial line (see Figure 4.8). The constants from these equations come from the analysis of our data. The intensity rule is defined by the following equation:

$$GSI \in (0.6633PSI + 10.24, 0.6633PSI + 12.24) \quad (4.1)$$

These rules have been extracted after a meticulous analysis of the corpus at hand, and they will be used in the synthesis phase in order to recreate synthetic animations. These rules are implemented in the cost function, which selects a GP according to the speech input. On one hand, the synchrony rule defines how the GPs and the pitch accents are synchronized and which are the desired GP durations. On the other hand, the intensity rule drives which GP is more appropriate according to the strength indicators.

### 4.4.4 Gesture synthesis using a Gesture Motion Graph (GMG)

#### Constructing a GMG

We implement a variant of the original motion graph (see Section 2.3) in order to generate gestures. A Gesture Motion Graph (GMG) consists of a set of nodes  $N$ , a set of edges  $E$  and edge weights  $W$ .  $N$  represents GPs as defined in Section 4.2. Elements of  $E$  are edges connecting GPs which are consecutive in the original motion capture recordings or have similar initial and final postures (extreme postures), so graph edges are directional.  $W$  is a set of edge weights that describe connections in order to drive gesture selection.

Before graph construction, we have manually segmented motion capture data into distinct GPs. These GPs are also segmented in different GP phases in order to detect stroke phases. Each stroke is analyzed to know its apex and GSI. All these outputs are achieved in the same way as explained in Section 4.4.2.

First, we connect GPs that can be displayed one after the other without any transition between them. We only permit this connection when GPs are consecutive in the original recordings. Then, we also connect GPs whose extreme postures do not exceed the similarity threshold which is pre-specified by the user. So, we compute posture and velocity similarities between final and initial frames of GPs in dataset. This computation depends on the chosen distance metric [van Basten and Egges, 2009]. In this work we use joint angles distance metric proposed in [Lee et al., 2002]. Our skeleton has 42 joints, but we only take into account a reduced set of joints to compute posture similarity. Our objective is to animate the upper-body of a character, for that reason, we do not need to use posture similarity from other parts of the body, nor need the global position. Just spines, neck, rightarm and leftarm, and rightforearm and leftforearm have a weight of 1, the rest is set to 0. Joint weights are referred in Figure 4.9. In this work, we do not assign specific weights to joints, to that effect, all joint considered have the same relevance in the similarity distance metric.

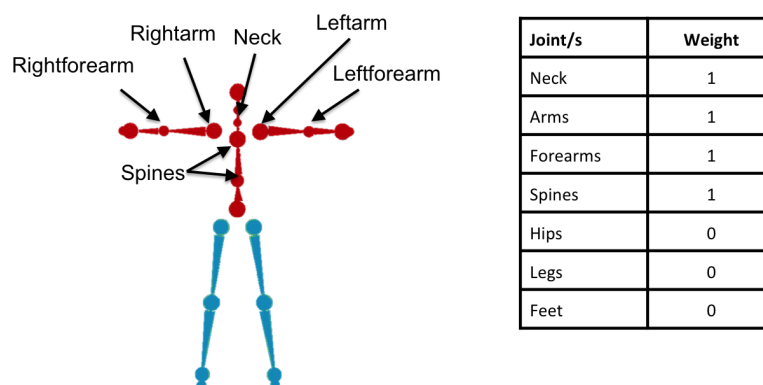


Figure 4.9: Joints weights. Spines, neck, rightarm and leftarm, rightforearm and leftforearm have a weight of 1, and the rest of body joints have a weight of 0.

Once we have computed pose distances we need a similarity threshold  $st$  to create edges between similar frames. Similarities under this threshold will lead to connect GPs with a directional edge. So, after connecting consecutive and similar GPs we obtain a GMG as described in Figure 4.10. As we can see, there are two type of edges:  $E_1$  and  $E_2$ .  $E_1$  denotes that GPs are consecutive. On the other hand,  $E_2$  denotes that GPs have similar initial and final poses, so a transition will be required.

In order to use that graph for gesture synthesis we set edge weights (not to be confused



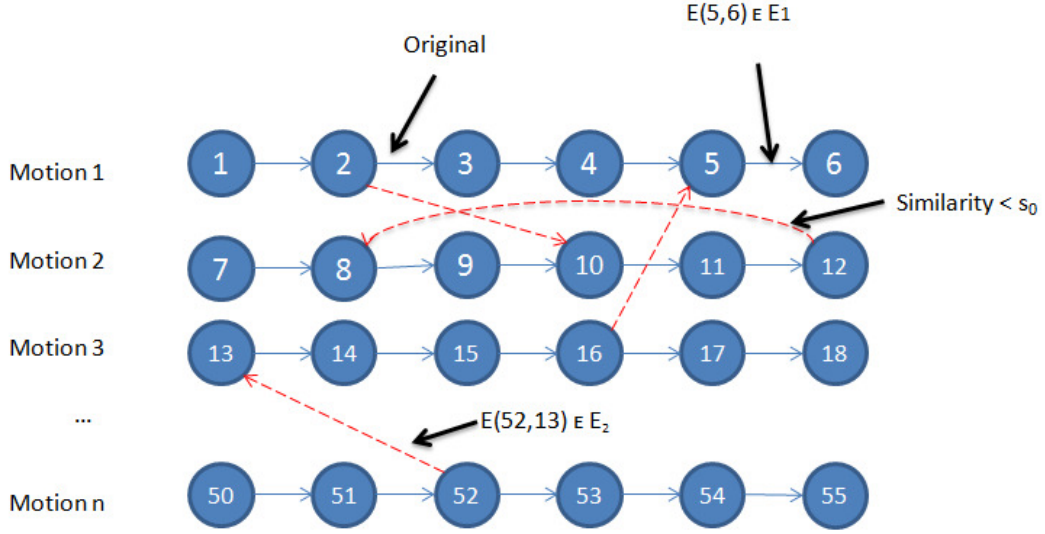


Figure 4.10: Edges generation. Edges in GMG connect consecutive GPs in original recordings ( $E_1$ ) as painted with continuous lines, and GPs with a posture similarity under a similarity threshold  $st$  ( $E_2$ ) as painted with dashed lines.

with joint weights  $\psi_k$ ) between GP  $m$  and GP  $n$  as

$$W(E(m, n)) = \begin{cases} 0 & \text{if } E(m, n) \in E_1 \\ d(m_\omega, n_\alpha) & \text{if } E(m, n) \in E_2 \end{cases} \quad (4.2)$$

where  $m_\omega$  is the final frame of the  $m$ -th GP,  $n_\alpha$  is the initial frame the  $n$ -th GP, and  $E_1$  and  $E_2$  are two edge subsets, which are illustrated in Figure 4.10. Smoothness weight is 0 when  $E(m, n) \in E_1$  and  $d(m_\omega, n_\alpha)$  in case of  $E(m, n) \in E_2$ .

Until now, several GPs are connected but some of them are dead ends and this will become a problem if they are selected in the gesture synthesis phase. In order to solve this, we use Tarjan's algorithm [Kovar et al., 2002] to compute the strongly connected components (SCC) of GMG. In this manner, we optimize our motion graph by deleting useless edges. Our dataset is formed by 261 gesture phrases allowing a wide variety of gestures. Table 4.2 shows the original size, original branching factor, SCC size and SCC branching factor for different graphs created from specified similarity thresholds.

As usual, branching factor increases as similarity threshold values increase due to the fact that less edges are deleted. In order to enrich as much as possible our dataset and at same time improve the quality of output motions, we chose the minimum similarity threshold (1.0) where all gesture phrases were included in a SCC graph. As a consequence, we

Similarity threshold	Original size	Original BF	SCC size	SCC BF
0.3	261	0.92	18	1.22
0.4	261	1.40	74	1.68
0.5	261	2.71	160	2.88
0.6	261	5.47	226	5.46
0.7	261	9.71	247	9.87
0.8	261	16.24	252	16.37
0.9	261	25.09	259	25.27
1.0	261	36.20	261	36.20

Table 4.2: Graph results varying similarity threshold from 0.3 to 1.0. Original size denotes the number of nodes initially belonging to GMG. Original BF defines the branching factor of graphs after deleting edges that overcome the specified similarity threshold. BF is computed as the number of edges divided by the number of nodes. SCC size describes the number of nodes belonging to the largest strongly connected component. Finally, SCC BF denotes the branching factor of the resulted SCC.

chose a blending window of 0.25 seconds and we apply start-end blending scheme [Wang and Bodenheimer, 2008] to create smooth transitions.

### Synthesizing gestures

In this subsection we describe how we select GPs in GMG. As input, we have the previous displayed GP (except at the beginning of motion stream, where rest pose is used) and the next pitch accent, which includes PAPT and PSI data. First, we search for all candidate GPs that are connected to the previous one selected. Then, we adjust candidate GPs to fit with speech anchor points (1. Temporal alignment). Finally, we compute a distance metric for all candidates and select the GP with the minimum cost (2. Selecting a gesture phrase), and display it with proper time adjustments.

#### 1. Temporal alignment

As mentioned in Section 4.4.3, we notice that stroke apex do not coincide exactly with PAPT and furthermore, apex is differently located for each GP despite that apex is usually at the end of stroke. We refer to the deviation between the apex and the PAPT as anticipation time ( $T_a$ ), and apex time ( $T_{ap}$ ) as the difference between the end of a GP and its apex. To match gestures with speech signal, first, we need to specify anchor points. Anchor points define the synchrony between gestures and speech.

In Figure 4.11 a possible gesture stream which is synchronized with anchor points is shown. To specify them, an anticipation time for each PAPT has to be sequentially deter-

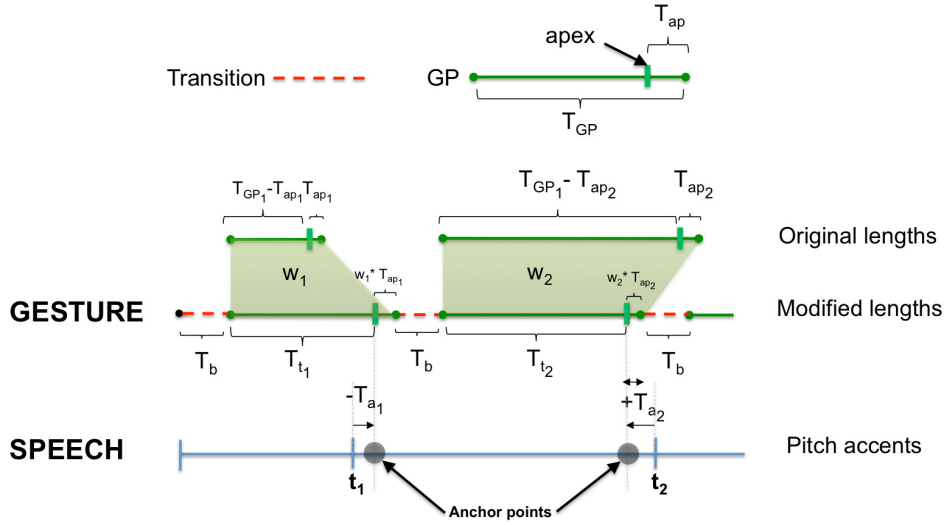


Figure 4.11: Gesture phrase alignment with anchor points. Dashed lines denote transition motion segments and continuous lines gesture phrases. Apex from gesture phrases are marked with vertical lines. There are two timelines, one for each cue (gesture and speech). Anchor points define the gaps where gestures have to be fitted.

mined. We used a random value within the time window defined in Section 4.4.3, where positive values denote that gesture anticipates to PAPT and negative values the contrary. Once we have a new anchor point, we have to fill the gap between this point and the previous one (or the initial time) with a candidate GP, adding a transition before it if  $E(m, n) \in E_2$ , where  $m$  is the previous GP and  $n$  is the next GP candidate. Rarely, the apex of any candidate GP will match with the anchor point. As we can see in Figure 4.11, we expand or compress candidate GPs to meet the required target length ( $T_t$ ). So, for each candidate GP there is a time warping factor ( $w$ ) that leads to fill the target length ( $T_t$ ) with the initial part of a candidate GP (up to its apex). The resulting length of the piece from the apex to the end ( $w \cdot T_{ap}$ ) is used to compute the next  $T_t$ . Therefore, for each pitch accent  $i$  we compute the target length ( $T_t$ ) by

$$T_{t_i} = (t_i - T_{a_i}) - (t_{i-1} - T_{a_{i-1}} + (w_{i-1} \cdot T_{ap_{i-1}})) - T_b \quad (4.3)$$

where  $t_i$  is the PAPT of the current pitch accent,  $T_{a_i}$  is the anticipation time of  $t_i$ ,  $w_{i-1}$  is the time warping factor and  $T_{ap_{i-1}}$  is the apex time, both from the previous GP. Finally,  $T_b$  is the blending window (0.25 seconds) for transitions. It is important to remark that  $T_b$  is only included in this formula when a candidate GP is connected by an edge belonging to  $E_2$ .

As we have mentioned, a time warping factor ( $w$ ) is computed for all candidate GPs. Time warping factor is defined by

$$w = \frac{T_t}{T_{GP} - T_{ap}} \quad (4.4)$$

where  $T_t$  is the target length of a GP up to its apex,  $T_{GP}$  is the length of a GP and  $T_{ap}$  is the distance from the end of a GP to its apex.

## 2. Selecting a gesture phrase

To select a GP we compute a distance metric for all candidate GPs taking into account 3 features: the similarity between the frames that will be joined with a transition (smoothness cost), the intensity relation between speech and gesture (intensity relation cost) and how the candidate GP will be affected by the time alignment process (time warp cost). Each feature is evaluated with a cost, and later a total cost is computed as our distance metric. We use Breadth-first search (BFS) algorithm [Tarjan, 1972] in order to search the minimum cost.

**Smoothness cost.** Achieving smooth transitions between GPs is the first criterion of selecting a gesture phrase. Smoothness cost  $S$  between  $m$ -th (previous selected GP) and  $n$ -th GPs (candidate GP) is the normalized edge weight  $W(E(m, n))$ . We use a max-min normalization from  $[0, s_0]$  to the range of  $[0, 1]$ , where  $st$  is the similarity threshold used in graph construction.

**Intensity relation cost.** Intensity relation cost is based on the intensity rule from Section 4.4.3 Formula (3) defines a region where GSI have a correlation with PSI, so intensity relation cost is

$$I(m, n) = \begin{cases} 0 & \text{if } GSI_n \in (0.6633 \cdot PSI_i + 10.24, 0.6633 \cdot PSI_i + 12.24) \\ p_I & \text{otherwise} \end{cases} \quad (4.5)$$

where  $GSI_n$  is the gesture strength indicator the  $n$ -th GP (candidate GP),  $PSI_i$  is the prosody strength indicator of pitch accent  $i$  and  $p_I$  is a penalty parameter. We set  $p_I$  as 10 and we just use it to discard GPs that do not meet the intensity rule. In this manner, we avoid to select penalized GP's due to its final score is significantly more than not penalized.

**Time warp cost.** We fix boundaries for time warping in order to preserve the maximum fidelity with the original GP. We use a  $MAX\_WARP$  as maximum time warping factor for expanding GPs and its inverse to compress them. Therefore, time warp cost is computed as

$$T(m, n) = \begin{cases} \max\_min(w) \text{ to } [0, 1] & \text{if } 1 \leq w \leq MAX\_WARP \\ \max\_min(w) \text{ to } [1, 0] & \text{if } \frac{1}{MAX\_WARP} \leq w < 1 \\ p_T & \text{otherwise} \end{cases} \quad (4.6)$$

where  $w$  is the time warp factor (see Formula 4.4) of the  $n$ -th GP (candidate GP) and  $MAX\_WARP$  is the MAXimum time WARP factor allowed. We empirically determine this value as 1.5. We set this value from the observation of the first gesture motion tests by paying attention to the credibility of the synthesized motion while varying  $MAX\_WARP$ . Finally,  $p_T$  is a penalty parameter set to 10. As in intensity relation cost, we use  $p_T$  to penalize GPs which exceed our boundaries due resulting motions are perceived as non-realistic.

Total cost. Finally we join all feature costs in a global distance metric which is defined by

$$C(m, n) = w_S S(m, n) + w_I I(m, n) + w_T T(m, n) \quad (4.7)$$

where  $w_S$ ,  $w_I$  and  $w_T$  are weights to define the importance of each feature in the total cost. In our implementation, we set all weights  $w_S$ ,  $w_I$ ,  $w_T$  as 1.0. So, we do not prioritize any weight above the other which could be subject of further investigations on generating appropriate gestures to speech input. Once we obtain all costs for candidate GPs, we select the GP with the minimum total cost.

#### 4.4.5 Evaluations

Each person gesticulates in a different manner when speaking, so there is no precise way to measure the quality of synthesized body language. Therefore, human observation is the best way to qualify our results. We have conducted a user test in order to compare our method using distinct combinations of extracted rules for synthesizing animations. Synthesis was driven by audio clips with different emphasis levels from 9 to 15 seconds in length. In this test, people were asked to discriminate between pairs of videos using the online platform Testing platfoRm for mUltimedia Evaluation (TRUE) [Planet et al., 2008]. TRUE is a free software platform oriented to the creation and management of online subjective and perceptual tests where different stimuli (audio, audiovisual, graphics and text) can be evaluated by human evaluators to validate or label them.

#### Animations setup

The test was composed of a set of synthetic and real captured animations. All animations belonged to one of these four types:

- Original animations, containing gesturing and speech of a real person.
- Intensity animations, which are synthesized using the time-synchrony and intensity rules.
- Time animations, which are synthesized using the time-synchrony rule only.
- Random animations, which are synthesized without using any of the synchrony rules. However, gestures are well formed and resulting animation is convincing.

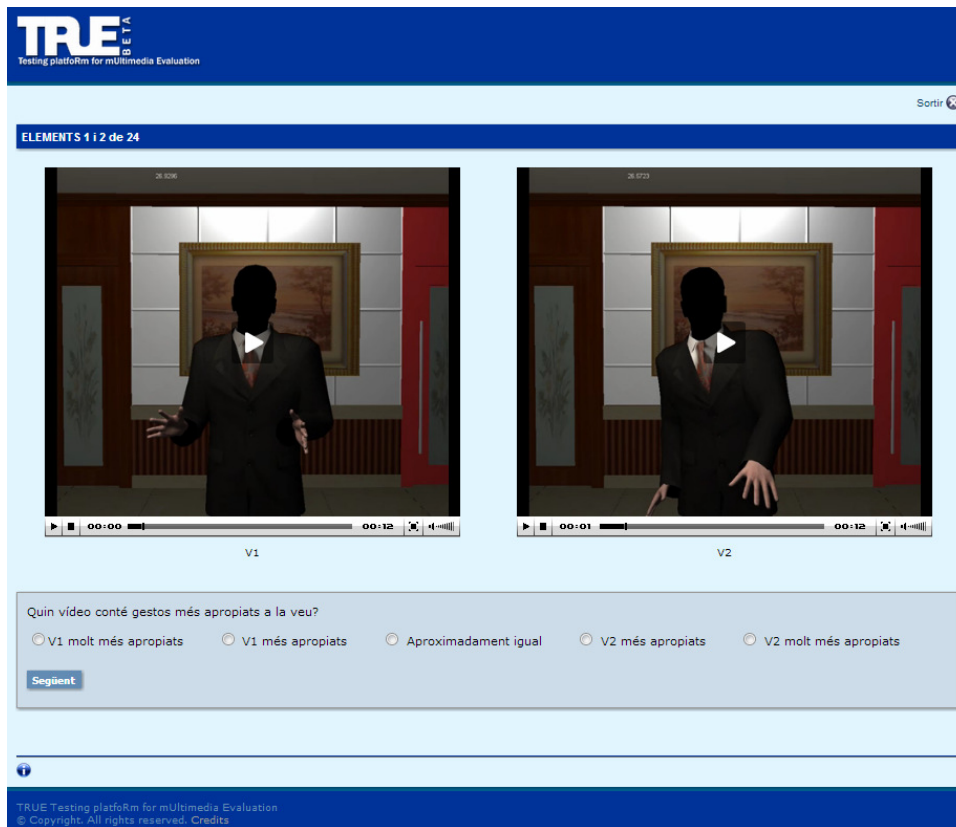


Figure 4.12: Snapshot of one step of the test using the online platform TRUE [Planet et al., 2008]. Every step contains two videos which have to be evaluated by the tester.

To generate synthesized animations we used a gesture motion graph (GMG) populated with gestures from aggressive and neutral styles of the corpus described in Section 4.4.2.

Two audio clips of one minute length were used to drive the synthesis. One audio contains speech with aggressive style and the other neutral style. These two styles cover different levels of emphasis. Aggressive speech is mainly emphatic, while neutral speech is generally unemphatic. Gestures of the speaker were captured in a mocap system at the same time as the audio was recorded. These motions are necessary for reproducing original animations.

Each type of animation required a different generation process: to create original animations, the gestures captured at the same time as speech were used. To create intensity animations, the system described in Section 4.4.4 was used. PSI values for the selected pitch accents were computed as described in Section 8.1.2. To create “type time” animations, the same process used for intensity animations was used. However, the PSI values are not computed based on corresponding pitch accents, but on a random variable. In this way, the

intensity rule is deactivated, and at the same time, the restrictions that are applied to intensity animations are not relaxed for time animations. Finally the desired gesture length and the assignation of PSI values were randomized in an attempt to create random animations.

To create the test, three utterances from 9 to 15 seconds were selected from each of the two audio clips. Utterances start and end with silences in the speech signal. Complete animations (one minute length) have been cut to coincide with the limits of the selected utterances. As mentioned in Section 4.4.4, the GMG chooses the best gesture phrase based on a set of restrictions and costs. However, due to a poor population of the GMG, in some cases the selected gesture may not be appropriate. That is, a gesture that exceeds the maximum warp or does not meet the intensity rule. The synthesizer sets a cost of 10 to the transitions to these type of gestures. In order to avoid the presence of invalid gestures in the test, animations containing transition costs of 10 are discarded.

### Test setup

The test is based in the P.800 recommendations of the International Telecommunication Union (ITU) [ITU-T, 1996]. More precisely, it was used the Comparison Category Rating (CCR) method, which consists of showing two side-by-side stimuli for every step of the test. In our test, stimulus are videos showing animations synthesized with different processes. Users are asked to rate the quality of each stimulus compared to the other stimulus of the same step. Therefore, this test allows to be measured the relative quality between the outputs of several processes.

The processes tested here are: random, time, and intensity, which are described in the previous section. These are compared between themselves, and also with unprocessed original videos. Users are asked which video contains more appropriate gesturing depending on the speech (see Figure 4.12). No more instructions were given to users, who were able to play videos as many times as they needed to answer each question. Every step of the test shows a video in the left side of the screen, tagged as "V1", and another one on the right side, tagged as "V2". For every pair of videos, users must choose one of the answers shown in Table 4.3. When an answer is chosen, the system automatically gives a quality value to both stimuli. Table 4.3 shows the values given to both videos for every answer. These values are used to compute a Comparative Mean Opinion Score (CMOS), which is an indicator of quality. Higher CMOS values mean higher quality outputs for a process.

Six different types of comparisons were considered. These are shown in Table 4.4. In order to avoid biased results due to prejudices, users did not know which type of comparison was being performed at every step of the test and the left-right ordering of the clips was randomized. Moreover, the six types of comparisons were randomly alternated to avoid the user learning patterns on how these are organized. Each user rated 12 comparisons from a total of 36 (6 utterances and 6 types of comparisons). Users rated different comparisons to cover the whole set.

Option name	Value for video1	Value for video2
V1 is much more appropriate	+2	-2
V1 is more appropriate	+1	-1
Approximately the same	0	0
V2 is more appropriate	-1	+1
V2 is much more appropriate	-2	+2

Table 4.3: Possible answers for each step of the subjective test and its corresponding quality values.

intensity vs. original
intensity vs. time
intensity vs. random
time vs. original
time vs. random
original vs. random

Table 4.4: List of all types of comparisons between animations.

#### 4.4.6 Results and discussion

The test was conducted by 62 evaluators, who performed two comparisons of each type. Thus, we obtained 124 preference values for each type of comparison. Evaluators come from different backgrounds, however, since the test was promoted from an engineering school, they are mostly engineer students or graduates. Their ages range from 16 to 64 years old with a mean age of 31, and 69% of them are male.

The answers of the evaluators are summarized in Figure 4.14. For each answer we obtained a value between -2 to 2, as shown in table 4.3. These values are used to compute the CMOS. Boxplot in Figure 4.13 depict the results of the pairwise comparisons (method A vs. method B) for the four types of animations, showing to what extent the former gesture synthesizer method is better (indicated as positive values) or worse (indicated as negative values) than the latter.

The users' preference for the intensity animations, obtained using the intensity and synchrony rules, can be clearly observed when compared to time animations or random animations. Intensity animations were valued as more appropriate than random animations in 65.3% of the cases, while only 8.0% valued them as worse. In 67.7% of cases, intensity animations were valued better than time animations, while in 19.4% were valued as worse.



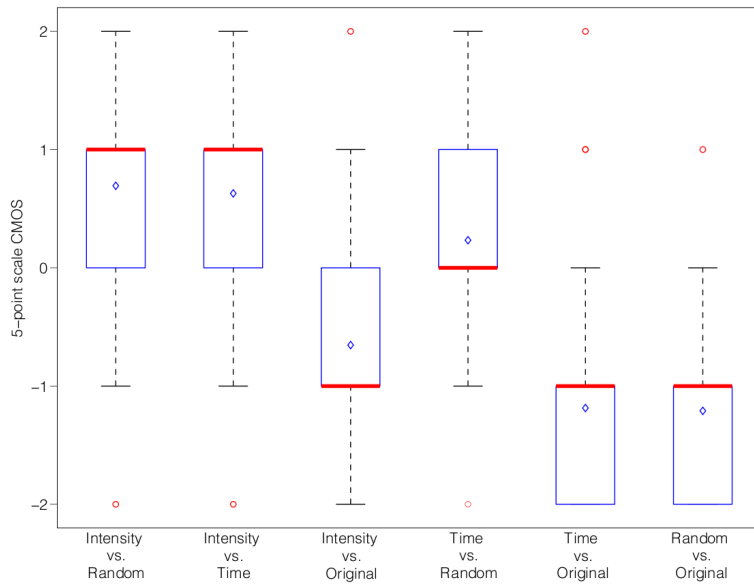


Figure 4.13: Five-point scale CMOS responses of users' preferences when comparing two videos generated with different methods (see Section 4.4.5 for methods descriptions). The diamond marker within the boxplots represents the distributions mean values.

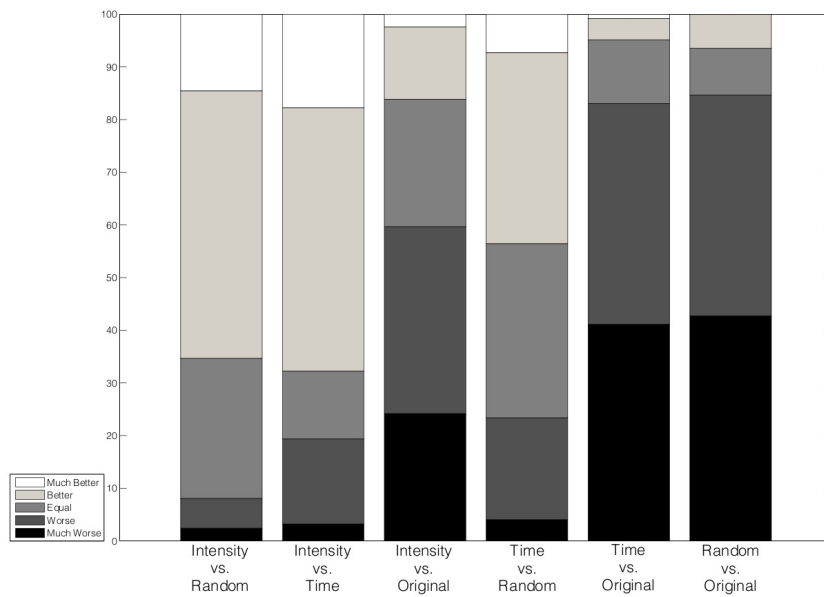


Figure 4.14: Preference bars. They show the percentages of users ratings for each comparison.

Furthermore, time animations, which only use the time synchrony rule, were valued better than random (43.5% better valued, and 23.4% worse valued), although the proportion of positive votes is smaller than for intensity animations.

Moreover, it can be depicted that original animations are more positively valued than any other type of animation. Even so, in 40.3% of the cases the original animations are not valued as well as intensity animations. This percentage drops to 16.9% and 15.3% when compared to time and random animations.

In order to evaluate the statistical significance of these results, a paired, two-tailed t-test comparing the users' preference for each animation synthesis method is computed. Results show that intensity animations are valued more positively than random animations (median = 1 and mean = 0.69) and better than time animations (median = 1 and mean = 0.63) with a confidence value of  $p < 0.001$ . Thus, these results lead us to conclude that the intensity rule, which is only applied in intensity animations, improves the appropriateness of gestures to its accompanying speech.

Moreover, all types of animations are lower rated than the originals (median = -1 and mean = -0.65 when compared to intensity animations, median = -1 and mean = -1.19 for time animations, and median = -1 and mean = -1.21 for the comparison to random), with a confidence value of  $p < 0.001$  for all cases. However, it can be observed that the intensity rule helps intensity animations to get closer to the quality of original animations than any other method.

We also observe that time animations are more positively valued than random animations (median = 0 and mean = 0.23) with a confidence value of  $p < 0.01$ . These are the least significant of all results, strengthening the idea that the temporal synchrony rule alone does not suffice to create gestures which are appropriate for its accompanying speech.

We have compared these results with the ones obtained by Levine at Gesture Controllers [Levine et al., 2010]. Their work presents a system which, like ours, is capable of generating gesture animation by using solely prosody parameters from speech signal as input. In Gesture Controllers, results are shown in a three-level scale ("prefer left", "undecided" and "prefer right"). In order to be able to compare results, our five-level scale has been reduced to three levels. "Much better" and "better" answers are assigned to +1 (equivalent to prefer left), and "worse" and "much worse" to -1 (equivalent to prefer right). This allows the computing of comparable CMOS values. Gesture Controllers present different results depending on whether gestures used in synthesis come from an actor or come from an untrained speaker. Since gestures of our corpus come from an amateur actor, who produced complex gesturing (most unit gestures are composed by two or more gestures, which produces more natural and trustworthy animations [Kipp et al., 2007]) our results are compared to an actor's results. Comparisons between a previous work by Levine [Levine et al., 2009] and original animations (labeled as Motion Capture) give a CMOS = -0.71. Comparisons between Gesture Con-

trollers [Levine et al., 2010] and original animations give a CMOS = -0.46. Finally, intensity animations from our work compared to original animations gives a CMOS = -0.44. Although all results are negative, our work gives a better score than the first work by Levine and similar to Gesture Controllers.

Results reveal that neither of the two systems (Gesture Controllers and ours) create gesture animations that are as natural as those created by a real person producing complex gesturing. However, both systems improve the results obtained by the previous work by Levine et al. [Levine et al., 2009]. The main advantage of Gesture Controllers is that it proposes a completely unsupervised system, while ours requires manual selection of pitch accents, and supervision in the annotation process for gestures and speech. A validation of the impact on the automatization of these processes would be required in order to perform a fair comparison between these two works. Anyhow, our work focuses in synthesizing gestures that accompany speech with varying levels of emphasis, and this variability is included in our test. Although Gesture Controllers is capable of generating gestures accompanying speech with different tones (sad, neutral or excited), it is not clear if their test incorporates such variety.

Our work, besides defining an animation synthesizer for gestures, uses this system to evaluate whether the intensity rule is relevant for gesture synthesis or not. This leads to a deeper understanding of the correlations between speech and gesture. In Gesture Controllers, this knowledge remains hidden inside the probabilistic models they use.

At the end of the test, users were asked which aspects of the videos they have paid attention to in order to answer the test. Although this was an open question, it is interesting to note the responses that were repeated most often. Many users (40 evaluators over 62) coincided in the appreciation that they watched the movements of hands and arms. However, there were several evaluators who also paid attention to head movements (9 evaluators), torso (4 evaluators), and shoulders (3 evaluators). Leg movements were deactivated to avoid foot sliding problems. This shows that although hands and arms play a major role in the perception of good quality body language, other parts of the body may also affect it. In our system, only hand and arm motions are labeled as gestures. It would be also interesting to annotate gestures produced by other parts of the body and evaluate improvements in animation quality.

Another aspect worth noting from the evaluators' comments is the type of synchronization between gesture and speech which they paid attention to. 29 evaluators mentioned that they valued the proper relation between the tone of the voice (relaxed or excited) and the intensity of gestures. This highlights the importance of a good correlation between emphasis levels of the two channels, which is the main contribution of our work. Far fewer evaluators (5) paid attention to the temporal synchrony. Results of the test also point to a difference in the relevance of these two synchronies, which shows a greater rating of intensity animations in comparison to time animations, than time animations in respect to random. All this leads us to conclude that the intensity rule is more relevant than the temporal synchronization

rule, at least in scenarios with varying emphasis.

It is also interesting to point out that only one evaluator mentioned using the quality of the animation as an evaluation parameter. This leads us to believe that the quality of animations produced by concatenating gestures with the GMG is generally similar to the original animations. If transitions were unrealistic, this would probably be a more common observation.

## 4.5 BodySpeech: Facial and gesture animations

In the previous Section, we have explained a semi-automatic gesture synthesis method due to speech signal have to be manually segmented. So, we have enhanced it in order to fully automatized gesture synthesis from a speech cue giving rise to BodySpeech.

We have implemented the BodySpeech system as a plugin for the Unity3D game engine [Unity Technologies, 2014b]. The Unity editor was used to create a visual interface that allows generating animations by selecting input speech audio files. In addition, the application uses Microsoft Speech API, Speech API (SAPI) [Microsoft, 2013] to detect speech phonemes, and the Tagarela plugin [Rodrigo Pegorari, 2013] for facial morphing.

The animation system is divided into two stages: an off-line preprocessing step and a runtime unsupervised step. Figure 4.15 shows an outline of the whole system. In the first stage, gesture mocap data is arranged in a motion graph structure as described in Section 4.5.1. On the other hand, we associate visemes (mouth shapes) with phonemes. Vowel phonemes have more than one associated viseme in order to capture emphasis in facial animation. Visemes parameterization is further explained in Section 4.5.1.

The second stage is where the output animation is generated. Speech is used to drive both gesture synthesis and facial animation. Input speech is analyzed in order to detect pitch accents (time occurrence and strength indicator) and the phoneme transcription of the message (see details in Section 8.2). Pitch accents drive gesture synthesis by selecting the most appropriate gesture unit for each one depending on strength levels (see more details in Section 4.5.2). Again, we use gestural phrases as gesture units. At the same time, phonemes intervals are matched with visemes to generate facial animation. Additionally, visemes are modified based on pitch accent strength indicators (see Section 4.5.2).

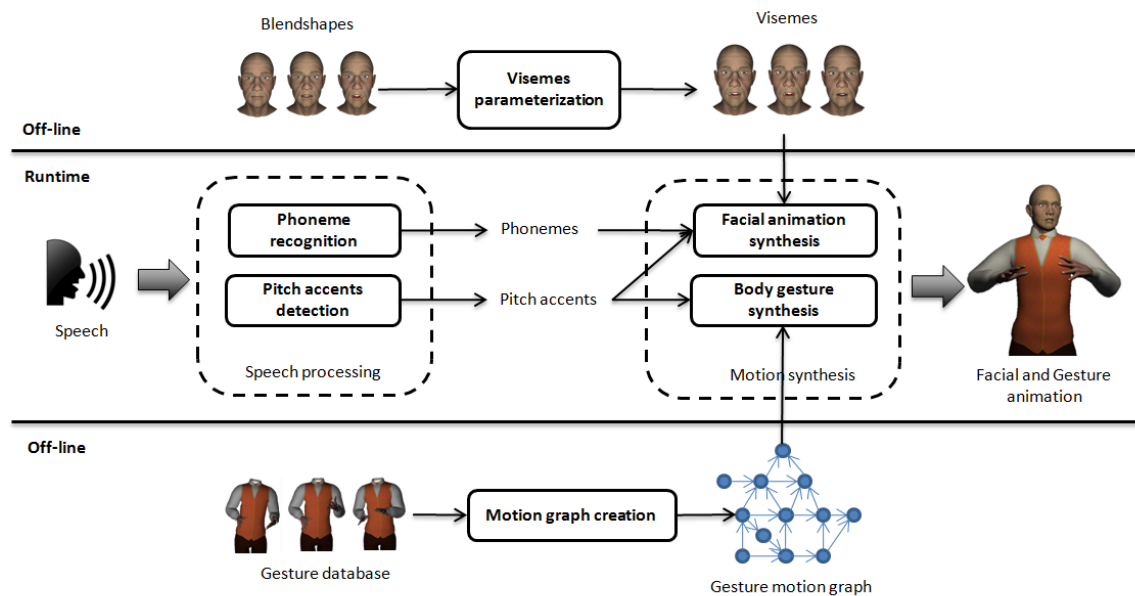


Figure 4.15: System overview. The off-line stage is used to generate a motion graph (at the bottom) and a set of visemes (on top). Then, body and facial animation are obtained from a speech signal in the runtime stage (on the middle).

#### 4.5.1 Setting up

##### Gesture motion graph construction

The application is able to parameterize the process of gesture motion graph (GMG) construction. The application is divided into three parts: New profile, Load profile and Player. New profile permits to generate a custom GMG (see Figure 4.16) and visemes; Load profile allows to select a saved profile; and Player lets to replay previous generated animations.

The process of generating new GMG's can be configured with the following parameters: joint weights (they are used in posture similarity distance metric) and similarity threshold. Altering these parameters the GMG is modified. Moreover, the user can select one or several motion capture databases to be used as source of GP's for the GMG. This allows increasing the size of the GMG which in turn improves animation richness. Branching factor is displayed in the interface to lead animators know the richness of generated graphs.

A labeled gesture motion database is used to construct GMG's. We use a database that consists of 6 clips that last slightly more than one minute each, in which an amateur actor with motion capture recording experience was asked to perform an improvised monologue with a concrete speaking style (aggressive and neutral) and performing only beat gestures. This is the same database as explained in 4.4.2. Furthermore, we have divided these clips

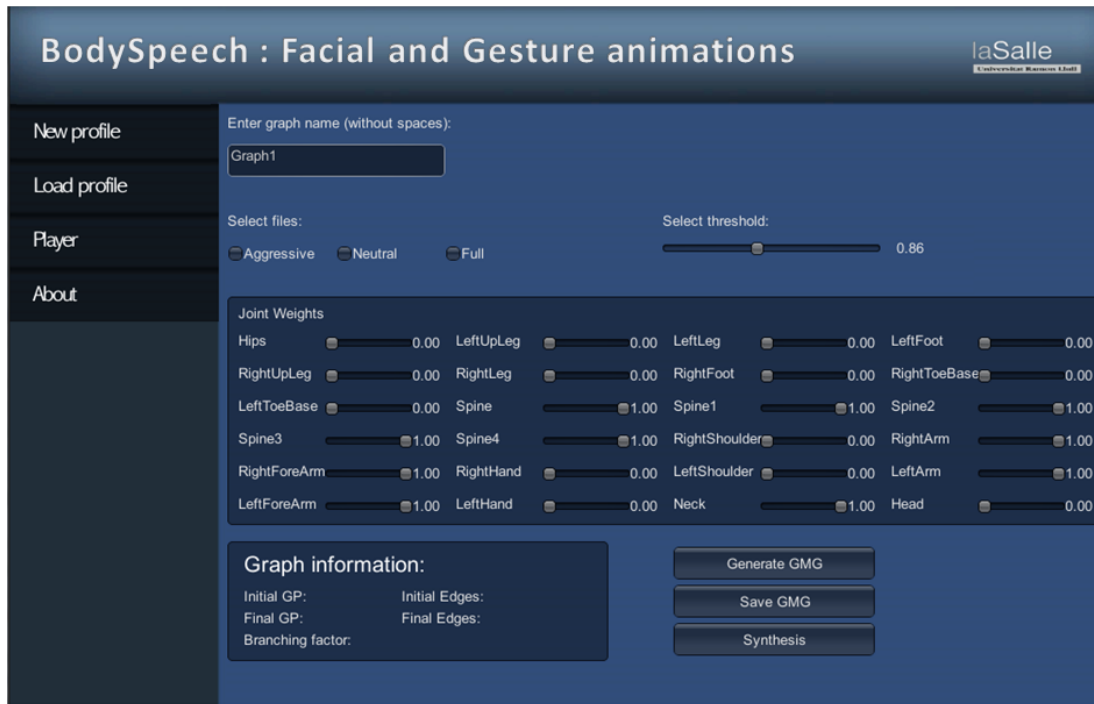


Figure 4.16: New graph screen. GMG can be parameterized by changing input databases (aggressive, neutral or full, which includes both previous), joint weights for posture similarity computation (dark blue box in the middle), similarity threshold that defines the existence of transitions between GP's (slider on top right). Once the GMG is generated, graph information is displayed at the bottom of the screen to know graph capabilities.

in two databases: aggressive style and neutral style; which could be used independently or jointly to construct a GMG.

As mentioned, gestural phrases and their corresponding gesture phases are annotated in this database. Also, stroke apexes (the maximum extension point) are annotated. So, we use gestural phrases (GP) motion clips to populate a gesture motion graph. Also, stroke phases are extracted and added as new gestural phrases rather than what was done in Section 4.4.4. In this way, we maximize the number of gestural phrases allowing more variety in gesture synthesis.

To optimize transitions, we compute the optimal blend length [Wang and Bodenheimer, 2008] for each pair of connected GP's. We include these data in edge weights in order to use it in the synthesis phase.

## Visemes parameterization

Facial animation is generated by concatenating and blending facial expressions over the time, linking these facial expressions with phonemes of the associated speech. To that effect, we relate each phoneme with facial expression, a viseme, which is represented by combination of blend shapes (shapes of the same mesh). To create a phoneme-viseme mapping we consider that multiple phonemes have similar mouth shapes when they are pronounced, therefore, they are linked to the same viseme. We use 15 categories (see Table 4.5).

/pau/	/r/	/k/, /g/, /ng/
/ae/, /ax/, /ah/, /aa/	/f/, /v/	/ch/, /sh/, /jh/
/ao/, /y/, /iy/, /ih/, /ay/, /aw/	/ow/, /oy/	/n/, /d/, /t/, /l/
/ey/, /eh/, /el/, /em/, /en/, /er/	/th/, /dh/	/s/, /z/, /zh/
/b/, /p/, /m/	/hh/	/w/, /uw/, /uh/

Table 4.5: 15 phoneme categories. Each category maps to a single viseme. Symbols are codified with MRPA (Machine Readable Phonemic Alphabet).

It is known that lip movements are linked to prosody [Cvejic et al., 2010]. Furthermore, the jaw lowers more in stressed syllables than in unstressed syllables [de Jong et al., 1993]. Based on these statements, we propose a modification of visemes based on pitch accents strength. To that effect, we define a viseme blending space between high emphatic and low emphatic facial expressions, each one with appropriate jaw positions. For each vowel, three visemes are defined (see /ah/ and /aw/ phonemes examples in Figure 4.17): neutral, high emphatic and low emphatic. Visemes can be customized by changing the weights of former blendshapes.

### 4.5.2 Facial and body gesture synthesis

#### Body gesture synthesis

As we have explained, gesture synthesis is driven by pitch accents. Distances between consecutive pitch accent times define the duration of selected GPs, and pitch accents strength are related to GP's strength. We adopt FMDistance [Onuma et al., 2008] to define gesture phrase strength again, as in Section 4.4.2, which is computed only for the stroke phase and is normalized to [0,1]. So, we iteratively evaluate each pitch accent and seek the most appropriate GP for each one. Gesture performance starts with a rest pose (which is also included in the motion graph as a node) and we use a breadth-first search algorithm to traverse the graph according to a proposed cost metric. Selected GP's are concatenated by motion blending to obtain a smooth motion stream. To finish the animation, the avatar

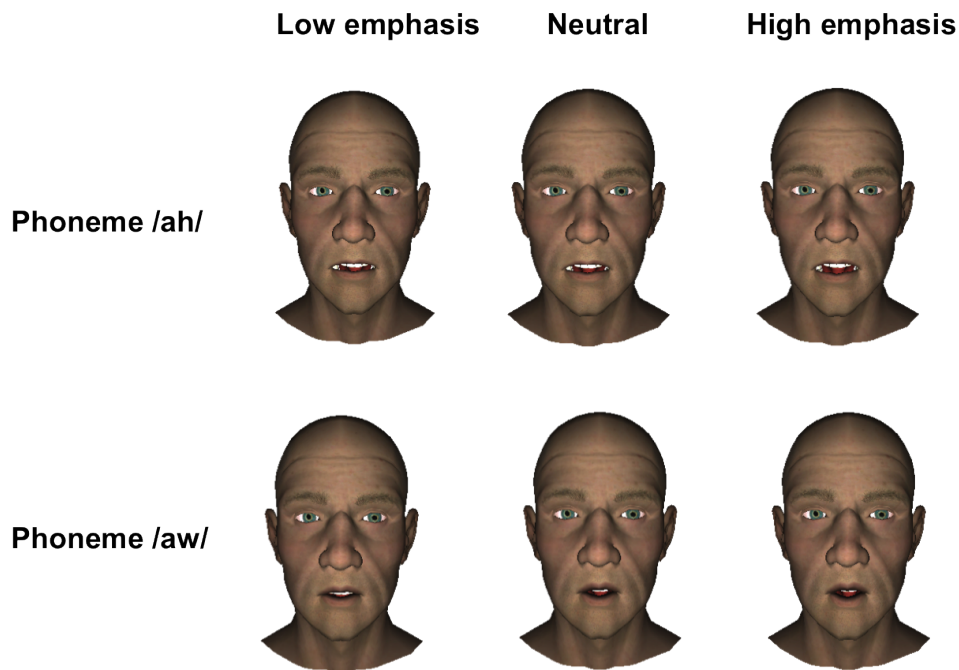


Figure 4.17: Emphatic visemes for /ah/ and /aw/ phonemes.

returns to the rest pose.

Furthermore, it is important to not modify strokes because they are the most significant part of gestures and emphasis relation in gesture selection is based on them. This differs from the temporal alignment applied in our previous version (see Section 4.4.4), where gestures are dealt as units without taking into account their parts. Hence, we manage two cases to align GPs with pitch accents times (see Figure 4.18): 'only stroke' and 'gestural phrase'. 'Only stroke' means that the gestural phrase is formed by a unique stroke, in this case, stroke length will be modified. 'Gestural phrase' case means that GP has more phases besides the stroke, so, we modify phases which are not the stroke phase. Then, GP length (and its phases length) is computed by taking into account the mentioned cases, anticipation time and blending length (included as an edge weight in GMG) between the current node and the candidate one. Therefore, we obtain a  $w$  warping factor (original length divided by target length) for each candidate GP. In the 'only stroke' case, we consider the stroke length to compute  $w$ . On the other hand, we consider the sum of non-stroke phases lengths in the 'gestural phrase' case.

Our cost metric is based on: length similarity between a GP and the interval to fill (time cost), posture similarity between candidate GP and the previous one (smooth cost) and pitch accent strength-stroke strength relation (emphasis cost). As a result, we define our



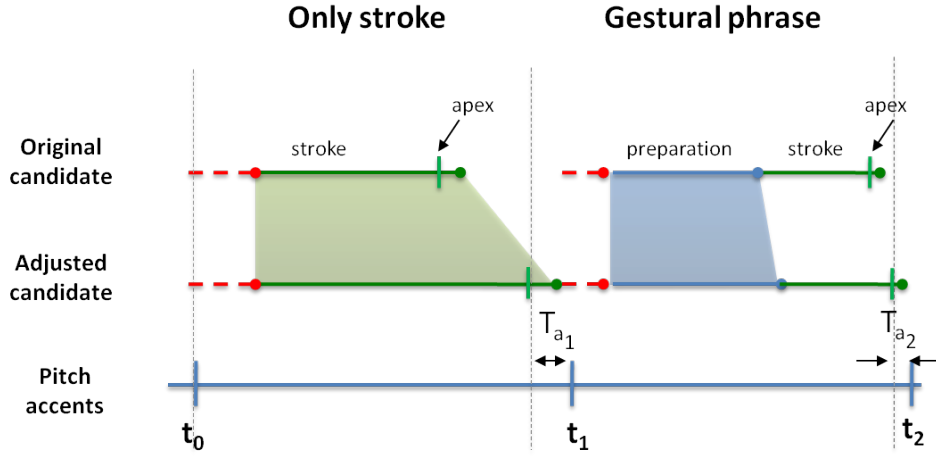


Figure 4.18: Gesture alignment. Body gestures are aligned with pitch accents by modifying their length. The goal is to match stroke apices with pitch accents times ( $t$ ), taking into account anticipation times ( $T_a$ ). We consider two cases: only stroke (on the left) and gestural phrase (on the right).

cost metric as

$$C(e(n_i, n_j), pa_k) = C_{smooth} + C_{emphasis} + C_{time} \quad (4.8)$$

where  $n_i$  is the previous GP,  $n_j$  is a candidate GP,  $e(n_i, n_j)$  is the transition edge between  $n_i$  and  $n_j$ , and  $pa_k$  is the  $k$ -th pitch accent in speech stream. Smooth cost ( $C_{smooth}$ ) is directly the posture similarity edge weight. Emphasis cost ( $C_{emphasis}$ ) is the absolute difference between pitch accent strength indicator and gesture strength indicator. We recompute gesture strength indicator in the 'only stroke' time alignment case due to its duration, and consequently, its FMDistance has changed. Time cost ( $C_{time}$ ) is defined by

$$C_{time} = \begin{cases} w' & \text{if } min < w < max \\ p & \text{otherwise} \end{cases} \quad (4.9)$$

where  $w'$  is the normalized warping factor from  $[min, max]$  to  $[0, 1]$ . We use  $min$  and  $max$  to not deteriorate motion quality by excessive changes on the original lengths.  $p$  is a penalty parameter that we use for penalizing GP's that exceed boundaries, avoiding their selection. Depending on the case of warping  $min$  and  $max$  take different values: 0.8 and 1.2 respectively for the 'gestural phrase' case, and 0.9 and 1.1 for the 'only stroke' case. Penalty parameter  $p$  is set to 10.

Once we have selected a gesture (the one with the minimum cost), this is concatenated with the previous one by linear motion blending. We use start-end blending scheme [Wang

and Bodenheimer, 2008] and the blending length included in the edge weights of the graph.

### Facial synthesis

We use phoneme transcription of the speech message to match phonemes with defined visemes. As usual, coarticulation between phonemes is generated by interpolating mesh points of visemes during initial and final times of phonemes. To include emphasis in facial expressions, we modify vowel visemes by blending them with its emphatic visemes. We relate pitch accent strength indicator with the amount of weight from neutral and high/low emphatic visemes. Pitch accent strength indicator is expressed in a 0 to 1 scale, so, we associate 0 values to low emphatic viseme, and 1 to high emphatic viseme as illustrated in Figure 4.19. So, pitch accent strength indicators that are lower than 0.5 will be represented by a combination of neutral and low emphatic viseme. Otherwise, neutral and high emphatic visemes will be used in the morphing process. In this way, we obtain the appropriate viseme according to speech intonation.

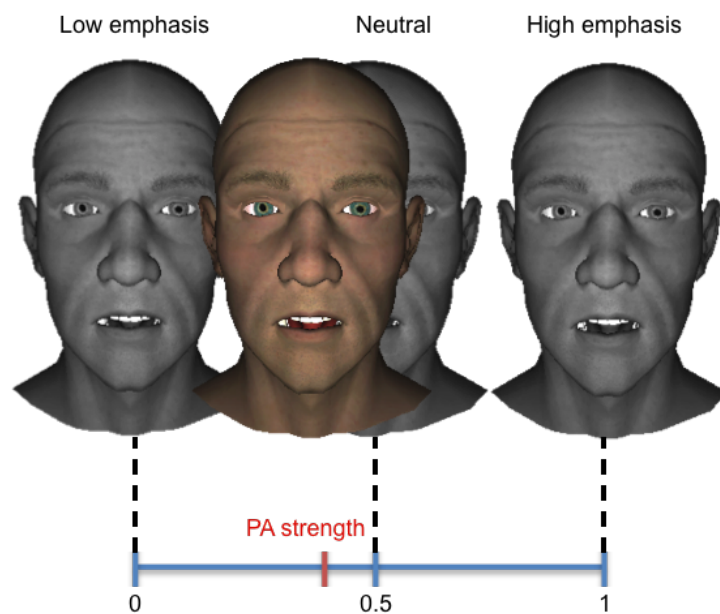


Figure 4.19: Output viseme generation for pitch accents. Pitch accent (PA) strength is used to weight low emphatic, neutral and high emphatic visemes in the morphing process. In this example, higher emphatic visemes have more opened mouths.

### 4.5.3 Editing gestures style

Also, the process that synthesizes animations can be parameterized in order to modify emphasis, both for gesture and facial animations. This way, animators can adjust output animations to satisfy plot requirements. As explained in Section 8.2, pitch accent detection can be parameterized by changing the strength indicator threshold and the time difference threshold. These two parameters can be modified in the application affecting the frequency of detected pitch accents and gestures. A greater gesture frequency is perceived as a more emphatic animation. Moreover, emphasis of gesture and facial animations can be also be adjusted independently with two moving sliders. The gesture style slider modifies the amount of strength that is added or diminish derived from pitch accent strength (from -1 to +1). 0 denotes that the input pitch accent strength remains equal, positive values increase pitch accent strength value up to 1, while negative values decrease strength value down to -1. This permits the generation of more prominent gestures from a low emphatic speech, or contrarily, to relax gesticulation in a high emphatic speech. Similarly, facial animation emphasis is controlled by an analogous slider. Sliders can be seen in Figure 4.20.

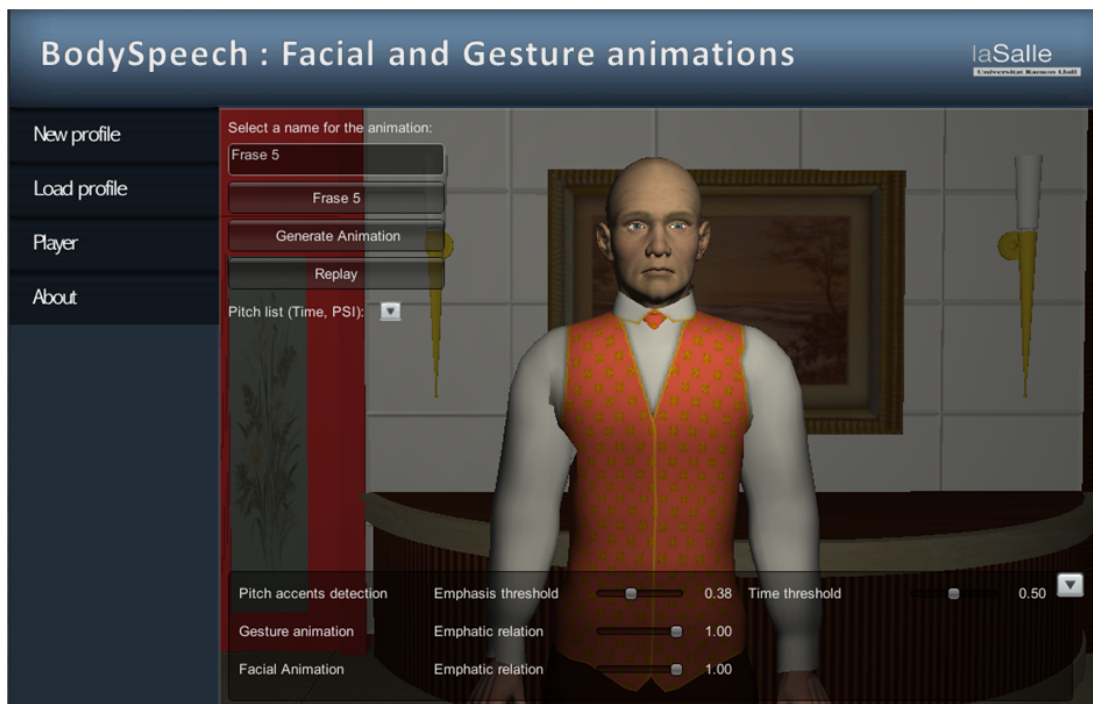


Figure 4.20: Synthesis screen. On the upper-left corner, there are the buttons to select an audio and generate the animations. At the bottom, there are the configurable pitch accent detection parameters, and sliders for adjusting gestural or facial animation emphasis.

#### 4.5.4 Summary

BodySpeech is a tool for animators that enables to generate gesture and facial animations according to an input speech. User has to specify the speech that will be used to automatically drive body and facial gestures, by relating parameters such as emphasis from speech, which is automatically extracted from an audio file. An improvement of GMG is used for synthesize body gestures (details in Section 4.5.2). Regarding to facial animation, BodySpeech implements lip synching based on the detection of speech phonemes plus a subtle adjustment of face expressions according to speech emphasis. Also, gesture and facial performances could be adjusted by modifying both the parameters used in the setting up process (Section 4.5.1) and/or changing control parameters from the synthesis phase (Section 4.5.3). In case of control parameters, they can be derived empirically by testing and observing the output gesture animations that the application generates.

In terms of software deployment, BodySpeech is a desktop application ready to be an authoring tool for animators. BodySpeech runs under a Windows Operating System (OS) [Microsoft, 2015c]. Output gesture animations which can be exported are coded in BioVision Hierarchy (BVH) character animation file format.

## Chapter 5

# Tangible Control and Interaction with Robots of Augmented Reality Virtual Characters

### Chapter Abstract

After working in motion synthesis, both pure and expressive synthesis, we face interactive control and interaction between virtual characters and the environment. Contrary to the site where the virtual characters lie, virtual worlds; and how they are controlled by the user, gamepads or some peripheral; we transfer them to the real world by changing the common rules. For this purpose, we present Augmented Reality (AR) virtual characters, also named AvatARs, and its interaction with character robots. AvatARs exist thanks to the combination of Tangible User Interfaces (TUI) and AR. An AvatAR is a virtual character controlled and represented by an object, a cube that acts as the tangible representation of a virtual character which also enables to interactively control its motion, which at same time occurs in the real world thanks to AR. Using TUI in AR environments for controlling virtual characters strengthens the link between the user and the virtual character providing a better sense of control and immersion. Moreover, AvatAR can be considered as a smart object, allowing their interaction in smart environments where other smart objects lies. Therefore, we also explore the interaction between an AvatAR (Vleo) and a Pleo, a social character robot. We create a fantastic scenario where both characters (virtual and physical) coexist, and so, an interactive narrative relationship between them make sense. Then, we have tested with a group of 8-12 year old children. Results from the test suggest that AvatAR are a good way to enhance HRI with physical character robots. This chapter is based on the published papers "AvatAR: Tangible interaction and augmented reality in character animation." IUI 2014 Workshop on Interacting with Smart Objects [Fernández-Baena and Miralles, 2014], "Interaction between Vleo and Pleo, a virtual social character and a social robot" RO-MAN 2015 [Fernández-Baena et al., 2015] and "Enhancing Long-term Children to Robot Interaction Engagement Through Cloud Connectivity" HRI 2015 Extended Abstracts [Albó-Canals et al., 2015].



## 5.1 Introduction

Bringing virtual characters closer to real world can be addressed in a variety of manners. In this work, we have started by generating smooth and responsive motions in order to enhance realism of virtual characters movements, and hence, improving user control over them. Then, we have explored the emotional connection between the user and virtual characters by generating expressive gestural animations linked to speech. Finally, in this chapter we focus on how to literally move virtual characters to real world with the aim to create a mixed and seamlessly integrated environment where reality and virtuality coexist

Virtual characters, as virtual entities, usually lie in virtual worlds as it is shown in Figure 5.1. This ensures aesthetic coherence between them and the environment around them, and also allows the design of credible behaviors that virtual characters can execute inside it. Because of this, we are used to find virtual characters in video games or in interactive applications displayed in screens where is easy to state their reason of being and existing. Although, if we change its natural habitat for the real world, we provide a great variety of options in how to visualize them outside displays, how to control their behaviors and movements and therefore, how they are able to interact with other objects (real or virtual). Thus, for moving virtual characters to the real world there are several challenges to address.



Figure 5.1: A screenshot from video game The Sims 3 University Life [EA, 2015]. As could be seen, virtual characters live in a virtual world that simulates a University campus, where they can walk, move on bicycle, and interrelate to each other as students. [JRC Gamecentrum, 2015]

Integrate virtual characters into the real world is the first we attempt. Augmented reality

(AR) is the way to add digital content to the real world, and so, it enhances the information provided by the physical world with virtual cues. Virtual objects and information need to be seamlessly integrated in the real world in order to be credible and useful. Some AR applications display virtual characters [Barakonyi et al., 2004][Bimber and Raskar, 2006] (see Figure 5.2) where they appear by the detection of a card marker and make autonomous decisions based on predefined behaviors. Also, card markers are used for commercial/entertainment purposes as in [Nintendo, 2015b]), but another time, virtual characters behavior is restricted, they commonly perform predefined animations in place. This causes an evident lack of control from the user over them by limiting the interaction of virtual characters with the real world. A reason is that users cannot control the direction of virtual character movement in the real world, and there is no real-time control of the behavior of the virtual characters. While some attempts have been made to drive character animation through card markers [Shin et al., 2005], introducing an intuitive and precise motion controller becomes a difficult task due to the characteristics of this interface, a card. In contrast, some AR mobile applications such as [Samsung Electronics Co., Ltd., 2015] provides motion control through touch-based interaction as it is illustrated in Figure 5.2. In these cases, the user can accurately drive character motion through a set of buttons that allow him to decide the direction of the virtual characters movements and their behaviors. Although, the user is out of where the interaction occurs. In order to approximate the user to the interaction area for increasing immersion and realism it is necessary to change the interface by fleeing of touch-based interfaces.



Figure 5.2: On the left, virtual characters displayed thanks to the detection of card markers [Bimber and Raskar, 2006]. On the right, AR EdiBear application for Samsung Apps [Samsung Electronics Co., Ltd., 2015] which provides a touch-screen interface for controlling the virtual bear.

Unlike touch-based interfaces, tangible user interfaces (TUI) permit to interact with digital content through a physical environment, allowing real experiences in mixed reality environments. TUI allows to design intuitive interactions using everyday objects [Ishii and



Ullmer, 1997], breaking the traditional approach of interacting with computers and giving a strong sense of immediacy to the user. So, using TUI interfaces for managing virtual characters enhance the integration between the real and the virtual world. One example is the game MonkeyBridge [Barakonyi et al., 2005], where autonomous agents are able to interact with physical and virtual game environment and, also physical objects are ready to act as active partners of virtual characters. In Figure 5.3 we can see how users introduce virtual entities (platforms) through placing physical objects (cardboards) on a table. However, they have not tangible control over the kind of motion that virtual characters perform due to they are driven by AI.



Figure 5.3: Screenshot of a two-player MonkeyBridge game. One of the players has already built a bridge for his character, which consists of virtual blocks (models with the dark wooden texture) and physical tiles (bright balsa-wood and stone cubes showing through the virtual objects). The monster character of the user standing in the middle has just hopped over from a virtual tile onto a physical platform. The user is holding the next building block in his hand. [Barakonyi et al., 2005]

Thus, we present Avatars, augmented reality (AR) virtual characters. Avatars are virtual characters which are displayed by hand-held devices thanks to augmented reality and controlled by a physical cube which acts as a tangible interface. Figure 5.4 illustrates it. We use the cube for driving virtual character movement and behavior, which in turn, it is used as target augmented reality 3D marker. In this manner, the cube is mixed in the virtual scene

providing more awareness in spatial perception, enhancing the character animation controllability of the user. To our best knowledge, there is no other realtime character controller that mixes tangible interaction and augmented reality. Furthermore, the cube is the physical embodiment of the virtual character, this could be viewed as a physical container of virtual content which came alive through a screen (a tablet or any other AR hand-held device). In our first prototype, we explore how to deal with the positioning of virtual and how to specify motion through the manipulation of the cube, which we will further explain in Section 5.2.

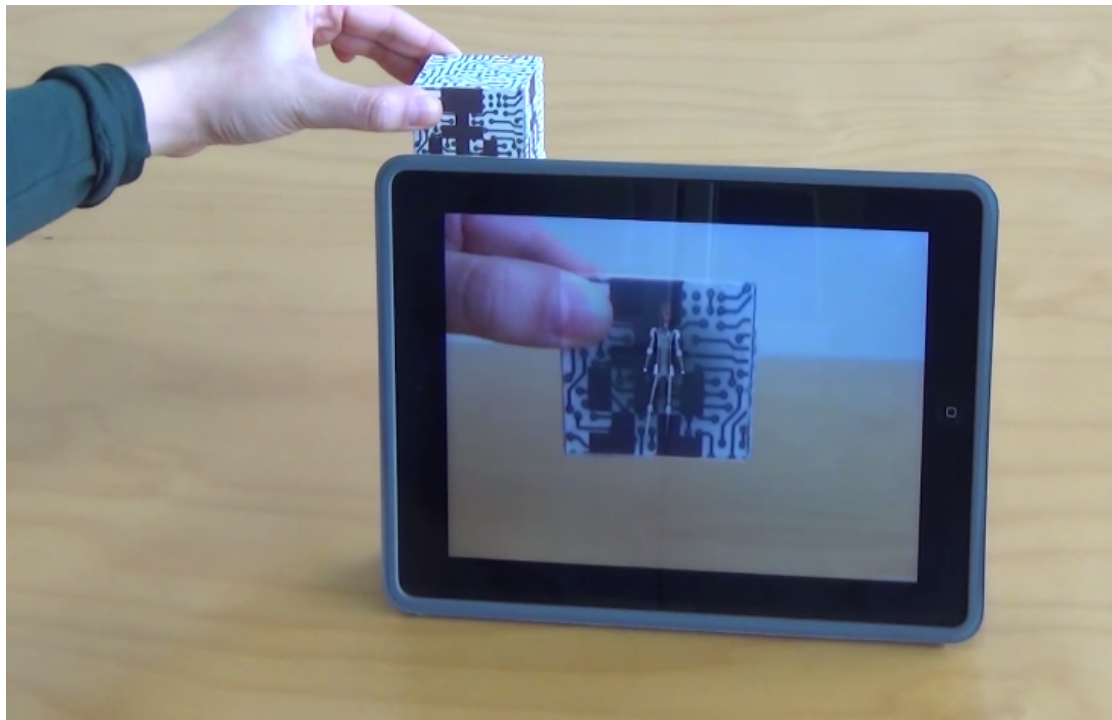


Figure 5.4: AvatAR interaction system which is formed by a physical cube and a tablet device. A virtual character is displayed on the screen inside the cube.

As it is known, objects which are involved in TUI applications, e.g. AvatARs, are regarded as a kind of smart objects. We have defined the properties, interaction information, object behavior and virtual character behavior in AvatAR concept. So, we have the proper tools for exploring the interaction of between virtual characters and other smart objects. Thus, we have automatically converted virtual characters in smart objects, ready to interact with the real environment. Smart objects interaction in augmented reality environments is lifelong investigating in character animation although some advances have been achieved in other topics like Internet of Things (Internet of Things (IoT)). Recently, Heun et al. [Heun et al., 2013] propose smarter objects, which among other things, it consists on associate a virtual object with physical object to support an easy means of modifying the behavior of that physical object. It uses augmented reality to match physical and virtual objects and supports

both tangible and graphic interaction (through a tablet). But there is no virtual character intervention in this work. Furthermore, Amores et al. [Amores et al., 2014] present SmartAvatars. SmartAvatars is a concept that also relies in a augmented reality environment and uses virtual characters to display system feedback in IoT scenarios. They expose a prototype called Flexo where virtual characters interact with objects, but their behaviors are constraint by the system reaction. However, they do not have neither tangible and direct control on virtual characters motions, which is precisely what we do in AvatAR concept described before.

AvatARs, in essence, are virtual characters so they are capable to move, talk, act, and interact with the virtual environment in a very rich way. Finding a real object that allows a meaningful and rich interaction with them is not easy due to common physical objects connected to smart environments are functionally limited, and they have been usually designed for an specific purpose (i.e. lamps or other appliances). So, In order to maximize AvatAR opportunities on the interaction with real objects, we choose a social robot to develop our second prototype. A robot as a virtual character, it is provided with physical embodiment, and it could be viewed as a smart object because fulfill the required features (tangibility and intelligence). Moreover, both entities are characters that could be linked in a mixed reality environment by allowing interaction between them. The fact of existing two characters sharing environment offers a new context for interactive storytelling between real and virtual characters. The creation of narratives, stories, could be used for several purposes such as entertainment or to confront diseases in children treatments, where nowadays are both virtual characters and robots used. We use Pleo [Inno Labs, 2014], a social robot for children that it is also used to reduce anxiety and stress to hospitalized children [Angulo et al., 2012][Larriba et al., 2015], to explore the interaction with AvatARs. Presently, children play with Pleo in order to relax, but their experience is short due to Pleo behavior tends to be repetitive as time goes by. Thus, this new scenario permits to design more experiences in order to increase the engagement in children during long-term. By adding a virtual character, and so the opportunity to include virtual objects in the Pleo former experience enables the generation of fantasy worlds, which involves visual effects, artificial intelligence, and more [Cavazza et al., 2004], breaking the physical bareer.

At this point, we propose to introduce AvatARs in the HRI in order to enrich and complement it, without the constraints of the physical world that limit the creation of new elements and reduce the immediacy. So, to that effect, we create Vleo. Vleo is a virtual Pleo, an AvatAR, it has the same appearance, but it is more expressive because its movement is not limited by the quantity of servos or the speed of their movements. In this context and from the child point of view, Pleo and Vleo can clearly communicate between them. In Figure 5.5 it is shown the setup of the scenario. Based on this, we hypothesize that the relation between the physical and virtual world is believable and as a consequence, we have created communication between them. In order to establish a relation between Pleo and Vleo, we have created narrative situations based on the behaviors of these characters, that are capable to perform actions and express emotions through movements and sounds. Their relationship is semantically emotion-based which is known that is a hard issue in HRI. Then, we have

addressed a user test to explore the engagement intensity and duration of this new interaction paradigm with a group of 8-12 year old children. Although being an early prototype, promising results make us confident that we still have a long way to go in this research area.

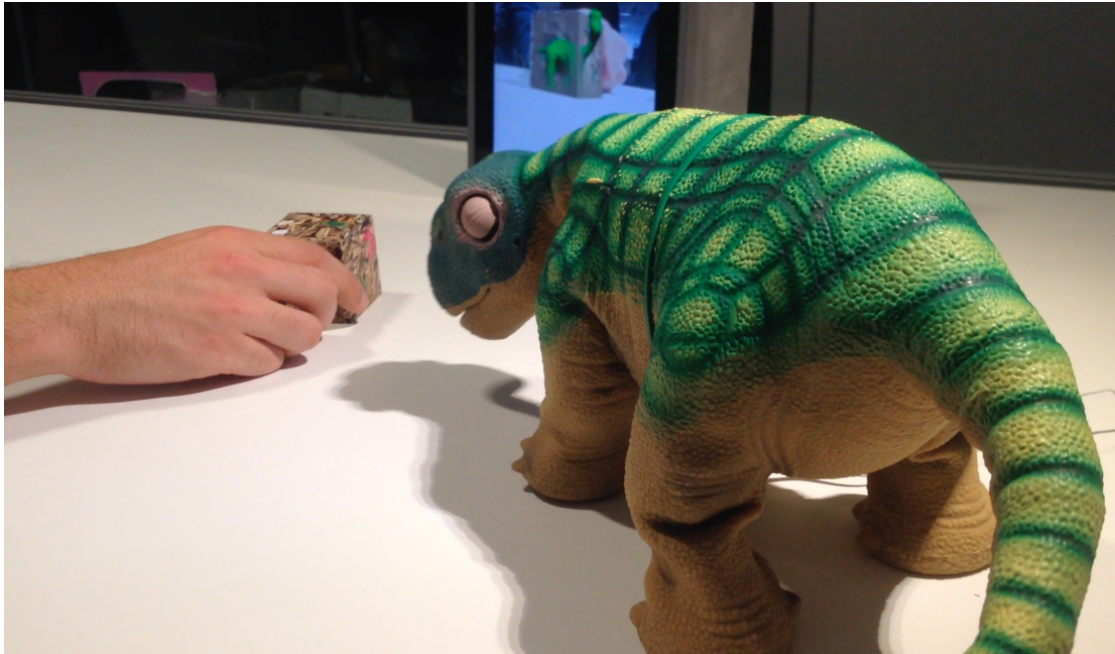


Figure 5.5: Vleo and Pleo mixed environment is formed by a Pleo, a physical cube and a tablet device. Vleo appears on the tablet screen when the cube is detected.

Having arrived at the final part of this work, we have been able to pass virtual characters from virtuality to reality thanks to augmented reality and tangible interfaces. Thus, virtual characters share interaction area with users by increasing the sense of reality and immersion in spite of the existence of a hand-held display device. In this sense, other configurations could be addressed to achieve a better experience. Anyway, Avatars can be used in a lot of HCI or HRI applications, encompassing several fields of application such as prototyping, training, communication, marketing or entertainment where virtual characters need to be in the real world.

## 5.2 Tangible control of augmented reality virtual characters

In computer animation, intuitively controlling the motion of a virtual character is considered as a difficult task. One reason for this is that a virtual character usually used in the field has a high degree-of-freedom (dof) for controlling the position and orientation of all body joints, thus making it difficult the design of an intuitive interface to manage the individual body joints. One intuitive approach to solve it is by using an instrumented puppet (see an

example in Figure 5.6) in order to retarget puppet posture to a virtual character [Mazalek and Nitsche, 2007][Numaguchi et al., 2011]. Along the same lines, Oshita et al. [Oshita et al., 2013] propose to use hand manipulation based on traditional puppet mechanism to control a character, whose work is extended in [Oshita et al., 2014] by using PCA to a set of sample poses, and assigned the extracted principal components to each dof of the hands. These approaches obtain a virtual character moving, but their move is restricted by the input. Even though the efficiency of motion retargeting is high, it makes the generation of real motion very difficult for the user.

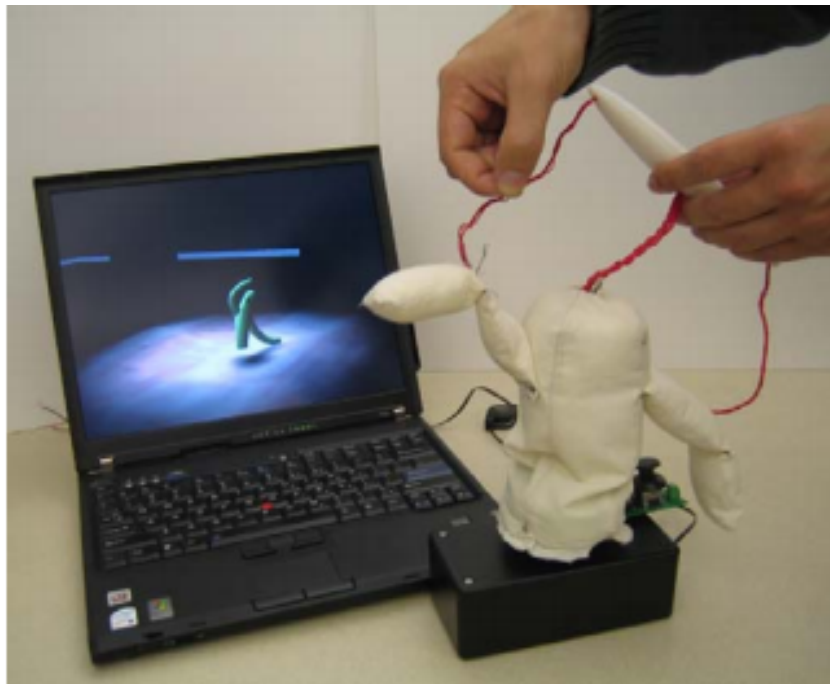


Figure 5.6: Users can pull directly on the marionette strings with a second hand to gain a greater range of independent arm movement beyond the paddle control. [Mazalek and Nitsche, 2007]

To obtain more natural complex motions (which involves a coordinated movement of different body parts) it is necessary to add a logical layer to select proper motion capture data. This is what Lockwood et al. [Lockwood and Singh, 2012] done. They use a touch-sensitive tabletop for generating full-body animations, where two fingers are used to pantomime leg movements. More recently, Seol et al. [Seol et al., 2013] present a novel real-time motion puppetry system that drives the motion of non-human characters using human motion input. Both works obtain a fine control of motion but they use hands-free control rather than tangible. Then, Shiratori [Shiratori, 2014] reviews user interfaces for character animation and character interaction by classifying them in: motion capture from body-mounted cameras [Shiratori et al., 2011], expressing animated performances through puppeteering [Mazalek



and Nitsche, 2007][Numaguchi et al., 2011][Oshita et al., 2013][Oshita et al., 2014], body avatar: creating freeform 3d avatars using first-person body gestures [Seol et al., 2013]. Thus, up to our knowledge, TUI animation controllers rather than puppets have not been addressed yet.

So, in this work, we would go a step further by controlling the motion of a virtual character using simple objects (see Figure 5.7) as in [Mazalek and Nitsche, 2007][Numaguchi et al., 2011][Oshita et al., 2013][Oshita et al., 2014], but even providing logical layer to reproduce plausible full-body motions like locomotions as in [Oshita et al., 2014][Lockwood and Singh, 2012][Seol et al., 2013], that interprets control gestures.

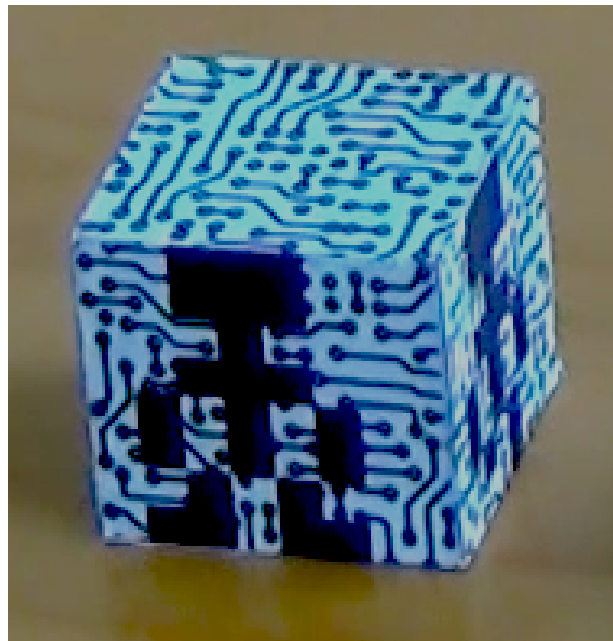


Figure 5.7: Cube for controlling an AvatAR. Note that faces denote the type of locomotion being the observed face reproduced on the virtual character.

### 5.2.1 AvatAR (Augmented Reality virtual character)

We define an AvatAR as an augmented reality virtual character which is controlled through a physical object. Then, AvatAR system is formed by a cube, a tablet (or a mobile device) and a surface. The cube is our character controller and at the same time our AR marker. We set the cube to stay on the surface (up to know, flying virtual characters are not considered). So, a virtual character appears inside the cube when this cube is detected by the tablet camera (see Figure 5.8). Then, the cube enables users the positioning and motion selection of the virtual character, providing a fine control for both. In this manner, augmented reality becomes tangible. From this moment, the virtual character is mixed with the reality viewed

by the camera of the tablet.

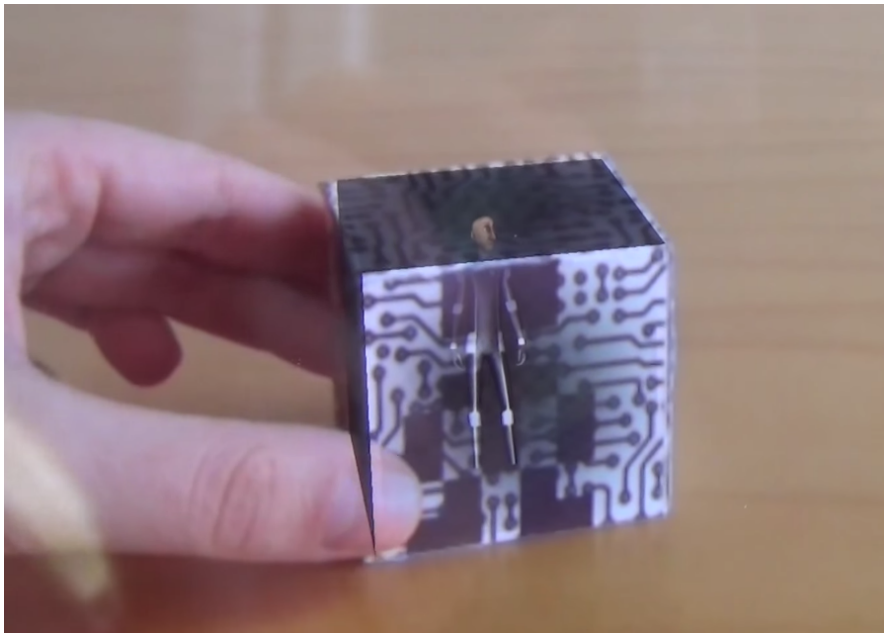


Figure 5.8: An augmented reality virtual character (AvatAR) is displayed inside the cube. Due to the camera is observing the idle face, AvatAR is in rest pose.

We face the design and conceptualization of AvatAR by taking into account the five properties of tangible systems which designers should consider in accordance with Antle [Antle, 2007], such properties are derived from the analysis of relevant literature from cognitive psychology for children. Despite we are not designing uniquely for children, we consider that following these principles will ensure the understanding of the system by users. First, she argues that designers have to take into account where users will utilize the TUI in order to relate their actions in space. In our case, users action consists on make gestures with a hand catching a cube while being sit on a chair as we illustrate in Figure 5.9. The second property is the perceptual mapping. Perceptual mapping refer to the mapping between the perceptual properties of the physical and digital aspects of the system. We afford it in the design of the cube (Section 5.2.4), which the graphic design of the faces recall to possible motions of the virtual character. Then, behavioral mapping is also considered. Behavioral mapping refer to the mapping between the input behaviors and output effect of the physical and digital aspects of the system. In our case, we design the control by finding analogies between the input and the effect on the virtual character, both for positioning and motion customization, as we explained in Section 5.2.2 and Section 5.2.3. Moreover, the mapping between the information carried in the physical and digital aspects of the system, known as semantic mapping, has been taken into account. Finally, allowing multiple users to interact is the last property, which we have not considered in this work.

In this prototype, we use a setup where the interaction with the cube occurs behind the screen (more detailed in Section 5.2.2). Otherwise, in our second prototype, where we explore the interaction of Avatars and robots, we change the interaction area in order to place in front of the screen (see in Figure 5.5). This decision affects on the controllability of the proposed systems which we will discuss in Section 6.3.

### 5.2.2 Interaction system

In this first version (see Figure 5.9), the tablet is placed on a stand in landscape position, looking in at the surface and in turn defining the interaction area. So, the interaction area is located behind the screen. Then, the virtual character is able to stand and move. The current motion state is picked rotating the cube over the same plane where the virtual character lies.

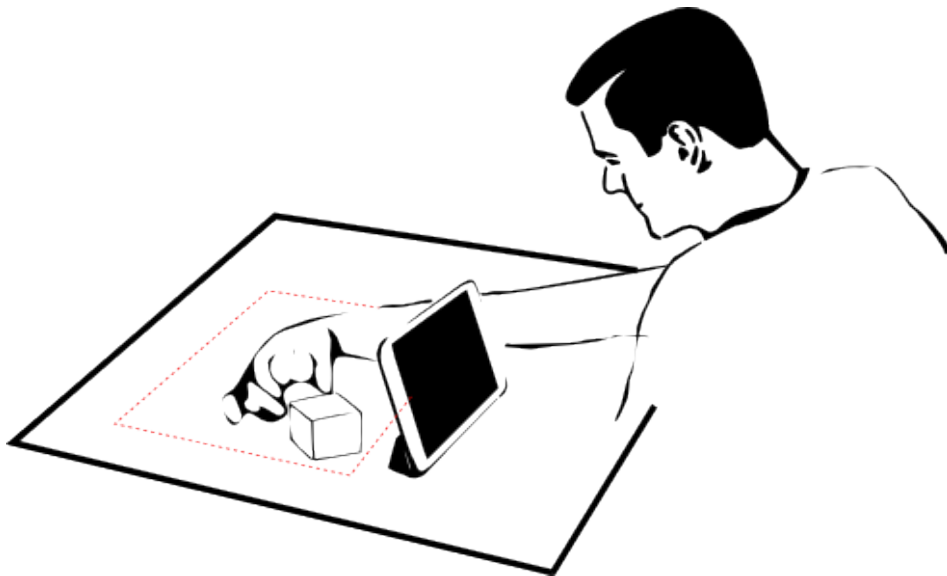


Figure 5.9: Interaction system. The system is formed by a tablet and a cube that user manipulates behind the tablet (inside the interaction area appointed in red slashed lines) enabling to see the virtual character displayed over the camera visualization of the cube.

Our character is able to move on a horizontal plane that in this prototype coincides with the tablet orientation (due its pose). For this purpose, we only consider idle motion and locomotion. We indicate through the cube the position where the virtual character has to move on. Indicating positions is straightforward due the cube lies in the scene jointly with the virtual character. The local orientation of the cube defines the type of motion providing a range that covers from walking, through running to sprinting. Establishing the analogy that likens the cube movement to a potentiometer. If we rotate the cube by 360 degrees,



the motions displayed are: idle, walking, running, walking, idle. So, the virtual character step displacement varies jointly with the type of locomotion.

### 5.2.3 Control design

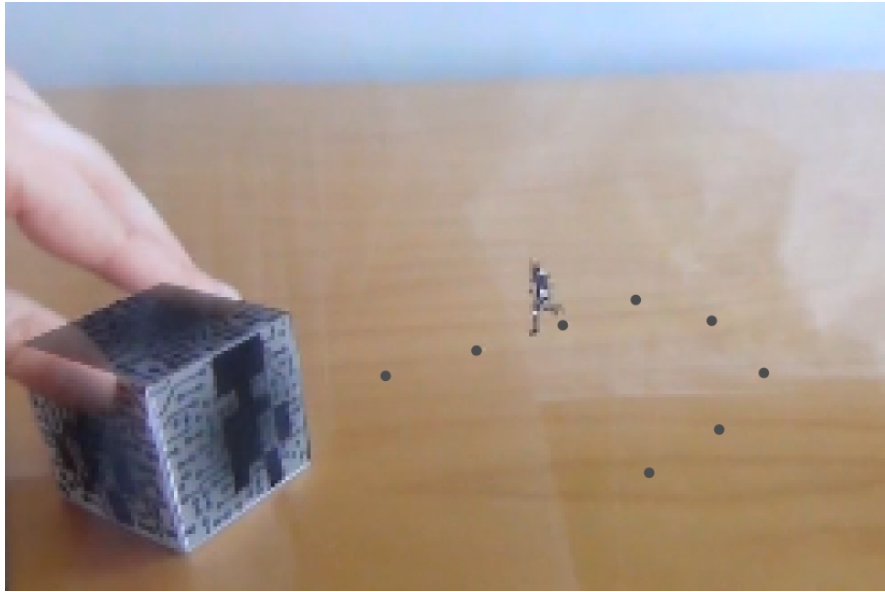
Up to this point, we can get locations and a variety of motion of our virtual character by moving and rotating the cube. Nevertheless it has to be decided how to use locations in order to design an easy-to-use character controller. To get started, we design two scenarios that can be applied in distinct applications which we leave for future investigations. We have implemented two scenarios in our prototype: a sketch-based controller and an interactive controller. The first one enables users to draw paths on the floor that the virtual characters follow; on the contrary, the second allows drive virtual characters position during all the time. So, we present two ways of using AvatAR.

#### Sketch control

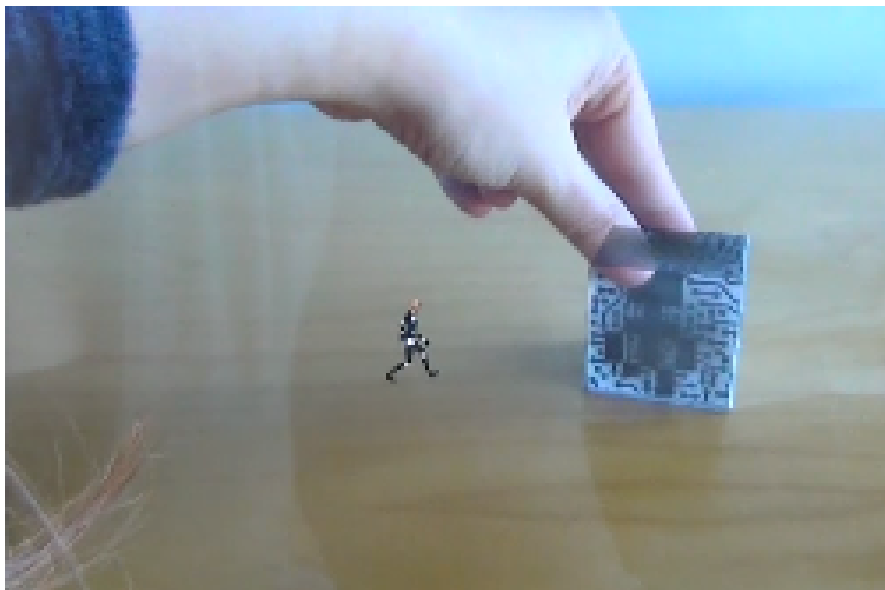
This scenario allows to define paths where the virtual character has to follow. So, we propose two steps. First, the path is defined by the translation of the cube across the scene. In order to guide the user in the path creation, we draw intermediate points (see Figure 5.10 (a)). Once we have the path defined (now we have restricted to eight points), the virtual character traces the path in loop mode. While it is moving, the user can control its behavior by rotating the cube. In this manner, it is possible to observe how the virtual character can covered the same path performing different motions. Moreover, the user can interrupt virtual characters movement by assigning idle motion which leads the virtual character to remain in a static position.

#### Interactive control

One of the most common uses of virtual characters is in video games. In this context, virtual characters are driven by players allowing positioning and behavior performing at all times. In the same way, we implement this scenario. In this case, the virtual character have always defined its target position (see Figure 5.10 (b)), which is the cube position. So, the virtual character follows the cube. How longs it takes to reach the cube depends on the type of locomotion. Obviously, if we set running locomotion, the character achieve faster the cubes, in case of setting walking locomotion, the character comes later.



(a) The AvatAR walk along the path which is defined by grey points. The orientation of the cube over the camera point of view indicates in real time the locomotion behavior that the AvatAR has to perform, changing the ground speed according to the selected locomotion.



(b) The AvatAR is walking in direction to the cube. As in sketch control, the cube orientation sets the AvatAR locomotion, and furthermore, the cube position defines the target position of the virtual character movement provoking continuous changes on its orientation.

Figure 5.10: Control scenarios.

## 5.2.4 Implementation

In order to make our first prototype we use Unity Game Engine [Unity Technologies, 2014b]. Unity is a game development ecosystem which includes Mecanim [Unity Technologies, 2014a], a powerful and flexible animation system which we use for create our animation controller. For augmented reality purposes, we use the software platform Vuforia [Qualcomm Connected Experiences, Inc., 2014] developed by Qualcomm. The following is a more detailed explanation of the animation controller, the motion control interface and the cube target we use.

### Animation controller

Mecanim enable easily construct and edit complex state machines and blend tree for complete control of how the virtual characters move. We want that our virtual character can perform different motion clips which are idle, walking and running. For this purpose, we have constructed a locomotion controller composed by two states: Idle and WalkRun. Idle state is an idle loop motion, and WalkRun is a blend tree which we will deeply explain later. To decide the current motion state we use "Speed" motion control parameter. Speed parameter balance between Idle and WalkRun states, assuming that if speed is greater than 0.5 the virtual character is moving, and so, walkRun must to be the current state. On the contrary, the virtual character is in Idle state.

As we have mentioned, WalkRun state (see Figure 5.11) is a blend tree which in turn is formed by Walks and Run blend trees. Blend trees provide variations of motion clips by blending similar phases of the input motions, so, motions have to be previously aligned. For both Walk and Run, we implement a classic blend tree which is composed by turn left, straight and turn right motions. The "Angular" motion control parameter for the turn key goes between -90 and +90, controlling which animation is being played. In case of walk blend tree, it is formed by short, medium and wide strides; and for run blend tree medium and wide. Moreover, "Speed" motion control parameter is also used to discriminate between Walk and Run blend trees, where its value goes from 0.5 to 5, remaining Walk state until 3.

As we have mentioned, WalkRun state (see Figure 5.11) is a blend tree which in turn is formed by Walks and Run blend trees. Blend trees provide variations of motion clips by blending similar phases of the input motions, so, motions have to be previously aligned. For both Walk and Run, we implement a classic blend tree which is composed by turn left, straight and turn right motions. The "Angular" motion control parameter for the turn key goes between -90 and +90, controlling which animation is being played. In case of walk blend tree, it is formed by short, medium and wide strides; and for run blend tree medium and wide. Moreover, "Speed" motion control parameter is also used to discriminate between Walk and Run blend trees, where its value goes from 0.5 to 5, remaining Walk state until 3.

Unlike in previous researches carried in this work, we have used a move tree to animate a virtual character rather than a motion graph, although we are synthesizing locomotions as

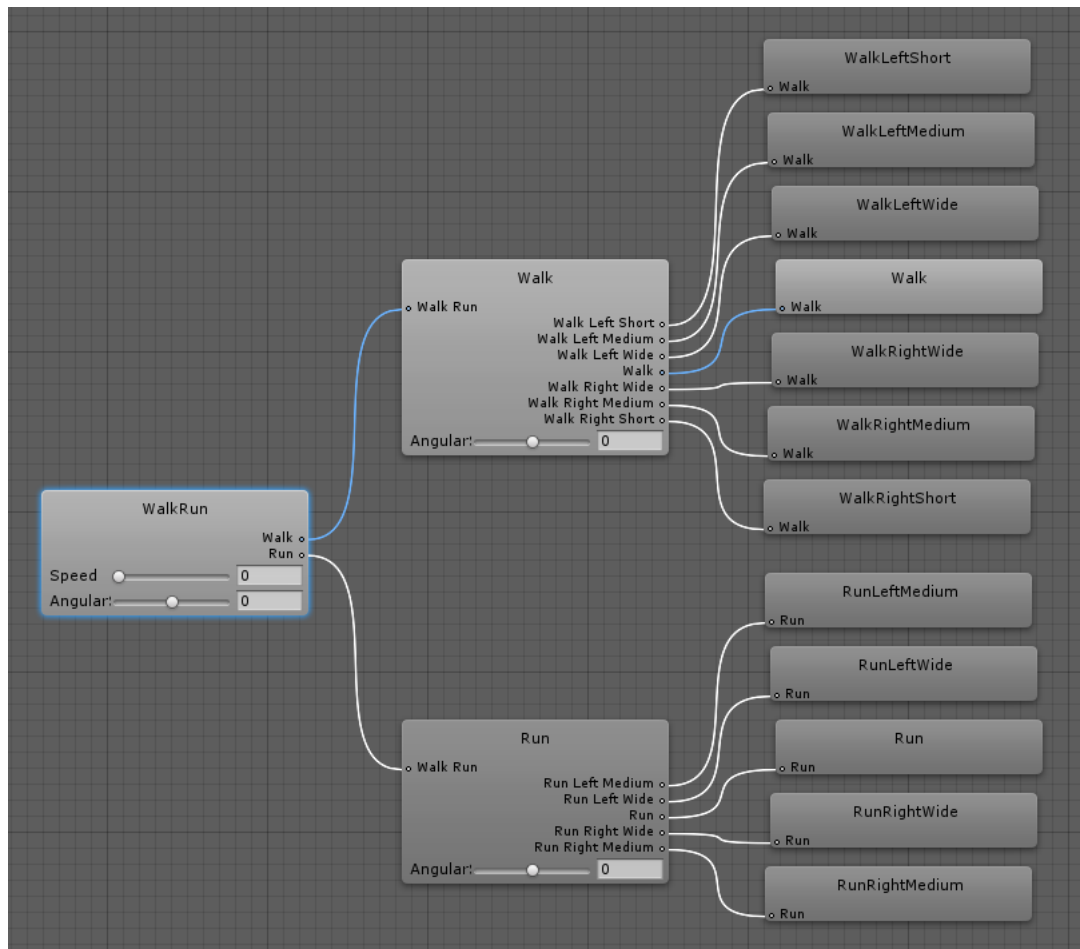


Figure 5.11: WalkRun blend tree state which at same time is formed by Walk and Run blends tree states. Walk blend tree state is formed by WalkLeftShort, WalkLeftMedium, WalkLeftWide, Walk, WalkRightWide, WalkRightMedium and WalkRightShort. Run blend tree state is formed by RunLeftMedium, RunLeftWide, Run, RunRightWide and RunRightMedium.

BPMGs method presented in Chapter 3. The objective of this part of the work is to research on interactive control of motion through AR and TUI rather than motion synthesis. So, we have used Mecanim to set up our character controller due to it eases and speeds up the configuration. Moreover, the fact that Mecanim belongs Unity3D Game Engine ensures its seamlessly integration with the game engine. Furthermore, Unity3D Game Engine eases the prototype development which includes AR and the need to be ready to run under a mobile device.

### Cube target

We design a cube with a set of image targets in its faces. Each face is illustrated by an icon that denotes one of the motion clips. We use 3 icons: idle, walking and running. The idea is to drive the virtual character by the cube faces. How to do this is addressed below. Moreover, we have texture the background with a chip pattern. In this manner, we improve the proper detection of the marker. In Figure 5.12 it is illustrated the graphic design of the marker.

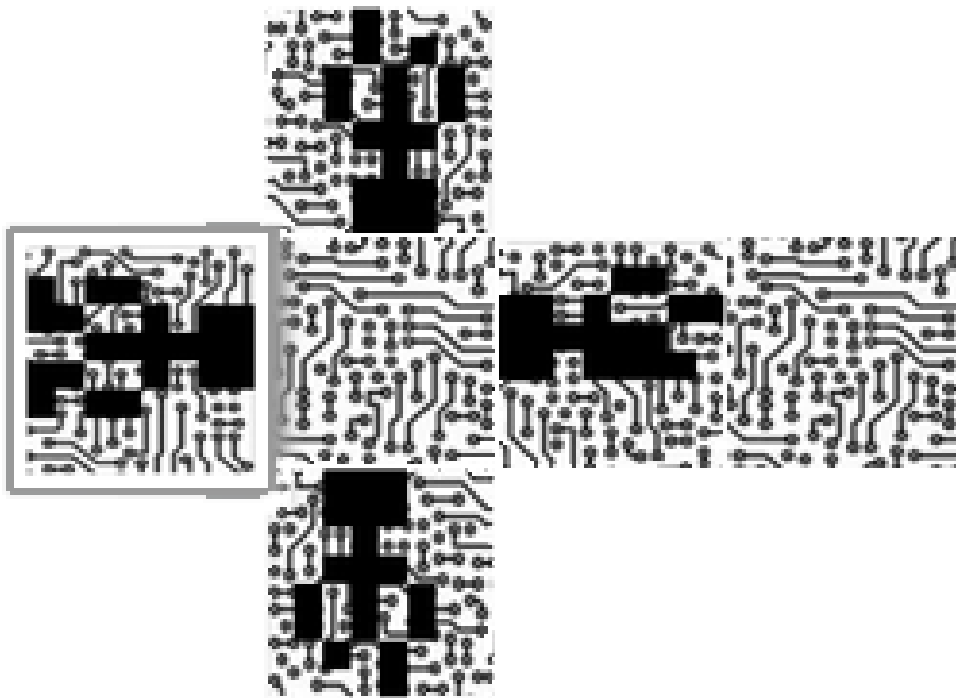


Figure 5.12: Graphic design of the target cube. Note that icon images denote idle (left), walking (top and down) and running (right) motion clips.

### Motion control interface

As we have mentioned, there are two motion control parameters in our animation controller which are "Speed" and "Angular". The way we infer "Speed" value is by computing the cube orientation respect to the camera orientation, taking into account the images on the cube faces. So, we compute the angle between the forward vector of the camera and the forward vector of the cube. Then, we transform this value depending on the section. If the obtained angle is between 0 and 45, "Speed" value is directly 0. In other cases, "Speed" value is obtained by linearly transforming the range of the value from 0 to 180 to 1.55 to

5.55. As for the "Angular" motion control parameter, it is extracted from the difference between the current direction of the movement and the previous one. This depends on the interaction scenario.

### 5.2.5 Discussion

This work comes up with a proposal to improve the intuitive control of virtual characters and its inclusion of them in daily environment. We use a cube as a tangible interface that permits positioning and to select motion clips for latter drive a virtual character. The graphic design of the cube and the interaction model allows to easily understand how it works. This is because motion control parameters have been taken into account in the design process.

Although the proposed interaction already fulfill the animation system requirements, it is limited by the cube rotation. The cube rotation only allows a degree of freedom, and therefore manage one motion control parameter. In our prototype, it is to only support locomotion control, which in terms of video games community, it is denoted as a gameplay. Change the gameplay by performing a gesture (i.e. shake the cube) and controlling it with rotation could be a possibility to increment the virtual character capabilities, and to improve the character controller. As a consequence, the graphic design of the marker should be changed for not confusing the users since the cube faces denote the active motion.

On the other hand, the fact of using a tangible controller and augmented reality to display the character performances produces several benefits. Firstly, the user can feel and see the digital content at same time. This way provides more visual and spatial awareness to the user. The user is located where interaction occurs, sharing the interaction space and being an active part of it. Moreover, the proposed interaction system allows the scene and the interaction area to move to different locations. This would allow to control the virtual characters in different places through using mobile devices. However, some changes may to be introduced in the current prototype for computing the appropriate poses of the virtual characters.

## 5.3 Interaction between Avatars and social robots

The goal of social robots is to interact and communicate with humans thanks to their shape, behavior and interaction capabilities [Breazeal, 2004]. Both physical embodiment and artificial intelligence have been evolving since 90s achieving great advances in the field of social robots. Some examples of advanced social robots are Aibo [Sony, 2014], Pleo [Inno Labs, 2014] or Nao [Aldebaran, 2014]. On the other hand, smart objects are entities that share information between them in order to release functional behaviors. So, we can consider social robots as kind of smart objects which is designed for a social purpose rather than for functional issues. Hence, we consider social robots as the best option to explore Avatar

interaction due to both are characters allowing narrative situations between them .

When children play with Pleo, Pleo gesticulates, moves and makes sounds, in turn, children touch and talk to Pleo. These set of actions define the scope of the interaction within Pleo and children. Clearly, not only what a Pleo could do has implications for the interaction, it is also lacks a degree of naturalness of its embodiment, physical movements and reactions. For example, Pleo is randomly behaving and suddenly a child touches or talks to it, Pleo does not perform a comprehensive or reactive movement or sound with the proper timing and expression as a consequence of the child input. In these cases, children's engagement lowers, because it is like Pleo fails at sensing children demands. There are several examples in the literature [Bainbridge et al., 2011] [Kidd and Breazeal, 2008] that suggest the use of physical agents to enhance engagement and long-term interaction between humans and robots. Furthermore, there are several research works about Pleo and its interaction with humans that are summarized in Section 5.3.1.

We define Vleo as a virtual social robot. This new virtual robot is based on AvatAR concept (see Section 5.2.1) and the Smart Avatar concept [Amores et al., 2014]. AvatARs are augmented reality characters that can be control through cubes. Alternatively, Smart Avatars are augmented reality characters linked to real objects. This link is an internet connection that allows a new kind of interaction between characters and objects. Smart Avatars can react to objects changing their states and vice versa. Up to now, Smart Avatars interact with very simple objects (e.g. a lamp). Interactions with smarter objects (like a robot) is one of the goals of this chapter. So, in order to design a virtual robot as Vleo we implemented two features: a physical controller and interaction with smart objects.

In this manner, we create a shared environment so called ecosystem. In order to enable their coexistence, apart from Pleo, we use a tablet facing the child that enables the AR visualization of Vleo, and also a physical cube. The cube is the tangible representation of Vleo that leads children to control it by gesturing with it. This setup is shown in Figure 5.5. In order to establish a relation between Pleo and Vleo, we have created narrative situations based on both characters behaviors, that are capable to perform actions and express emotions through movements and sounds. Then, we have addressed a user test to explore the engagement intensity and duration of this new interaction paradigm with a group of 8-12 year old children.

The following sections are structured as follows. After explaining the interaction design of this prototype, we describe the implementation. Then, we present the experiment we have conducted, and results from the test are shown. After that, we discuss and conclude this part of the work.

### 5.3.1 Pleo robot state of the art

Previously to get in to our HRI proposal, we summarize Pleo Robot research works in order to discuss our contribution in this field. Pleo [Inno Labs, 2014] (see Figure 5.13) is an autonomous toy robot modeled and bioinspired on a one-week-old Camarasaurus dinosaur. Originally created by Ugobe, it has been bought by Innvo Labs who are continuously developing it. Since its appearance in the industrial market, several researchers in the field of social robots have performed different studies that focus on understanding human-robot interaction [Jacobsson, 2009][Pitsch and Koch, 2010][Díaz Boladeras et al., 2011][Paepcke and Takayama, 2010][Fernaesus et al., 2010] [Heerink et al., 2012], exploring ways of interaction with Pleo [Kim et al., 2009] [Ryokai et al., 2009] [Curtis et al., 2011] or how to increase the engagement/interaction where playing with Pleo by adding digital content [Dimas et al., 2010][Gomes et al., 2011][Segura et al., 2012], and others not considered in this work. To that effect, we have divided the following sections from these points of view.



Figure 5.13: Pleo "eating". When you buy Pleo, it comes with an electronic dna green leaf. If you call your Pleo and show it the leaf, it will come and get close to the leaf and sniff it (actually at this moment it is watching the leaf with a small camera on his nose) then finally bite the leaf and a crunch sound will be heard.

#### Understanding Pleo

In order to understand human-robot interaction with Pleo, Jakobsson et al. [Jacobsson, 2009] report a qualitative study based on the publicly available blogs and forums hosted by Ugobe, the manufacturer of Pleo. They realize that the interaction experience with Pleo includes staging, performing and playing. With the same goal, Pitsch et al. [Pitsch and Koch, 2010] present a study where 3-8 year old infants play with Pleo. They explore infant interaction considering Pleo as different ontological categories: inanimate object, animate object that is potentially threatening, animate object that responds to interactional patterns, and polyfunctional object. They conclude that Pleo seems to blur foundational ontological



categories in children, such as animate vs. inanimate.

The study addressed in [Díaz Boladeras et al., 2011] shows that robot appearance features affect children preferences and are social cues in role attribution. Functional and social characteristics are derived from appearance and performance of social robots. In the case of Pleo, children expect animal-like behaviors such as making sounds and eating. Similarly, Paepcke et al. [Paepcke and Takayama, 2010] explore what a robot can do to such an end-user and what the user expects Pleo to do. Obviously, the fact that erring on the side of setting expectations lower rather than higher led to less disappointment and more positive appraisals of the robot's competence.

More recent studies [Fernaesus et al., 2010] [Heerink et al., 2012] have also focused on how users perceived Pleo. Results from [Fernaesus et al., 2010] suggests an apparent tension between participants expecting the robot to work as a toy and compared the Pleo with real pet animals. The results from [Heerink et al., 2012] show that users interaction basically consists on petting the robot and showing it objects.

#### Interacting with Pleo

Most users interact with Pleo using common physical and verbal communication. Kim et al. [Kim et al., 2009] studied how untrained users talk to Pleo in order to teach it how to perform some tasks. They realized that users use strongly positive and negative affective prosody when talking to Pleo. The fact that users act in this way spontaneously suggests a high degree of empathy between users and Pleo. In our work we have not considered speech as input, although it could be added to enrich future work.

Alternatively, [Ryokai et al., 2009] introduced a mixed physical and virtual authoring environment in order to enable children to create stories acted out by Pleo. In this work, they introduce tangible tools used for symbolic and abstract manipulations in storytelling. A set of colored cards each with an assigned programmed behavior are used to drive Pleo by putting them in front of it. In that manner, children decide what behavior Pleo performs and when. We also use tangible interaction to drive the storytelling, but it is used to drive Vleo, and its operation is very different due to the differences of the controllers.

Furthermore, Curtis et al. [Curtis et al., 2011] propose a low cost system for child interaction through turn taking and dance based on Pleo. Beyond the way to create new dance movements, there has been no advances in how Pleo interacts with children. Instead of performing walking or behavior movements, Pleo dances and users talk and touch them as usual.

## Interacting with Pleo and digital content

Past research involving Pleo only considered the user and the Pleo as interaction elements (with the exception of [Ryokai et al., 2009] that introduces physical cards). Because of this, robotic pet are not capable of engaging users for extended periods of time. This is where virtual content comes in. In [Dimas et al., 2010] they extend the identity of Pleo by creating a virtual representation of it on a mobile device. In a similar way, [Gomes et al., 2011] addresses two versions of Pleo, a virtual and a physical one. However, Gomes et al. consider a migration between the two embodiments in order to strengthen the idea that both versions are the same entity.

More recently, Segura et al. [Segura et al., 2012] did a further examine migration effects. They show how seemingly subtle variations on the migration process can affect the children's perception on the character and its embodiments, although, they argue that there are a lot of unsolved design and implementation challenges for migration process.

In our case, we benefit from digital content which enhances and boost the interaction between Pleo and children. However, we do not consider adding more embodiments to Pleo, or to extend the experience in different platforms such as mobile devices. Alternatively, we propose to add more Pleos with the shape of virtual characters in the interaction experience. Furthermore, we want both worlds to coexist in time and space. Thus we create a mixed reality experience where elements from different worlds, real and virtual interact.

### 5.3.2 Interaction design

A mixed reality space for Pleo and Vleo has been designed in order to enable interaction between them. This type of spaces allows users to interact with both physical and virtual objects in real time. Thus, Pleo (physical) is an autonomous agent, it moves and acts driven by artificial intelligence. On the other hand, Vleo (virtual) is controlled by the child. This point is important because children can decide what Vleo has to do. WeWe chose this to emphasize the relevance of the children in this experience. Based on this, we can summarize the flow in interaction in Figure 5.14. For controlling Vleo we introduce a tangible object, a cube. This is very similar to what has been done in [Fernández-Baena and Miralles, 2014], the of which details are further explained below. The cube connects the virtual and the physical spaces providing more awareness in spatial perception versus a touchscreen interface, also enhancing the sense of immediacy to the user in interaction. So, by moving and gesturing with the cube, children can precisely choose Vleo behavior. As a consequence, Pleo reacts to what Vleo demands. Therefore, children can observe how Pleo reacts to Vleo, closing the interaction loop. For that purpose, we have designed a Vleo-Pleo interaction and a User-Vleo interaction.

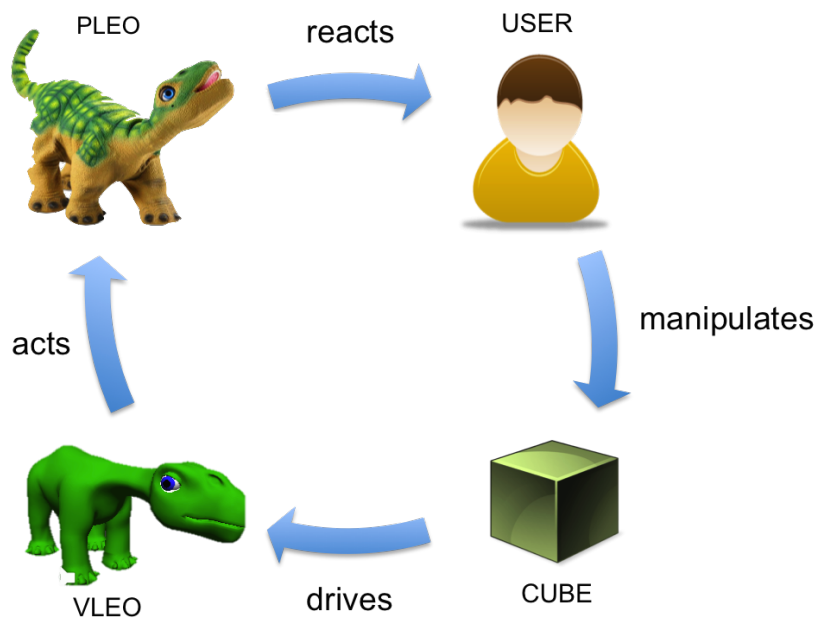


Figure 5.14: Interaction flow system. User manipulates a physical cube that drives a virtual character. This virtual character acts on Pleo robot and user perceives the robot's reactions.

#### Vleo-Pleo interaction

We design the interactive storytelling of his work under character-driven storytelling model [Cavazza et al., 2002]. Thus, Pleo reacts to Vleo behaviors which are controlled by the user. Therefore, Vleo begins any narrative situation between them. In order to create a relation between Vleo and Pleo that can be understood by children and, even more fundamentally, that Vleo and Pleo can express through their animation, we have chosen basic actions and emotional states for both. Thus, we define a lineal and logical causal progression of plot and we specify characters behaviors to be believable. Character believability [Bates, 1994] is the perception by users that the actions performed by characters do not affects negatively to the audience's credibility feeling. On the basis of this, we select the following actions and emotions for both characters: Vleo and Pleo.

Vleo can perform three actions: idle, throwing kisses or shouting. Additionally, Pleo can be in three emotional states: neutral, happy or angry. So, Vleo actions trigger Pleo emotional states. The affect of Vleos actions in Pleo states are shown in Figure 5.15. As logic suggests, when Vleo is throwing kisses, Pleo's emotional state moves from angry to happy. On the contrary, if Vleo is shouting, Pleo's state moves from happy to angry. However, if Vleo is doing nothing (idle), Pleo does not react and remains in the current state. In this manner, we create a continuous and comprehensive relationship between Pleo's emotional states and

Vleo's actions. So, we combine logical and aesthetic factors to ensure understandability in the narrative as Riedl et al. suggest [Riedl and Young, 2010].

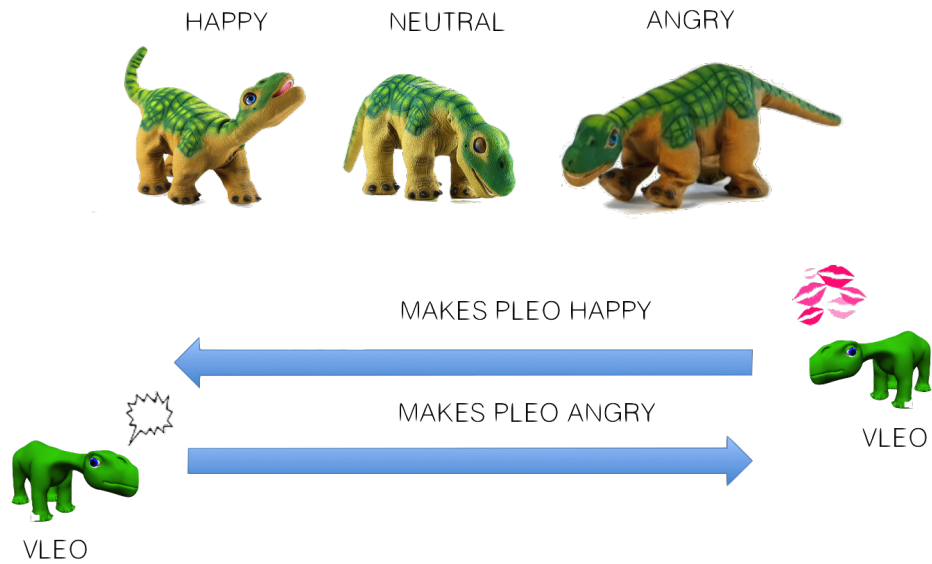
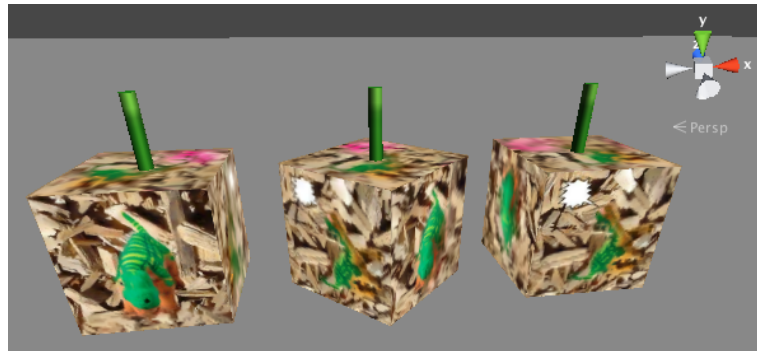


Figure 5.15: Vleo-Pleo interaction. Vleo makes Pleo happy when it is throwing kisses, so Pleo's emotional states shifts from angry to neutral and happy emotional state. Contrarily, Vleo makes Pleo angry when it is shouting so Pleo's emotional state shifts from happy to neutral and angry.

## User-Vleo interaction

As we have explained, Vleo is a virtual character driven by a child. The way a child moves and controls Vleo is by moving a cube. So, we define a second prototype which is also formed by a cube, a tablet and a plane surface. The tablet is placed on a stand in portrait position. The cube is visible to the front camera of the tablet. So, the interaction area is located in front of the screen, between the child and the tablet. The cube is our virtual character controller and at the same time our AR marker as AvatAR concept defines. So, Vleo appears inside the cube when it is detected. In this manner, augmented reality becomes tangible and controllable from the moment Vleo is included in the view of the tablet's front camera.

Each of the faces of the cube represent one of the actions that Vleo can perform. The top face indicates the current action. By rotating the cube in a clockwise direction, the intensity of the animation increase (see Figure 5.16 (b)); otherwise, by rotating the cube in a counter clockwise direction, we can decrease this intensity (see Figure 5.16 (a)). This is similar to our first prototype where the rotation of the cube over the plane where it is placed allows us to define animation parameters. These parameters define the intensity of the actions again. However, in this case the current action is indicated by the top face rather than in the previous prototype where the current action was indicated by the front face. This change is due to the placement of the interaction area respect to the tablet device. In this prototype the interaction area is defined in front of the screen, being the user's gaze fixed between the cube and the screen, and so, being the top face more easy to be seen. Moreover, it is important to note that each time the cube changes its top face by turning the cube rising from the surface, the intensity of the action is reset to the average value. Thus, actions are launched when the cube rests on the floor after a raising rotation (see Figure 5.16 (c)), which is not permitted in the first prototype of AvatAR.



(a) Decreasing intensity of Vleo state by rotating in a counter clockwise direction over the  $y$  axis.



(b) Increasing intensity of Vleo state by rotating in a clockwise direction over the  $y$  axis.



(c) Changing Vleo state by rotating over the  $z$  axis (it also could be done by rotating  $x$  axis, it does not matters the direction of the rotation) and leaving the cube over the floor.

Figure 5.16: Cube gestures for controlling Vleo animation.

### 5.3.3 Implementation

In order to carry out the interaction design, several software applications have been developed to manage Vleo, Pleo and to communicate them. First, we have implemented a controller to drive Vleo jointly with an animation system in order to enable Vleo animation. This controller runs on a tablet device, where Vleo can be visualized. Apart from that, we have coded a scripted behavior to animate the physical Pleo and we have added a bluetooth module in order to enable communication between Pleo and Vleo controller. In the following subsections there are details about how each part was implemented.

#### System

The communication between Pleo and Vleo has been done through a server. This server is connected to Pleo and Vleo with different protocols, being a Transmission Control Protocol / Internet Protocol (TCP/IP) protocol for communicating with Vleo and Bluetooth protocol for communicating with Pleo. In this way, we put all the logic of the Vleo-Pleo interaction on the server. This will allow us to use a cloud system for managing the interaction if necessary in the future.

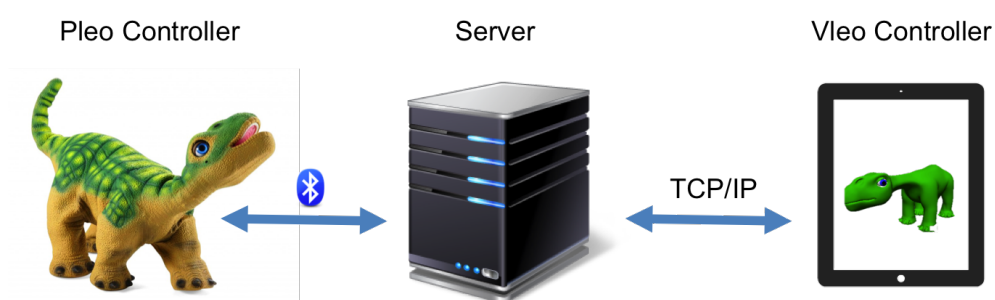


Figure 5.17: Communication system. Pleo controller communicates with the server through bluetooth technology, on the other hand, Vleo controller communicates with the server through wifi using TCP/IP protocol.

#### Vleo

Vleo's controller has been implemented with a Unity [Unity Technologies, 2014b] game engine, as it enables graphic rendering and skeletal animation from Mecanim [Unity Technologies, 2014a] animation system. To track the physical cube, we use Vuforia [Qualcomm Connected Experiences, Inc., 2014] augmented reality library, which could be easily integrated in Unity as a plugin. Thus, we use the same technology as in our first AvatAR prototype.

As we have mentioned, we want Vleo to perform different action states: idle, throwing kisses and shouting. For this purpose, we have constructed a behavior controller composed

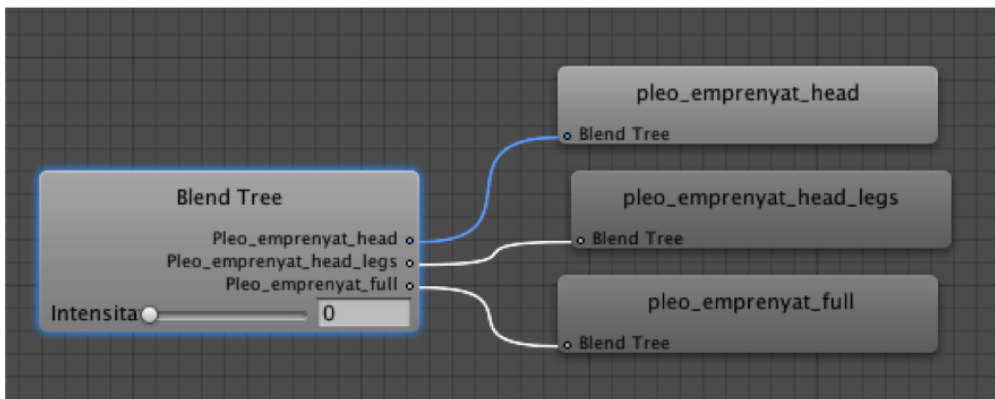
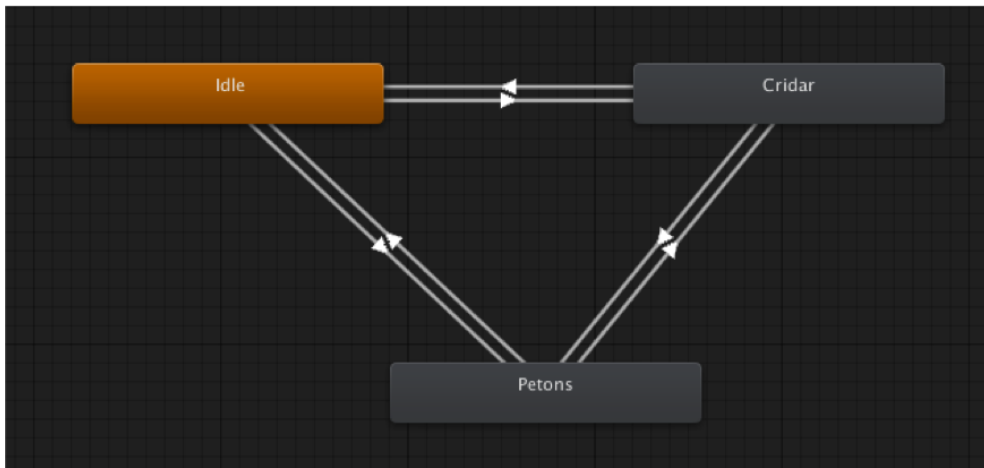


Figure 5.18: Top, the state machine of Vleo actions; bottom, the blend tree used to animate Vleo during shouting action. This last blend tree is formed by a head animation ('pleo\_emprenyat\_head' in the image), an animation that contain head and legs movement ('pleo\_emprenyat\_head\_legs') and finally an animation that brings together the last two plus a tail movement ('pleo\_emprenyat\_full').

of three states as Figure 5.18 shows. The Idle state is a unique loop motion, and the other two states are analogous blend trees. Both are composed of three loop motion clips, each progressively more intense. So, kisses and shouting state actions (Petons and Cridar in Figure 5.18) range from less active to more active head and leg movements. To manage the different intensity of each motion when motion blending occurs, we use a weight parameter that ranges from -1 (less active) to 0 (neutral) and 1 (more active), which defines the amount of activation of each state. This parameter is inferred by computing the increase or decrease of the cube orientation over the plane, setting to 0 each time that the cube is placed over the plane with a different face. We set two full turns of the cube to change the weight parameter from -1 to 1.



We design the cube for recognition and control purposes. Figure 5.19 illustrates the constructed prototype. The cube represents actions in its faces. Each action has two faces. We teach to children how to place and manipulate the cube in order to control Vleo. Moreover, we have textured the background of the cube faces in order to improve robustness of image recognition.



Figure 5.19: Physical cube built to drive Vleo. Their faces are illustrated with Pleo images that represents the actions that Vleo can perform.

## Pleo

Pleo's connectivity has been enhanced through the bluetooth module RN-41 (see Figure 5.20) that allows Pleo to communicate with the server. To improve the connectivity range we have attached an additional antenna. Finally, we have embedded everything in a box below Pleo's stomach as an extension of its body.

As it is known, Pleo can behave according to its operating system or a coded behavior that lies in a miniSD card. So, we have coded a .ply script which includes the three mentioned Pleo states (happy, angry and neutral). Each one is represented by a unique loop motion clip and an appropriate sound.

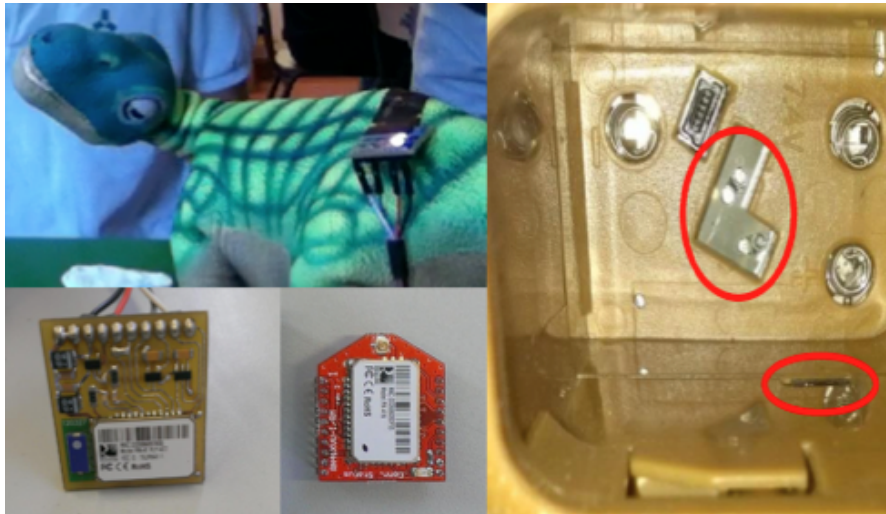


Figure 5.20: The Pleo's connectivity has been enhanced adding a bluetooth board and antenna through the Pleo UART (Universal Asynchronous Receiver/Transmitter) system. The two points to access the UART are circled in red.

#### 5.3.4 Experiment design

We have conducted an experiment with children in order to verify our interaction proposal. We target it children between the ages of 8 to 12 as Ugobe does with Pleo users. The following is a detailed explanation of our hypothesis, test and results.

##### Hypothesis

The hypothesis of the study was focused on the classic HCI usability evaluation factors (International Organization for Standardization (ISO) 92411-11), that is: efficiency, effectiveness and user satisfaction. These three factors will be evaluated as follows:

1. The cube controller for Vleo is easy to use for 8-12 year old children.
2. 8-12 year old children understand and realize the interaction between Vleo and Pleo.
3. 8-12 year old children enjoy their time by playing with Vleo and Pleo.

##### Procedures

The experiment was performed in three different phases: familiarization, practice and test. Before starting the experiments with Vleo and Pleo, the children and the experimenter took part in a familiarization phase. The goal of this phase was to get acquainted with the children to know Pleo and Vleo actions. First, the experimenter lets the children play for 2-3 minutes with Pleo. Then, the experimenter spent 1-2 minutes explaining what Vleo is, what

the cube is, and how he or she can control Vleo using the cube. Pleo is not present during this explanation.

After that, children were asked to practice making Vleo throwing kisses or shout. Once all the children understood how to control Vleo the test phase started. Then, in the test the children were asked to complete two tasks: Task 1: Make Pleo feel happy; Task 2: Make Pleo feel angry. For each task, we let the children control Vleo with no instructions and with a maximum time of 2 minutes. The task finished when the children believed that the goal was accomplished. It is there for possible that they finish it without success.

### Participants and Settings

The fifteen participants in the experiments were children (11 boys and 4 girls) 8-12 year old. All them enrolled in a summer robotics program organized by La Salle Campus Barcelona - URL. An experimenter was on a table with Vleo and Pleo to facilitate the test, and to intervene in case of difficulties. He was also involved in the activity as a facilitator of the interaction, providing guidance and ensuring that children did not became frustrated or harmed during the activity.

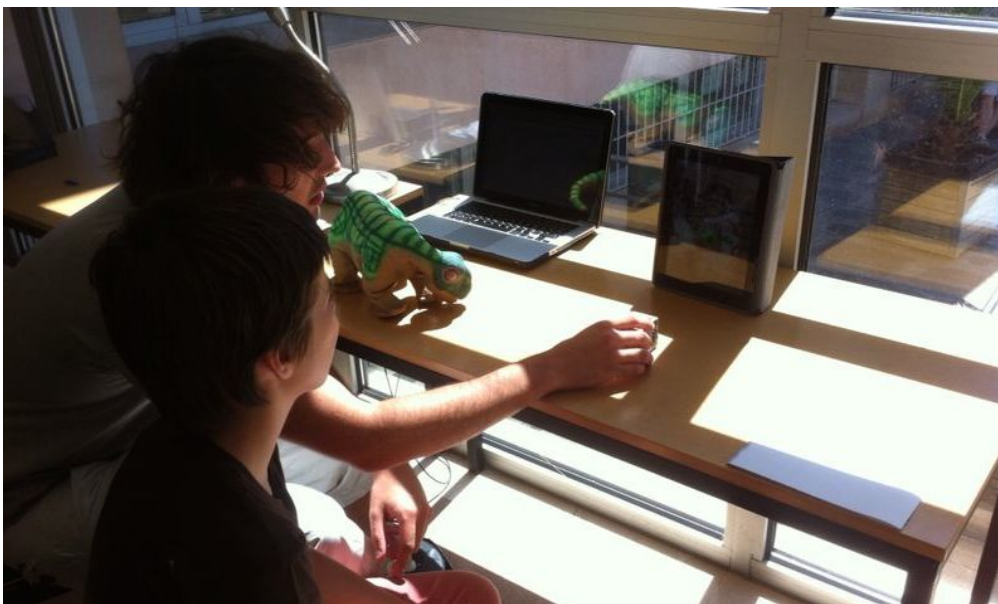


Figure 5.21: The facilitator explaining to a subject how to control Vleo.

The experiment took place in a wide corridor at the university (see Figure 5.21), in front of the classrooms where the participants were taking the Robotics course. An iPad, a cube and a Pleo were placed on a table. The children were sitting or standing facing the iPad, doing the test one at a time. Two experimenters were present during the test, one with the

mentioned role of facilitator, and the other with the role of observer who track results and was in charge of the proper technical performance of the interaction system.

### Questionnaire and evaluations

We use a questionnaire and a set of evaluations to assess the children's impressions.

- Questionnaire: The questionnaire measured the perception of the children about the accomplishments of the tasks. The questionnaire was asked at the end of the trials, the experimenter records the questions with the children's answers. Some questions used a 5 point Likert-scale and with a space available for literals, providing information not covered by the response categories. The questions are summarized in Table 5.1.
- Evaluations: Experimenters took some evaluations to relate user performance and answers. For each trial it was noted if the task was solved with success or not, and how many times the user needed to perform it (whether the user did it right or not).

Table 5.1: Assessment questionnaire children perception

Test phase	Question	Type of answer
After task 1	'Do you think that Pleo is happy?'	5 point Likert-scale
After task 1	Without Vleo, how would you make Pleo happy'	Literal
After task 2	'Do you think that Pleo is angry?'	5 point Likert-scale
After task 2	'Without Vleo, how would you make Pleo angry'	Literal
End of the test	'Do you like to play with Vleo'	5 point Likert-scale and literal
End of the test	'How do you like to play most?'	Only with Pleo or with Pleo and Vleo

### 5.3.5 Results and discussion

The collected data from the questionnaires related to each task is summarized in Figure 5.22. Unfortunately, we have discarded 3 participants due to technical constraints. In the first task, 91.7% of the participants believed that Pleo was happy by giving the highest rating, the rest (8.3%) believed that Pleo was happy, but not much. In case of the second task, all participants agreed that Pleo was angry without exemption.

The user perception of Pleo's emotional state slightly depends on the state itself. Happy emotional state is not perceived as clear as angry emotional state. One possible cause of this could be related to the selected animation clip and sound for this emotional state. It seems that representing the angry state is a little easier than the happy state. Still, we can validate



Figure 5.22: Happy feedback perception and angry feedback perception.

our second hypothesis since children always correctly perceived Pleo's and Vleo's behaviors.

Figure 5.23 show the amount of time taken to perform the tasks. We obtained a wide range of values from both. Values range from 5 to 40 seconds in the first task. The average was 22 seconds and a standard deviation of 10,2 seconds. In the second task, values range from 6 to 26 seconds with an average value of 16,8 seconds and a standard deviation of 6,2 seconds. On both tasks, all participants could finish with success.

The time needed to change Pleo's emotional state is linked with Vleo control system usability. This change is achieved through setting Vleo's actions and the appropriate intensity. During the test, the experimenters have detected some technical problems and this resulted in high values obtained in the first task proposed. Some users needed more than 25 seconds to reach the objective. It is noticeable that children improved their performance

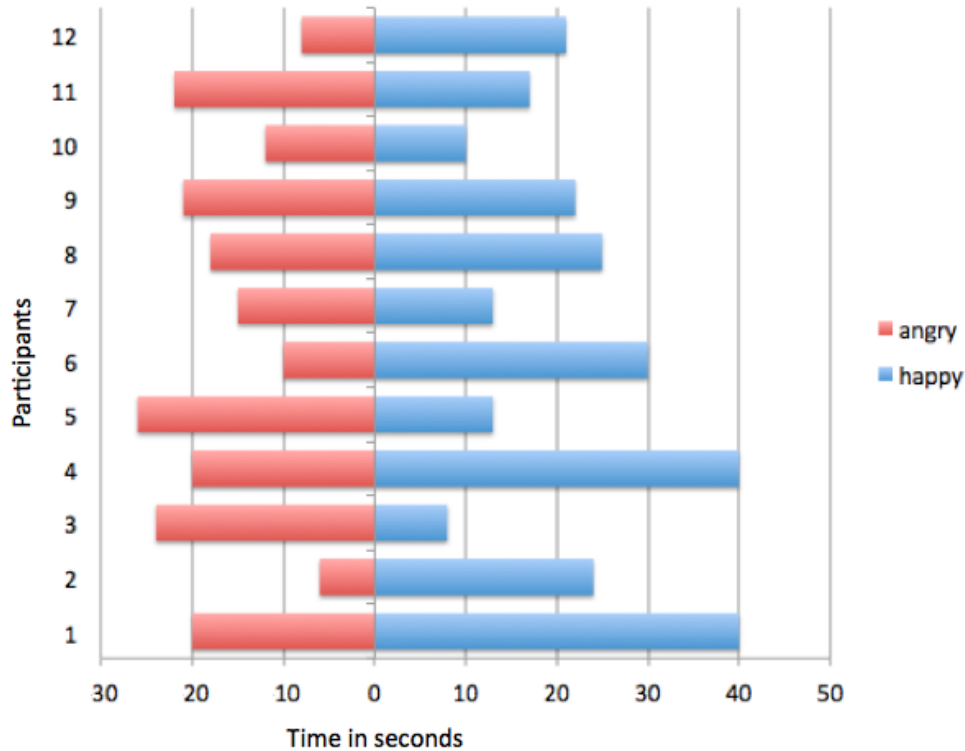


Figure 5.23: Time taken to get Pleo happy and time taken to get Pleo angry.

in the second task, as demonstrated by the decrease of the average and standard deviation values, with improvements of 27% and 39 %, respectively. So, the Vleo controller must be adjusted to be easier to use allowing faster execution by decreasing the number of turns of the cube. This will improve execution and facilitate controller operations.

Finally, we summarize the play preference in Figure 5.24. As shown, a significant majority (83.3%) preferred to play with Vleo and Pleo, otherwise, only a 16.7% prefer to play with Pleo alone. In this case, the results are meaningful. More children prefer playing with Vleo and Pleo rather than with only Pleo, as we have addressed in our third hypothesis. A number of different factors may be the cause. One of them could be the Wow effect of the interaction setup. It is not familiar to children to drive a virtual character with a cube. It is even less to visualize this virtual character overlaid on the reality through a tablet camera view. These elements make the experience more innovative, favoring engagement. In addition, the fact that being two characters in the interaction system enables the creation of a more complex scenarios than with only one character, and in this way it prolongs the time of use. Furthermore, this enables the creation of any type of situations by generating more interactive content for children.

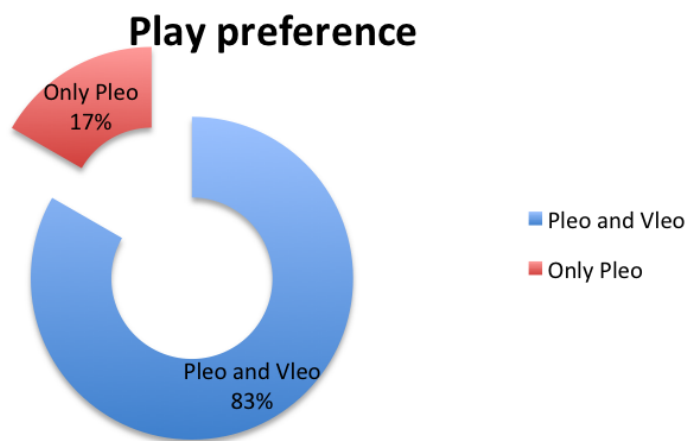


Figure 5.24: Play preference





## Chapter 6

# Conclusions, Limitations, and Future Directions

### Chapter Abstract

This chapter exposes the conclusions, limitations, and future directions of this thesis. This thesis contributes to face challenges related to the creation of natural 3D virtual characters for HCI and HRI applications, by making them responsive, expressive and tangible. In particular, we deal with locomotion synthesis by presenting Body Part Motion Graphs (BPMGs) data-driven method. The aim of this method is to get more responsive virtual characters to user input. To test it, we have compared BPMGs against SMG which is the well-known method of the state of the art. Then, we focus on expressive animation of gestures. Gesture Motion Graph (GMG) and BodySpeech are developed for synthesizing gesture animations according to an input speech, after the analysis of the correlation between speech and gestures. Apart from working on the generation of movements, we dealt with interaction with users by assessing the credibility perceived by them on the synthesized animations of gestures according to a speech against, then compared to the original ones. Finally, we propose AvatARs concept that leads virtual characters to rest over the real world and to become tangible thanks to AR and TUI. After this point, interactive control of AvatAR animation and its interaction with character robots are examined. For that purpose, we have designed a TUI to control virtual character motion. This TUI is a simple cube that allows to position and precisely animate an AvatAR. Then, we have used it to drive an AvatAR, called Vleo, in an interactive storytelling scenario that includes a Pleo, a character robot. We have tested this scenario with a group of children in order to explore AvatARs importance in HRI applications. In order to discuss on all these concerns, the conclusions are organized into three sections: Animation using BPMGs; Animation and interaction through GMG and BodySpeech; and AvatARs animation and interaction with character robots. In these sections, we also expose future work corresponding to each of the research lines.



## 6.1 Animation using Body Part Motion Graphs (BPMGs)

Virtual characters have to quickly respond to user demands when they select the action that the virtual character have to perform, which means, that character controllers must be able to transition between different motion clips while preserving the smoothness of the motion. In the third chapter, we have presented a locomotion synthesis system that works with partial motions rather than full-body motions in order to speed up response in transitions. For that purpose, we have defined a framework to deal with motion capture data that which includes the construction of BPMGs (see Section 3.3.3) and the generation of progressive transitions (see Section 3.3.4) between motion clips. In our experiments we demonstrate that transitioning by parts using BPMGs improves connectivity between the input motion dataset, responsiveness, and smoothness of transitions of the synthesized locomotions respects to SMG. Although, a lot of parameters belonging to BPMGs method such as number of body parts where full-body movements are divided, group of joints belonging to each body part, or the duration of the progressiveness window could be analyzed and optimized in order to obtain the desired transition performance according to the animator requirements. We consider our method an incremental improvement over SMG method focused on achieving quick transitions, that can be implemented together with other motion graph improvements in order to create a powerful automatic animation system based on motion graphs. Good examples are the ones proposed in WcMG [Zhao and Safonova, 2009] of interpolating between poses from the input dataset to increase connectivity, or FMG [Mahmudi and Kallmann, 2013] where they calculate similar postures only between detected feature postures in the input dataset to reduce construction time.

In Section 3.4, we have used a set of measurements to evaluate progressive transitions using BPMGs. We have focused on how quickly transitions can be executed and how long the system takes to respond to transition demands. For that purpose, TBF and LM metrics are computed as proposed in [Zhao and Safonova, 2009] and [Reitsma and Pollard, 2007], respectively. We have evaluated whole dataset (details in Section 3.4.1) and motion to motion transitions in order to have a global perspective and a specific one focused on motion clips (details in Section 3.4.2). These measurements have been done over BPMGs and SMG in order to compare both methods. Results show that progressive transitions using BPMGs have better response and better execution than SMG. Moreover, it should be noted that these results are achieved using similarity threshold values for BPMGs lower than SMG as observed in Table 3.2. So, BPMGs transitions are smoother than SMG. Apart from this, the fact that BPMGs work with body part motions allows better connectivity between motion clips, although similarity thresholds are lower. To demonstrate these conclusions we have analyzed the size of the largest SCC (details in Section 3.5.1) as a percentage of the original motion data set size for both methods varying threshold values as Zhao and Safonova suggest in [Zhao and Safonova, 2009]. Because of these, we consider that BPMGs method improves some of the deficiencies in motion control of SMG and the method also offers a new way to generate quick transitions between locomotions in other discrete methods of the state of the art.

Body part segmentation is the main advantage to overcome in performance that BPMGs method possesses over SMG. We evaluate how body segmentation (the SPs used are illustrated In Figure 3.13) affects to BPMGs method in Section 3.5. From there, we conclude the number of body parts and the joints belonging to each of the BP that conforms with SP affects the connectivity and transition cost of BPMGs. From Section 3.5.2 we point out that transition cost has an inverse relation with the number of body parts. But more body parts do not imply better connectivity with lower similarity thresholds. In some cases, more segmentation provokes an increase of connectivity but in other cases it does not. So, connectivity with different split profiles depends on locomotions in dataset and how similar the partial poses of body parts are (see details in Section 3.5.1). Therefore, connectivity and transition cost values show which split profile is the most appropriate for an specific input motion dataset as discussed in Section 3.5.3. So, selecting an appropriate split profile between two specific motion clips or for an specific input motion dataset could be done by minimizing connectivity (through the percentage of the largest SCC), response (through TBF and LM) and transition cost.

In character animation what is truly important is how plausible the output motions are. Creating a full-body transition through the composition of body part transitions (details in Section 3.3.4), in addition to match them in time, makes the need of synchronizing them. The physical correctness of the generated poses could not be preserved, and the full-body movement could not be coherent. With this in mind, we tested BPMGs without over using the desynchronization allowed in the search process. We selected proper parameters, or not creating BP s that do not belong to independent human kinematic chains. In these manners, we avoid the potential problems caused from these issues. On the other hand, although we improve connectivity, responsiveness and transition cost over SMG, sometimes our transitions look a little "shaky" because we connect individual frames to transition rather than motion segments. Searching algorithm is based on the sum of edges weights that traverse a path to find the shortest path. So, an output path only ensures little steps (restricted by the similarity threshold) between frames but not continuity between more than two steps. This fact produces quick changes on joint orientations. Global transition draws a shape which we conserve although we remove local noise with a low pass filter. Furthermore, it would be interesting to address a perceptual validity of the output motion in terms of naturalness as Etemad et al. [Etemad et al., 2015] suggest.

To show the performance of our method we have compared to SMG. Comparing both methods have constrained the performance of our method due to some parameters that can improve more the obtained results. Body part weights permit more flexibility for deciding which parts are more important in comparing the similarity of two postures. This structure allows a hierarchical approach with more control of how important are joints individually and grouped with others. Another parameter to take into account is time of the progressiveness window. This parameter defines progressiveness of transitions, as described in Section 3.3.4, and it will be interesting to analyze how progressive could be a transition while maintaining

realism. A similar approach to [Wang and Bodenheimer, 2008] could be used to address it. Wang and Bodenheimer use full leave-one-out cross-validation study [Duda et al., 2012] to estimate the generalization rate of the optimized weights, referring to joint weights, then, a cross-validation study, was performed to evaluate the weighting determined by the optimization. In BPMGs, the value of the progressiveness window should be derived while preserving naturalness and correctness of the motions, of course, a study will be needed to assess this value depending on the input motion dataset, presumably, this parameter could be dependent on the type of motion clips (e.g. linked to walking locomotions).

As we have mentioned before, motion to motion transition can be more optimal if specific split profiles are used to transition between motion clips as discussed in Section 3.5.3. But BPMGs are constructed in an offline process which takes time, and to change body parts will mean to recompute some graphs from scratch. Therefore, we plan to implement a two level framework to select appropriate graphs depending on origin and target motion clips. This approach could require a considerable amount of disk space, so a hierarchical segmentation will be addressed to optimize the space required. For example, we could create a set of BPMGs belonging to each human kinematic chain (see Section 3.3.2), then, by summing their edge weights that carries the information needed to search for transitions, we could create the BPMGs needed in practically real time due to the low computational cost that implies. In this manner, by first selecting and generating the graphs belonging to the optimal split profile and later performing body part transitions with them, transitions between motions could be better than ones presented in this work.

## 6.2 Animation and interaction through Gesture Motion Graph (GMG) and BodySpeech

Expressive animation of virtual characters is needed in many situations, for example, when a virtual character communicates a speech to a user/s. Speech carries a lot of meaning that have to match with the animation of the virtual character. In this context, HCI is at the expense of the credibility of the virtual character performance. Thus, in Chapter 4, we have presented a study of the relationship between gestures and speech in terms of temporal alignment and emphasis. Following, we present an algorithm capable of generating a gesture stream driven by an annotated speech signal. The generating of gesture animations is based on a Gesture Motion Graph (GMG). Then, we have presented an automatic system and a desktop application based on the previous algorithm to generate body gestures according to an audio speech, as well as to facial animation, called BodySpeech. In this context, we only rely on prosodic features of speech [Chiu and Marsella, 2011][Chiu and Marsella, 2014][Levine et al., 2009] [Levine et al., 2010][Marsella et al., 2013] in order to generate appropriate gestures rather than semantics [Sadoughi et al., 2014] [Lhommet and Marsella, 2014], or using both prosody and semantics [Marsella et al., 2013]. Moreover we only based our algorithm on "beat" gestures as in [Levine et al., 2009] [Levine et al., 2010] rather than using also iconic [Stone et al., 2004] or metaphoric gestures [Stone et al., 2004][Lhommet and Marsella, 2014].

For the emphasis analysis between gestures and speech, we have introduced strength indicators for gesture (GSI) and speech (PSI), which have been validated as good candidates for representing perceived strength as discussed in Section 4.4.2 and Section 8.1.2 respectively. The most fitted parameter for defining the gesture strength was the FMDistance [Onuma et al., 2008] (details in Section 4.4.2) that has become in the GSI, although its primary use was motion classification. After a correlation analysis of both indicators, we have derived an intensity rule. As discussed in Section 4.4.3, this rule defines that for a given PSI value, the GSI has to be near to the linear polynomial line, with a margin of 1, defined in Equation 4.1 (see Figure 4.8). Regarding the temporal alignment analysis, we have obtained a synchrony rule after a meticulous analysis of anchor points from gesture (apexes) and speech (PAPT). We obtained a synchrony window of -0.03 to 0.22 seconds (see Figure 4.7), which is similar to [Loehr, 2004], where they obtained a mean of 0 seconds and a standard deviation of 0.27 seconds, although in our tests the negative values (a delay of the stroke apex relative to the PAPT) are not as noticeable. In order to evaluate the performance of these rules, they have been used to drive a GMG which selects the most appropriate GP for each pitch accent that have to be accompanied by a gesture.

GMG synthesizes a continuous motion stream of gestures (see Section 4.4.4). Gesture synthesis is driven by speech which is analyzed in order to extract pitch accents and their associated PAPT and GSI. Then, GMG continuously associates GPs to selected pitch accents to be accompanied by a gesture. In order to select a GP, GMG takes into account its intrinsic configuration (the weights of the graph that denotes similarity between postures), and how synchrony rule and intensity rules affect to candidate GP in terms of appropriateness. Thus, cost function (see Formula 4.7) includes motion smoothness, how much synchrony rule warps the original lengths of GPs and how much appropriate are GPs while considering GSI-PSI relationship. In terms of animation, GMG provides flexibility in the generation of gestures adapted to speech emphasis as obtained Branching Factor (BF) denotes ( $BF = 36.20$ ), which means that in average there are 36 candidate GPs to transition for each pitch accent, being 261 the total of existing GPs. However, our gesture synthesizer concatenates GPs from a mocap database that also includes the annotation of the phases (e.g. preparation) belonging to these GPs. This offers the opportunity of composing GPs from phases belonging to others, which means an increment in the flexibility of the synthesizer while preserving its size. Whatever a more complex search function would be needed for that purpose.

Both GMG and BPMGs are based on motion graphs, however, there exists differences between them. To begin with, nodes from graphs do not represent the same motion unit. BPMGs nodes are motion frames rather than GMG nodes, which are GPs. The main difference is that BPMGs deal with body part motions thus requiring a composition phase, in addition to a synchronization phase due to the allowed progressiveness of transitions. BPMGs is a method for motion concatenation that could be used for any kind of motions although we only have tested with locomotions. On the other hand, GMG is a synthesizer that includes an appropriate cost metric for gesture selection, that uses common transition

techniques such as motion blending for transitioning between its nodes (GPs). In case of BPMGs search process a straightforward cost metric is used which only includes similarity between postures.

Once the gesture animations were generated, they were presented to several users as a perceptual test (see details on Section 4.4.5). Results show that the use of our intensity rule clearly improves the naturalness of animations with respect to animations with only a synchrony rule implemented. We conclude that the use of our intensity rule is essential to adapt gesture synthesis to speech emphasis, as the synchrony rule by itself is not capable of dealing successfully with such scenario. However, animations with original motion capture data are still preferred by users. Specifically, intensity animations (with synchrony and intensity rules) compared to original animations gives a CMOS = -0.44. Comparisons between a previous work by Levine [Levine et al., 2009] give a CMOS = -0.71, and between Gesture Controllers [Levine et al., 2010], CMOS = -0.46. As results show, our work gives a better score than the first work by Levine and similar to Gesture Controllers. In order to get closer to the quality of original animations it would be interesting to add gestures generated by other parts of the body besides hands and arms. Moreover, in order to relax the restriction of using only beat gestures, the intensity rule could be used to extend other works that are capable of generating other types of gestures. For example, the work presented by Stone et al. [Stone et al., 2004] could employ the intensity rule in their unit selection algorithm to generate appropriate gestures according to speech emphasis. Although the quality of our animations is quite remarkable (as explained in Section 4.4.6) there is still room for improvement. Thus, transitions between motion segments could be improved. In order to do that, we have some options like searching for the optimal transition length [Wang and Bodenheimer, 2008] or using body part motion graphs [Fernández-Baena and Miralles, 2012]. The first option is what we have addressed in BodySpeech system.

BodySpeech system is an automatic method to generate body gestures and facial animation according to speech input. Our animation system is based on GMG for synthesizing gesture performances and on lip synching techniques for facial expressions. Gesture animation stream is produced by concatenating gesture phrases aligned with pitch accents. Gestures are selected in order to maintain motion smoothness, preserve as many original motion clips as possible, and obey emphasis relation with speech, as done in the first version of GMG. Some improvements on GMG were included such as a better reuse of the input motion data and a better adaptation of GPs that preserves the strength information of the GP intact (see details in Section 4.5.2). Lip synching is generated following a standard algorithm. However, we relate speech strength with facial expressions to improve realism. Moreover, we have implemented a tool for animators that allows controlling the output animations via parameterization. A set of straightforward parameters are presented which permit a change in animation emphasis by adjusting pitch accents detection or emphasis relation between gestures/visemes with speech as explained in Section 4.5.3.

In future works we plan to improve facial animation by adding more rules related to

speech intonation. A similar study such as performed on speech-gesture analysis in Section 4.4.3 could be addressed. At the moment Kinect Face Tracking Software Development Kit (SDK) (see Figure 6.1) supports the extraction of 6 different Action Units (AU) deciphered from depth data captured by Kinect device. These AUs provides floating point values ranging from -1 to 1, representing the extreme of each facial mimic. In addition, it provides the functionality to obtain 3 rotation angles of the head movement. Based on these facial parameters and speech ones, some rules could be derived.

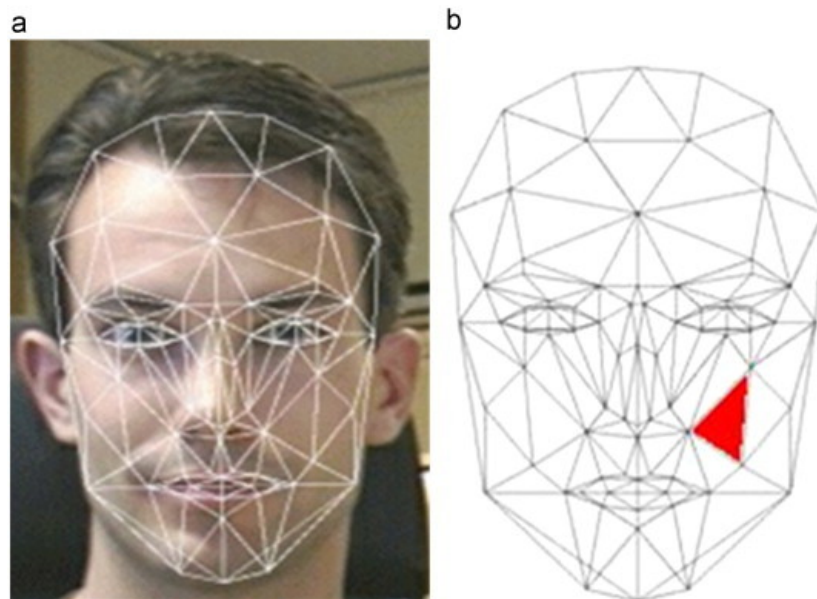


Figure 6.1: On the left, a virtual mesh fitted on to a face thanks to the detection of key face points [Behance, 2015]. On the right, a colored face of the virtual mesh whose shape denotes relaxation in the facial expression of the subject.

In addition, we plan to include independent head motion [Busso et al., 2005], head-and-eye motion [Le et al., 2012], and/or finger motion [Jörg et al., 2012] to further increase realism of the overall animations. It would also be interesting to augment gesture animation by providing lower body motion, as Luo et al. done in [Luo et al., 2009]. At this stage, we have only considered upper motion to generate body language. Due to our gesture synthesizer based on motion graphs, this improvement could be addressed by implementing a synchronized motion graph [Ng et al., 2010] that will be plugged into the GMG. Synchronized motion graphs method uses cross partial correlation defined in [Heck et al., 2006] to ensure coherence between upper and lower body part animations.

Body language involves, or can involve, upper body movements mainly by the hands, head movements, facial animation, lower body movements and even finger motion, among



others. Adding more details on synthesized animations by dealing independently with the synthesis of the movement of different parts of the body could not produce the desired results. As we mentioned in the Motivation of this manuscript, one reason that causes the Uncanny Valley is the desynchronization between body part animations. Thus, studying the relation between different type of movements involved in body language, e.g. the ones mentioned above, could serve to avoid desynchronization, beyond for driving motion synthesis. As an example, the relation between gestures and postures was studied in [Luo and Neff, 2012]. So, modeling the appropriate body language generated is the most challenging open issue, which needs powerful animation systems to be able to sustain modeling requirements, to synthesize realistic, and plausible animations. Moreover, the knowledge of how body language generation is important to achieve expressive animations. Furthermore, this knowledge could be applied to parameterize body language behavior, as we have done in Section 4.5.3, enabling to have control over the intention of the speaker. This extends the use of our gesture synthesizer by allowing to create custom animations as Stone et al. done in [Stone et al., 2004], whose dependency with speech could be used for exaggerating or relaxing the intention of the speaker. So, the combination of flexible motion control and emotional behavior is achieved.

### 6.3 AvatARs animation and interaction with character robots

Augmented reality virtual characters, or AvatARs, have a long path of further research to be done. AvatARs are displayed through AR and controlled by a TUI. Hence, they permit users to have an active role in mixed reality HCI or HRI experiences, being in charge of controlling virtual characters without breaking the immersion by using some graphical user interface. This allows to seamlessly include and control virtual characters in real world are the two contributions of AvatAR concept exposed in Chapter 5. Then, we proposed a new way of interacting with virtual characters and social robots, in our case with Pleo robot. We combined together the physical and the virtual world through this proposal. To that effect, we have created Vleo, an AvatAR with with Pleo's appearance. And then, we put them together in a mixed environment. The distinguishing feature of our work is the relationship between a virtual character and a social robot, which coexist in space and time in the interaction experience, and furthermore they also interact between them. Section 5.3.5 results demonstrated the introduction of AvatARs in the interaction between users and robots increased the engagement.

AvatARs are virtual characters with two representations: virtual and physical. The physical representation is a cube which is properly decorated with representative images, and the virtual representation is 3D virtual character with the appearance of a cartoon human. The animation of the virtual character depends on the manipulation of the physical cube, thus, the semantic mapping that exists between the cube and the virtual character is evident through the mobile screen that displays the animated virtual character. This strengthens the idea that both representations responds to one character. Another example of the se-

mantic relation between a physical character and a virtual character are Amiibo [Nintendo, 2015a] characters from Nintendo. Amiibo works with the idea of transferring the properties of physical characters to their analogues in the virtual world (i.e. Nintendo video games). In this case, although Amiibo and AvatAR have been designed for different purposes, in Amiibo only the transfer of information through the physical representation is considered, instead of allowing interactive tangible control of motion that animates the virtual characters, such as in AvatARs.

As explained in Section 5.2.3 and in Section 5.3.2, a cube is used to interactively control AvatARs motion. A cube is a simple physical object and so its manipulation is intuitive. Although, the way that the manipulation of the cube affects AvatARs motion could be not as intuitive nor evident. Because of this, in order to design intuitive interactions we only face two gestures which have analogies with other controls or common sense [Blackler et al., 2005]. First, we deal with the rotation of the cube over a plane. We use this interaction to change AvatARs locomotion (first prototype) and behavior intensity (second prototype). In both cases, the analogy is based on the increase or decrease of some control value as occurs in a myriad of tangible controls such as a volume wheel. In the first prototype, locomotion velocity is the value controlled by the rotation, being an increase of this associated with a progression from walking to jogging and running motion clips. On the other hand, a change of activation is defined by control value of the second prototype. An increase of this value brings to exaggerate movements by extending its trajectories and adding progressively more body parts that lie still immobile. Furthermore, in our second prototype we also consider the full gesture through the rotation of 90 degrees over any parallel axis of the plane in order to change the basis of the cube (see Figure 5.16 (c)). This gesture should be understood as a change of mode, which we apply for varying Vleo behavior. To that effect, top face is graphically related with the current behavior in order to easily understand its functioning (perceptual mapping [Antle, 2007]), ensuring that consistency is maintained throughout the real and the virtual world (semantic mapping [Antle, 2007]). In both interactions, we have not detected difficulties in users for understanding them as discussed in Section 5.3.5. More gestures could also be addressed to manipulate the cube, therefore, it could help with more complex character controllers than the mentioned above. In this research, we only consider character control when the cube is placed on the table, not when it is suspended in the air over the surface defined by the interaction area. Then, in order to design a more usable and intuitive tangible character controller, a user test would be necessary to assess the most appropriate gestures for controlling AvatARs.

On the other hand, the usage of a physical cube to drive Vleo, a virtual element, in order to interact with Pleo, a physical element, suggests that the relationship between both elements be more credible. Indeed, the experience we propose is similar to traditional puppetry, but giving more flexibility and expressivity in interaction due to the virtual world nature [Marshall et al., 2002] and the presence of Pleo robot, allowing us to create an interactive storytelling experience with virtual characters and robots.

The interaction design of the second prototype (see Section 5.3.2) only regarded two interactive elements: Vleo and Pleo. The Vleo cube is a tangible extension of Vleo's virtual character and together form one unified character. Accordingly, the interaction experience could be improved by adding more virtual or physical elements. In case of virtual elements, a virtual world could be designed [Chaturvedi et al., 2011]. In case of physical elements, a set of tangible objects could be used for representing virtual entities ready to be animated in the virtual world [Held et al., 2012]. These kinds of improvements will lead to complex and collaborative tasks that involve new elements, physical or virtual, to be operated by Pleo and/or Vleo. In this manner, the experience will be richer and more open to the creation of new narrative situations for interactive storytelling.

Another future line of work is to include more Pleo robots and Vleo cubes. Thus, several changes must be applied to the current interaction system setup. Using a bigger display could lead to have more Pleos and Vleos in the same location without decreasing the usability avoiding interfering with other users. However, AR performance is difficult to achieve, and we would suggest to change the detection of the cube orientation, by enhancing with electronics such as an accelerometer sensor (as in [Kranz et al., 2005]). Conversely, if we take advantage of the remote control, there is no need for more space or having to change the technology. Several children could control their Vleo in their places, and they could visualize other Pleos and Vleos through their virtual windows e.g. tablet screens or larger displays. Moreover, the inflow of more characters both virtual and physical creates the need to monitor them potentially through a cloud computing system like proposed in [Navarro et al., 2013]. Furthermore, interactive storytelling between more characters and/or elements would require the use of a planning algorithm. An example is Intent-based Partial Order Causal Link (IPOCL). Riedl et al. [Riedl and Young, 2010] which suggests the IPOCL planner that, creates causal plot progression, and structured plans that reasons on characters intentions by identifying possible character goals and why they commit to those goals. In this manner, richer experiences would be possible from the interactive storytelling point of view.

The fact of considering Avatars as smart objects lead to explore their interaction with other smart objects, in this case, we explore its interaction with Pleo robot by enabling narrative situations between both characters (virtual and robot). Although, this scenario could also be addressed in other use cases where Avatars would interact with other types of smart objects, not characters. Up to this point, the constraints of the interaction system a tablet with a camera pointing a cube and its screen faced to the user makes it difficult to imagine appropriate use cases and so how to plan it. However, the emergence of HoloLens technology [Microsoft, 2015a] (see Figure 6.2), which includes a high-definition 3D optical Head-Mounted Display (HMD) to allow for AR applications, may overcome the problems of our interaction system. So, with HoloLens, 3D virtual content presumably can be located in any place of the real world which is included within the user field of view. To that effect, a large and profit research in how virtual characters could interact with smart objects could be conducted.



Figure 6.2: A depiction of a Microsoft HoloLens user navigating Windows Holographic, with an application window on the left, and the Holographic Start menu on the right. [Wikipedia, 2015a]

## Chapter 7

# Research Outputs

### Chapter Abstract

In this chapter we summarize the main contributions of this thesis. Moreover, we emphasize the effort that author has been made in pursuing the cooperative researches of Chapter 4 and Chapter 5, which are extracted from the work of the cooperative research projects where other colleagues and researchers were involved. Then, we review the papers that have been published for the pursuit of this work.



## 7.1 Summary of contributions

In this thesis we present several new concepts and techniques that represent an improvement over the existing state of the art related to data-driven locomotion synthesis, data-driven expressive gesture synthesis and interactive control of virtual characters and its interaction with character robots. Following the same order, we can summarize our contributions in:

### 1. Data-driven locomotion synthesis

- *Locomotion synthesis method.* We propose a motion concatenation algorithm which is based on motion graphs [Lee et al., 2002] (SMG), we name it as progressive transitions using Body Part Motion Graphs (BPMGs). This algorithm achieves faster and shorter transitions between locomotion clips, and furthermore, its representation that is based in a split profile of body parts allows to find more transitions than SMG method.

"Progressive transitions using body part motion graphs" in SIGGRAPH Asia 2011 Posters, ACM [Fernández-Baena and Miralles, 2011] and "Fast response and quick progressive transitions using body part motion graphs" in Eurographics 2012 Short Papers, The Eurographics Association [Fernández-Baena and Miralles, 2012] have been published from this contribution.

### 2. Data-driven expressive gesture synthesis

- *Speech-gesture relationship analysis.* We address a study in order to understand the relationship between gestures and speech. It is done by comparing kinematic parameters of upper-body movement and prosodic parameters of speech, that describes the time and emphasis of the matching occurrences, apices and pitch accents from both cues respectively. Emphasis indicators were defined from classification being GSI the strength indicator for gestures, which coincides with FMDistance [Onuma et al., 2008] (previously used for motion retrieval). From the results, we have derived intensity and synchrony rules that relates speech with gesture.
- *Gesture synthesis system.* We propose a gesture synthesis system which is based on motion graphs [Lee et al., 2002] structure. We name it as Gesture Motion Graph (GMG). This synthesizer leads to generate a smooth and continuous flow of gestures according to an input speech. Gestures are synchronized with speech thanks to mentioned rules from the speech and gesture analysis. Results from a

user test validate the synthesis method and show the importance of synchronizing gestures with speech emphasis in order to perceive plausible performances of gestures. Then, we present BodySpeech that includes gesture synthesis through GMG and facial animation. It also enables to customize gestural performances of virtual characters.

"Gesture synthesis adapted to speech emphasis" in *Speech Communication Journal* (Special Issue on Gesture and Speech in Interaction) [Fernández-Baena et al., 2014] and "BodySpeech: A configurable gesture and facial animation system for speaking avatars" in *Computer Graphics and Virtual Reality 2013 (CGVR-2013)* [Fernández-Baena et al., 2013] have been published from these two contributions.

This part of the work was derived from THOFU project ("Tecnologías del HOtel del FUturo", CEN-20101019) granted by the Ministry of Science and Innovation of Spain, specifically, from a work package that involves the generation of gestures for a waiter virtual character. This project includes the classification of strength for gestures and speech, the correlation analysis between speech and gestures, and the development of a synthesizer of gestures. The author was in charge of the leading the research, development, follow-up and documentation/dissemination of the project. Concisely, the author take part of research design and planning of the overall project; the research of the classification of gesture strength and its correlation with speech; the research, design and implementation of the algorithm for gesture synthesis and its testing; and managing software development to end-users. However, as regards to speech issues, the author did not have any relevant incidence.

### 3. Interactive control of virtual characters and its interaction with character robots

- *Tangible representation and interactive control of virtual characters.* We propose a TUI interface, which is a physical cube, to drive locomotion synthesis of virtual characters in AR environments. We name these virtual characters as AvatARs, augmented reality (AR) virtual characters. Two ways of controlling AvatARs animation through the gesturing of the cube are presented.

"AvatAR: Tangible interaction and augmented reality in character animation" in *IUI 2014 Workshop on Interacting with Smart Objects* [Fernández-Baena and Miralles, 2014] has been published from this contribution.

- *Interaction between AvatAR and character robots.* We propose to introduce AvatARs in HRI in order to enhance engagement in children when they are playing with Pleo character robot through interactive storytelling. To that effect, we present Vleo, an AvatAR, which shares interaction area with Pleo. We design a behavioral relationship between both characters, and then we test it with children.



Results denote that engagement is increased when children interacts with Pleo through AvatARs.

"Enhancing Long-term Children to Robot Interaction Engagement Through Cloud Connectivity" in HRI 2015 Extended Abstracts [Albó-Canals et al., 2015] and "Interaction between Vleo and Pleo, a virtual social character and a social robot" in RO-MAN 2015 [Fernández-Baena et al., 2015] have been published from this contribution. This last one was awarded the Best Paper Award of the conference.

This part of the work was derived from PATRICIA project (Pain and Anxiety Treatment based on social Robot Interaction with Children to Improve pAtient experience, TIN2012-38416-C03-01,02) supported by the Ministry of Economy and Competitiveness of Spain, specifically, from a work package that explores new interactions through AR with Pleo robot where the author have been working together another contributor, Roger Boldú, who was in charge of the electronic issues, server implementation and Pleo programming.

## 7.2 Publications and additional material

During this thesis several publications have been achieved. It is important to note that all publications belonging to this thesis have been published in peer-reviewed processes and also the author of this thesis is the first author of almost all (only in one publication is not the first author) publications being its principle contributor.

List of publications:

[Fernández-Baena and Miralles, 2011] Fernández-Baena, A. and Miralles, D. (2011). Progressive transitions using body part motion graphs. In *SIGGRAPH Asia 2011 Posters*, SA '11, pages 3:1–3:2, New York, NY, USA. ACM.

[Fernández-Baena and Miralles, 2012] Fernández-Baena, A. and Miralles, D. (2012). Fast response and quick progressive transitions using body part motion graphs. In *EUROGRAPHICS ICS 2012*, pages 77–80, Cagliari, Sardinia, Italy. Eurographics Association.

[Fernández-Baena et al., 2013] Fernández-Baena, A., Antonijoan, M., Montaña, R., Fusté, A., and Amores, J. (2013). Bodyspeech: A configurable gesture and facial animation system for speaking avatars. In *Computer Graphics and Virtual Reality 2013 (CGVR -2013)*, Las Vegas, USA.

[Fernández-Baena and Miralles, 2014] Fernández-Baena, A., Montaña, R. M., Antonijoan, M., Roversi, A., Miralles, D., and Alias, F. (2014). Gesture synthesis adapted to speech emphasis. *Speech Communication*, 57(0):331–350.

[Fernández-Baena and Miralles, 2014] Fernández-Baena, A. and Miralles, D. (2014). Avatar: Tangible interaction and augmented reality in character animation. In *IUI 2014 Workshop on Interacting with Smart Objects*. Haifa, Israel.

[Albó-Canals et al., 2015] Albó-Canals, J., Fernández-Baena, A., Boldú, R., Barco, A., Navarro, J., Miralles, D., Raya, C., and Angulo, C. (2015). Enhancing long-term children to robot interaction engagement through cloud connectivity. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts, HRI'15 Extended Abstracts*, pages 105–106, New York, NY, USA. ACM.

[Fernández-Baena et al., 2015] Fernández-Baena, A., Boldú, R., Albó-Canals, J., and Miralles, D. (2015). Interaction between vleó and pleó, a virtual social character and a social robot. In *RO-MAN (to be issued)*, 2015 IEEE. IEEE.

[Fernández-Baena and Miralles, 2011] Fernández-Baena, A. and Miralles, D. (2011). Progressive transitions using body part motion graphs. In *SIGGRAPH Asia 2011 Posters*, SA '11, pages 3:1–3:2, New York, NY, USA. ACM.

**Abstract:** In this work we describe a preliminary method for progressive transitions in human locomotions. To achieve this, motion graphs have been used to synthesize body part transitions and every part has been synchronized with the other parts using time scaling. Lastly, we have compared transition costs and graph connectivity of our method to standard motion graphs. The results are promising to allow better smoothness and response in motion clips concatenation.

**PROGRESSIVE TRANSITIONS USING BODY PART MOTION GRAPHS**

Adso Fernández-Baena<sup>1</sup>  
UTM - Grup de Tecnologies Mèdies  
La Salle, Universitat Ramon Llull, Barcelona

David Miralles<sup>2</sup>  
GTM - Grup de Tecnologies Mèdies  
La Salle, Universitat Ramon Llull, Barcelona

**Abstract**

In this work we describe a preliminary method for progressive transitions in human locomotions. To achieve this, motion graphs have been used to synthesize body part transitions and every part has been synchronized with the other parts using time scaling. Lastly, we have compared transition costs and graph connectivity of our method to standard motion graphs. The results are promising to allow better smoothness and response in motion clips concatenation.

**1 Introduction**

Automatic concatenation from motion capture clips in order to create a larger stream of movements has been a wide research area in 2D character animation. In 2002, motion graphs were introduced in computer games such as [Klein et al., 2002][Lee et al., 2002]. Motion graphs consist on the concatenation of motion capture clips as a graph. Distance metrics are computed to know the similarity. Connection distance metrics compute some function to compare frames, taking into account full body poses. To refine this comparison, weights are assigned to joints giving more or less importance to them. These weights are computed using the transition points between motion clips [Wang and Hutchinson, 2008]. Finally, motions are concatenated using new queries.

Searching similar full body postures, or in standard motion graphs (SMG), can sometimes be tricky. Two frames having the same pose, except for some joints, could overcome the threshold similarity even though the whole body is similar enough. Not being able to find similar poses can affect graph connectivity and reduce the possibility of transitions between different kind of motions. In Fig. 1 there are some transition points (distance between frames) changing joints in both parts (BP). As we can see, good transition points (dark ones) are located in close positions of each step, but not exactly in the same plane. These differences show that transitioning all parts at one time (in a SMG way) could be not optimal. Therefore, we propose to divide the body into body parts and transition each part independently. Since there is no standard body part segmentation [Jang et al., 2009][Niu et al., 2010], we have focused on using the body part system from different motion capture to create a motion capture database. However, since we do not want to be focused in searching good transition points for every body part and generating their own transitions. The main problem with this approach is synchronization. Locations are the sum of body part movements with a specific resolution and cadence. We deal with this issue searching transitions for each part of the body in a time window, and then we will address an inverse synchronization and cadence. This process will be properly described in next section. We have called our method progressive transitions using body part motion graphs (BPBMG).

**2 Progressive Transitions using BPBMG's**

In this work we describe a preliminary method for progressive transitions in human locomotions. To achieve this, motion graphs have been used to synthesize body part transitions and every part has been synchronized with the other parts using time scaling. Lastly, we have compared transition costs and graph connectivity of our method to standard motion graphs. The results are promising to allow better smoothness and response in motion clips concatenation.

<sup>1</sup> e-mail: adso@utlleu.edu  
<sup>2</sup> e-mail: davidm@utlleu.edu

**Figure 1:** Illustrative matrices of different body parts. Dark ones are optimal transition points and light ones are not optimal.

**Figure 2:** Body parts diagram showing head, torso, left arm, right arm, left leg, and right leg.

At this point, we have four motion graphs for the body parts. So, we need to find transition points for each BP and transition them synchronously. The synchronization process is shown in Fig. 3. We can search for the best transition in the lower body part. Every transition consists of a set of frames: an origin frame (O), a target frame (T) and some path frames. Then, we look for the best position of its other body parts beginning with the origin point of LB part transition and ending with the target point using a 1000 second (1000) window in our case. Finally, we get several combinations of BP transitions and we use the combination with the minimum cost. We present the final body part transition in a similar manner to the body in human locomotion. Each body part has its own transition, but origin frames, path frames and target frames may be different. If we do not modify BP transitions the result might look jerky. Our goal is that all body parts coincide with one frame of the target motion. To achieve this objective, BP transition paths are finally time scaled to ensure motion synchronization.

APRIL 14, 2014, CMU

AND LEE, J. 2002. Efficient combination of partial motion graphs. In *ACM SIGGRAPH*, 473–482.

K. BODENHORN, J. K. AND CONTROL OF STATES. *ANNALS OF THE NEW YORK ACADEMY OF SCIENCES*, 2002. Motion capture. p. 2002.

DI, F. K., AND CHEN, L. 2008. Synthesis and evaluation of motion graphs. In *ACM SIGGRAPH*, 27–34.

ACHIEVING GOOD CONNECTIVITY (July), 139–152.

Percentage of graph connectivity for different methods.

**PROGRESSIVE TRANSITIONS USING BODY PART MOTION GRAPHS**

Adso Fernández-Baena  
adso@utlleu.edu

David Miralles  
davidm@utlleu.edu

laSalle  
UNIVERSITAT RAMON LLULL, BARCELONA (SPAIN)

**SIGGRAPH ASIA 2011  
HONG KONG**

**ABSTRACT**

In this work we describe a preliminary method for progressive transitions in human locomotions. To achieve this, motion graphs have been used to synthesize body part transitions and every part has been synchronized with the other parts using time scaling. Lastly, we have compared transition costs and graph connectivity of our method to standard motion graphs. The results are promising to allow better smoothness and response in motion clips concatenation.

**MOTIVATION**

Motion graphs [Klein et al., 2002][Lee et al., 2002] concatenate joint weights to generate [Baena and Bodenhorn, 2008].

- Create graph poses.
- Compute the distance between frames [Lee et al., 2002] considering joint weights to generate [Baena and Bodenhorn, 2008].
- Create graph poses.
- Find out nodes of BPBMG's and edges where there is no foot contact (only in LB).

**EXPERIMENTAL RESULTS**

Three walking motions have been used from [CMU] to compare our method with SMG results.

**Dataset1:** Normal walking and low speed walking  
**Dataset2:** Normal walking and low walking

Transition cost involves the sum of move's distances between frames, from our steps until the target motion is missing completely.

In all cases, progressive transitions have less cost than SMG, and in some cases SMG cost is more than twice as much.

**3) Search and Synchronization process**

We search for the best transition in LB. Look for the best paths of the other BPs using a 1000 second (1000) window.

• Select the combination with the minimum cost.

• BP transition paths are linearly time scaled to ensure motion synchronization.

**CONCLUSIONS**

- Specific body part transition paths.
- More possible transitions.
- Less transition cost.
- More connectivity.
- More connectivity.

**FUTURE WORK**

- Other ways to split whole body locomotion into body parts.
- Improve transition searching process.
- Improve synchronization process.
- Parameterize transitions with different factors, such as duration, time relation of BP parts.
- Measure the readability of the method.

**REFERENCES**

BODENHORN, K., AND CHEN, L. 2008. Synthesis and evaluation of motion graphs. In *ACM SIGGRAPH*, 27–34.

BAENA, A., AND MIRALLES, D. 2008. Synthesis and evaluation of motion graphs. In *ACM SIGGRAPH*, 27–34.

BAENA, A., AND MIRALLES, D. 2008. Synthesis and evaluation of motion graphs. In *ACM SIGGRAPH*, 27–34.

BAENA, A., AND MIRALLES, D. 2008. Synthesis and evaluation of motion graphs. In *ACM SIGGRAPH*, 27–34.

BAENA, A., AND MIRALLES, D. 2008. Synthesis and evaluation of motion graphs. In *ACM SIGGRAPH*, 27–34.

This publication is accompanied by a video that can be found at [goo.gl/hYz8nA](http://goo.gl/hYz8nA).

[Fernández-Baena and Miralles, 2012] Fernández-Baena, A. and Miralles, D. (2012). Fast response and quick progressive transitions using body part motion graphs. In *EUROGRAPHICS 2012*, pages 77–80, Cagliari, Sardinia, Italy. Eurographics Association.

**Abstract:** Interactive applications where 3D character animation plays an important role need avatars ready to perform different activities. This objective has been accomplished in different works [Lee et al., 2002][Zhao and Safonova, 2009] that look for transition points in motion capture clips to allow transitions between them. These works ensure realism and smoothness but their responses and transition durations depend on transition points. Working with partial motions, such as body part motions, allows finding specific transition points for each part in order to optimize whole body transitions in a progressive way. This can be achieved with body part motion graphs (BPMGs) [Fernández-Baena and Miralles, 2011]. In this work we want to show that progressive transitions generated by BPMGs have fast response and quick execution. In order to demonstrate this we have compared BPMGs' transitions against standard motion graphs (SMG) transitions. The results we have obtained show that our method allows more reaction velocity and execution. Moreover BPMGs' transitions are smoother than SMG transitions.

<p>EUROGRAPHICS 2012 / C. Antiga, E. Pappas</p> <p>Short Paper</p> <h3>Fast Response and Quick Progressive Transitions using Body Part Motion Graphs</h3> <p>Albio Fernández-Baena and David Miralles GTM - Grup de Tecnologia Mèdia, La Salle - Universitat Ramon Llull, Barcelona</p> <p><b>Abstract</b> Interactive applications where 3D character animation plays an important role need avatars ready to perform different activities. This objective has been accomplished in different works [Lee et al., 2002][Zhao and Safonova, 2009] that look for transition points in motion capture clips to allow transitions between them. These works ensure realism and smoothness but their responses and transition durations depend on transition points. Working with partial motions, such as body part motions, allows finding specific transition points for each part in order to optimize whole body transitions in a progressive way. This can be achieved with body part motion graphs (BPMGs) [Fernández-Baena and Miralles, 2011]. In this work we want to show that progressive transitions generated by BPMGs have fast response and quick execution. In order to demonstrate this we have compared BPMGs' transitions against standard motion graphs (SMG) transitions. The results we have obtained show that our method allows more reaction velocity and execution. Moreover BPMGs' transitions are smoother than SMG transitions.</p> <p>Categories and Subject Descriptors according to ACM CCS: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation</p> <p><b>1. Introduction</b> Character animation in videogames and interactive applications is becoming more important and a mark of quality. Achieving real and plausible motions to drive characters is made done in two different ways: animating motion by hand or concealing motion capture clips. The second option has motivated the development of several motion construction systems in the research community. One of the most important is motion graph. Motion graph appears in 3D in various works such as [KIGUCHI][LAFONT][LAFONT] and consists in subdividing motion capture data in a graph structure associating frames under some constraints. These frames are connected subdividing user input using specific algorithms of graph theory. In this manner, transitions between different behaviors are achieved.</p> <p>A variety of graph-based motion synthesis systems have been developed after motion graphs appeared such as [KIGUCHI][KIGUCHI][KIGUCHI][LAFONT]. These systems have different peculiarities that hinder comparison between them. In [LAFONT] some methods are proposed for comparing performance of graph-based methods. This comparison is conducted based on a set of tasks and environment capabilities.</p> <p>In interactive applications, what is important is the response of characters and how quickly they change their behavior satisfying user commands. Wang and Bodenheimer [WANG] made an exhaustive study of transition length in these transitions which is related with transition response. They focused on determining which frame length to use in order to achieve pleasant animations.</p> <p>More recent works such [MIRALLES][MIRALLES] are more focused on generating animations that respond quickly to application requests. Both works use continuous control of motion generation, unlike graph-based methods. Graph-based methods are discrete and they are focused on how to transition from one clip to another. [MIRALLES] applied their discrete methods such as motion graph are slower in response by a factor of two than continuous one. We will show that body part motion graphs are up to three times faster than standard motion graphs in terms of response time.</p> <p><b>2. Body Part Motion Graphs</b> Body part motion graphs (BPMG) or [FERNÁNDEZ-BAENA] is a motion synthesis method based on standard motion graph (SMG).</p>	<p>Video Graphs have a framework of 60 graphs and body part motion graphs using this framework to generate the performance of Behavior Transition and transition</p> <p>It is important to know SMG's. After searching 60 clips to get a standard body part motion graph motion graph <math>P_1</math>, an end frame by <math>J_1</math>, <math>S_1</math>, each clip receive <math>J_1</math> transition position. We will use <math>min(T_1, L_1)</math></p>  <p>SMG results, we notice that only 5% of the clips are used while BPMG's the 40% of clips are used. This is because the transition between clips in SMG is not progressive. In BPMG's the transition time is close to zero while SMG transition time is long many frames. In our method, we notice that the transition time is close to zero while SMG transition time is long many frames. In our method, we notice that the transition time is close to zero while SMG transition time is long many frames.</p> <p><math>min(T_1, L_1)</math> (2)</p> <p>is the target behavior, <math>C_1</math> and <math>min(T_1, L_1)</math> from first instance of <math>C_1</math> and <math>min(T_1, L_1)</math> from first instance of <math>C_1</math>. We have also in behaviors. In order to get transition duration time, we can evaluate <math>min(T_1, L_1)</math> and response times between thresholds for both minimum values that the distance. Threshold and BPMG's response are better than</p> <p>SMG results, we notice that only 5% of the clips are used while BPMG's the 40% of clips are used. This is because the transition between clips in SMG is not progressive. In BPMG's the transition time is close to zero while SMG transition time is long many frames. In our method, we notice that the transition time is close to zero while SMG transition time is long many frames.</p> <p><math>min(T_1, L_1)</math> (1)</p> <p>to represent the transition duration time between body part</p> <p>© Eurographics Association 2012.</p>	<p>Video Graphs one and transition response of these both methods information about behavior [MIRALLES] proposed method to evaluate how fast other actions of the character with action fragments and the end of evaluating local maximum in a subsequence. This measure is human the motion is related to the location. So, we can be location to any behavior to behavior LM</p> <p><math>min(T_1, L_1)</math> (2)</p> <p>is the target behavior, <math>C_1</math> and <math>min(T_1, L_1)</math> from first instance of <math>C_1</math> and <math>min(T_1, L_1)</math> from first instance of <math>C_1</math>. We have also in behaviors. In order to get transition duration time, we can evaluate <math>min(T_1, L_1)</math> and response times between thresholds for both minimum values that the distance. Threshold and BPMG's response are better than</p> <p>SMG results, we notice that only 5% of the clips are used while BPMG's the 40% of clips are used. This is because the transition between clips in SMG is not progressive. In BPMG's the transition time is close to zero while SMG transition time is long many frames. In our method, we notice that the transition time is close to zero while SMG transition time is long many frames.</p> <p><math>min(T_1, L_1)</math> (1)</p> <p>to represent the transition duration time between body part</p> <p>© Eurographics Association 2012.</p>	<p>Video Graphs take on Computer Animation 2009, ACM SIGGRAPH 2009, ACM SIGGRAPH 2010, ACM SIGGRAPH 2011, ACM SIGGRAPH 2012, ACM SIGGRAPH 2013, ACM SIGGRAPH 2014, ACM SIGGRAPH 2015, ACM SIGGRAPH 2016, ACM SIGGRAPH 2017, ACM SIGGRAPH 2018, ACM SIGGRAPH 2019, ACM SIGGRAPH 2020, ACM SIGGRAPH 2021, ACM SIGGRAPH 2022, ACM SIGGRAPH 2023, ACM SIGGRAPH 2024, ACM SIGGRAPH 2025, ACM SIGGRAPH 2026, ACM SIGGRAPH 2027, ACM SIGGRAPH 2028, ACM SIGGRAPH 2029, ACM SIGGRAPH 2030, ACM SIGGRAPH 2031, ACM SIGGRAPH 2032, ACM SIGGRAPH 2033, ACM SIGGRAPH 2034, ACM SIGGRAPH 2035, ACM SIGGRAPH 2036, ACM SIGGRAPH 2037, ACM SIGGRAPH 2038, ACM SIGGRAPH 2039, ACM SIGGRAPH 2040, ACM SIGGRAPH 2041, ACM SIGGRAPH 2042, ACM SIGGRAPH 2043, ACM SIGGRAPH 2044, ACM SIGGRAPH 2045, ACM SIGGRAPH 2046, ACM SIGGRAPH 2047, ACM SIGGRAPH 2048, ACM SIGGRAPH 2049, ACM SIGGRAPH 2050, ACM SIGGRAPH 2051, ACM SIGGRAPH 2052, ACM SIGGRAPH 2053, ACM SIGGRAPH 2054, ACM SIGGRAPH 2055, ACM SIGGRAPH 2056, ACM SIGGRAPH 2057, ACM SIGGRAPH 2058, ACM SIGGRAPH 2059, ACM SIGGRAPH 2060, ACM SIGGRAPH 2061, ACM SIGGRAPH 2062, ACM SIGGRAPH 2063, ACM SIGGRAPH 2064, ACM SIGGRAPH 2065, ACM SIGGRAPH 2066, ACM SIGGRAPH 2067, ACM SIGGRAPH 2068, ACM SIGGRAPH 2069, ACM SIGGRAPH 2070, ACM SIGGRAPH 2071, ACM SIGGRAPH 2072, ACM SIGGRAPH 2073, ACM SIGGRAPH 2074, ACM SIGGRAPH 2075, ACM SIGGRAPH 2076, ACM SIGGRAPH 2077, ACM SIGGRAPH 2078, ACM SIGGRAPH 2079, ACM SIGGRAPH 2080, ACM SIGGRAPH 2081, ACM SIGGRAPH 2082, ACM SIGGRAPH 2083, ACM SIGGRAPH 2084, ACM SIGGRAPH 2085, ACM SIGGRAPH 2086, ACM SIGGRAPH 2087, ACM SIGGRAPH 2088, ACM SIGGRAPH 2089, ACM SIGGRAPH 2090, ACM SIGGRAPH 2091, ACM SIGGRAPH 2092, ACM SIGGRAPH 2093, ACM SIGGRAPH 2094, ACM SIGGRAPH 2095, ACM SIGGRAPH 2096, ACM SIGGRAPH 2097, ACM SIGGRAPH 2098, ACM SIGGRAPH 2099, ACM SIGGRAPH 2100, ACM SIGGRAPH 2101, ACM SIGGRAPH 2102, ACM SIGGRAPH 2103, ACM SIGGRAPH 2104, ACM SIGGRAPH 2105, ACM SIGGRAPH 2106, ACM SIGGRAPH 2107, ACM SIGGRAPH 2108, ACM SIGGRAPH 2109, ACM SIGGRAPH 2110, ACM SIGGRAPH 2111, ACM SIGGRAPH 2112, ACM SIGGRAPH 2113, ACM SIGGRAPH 2114, ACM SIGGRAPH 2115, ACM SIGGRAPH 2116, ACM SIGGRAPH 2117, ACM SIGGRAPH 2118, ACM SIGGRAPH 2119, ACM SIGGRAPH 2120, ACM SIGGRAPH 2121, ACM SIGGRAPH 2122, ACM SIGGRAPH 2123, ACM SIGGRAPH 2124, ACM SIGGRAPH 2125, ACM SIGGRAPH 2126, ACM SIGGRAPH 2127, ACM SIGGRAPH 2128, ACM SIGGRAPH 2129, ACM SIGGRAPH 2130, ACM SIGGRAPH 2131, ACM SIGGRAPH 2132, ACM SIGGRAPH 2133, ACM SIGGRAPH 2134, ACM SIGGRAPH 2135, ACM SIGGRAPH 2136, ACM SIGGRAPH 2137, ACM SIGGRAPH 2138, ACM SIGGRAPH 2139, ACM SIGGRAPH 2140, ACM SIGGRAPH 2141, ACM SIGGRAPH 2142, ACM SIGGRAPH 2143, ACM SIGGRAPH 2144, ACM SIGGRAPH 2145, ACM SIGGRAPH 2146, ACM SIGGRAPH 2147, ACM SIGGRAPH 2148, ACM SIGGRAPH 2149, ACM SIGGRAPH 2150, ACM SIGGRAPH 2151, ACM SIGGRAPH 2152, ACM SIGGRAPH 2153, ACM SIGGRAPH 2154, ACM SIGGRAPH 2155, ACM SIGGRAPH 2156, ACM SIGGRAPH 2157, ACM SIGGRAPH 2158, ACM SIGGRAPH 2159, ACM SIGGRAPH 2160, ACM SIGGRAPH 2161, ACM SIGGRAPH 2162, ACM SIGGRAPH 2163, ACM SIGGRAPH 2164, ACM SIGGRAPH 2165, ACM SIGGRAPH 2166, ACM SIGGRAPH 2167, ACM SIGGRAPH 2168, ACM SIGGRAPH 2169, ACM SIGGRAPH 2170, ACM SIGGRAPH 2171, ACM SIGGRAPH 2172, ACM SIGGRAPH 2173, ACM SIGGRAPH 2174, ACM SIGGRAPH 2175, ACM SIGGRAPH 2176, ACM SIGGRAPH 2177, ACM SIGGRAPH 2178, ACM SIGGRAPH 2179, ACM SIGGRAPH 2180, ACM SIGGRAPH 2181, ACM SIGGRAPH 2182, ACM SIGGRAPH 2183, ACM SIGGRAPH 2184, ACM SIGGRAPH 2185, ACM SIGGRAPH 2186, ACM SIGGRAPH 2187, ACM SIGGRAPH 2188, ACM SIGGRAPH 2189, ACM SIGGRAPH 2190, ACM SIGGRAPH 2191, ACM SIGGRAPH 2192, ACM SIGGRAPH 2193, ACM SIGGRAPH 2194, ACM SIGGRAPH 2195, ACM SIGGRAPH 2196, ACM SIGGRAPH 2197, ACM SIGGRAPH 2198, ACM SIGGRAPH 2199, ACM SIGGRAPH 2200, ACM SIGGRAPH 2201, ACM SIGGRAPH 2202, ACM SIGGRAPH 2203, ACM SIGGRAPH 2204, ACM SIGGRAPH 2205, ACM SIGGRAPH 2206, ACM SIGGRAPH 2207, ACM SIGGRAPH 2208, ACM SIGGRAPH 2209, ACM SIGGRAPH 2210, ACM SIGGRAPH 2211, ACM SIGGRAPH 2212, ACM SIGGRAPH 2213, ACM SIGGRAPH 2214, ACM SIGGRAPH 2215, ACM SIGGRAPH 2216, ACM SIGGRAPH 2217, ACM SIGGRAPH 2218, ACM SIGGRAPH 2219, ACM SIGGRAPH 2220, ACM SIGGRAPH 2221, ACM SIGGRAPH 2222, ACM SIGGRAPH 2223, ACM SIGGRAPH 2224, ACM SIGGRAPH 2225, ACM SIGGRAPH 2226, ACM SIGGRAPH 2227, ACM SIGGRAPH 2228, ACM SIGGRAPH 2229, ACM SIGGRAPH 2230, ACM SIGGRAPH 2231, ACM SIGGRAPH 2232, ACM SIGGRAPH 2233, ACM SIGGRAPH 2234, ACM SIGGRAPH 2235, ACM SIGGRAPH 2236, ACM SIGGRAPH 2237, ACM SIGGRAPH 2238, ACM SIGGRAPH 2239, ACM SIGGRAPH 2240, ACM SIGGRAPH 2241, ACM SIGGRAPH 2242, ACM SIGGRAPH 2243, ACM SIGGRAPH 2244, ACM SIGGRAPH 2245, ACM SIGGRAPH 2246, ACM SIGGRAPH 2247, ACM SIGGRAPH 2248, ACM SIGGRAPH 2249, ACM SIGGRAPH 2250, ACM SIGGRAPH 2251, ACM SIGGRAPH 2252, ACM SIGGRAPH 2253, ACM SIGGRAPH 2254, ACM SIGGRAPH 2255, ACM SIGGRAPH 2256, ACM SIGGRAPH 2257, ACM SIGGRAPH 2258, ACM SIGGRAPH 2259, ACM SIGGRAPH 2260, ACM SIGGRAPH 2261, ACM SIGGRAPH 2262, ACM SIGGRAPH 2263, ACM SIGGRAPH 2264, ACM SIGGRAPH 2265, ACM SIGGRAPH 2266, ACM SIGGRAPH 2267, ACM SIGGRAPH 2268, ACM SIGGRAPH 2269, ACM SIGGRAPH 2270, ACM SIGGRAPH 2271, ACM SIGGRAPH 2272, ACM SIGGRAPH 2273, ACM SIGGRAPH 2274, ACM SIGGRAPH 2275, ACM SIGGRAPH 2276, ACM SIGGRAPH 2277, ACM SIGGRAPH 2278, ACM SIGGRAPH 2279, ACM SIGGRAPH 2280, ACM SIGGRAPH 2281, ACM SIGGRAPH 2282, ACM SIGGRAPH 2283, ACM SIGGRAPH 2284, ACM SIGGRAPH 2285, ACM SIGGRAPH 2286, ACM SIGGRAPH 2287, ACM SIGGRAPH 2288, ACM SIGGRAPH 2289, ACM SIGGRAPH 2290, ACM SIGGRAPH 2291, ACM SIGGRAPH 2292, ACM SIGGRAPH 2293, ACM SIGGRAPH 2294, ACM SIGGRAPH 2295, ACM SIGGRAPH 2296, ACM SIGGRAPH 2297, ACM SIGGRAPH 2298, ACM SIGGRAPH 2299, ACM SIGGRAPH 2300, ACM SIGGRAPH 2301, ACM SIGGRAPH 2302, ACM SIGGRAPH 2303, ACM SIGGRAPH 2304, ACM SIGGRAPH 2305, ACM SIGGRAPH 2306, ACM SIGGRAPH 2307, ACM SIGGRAPH 2308, ACM SIGGRAPH 2309, ACM SIGGRAPH 2310, ACM SIGGRAPH 2311, ACM SIGGRAPH 2312, ACM SIGGRAPH 2313, ACM SIGGRAPH 2314, ACM SIGGRAPH 2315, ACM SIGGRAPH 2316, ACM SIGGRAPH 2317, ACM SIGGRAPH 2318, ACM SIGGRAPH 2319, ACM SIGGRAPH 2320, ACM SIGGRAPH 2321, ACM SIGGRAPH 2322, ACM SIGGRAPH 2323, ACM SIGGRAPH 2324, ACM SIGGRAPH 2325, ACM SIGGRAPH 2326, ACM SIGGRAPH 2327, ACM SIGGRAPH 2328, ACM SIGGRAPH 2329, ACM SIGGRAPH 2330, ACM SIGGRAPH 2331, ACM SIGGRAPH 2332, ACM SIGGRAPH 2333, ACM SIGGRAPH 2334, ACM SIGGRAPH 2335, ACM SIGGRAPH 2336, ACM SIGGRAPH 2337, ACM SIGGRAPH 2338, ACM SIGGRAPH 2339, ACM SIGGRAPH 2340, ACM SIGGRAPH 2341, ACM SIGGRAPH 2342, ACM SIGGRAPH 2343, ACM SIGGRAPH 2344, ACM SIGGRAPH 2345, ACM SIGGRAPH 2346, ACM SIGGRAPH 2347, ACM SIGGRAPH 2348, ACM SIGGRAPH 2349, ACM SIGGRAPH 2350, ACM SIGGRAPH 2351, ACM SIGGRAPH 2352, ACM SIGGRAPH 2353, ACM SIGGRAPH 2354, ACM SIGGRAPH 2355, ACM SIGGRAPH 2356, ACM SIGGRAPH 2357, ACM SIGGRAPH 2358, ACM SIGGRAPH 2359, ACM SIGGRAPH 2360, ACM SIGGRAPH 2361, ACM SIGGRAPH 2362, ACM SIGGRAPH 2363, ACM SIGGRAPH 2364, ACM SIGGRAPH 2365, ACM SIGGRAPH 2366, ACM SIGGRAPH 2367, ACM SIGGRAPH 2368, ACM SIGGRAPH 2369, ACM SIGGRAPH 2370, ACM SIGGRAPH 2371, ACM SIGGRAPH 2372, ACM SIGGRAPH 2373, ACM SIGGRAPH 2374, ACM SIGGRAPH 2375, ACM SIGGRAPH 2376, ACM SIGGRAPH 2377, ACM SIGGRAPH 2378, ACM SIGGRAPH 2379, ACM SIGGRAPH 2380, ACM SIGGRAPH 2381, ACM SIGGRAPH 2382, ACM SIGGRAPH 2383, ACM SIGGRAPH 2384, ACM SIGGRAPH 2385, ACM SIGGRAPH 2386, ACM SIGGRAPH 2387, ACM SIGGRAPH 2388, ACM SIGGRAPH 2389, ACM SIGGRAPH 2390, ACM SIGGRAPH 2391, ACM SIGGRAPH 2392, ACM SIGGRAPH 2393, ACM SIGGRAPH 2394, ACM SIGGRAPH 2395, ACM SIGGRAPH 2396, ACM SIGGRAPH 2397, ACM SIGGRAPH 2398, ACM SIGGRAPH 2399, ACM SIGGRAPH 2400, ACM SIGGRAPH 2401, ACM SIGGRAPH 2402, ACM SIGGRAPH 2403, ACM SIGGRAPH 2404, ACM SIGGRAPH 2405, ACM SIGGRAPH 2406, ACM SIGGRAPH 2407, ACM SIGGRAPH 2408, ACM SIGGRAPH 2409, ACM SIGGRAPH 2410, ACM SIGGRAPH 2411, ACM SIGGRAPH 2412, ACM SIGGRAPH 2413, ACM SIGGRAPH 2414, ACM SIGGRAPH 2415, ACM SIGGRAPH 2416, ACM SIGGRAPH 2417, ACM SIGGRAPH 2418, ACM SIGGRAPH 2419, ACM SIGGRAPH 2420, ACM SIGGRAPH 2421, ACM SIGGRAPH 2422, ACM SIGGRAPH 2423, ACM SIGGRAPH 2424, ACM SIGGRAPH 2425, ACM SIGGRAPH 2426, ACM SIGGRAPH 2427, ACM SIGGRAPH 2428, ACM SIGGRAPH 2429, ACM SIGGRAPH 2430, ACM SIGGRAPH 2431, ACM SIGGRAPH 2432, ACM SIGGRAPH 2433, ACM SIGGRAPH 2434, ACM SIGGRAPH 2435, ACM SIGGRAPH 2436, ACM SIGGRAPH 2437, ACM SIGGRAPH 2438, ACM SIGGRAPH 2439, ACM SIGGRAPH 2440, ACM SIGGRAPH 2441, ACM SIGGRAPH 2442, ACM SIGGRAPH 2443, ACM SIGGRAPH 2444, ACM SIGGRAPH 2445, ACM SIGGRAPH 2446, ACM SIGGRAPH 2447, ACM SIGGRAPH 2448, ACM SIGGRAPH 2449, ACM SIGGRAPH 2450, ACM SIGGRAPH 2451, ACM SIGGRAPH 2452, ACM SIGGRAPH 2453, ACM SIGGRAPH 2454, ACM SIGGRAPH 2455, ACM SIGGRAPH 2456, ACM SIGGRAPH 2457, ACM SIGGRAPH 2458, ACM SIGGRAPH 2459, ACM SIGGRAPH 2460, ACM SIGGRAPH 2461, ACM SIGGRAPH 2462, ACM SIGGRAPH 2463, ACM SIGGRAPH 2464, ACM SIGGRAPH 2465, ACM SIGGRAPH 2466, ACM SIGGRAPH 2467, ACM SIGGRAPH 2468, ACM SIGGRAPH 2469, ACM SIGGRAPH 2470, ACM SIGGRAPH 2471, ACM SIGGRAPH 2472, ACM SIGGRAPH 2473, ACM SIGGRAPH 2474, ACM SIGGRAPH 2475, ACM SIGGRAPH 2476, ACM SIGGRAPH 2477, ACM SIGGRAPH 2478, ACM SIGGRAPH 2479, ACM SIGGRAPH 2480, ACM SIGGRAPH 2481, ACM SIGGRAPH 2482, ACM SIGGRAPH 2483, ACM SIGGRAPH 2484, ACM SIGGRAPH 2485, ACM SIGGRAPH 2486, ACM SIGGRAPH 2487, ACM SIGGRAPH 2488, ACM SIGGRAPH 2489, ACM SIGGRAPH 2490, ACM SIGGRAPH 2491, ACM SIGGRAPH 2492, ACM SIGGRAPH 2493, ACM SIGGRAPH 2494, ACM SIGGRAPH 2495, ACM SIGGRAPH 2496, ACM SIGGRAPH 2497, ACM SIGGRAPH 2498, ACM SIGGRAPH 2499, ACM SIGGRAPH 2500, ACM SIGGRAPH 2501, ACM SIGGRAPH 2502, ACM SIGGRAPH 2503, ACM SIGGRAPH 2504, ACM SIGGRAPH 2505, ACM SIGGRAPH 2506, ACM SIGGRAPH 2507, ACM SIGGRAPH 2508, ACM SIGGRAPH 2509, ACM SIGGRAPH 2510, ACM SIGGRAPH 2511, ACM SIGGRAPH 2512, ACM SIGGRAPH 2513, ACM SIGGRAPH 2514, ACM SIGGRAPH 2515, ACM SIGGRAPH 2516, ACM SIGGRAPH 2517, ACM SIGGRAPH 2518, ACM SIGGRAPH 2519, ACM SIGGRAPH 2520, ACM SIGGRAPH 2521, ACM SIGGRAPH 2522, ACM SIGGRAPH 2523, ACM SIGGRAPH 2524, ACM SIGGRAPH 2525, ACM SIGGRAPH 2526, ACM SIGGRAPH 2527, ACM SIGGRAPH 2528, ACM SIGGRAPH 2529, ACM SIGGRAPH 2530, ACM SIGGRAPH 2531, ACM SIGGRAPH 2532, ACM SIGGRAPH 2533, ACM SIGGRAPH 2534, ACM SIGGRAPH 2535, ACM SIGGRAPH 2536, ACM SIGGRAPH 2537, ACM SIGGRAPH 2538, ACM SIGGRAPH 2539, ACM SIGGRAPH 2540, ACM SIGGRAPH 2541, ACM SIGGRAPH 2542, ACM SIGGRAPH 2543, ACM SIGGRAPH 2544, ACM SIGGRAPH 2545, ACM SIGGRAPH 2546, ACM SIGGRAPH 2547, ACM SIGGRAPH 2548, ACM SIGGRAPH 2549, ACM SIGGRAPH 2550, ACM SIGGRAPH 2551, ACM SIGGRAPH 2552, ACM SIGGRAPH 2553, ACM SIGGRAPH 2554, ACM SIGGRAPH 2555, ACM SIGGRAPH 2556, ACM SIGGRAPH 2557, ACM SIGGRAPH 2558, ACM SIGGRAPH 2559, ACM SIGGRAPH 2560, ACM SIGGRAPH 2561, ACM SIGGRAPH 2562, ACM SIGGRAPH 2563, ACM SIGGRAPH 2564, ACM SIGGRAPH 2565, ACM SIGGRAPH 2566, ACM SIGGRAPH 2567, ACM SIGGRAPH 2568, ACM SIGGRAPH 2569, ACM SIGGRAPH 2570, ACM SIGGRAPH 2571, ACM SIGGRAPH 2572, ACM SIGGRAPH 2573, ACM SIGGRAPH 2574, ACM SIGGRAPH 2575, ACM SIGGRAPH 2576, ACM SIGGRAPH 2577, ACM SIGGRAPH 2578, ACM SIGGRAPH 2579, ACM SIGGRAPH 2580, ACM SIGGRAPH 2581, ACM SIGGRAPH 2582, ACM SIGGRAPH 2583, ACM SIGGRAPH 2584, ACM SIGGRAPH 2585, ACM SIGGRAPH 2586, ACM SIGGRAPH 2587, ACM SIGGRAPH 2588, ACM SIGGRAPH 2589, ACM SIGGRAPH 2590, ACM SIGGRAPH 2591, ACM SIGGRAPH 2592, ACM SIGGRAPH 2593, ACM SIGGRAPH 2594, ACM SIGGRAPH 2595, ACM SIGGRAPH 2596, ACM SIGGRAPH 2597, ACM SIGGRAPH 2598, ACM SIGGRAPH 2599, ACM SIGGRAPH 2600, ACM SIGGRAPH 2601, ACM SIGGRAPH 2602, ACM SIGGRAPH 2603, ACM SIGGRAPH 2604, ACM SIGGRAPH 2605, ACM SIGGRAPH 2606, ACM SIGGRAPH 2607, ACM SIGGRAPH 2608, ACM SIGGRAPH 2609, ACM SIGGRAPH 2610, ACM SIGGRAPH 2611, ACM SIGGRAPH 2612, ACM SIGGRAPH 2613, ACM SIGGRAPH 2614, ACM SIGGRAPH 2615, ACM SIGGRAPH 2616, ACM SIGGRAPH 2617, ACM SIGGRAPH 2618, ACM SIGGRAPH 2619, ACM SIGGRAPH 2620, ACM SIGGRAPH 2621, ACM SIGGRAPH 2622, ACM SIGGRAPH 2623, ACM SIGGRAPH 2624, ACM SIGGRAPH 2625, ACM SIGGRAPH 2626, ACM SIGGRAPH 2627, ACM SIGGRAPH 2628, ACM SIGGRAPH 2629, ACM SIGGRAPH 2630, ACM SIGGRAPH 2631, ACM SIGGRAPH 2632, ACM SIGGRAPH 2633, ACM SIGGRAPH 2634, ACM SIGGRAPH 2635, ACM SIGGRAPH 2636, ACM SIGGRAPH 2637, ACM SIGGRAPH 2638, ACM SIGGRAPH 2639, ACM SIGGRAPH 2640, ACM SIGGRAPH 2641, ACM SIGGRAPH 2642, ACM SIGGRAPH 2643, ACM SIGGRAPH 2644, ACM SIGGRAPH 2645, ACM SIGGRAPH 2646, ACM SIGGRAPH 2647, ACM SIGGRAPH 2648, ACM SIGGRAPH 2649, ACM SIGGRAPH 2650, ACM SIGGRAPH 2651, ACM SIGGRAPH 2652, ACM SIGGRAPH 2653, ACM SIGGRAPH 2654, ACM SIGGRAPH 2655, ACM SIGGRAPH 2656, ACM SIGGRAPH 2657, ACM SIGGRAPH 2658, ACM SIGGRAPH 2659, ACM SIGGRAPH 2660, ACM SIGGRAPH 2661, ACM SIGGRAPH 2662, ACM SIGGRAPH 2663, ACM SIGGRAPH 2664, ACM SIGGRAPH 2665, ACM SIGGRAPH 2666, ACM SIGGRAPH 2667, ACM SIGGRAPH 2668, ACM SIGGRAPH 2669, ACM SIGGRAPH 2670, ACM SIGGRAPH 2671, ACM SIGGRAPH 2672, ACM SIGGRAPH 2673, ACM SIGGRAPH 2674, ACM SIGGRAPH 2675, ACM SIGGRAPH 2676, ACM SIGGRAPH 2677, ACM SIGGRAPH 2678, ACM SIGGRAPH 2679, ACM SIGGRAPH 2680, ACM SIGGRAPH 2681, ACM SIGGRAPH 2682, ACM SIGGRAPH 2683, ACM SIGGRAPH 2684, ACM SIGGRAPH 2685, ACM SIGGRAPH 2686, ACM SIGGRAPH 2687, ACM SIGGRAPH 2688, ACM SIGGRAPH 2689, ACM SIGGRAPH 2690, ACM SIGGRAPH 2691, ACM SIGGRAPH 2692, ACM SIGGRAPH 2693, ACM SIGGRAPH 2694, ACM SIGGRAPH 2695, ACM SIGGRAPH 2696, ACM SIGGRAPH 2697, ACM SIGGRAPH 2698, ACM SIGGRAPH 2699, ACM SIGGRAPH 2700, ACM SIGGRAPH 2701, ACM SIGGRAPH 2702, ACM SIGGRAPH 2703, ACM SIGGRAPH 2704, ACM SIGGRAPH 2705, ACM SIGGRAPH 2706, ACM SIGGRAPH 2707, ACM SIGGRAPH 2708, ACM SIGGRAPH 2709, ACM SIGGRAPH 2710, ACM SIGGRAPH 2711, ACM SIGGRAPH 2712, ACM SIGGRAPH 2713, ACM SIGGRAPH 2714, ACM SIGGRAPH 2715, ACM SIGGRAPH 2716, ACM SIGGRAPH 2717, ACM SIGGRAPH 2718, ACM SIGGRAPH 2719, ACM SIGGRAPH 2720, ACM SIGGRAPH 2721, ACM SIGGRAPH 2722, ACM SIGGRAPH 2723, ACM SIGGRAPH 2724, ACM SIGGRAPH 2725, ACM SIGGRAPH 2726, ACM SIGGRAPH 2727, ACM SIGGRAPH 2728, ACM SIGGRAPH 2729, ACM SIGGRAPH 2730, ACM SIGGRAPH 2731, ACM SIGGRAPH 2732, ACM SIGGRAPH 2733, ACM SIGGRAPH 2734, ACM SIGGRAPH 2735, ACM SIGGRAPH 2736, ACM SIGGRAPH 2737, ACM SIGGRAPH 2738, ACM SIGGRAPH 2739, ACM SIGGRAPH 2740, ACM SIGGRAPH 2741, ACM SIGGRAPH 2742, ACM SIGGRAPH 2743, ACM SIGGRAPH 2744, ACM SIGGRAPH 2745, ACM SIGGRAPH 2746, ACM SIGGRAPH 2747, ACM SIGGRAPH 2748, ACM SIGGRAPH 2749, ACM SIGGRAPH 2750, ACM SIGGRAPH 2751, ACM SIGGRAPH 2752, ACM SIGGRAPH 2753, ACM SIGGRAPH 2754, ACM SIGGRAPH 2755, ACM SIGGRAPH 2756, ACM SIGGRAPH 2757, ACM SIGGRAPH 2758, ACM SIGGRAPH 2759, ACM SIGGRAPH 2760, ACM SIGGRAPH 2761, ACM SIGGRAPH 2762, ACM SIGGRAPH 2763, ACM SIGGRAPH 2764, ACM SIGGRAPH 2765, ACM SIGGRAPH 2766, ACM SIGGRAPH 2767, ACM SIGGRAPH 2768, ACM SIGGRAPH 2769, ACM SIGGRAPH 2770, ACM SIGGRAPH 2771, ACM SIGGRAPH 2772, ACM SIGGRAPH 2773, ACM SIGGRAPH 2774, ACM SIGGRAPH 2775, ACM SIGGRAPH 2776, ACM SIGGRAPH 2777, ACM SIGGRAPH 2778, ACM SIGGRAPH 2779, ACM SIGGRAPH 2780, ACM SIGGRAPH 2781, ACM SIGGRAPH 2782, ACM SIGGRAPH 2783, ACM SIGGRAPH 2784, ACM SIGGRAPH 2785, ACM SIGGRAPH 2786, ACM SIGGRAPH 2787, ACM SIGGRAPH 2788, ACM SIGGRAPH 2789, ACM SIGGRAPH 2790, ACM SIGGRAPH 2791, ACM SIGGRAPH 2792, ACM SIGGRAPH 2793, ACM SIGGRAPH 2794, ACM SIGGRAPH 2795, ACM SIGGRAPH 2796, ACM SIGGRAPH 2797, ACM SIGGRAPH 2798, ACM SIGGRAPH 2799, ACM SIGGRAPH 2800, ACM SIGGRAPH 2801, ACM SIGGRAPH 2802, ACM SIGGRAPH 2803, ACM SIGGRAPH 2804, ACM SIGGRAPH 2805, ACM SIGGRAPH 2806, ACM SIGGRAPH 2807, ACM SIGGRAPH 2808, ACM SIGGRAPH 2809, ACM SIGGRAPH 2810, ACM SIGGRAPH 2811, ACM SIGGRAPH 2812, ACM SIGGRAPH 2813, ACM SIGGRAPH 2814, ACM SIGGRAPH 2815, ACM SIGGRAPH 2816, ACM SIGGRAPH 2817, ACM SIGGRAPH 2818, ACM SIGGRAPH 2819, ACM SIGGRAPH 2820, ACM SIGGRAPH 2821, ACM SIGGRAPH 2822, ACM SIGGRAPH 2823, ACM SIGGRAPH 2824, ACM SIGGRAPH 2825, ACM SIGGRAPH 2826, ACM SIGGRAPH 2827, ACM SIGGRAPH 2828, ACM SIGGRAPH 2829, ACM SIGGRAPH 2830, ACM SIGGRAPH 2831, ACM SIGGRAPH 2832, ACM SIGGRAPH 2833, ACM SIGGRAPH 2834, ACM SIGGRAPH 2835, ACM SIGGRAPH 2836, ACM SIGGRAPH 2837, ACM SIGGRAPH 2838, ACM SIGGRAPH 2839, ACM SIGGRAPH 2840, ACM SIGGRAPH 2841, ACM SIGGRAPH 2842, ACM SIGGRAPH 2843, ACM SIGGRAPH 2844, ACM SIGGRAPH 2845, ACM SIGGRAPH 2846, ACM SIGGRAPH 2847, ACM SIGGRAPH 2848, ACM SIGGRAPH 2849, ACM SIGGRAPH 2850, ACM SIGGRAPH 2851, ACM SIGGRAPH 2852, ACM SIGGRAPH 2853, ACM SIGGRAPH 2854, ACM SIGGRAPH 2855, ACM SIGGRAPH 2856, ACM SIGGRAPH 2857, ACM SIGGRAPH 2858, ACM SIGGRAPH 2859, ACM SIGGRAPH 2860, ACM SIGGRAPH 2861, ACM SIGGRAPH 2862, ACM SIGGRAPH 2863, ACM SIGGRAPH 2864, ACM SIGGRAPH 2865, ACM SIGGRAPH 2866, ACM SIGGRAPH 2867, ACM SIGGRAPH 2868, ACM SIGGRAPH 2869, ACM SIGGRAPH 2870, ACM SIGGRAPH 2871, ACM SIGGRAPH 2872, ACM SIGGRAPH 2873, ACM SIGGRAPH 2874, ACM SIGGRAPH 2875, ACM SIGGRAPH 2876, ACM SIGGRAPH 2877, ACM SIGGRAPH 2878, ACM SIGGRAPH 2879, ACM SIGGRAPH 2880, ACM SIGGRAPH 2881, ACM SIGGRAPH 2882, ACM SIGGRAPH 2883, ACM SIGGRAPH 2884, ACM SIGGRAPH 2885, ACM SIGGRAPH 2886, ACM SIGGRAPH 2887, ACM SIGGRAPH 2888, ACM SIGGRAPH 2889, ACM SIGGRAPH 2890, ACM SIGGRAPH 2891, ACM SIGGRAPH 2892, ACM SIGGRAPH 2893, ACM SIGGRAPH 2894, ACM SIGGRAPH 2895, ACM SIGGRAPH 2896, ACM SIGGRAPH 2897, ACM SIGGRAPH 2898, ACM SIGGRAPH 2899, ACM SIGGRAPH 2900, ACM SIGGRAPH 2901, ACM SIGGRAPH 2902, ACM SIGGRAPH 2903, ACM SIGGRAPH 2904, ACM SIGGRAPH 2905, ACM SIGGRAPH 2906, ACM SIGGRAPH 2907, ACM SIGGRAPH 2908, ACM SIGGRAPH 2909, ACM SIGGRAPH 2910, ACM SIGGRAPH 2911, ACM SIGGRAPH 2912, ACM SIGGRAPH 2913, ACM SIGGRAPH 2914, ACM SIGGRAPH 2915, ACM SIGGRAPH 2916, ACM SIGGRAPH 2917, ACM SIGGRAPH 2918, ACM SIGGRAPH 2919, ACM SIGGRAPH 2920, ACM SIGGRAPH 2921, ACM SIGGRAPH 2922, ACM SIGGRAPH 2923, ACM SIGGRAPH 2924, ACM SIGGRAPH 2925, ACM SIGGRAPH 2926, ACM SIGGRAPH 2927, ACM SIGGRAPH 2928, ACM SIGGRAPH 2929, ACM SIGGRAPH 2930, ACM SIGGRAPH 2931, ACM SIGGRAPH 2932, ACM SIGGRAPH 2933, ACM SIGGRAPH 2934, ACM SIGGRAPH 2935, ACM SIGGRAPH 2936, ACM SIGGRAPH 2937, ACM SIGGRAPH 2938, ACM SIGGRAPH 2939, ACM SIGGRAPH 2940, ACM SIGGRAPH 2941, ACM SIGGRAPH 2942, ACM SIGGRAPH 2943, ACM SIGGRAPH 2944, ACM SIGGRAPH 2945, ACM SIGGRAPH 2946, ACM SIGGRAPH 2947, ACM SIGGRAPH 2948, ACM SIGGRAPH 2949, ACM SIGGRAPH 2950, ACM SIGGRAPH 2951, ACM SIGGRAPH 2952, ACM SIGGRAPH 2953, ACM SIGGRAPH 2954, ACM SIGGRAPH 2955, ACM SIGGRAPH 2956, ACM SIGGRAPH 2957, ACM SIGGRAPH 2958, ACM SIGGRAPH 2959, ACM SIGGRAPH 2960, ACM SIGGRAPH 2961, ACM SIGGRAPH 2962, ACM SIGGRAPH 2963, ACM SIGGRAPH 2964, ACM SIGGRAPH 2965, ACM SIGGRAPH 2966, ACM SIGGRAPH 2967, ACM SIGGRAPH 2968, ACM SIGGRAPH 2969, ACM SIGGRAPH 2970, ACM SIGGRAPH 2971, ACM SIGGRAPH 2972, ACM SIGGRAPH 2973, ACM SIGGRAPH 2974, ACM SIGGRAPH 2975, ACM SIGGRAPH 2976, ACM SIGGRAPH 2977, ACM SIGGRAPH 2978, ACM SIGGRAPH 2979, ACM SIGGRAPH 2980, ACM SIGGRAPH 2981, ACM SIGGRAPH 2982, ACM SIGGRAPH 2983, ACM SIGGRAPH 2984, ACM SIGGRAPH 2985, ACM SIGGRAPH 2986, ACM SIGGRAPH 2987, ACM SIGGRAPH 2988, ACM SIGGRAPH 2989, ACM SIGGRAPH 2990, ACM SIGGRAPH 2991, ACM SIGGRAPH 2992, ACM SIGGRAPH 2993, ACM SIGGRAPH 2994, ACM SIGGRAPH 2995, ACM SIGGRAPH 2996, ACM SIGGRAPH 2997, ACM SIGGRAPH 2998, ACM SIGGRAPH 2999, ACM SIGGRAPH 3000, ACM SIGGRAPH 3001, ACM SIGGRAPH 3002, ACM SIGGRAPH 3003, ACM SIGGRAPH 3004, ACM SIGGRAPH 3005, ACM SIGGRAPH 3006, ACM SIGGRAPH 3007, ACM SIGGRAPH 3008, ACM SIGGRAPH 3009, ACM SIGGRAPH 3010, ACM SIGGRAPH 3011, ACM SIGGRAPH 3012, ACM SIGGRAPH 3013, ACM SIGGRAPH 3014, ACM SIGGRAPH 3015, ACM SIGGRAPH 3016, ACM SIGGRAPH 3017, ACM SIGGRAPH 3018, ACM SIGGRAPH 3019, ACM SIGGRAPH 3020, ACM SIGGRAPH 3021, ACM SIGGRAPH 3022, ACM SIGGRAPH 3023, ACM SIGGRAPH 3024, ACM SIGGRAPH 3025, ACM SIGGRAPH 3026, ACM SIGGRAPH 3027, ACM SIGGRAPH 3028, ACM SIGGRAPH 3029, ACM SIGGRAPH 3030, ACM SIGGRAPH 3031, ACM SIGGRAPH 3032, ACM SIGGRAPH 3033, ACM SIGGRAPH 3034, ACM SIGGRAPH 3035, ACM SIGGRAPH 3036, ACM SIGGRAPH 3037, ACM SIGGRAPH 3038, ACM SIGGRAPH 3039, ACM SIGGRAPH 3040, ACM SIGGRAPH 3041, ACM SIGGRAPH 3042, ACM SIGGRAPH 3043, ACM SIGGRAPH 3044, ACM SIGGRAPH 3045, ACM SIGGRAPH 3046, ACM SIGGRAPH 3047, ACM SIGGRAPH 3048, ACM SIGGRAPH 3049, ACM SIGGRAPH 3050, ACM SIGGRAPH 3051, ACM SIGGRAPH 3052, ACM SIGGRAPH 3053, ACM SIGGRAPH 3054, ACM SIGGRAPH 3055, ACM SIGGRAPH 3056, ACM SIGGRAPH 3057, ACM SIGGRAPH 3058, ACM SIGGRAPH 3059, ACM SIGGRAPH 3060, ACM SIGGRAPH 3061, ACM SIGGRAPH 3062, ACM SIGGRAPH 3063, ACM SIGGRAPH 3064, ACM SIGGRAPH 3065, ACM SIGGRAPH 3066, ACM SIGGRAPH 3067, ACM SIGGRAPH 3068, ACM SIGGRAPH 3069, ACM SIGGRAPH 3070, ACM SIGGRAPH 3071, ACM SIGGRAPH 3072, ACM SIGGRAPH 3073, ACM SIGGRAPH 3074, ACM SIGGRAPH 3075, ACM SIGGRAPH 3076, ACM SIGGRAPH 3077, ACM SIGGRAPH 3078, ACM SIGGRAPH 3079, ACM SIGGRAPH 3080, ACM SIGGRAPH 3081, ACM SIGGRAPH 3082, ACM SIGGRAPH 3083, ACM SIGGRAPH 3084, ACM SIGGRAPH 3085, ACM SIGGRAPH 3086, ACM SIGGRAPH 3087, ACM SIGGRAPH 3088, ACM SIGGRAPH 3089, ACM SIGGRAPH 3090, ACM SIGGRAPH 3091, ACM SIGGRAPH 3092, ACM SIGGRAPH 3093, ACM SIGGRAPH 3094, ACM SIGGRAPH 3095, ACM SIGGRAPH 3096, ACM SIGGRAPH 3097, ACM SIGGRAPH 3098, ACM SIGGRAPH 3099, ACM SIGGRAPH 3100, ACM SIGGRAPH 3101, ACM SIGGRAPH 3102, ACM SIGGRAPH 3103, ACM SIGGRAPH 3104, ACM SIGGRAPH 3105, ACM SIGGRAPH 3106, ACM SIGGRAPH 3107, ACM SIGGRAPH 3108, ACM SIGGRAPH 3109, ACM SIGGRAPH 3110, ACM SIGGRAPH 3111, ACM SIGGRAPH 3112, ACM SIGGRAPH 3113, ACM SIGGRAPH 3114, ACM SIGGRAPH 3115, ACM SIGGRAPH 3116, ACM SIGGRAPH 3117, ACM SIGGRAPH 3118, ACM SIGGRAPH 3119, ACM SIGGRAPH 3120, ACM SIGGRAPH 3121, ACM SIGGRAPH 3122, ACM SIGGRAPH 312</p>
---	---	--	---

[Fernández-Baena and Miralles, 2014] Fernández-Baena, A., Montañó, R. M., Antonijoan, M., Roversi, A., Miralles, D., and Alias, F. (2014). Gesture synthesis adapted to speech emphasis. *Speech Communication*, 57(0):331–350.

Abstract: Avatars communicate through speech and gestures to appear realistic and to enhance interaction with humans. In this context, several works have analyzed the relationship between speech and gestures, while others have been focused on their synthesis, following different approaches. In this work, we address both goals by linking speech to gestures in terms of time and intensity, to then use this knowledge to drive a gesture synthesizer from a manually annotated speech signal. To that effect, we define strength indicators for speech and motion. After validating them through perceptual tests, we obtain an intensity rule from their correlation. Moreover, we derive a synchrony rule to determine temporal correspondences between speech and gestures. These analyses have been conducted on aggressive and neutral performances to cover a broad range of emphatic levels, whose speech signal and motion have been manually annotated. Next, intensity and synchrony rules are used to drive a gesture synthesizer called gesture motion graph (GMG). These rules are validated by users from GMG output animations through perceptual tests. Results show that animations using intensity and synchrony rules perform better than those only using the synchrony rule (which in turn enhance realism with respect to random animation). Finally, we conclude that the extracted rules allow GMG to properly synthesize gestures adapted to speech emphasis from annotated speech.

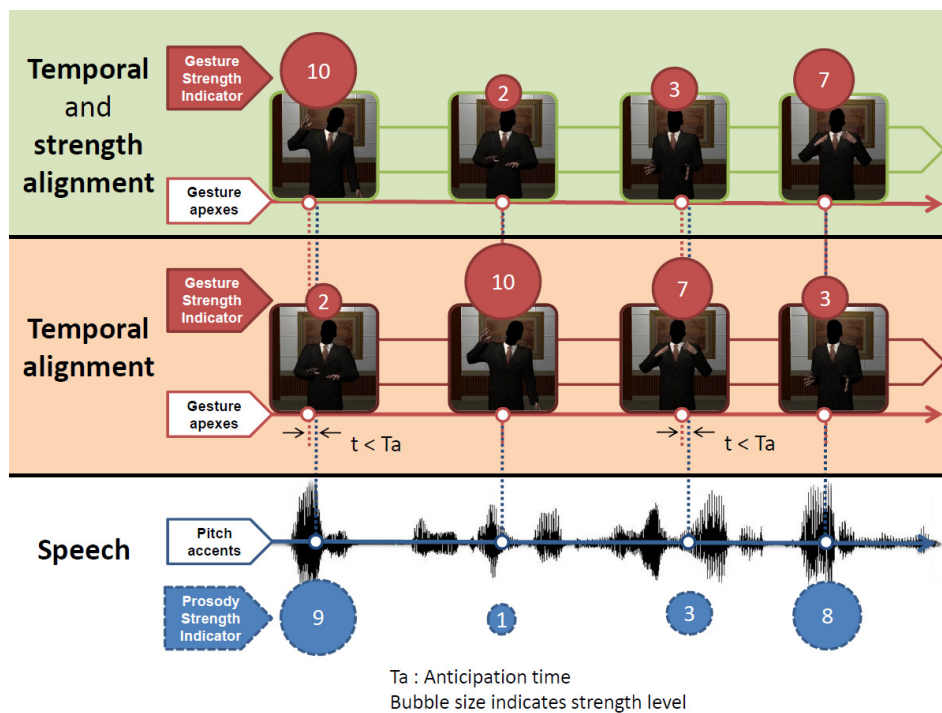



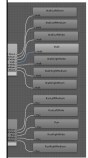

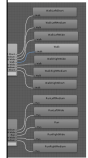

Figure 7.1: Graphical abstract.





[Fernández-Baena and Miralles, 2014] Fernández-Baena, A. and Miralles, D. (2014). Avatar: Tangible interaction and augmented reality in character animation. In *IUI 2014 Workshop on Interacting with Smart Objects*. Haifa, Israel.

**Abstract:** In this paper, we present a novel interaction system, which combines tangible interaction and augmented reality for controlling a virtual avatar. By physically interacting with a cube, it is possible to drive avatars motion that occurs in the real world. The cube acts as a motion controller and as an AR marker reaching input and rendering purposes. The cube facilitates users the avatar positioning and motion customization, providing a fine control for both. In this first version, the avatar is able to stand and move. The current motion state is picked rotating the cube over the same plane where the avatar lies. We have implemented two scenarios in our prototype: a sketch-based controller and an interactive controller. The first one enables users to draw paths on the floor that the avatars follow; on the contrary, the second allows drive avatars position during all the time. The idea of using tangible objects in augmented reality environments for controlling avatars strengthens the link between the user and the avatar providing a better sense of control and immersion.

<p><b>Avatar: Tangible interaction and augmented reality in character animation</b></p> <p><b>Acho Fernández-Baena</b> Seamless Interaction, GTM La Salle RD, La Salle - URL 08022 Barcelona, Spain acho@salleurl.edu</p> <p><b>David Miralles</b> Seamless Interaction, GTM La Salle RD, La Salle - URL 08022 Barcelona, Spain davime@saleurl.edu</p> <p><b>ABSTRACT</b> In this paper, we present a novel interaction system, which combines tangible interaction and augmented reality for controlling a virtual avatar. By physically interacting with a cube, it is possible to drive avatars motion that occurs in the real world. The cube acts as a motion controller and as an AR marker reaching input and rendering purposes. The cube facilitates users the avatar positioning and motion customization, providing a fine control for both. In this first version, the avatar is able to stand and move. The current motion state is picked rotating the cube over the same plane where the avatar lies. We have implemented two scenarios in our prototype: a sketch-based controller and an interactive controller. The first one enables users to draw paths on the floor that the avatars follow; on the contrary, the second allows drive avatars position during all the time. The idea of using tangible objects in augmented reality environments for controlling avatars strengthens the link between the user and the avatar providing a better sense of control and immersion.</p> <p><b>Author Keywords</b> tangible interaction; augmented reality; character animation</p> <p><b>ACM Classification Keywords</b> H.5.2. Information Interfaces and Presentation (e.g. HCI); Input devices and strategies</p> <p><b>General Terms</b> Human Factors; Design.</p> <p><b>INTRODUCTION</b> User interfaces have been evolved a great deal from command line interfaces, through graphical interfaces to last generation. This last generation brings together tangible and natural interfaces. Tangible user interfaces (TUI) [3] permit to interact with digital content through a physical environment. TUI allows to design interactions using everyday objects, breaking the traditional approach of interacting with computers and giving a strong sense of immediacy to the user.</p> <p><small>Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted by ACM, provided that the fee code for users to reproduce and distribute reprints for non-commercial purposes and that the fee code for users to reproduce and distribute reprints for non-commercial purposes is paid to the copyright owner. Copyright © 2014, ACM. Workshop on Interacting with Smart Objects, February 24, 2014, Haifa, Israel. Copyright is held by the author(s).</small></p>	 <p>formed by short, medium and long term motion and wide parameter is also used to form final trees, where its to Walk (see next).</p> <p>ing the cube. Nevertheless buttons in order to design To get started, we design</p>  <p>paths where the virtual propose two steps. First, ion of the cube across the the path creation, we draw it. Once we have the path to eight points, the virtual body. While it is moving, by rotating the cube. In this low the avatar can covered at position. Moreover, the by by assigning side motion to a static position.</p>  <p>image targets in its faces, so that denotes one of the walking and running. The get by the cube faces. How precise, we have to ensure the in this manner, we improve in Figure 6 is illustrated</p> <p>of the target cube</p>	<p>formed by short, medium and long term motion and wide parameter is also used to form final trees, where its to Walk (see next).</p> <p>ing the cube. Nevertheless buttons in order to design To get started, we design</p>  <p>paths where the virtual propose two steps. First, ion of the cube across the the path creation, we draw it. Once we have the path to eight points, the virtual body. While it is moving, by rotating the cube. In this low the avatar can covered at position. Moreover, the by by assigning side motion to a static position.</p>  <p>image targets in its faces, so that denotes one of the walking and running. The get by the cube faces. How precise, we have to ensure the in this manner, we improve in Figure 6 is illustrated</p> <p>of the target cube</p>
--	---	---

This publication is accompanied by a video that can be found at <https://goo.gl/BD5Sgj>.







## Chapter 8

# Appendix

### Chapter Abstract

In these appendices we expose some extra information which is out of the scope of this work in order to understand deeply some parts. Thus, we relate the speech analysis carried out that feeds Section 4.4.3 and the automatic pitch accents detector implemented to drive body gesture synthesis (see Section 4.5.2).



## 8.1 Speech analysis

In this section we detail the analysis of speech carried out in order to later determine synchrony and intensity rules (see Section 4.4.3). This analysis is performed entirely for Spanish speech data, so it can only be validated for this language. Other studies should be carried out in the future in order to generalize our analysis to other languages. Moreover, it is worth noting that our rules are restricted to our corpus (details in Section 4.4.2), i.e., they are not general rules applicable to any voice (at least this has not been tested). Our goal was not to search for a general rule but to look for correlations between gesture and speech and test if an intensity rule together with a synchrony rule improves the naturalness of synthetic animations.

### 8.1.1 Intonation in Spanish

Previously to explain the speech analysis, we lay some foundations about intonation in Spanish, as it is the language of our speech cues. Pierrehumbert introduced the Autosegmental and Metrical model (Autosegmental and Metrical model (AM)) in her PhD thesis [Pierrehumbert, 1980]. It was used to analyze intonation in English. The objective of AM is the identification of contrastive elements of the intonation system. The combination of these elements produces the melodic contours found in language. In AM theory, the melody or tonal modulation is treated as a separate phenomena with respect to other phonological features. Tones are treated as autosegments, which relate to text following a set of rules that may vary from language to language.

The AM theory says that the intonation contour corresponds to the interpolation between tonal events associated to certain syllables. In Spanish, these tonal events can be associated to lexical accents (pitch accents), or to the end of certain sentences (boundary tones).

Stressed syllables are characterized by having a pitch prominence, relative longer duration, and relative higher intensity than other syllables in the same word. However, this prominence is not always associated with a high pitch. In Spanish, almost all words have a stressed syllable. Only functional words such as prepositions, determinants and alike are rarely accentuated. Although, not all stressed syllables in Spanish have a pitch accent. This mechanism has a pragmatic function, which allows speakers to highlight some words over others. This unstressing technique is also used in English.

The types of pitch accents are defined as:

- H\*: Pitch peak on the stressed syllable
- L\*: Pitch valley on the stressed syllable
- L+H\*: Pitch peak on the stressed syllable preceded by a valley
- L\*+H: Pitch valley on the stressed syllable followed by a peak
- H+L\*: Pitch valley on the stressed syllable preceded by a peak
- H\*+L: Pitch peak on the stressed syllable followed by a valley

Not all languages use all these types of pitch accents. For example, according to Beckman et al. [Beckman et al., 2002], in Spanish only L\*+H, L+H\*, H+L\*, H\* are present. Moreover, L\* may be used in interrogative sentences. Nevertheless, other works added more tags for Spanish [Vilaplana and Prieto, 2009] like the L+>H\* tag, also used in later works [Escudero-Mancebo and Aguilar, 2010].

## 8.1.2 Speech analysis

### Speech corpus annotation

We used the well-known speech analysis software Praat [Boersma and Weenink, 2011] for the annotation of the Spanish speech corpus at hand. First, the speech corpus was hierarchically segmented into sentences, words, syllables and phonemes. In order to do that, we used the EasyAlign tool [Goldman, 2011], which worked remarkably well, although some manual corrections were needed.

In order to annotate the intonation, we used the Tones and Break Indices (ToBI) (Tones and Break Indices) system, as it describes pitch accents, besides being a commonly used tagging system [Kalinli and Narayanan, 2009] [Loehr, 2004]. ToBI was originally designed to describe standard American English intonation [Silverman et al., 1992b]. However, a “Spanish ToBI” has already been used in some previous works [Escudero-Mancebo and Aguilar, 2010] [Ortiz-Lira, 1999] [Vilaplana and Prieto, 2009]. However, work remains to be done in order to achieve a complete consensus about how to adapt ToBI tags to Spanish [Beckman et al., 2002].

As this annotation may entail between 100 or 200 times the real duration of the video to analyze, some works in the literature have tried to automate (or semi-automate) the annotation process [Syrdal et al., 2001]. However, we opted for a manual tagging since it obtains reliable results while being aware of some delicate situations (e.g. pitch - doubling / halving) which, if not well processed, could distort the analysis.

Finally, two more annotation tiers were manually added to Praat. One of them is an annotation of the syllable nucleus inside the pitch accent. The other consists of a point tier which indicates the time where the pitch achieves its maximum value inside the syllable as usually the peak is taken as a point of reference [Leonard and Cummins, 2011] [Loehr, 2004]. We define this point as the pitch accent peak time (PAPT). This labeling point is used to analyze the correlation between gestures and speech (distance between apex time and PAPT). As an example, the complete annotation of a speech fragment is depicted in Figure 8.1.

### Classification of pitch accent strength

In order to measure prosodic prominence, we considered two measures [Silipo and Greenberg, 1999] [Abete et al., 2010] that could be appropriate for reaching our goal. In [Silipo and Greenberg, 1999], the authors proposed an ‘evidence variable’ (Evidence Variable (EV)) for marking prosodic stress computed with the multiplication of mean energy and duration of the syllable nucleus. More recent work has added to the ‘evidence variable’ a new parameter (called ‘m’) which described the contribution of the pitch range [Abete et al., 2010]. This variable will be referred from now on as Evidence Variable 2 (EV2). This parameter rewards situations where pitch variation inside the nucleus has a rising pattern. In our experience, situations with a decreasing pitch pattern were perceptually weak, so these cases (labeled with H+L\* ToBI tags) have been penalized in the same way as in [Abete et al., 2010], which consists in forcing these cases to the inverse maximum pitch range value.

The major acoustic correlates of prosodic prominence reported in the literature are pitch movements, global energy and duration [Streefkerk et al., 1999], although some works obtained good results leaving out the pitch [Silipo and Greenberg, 1999] or when adding spectral emphasis [Tamburini and Wagner, 2007]. In this work, we have omitted spectral emphasis as it does not carry

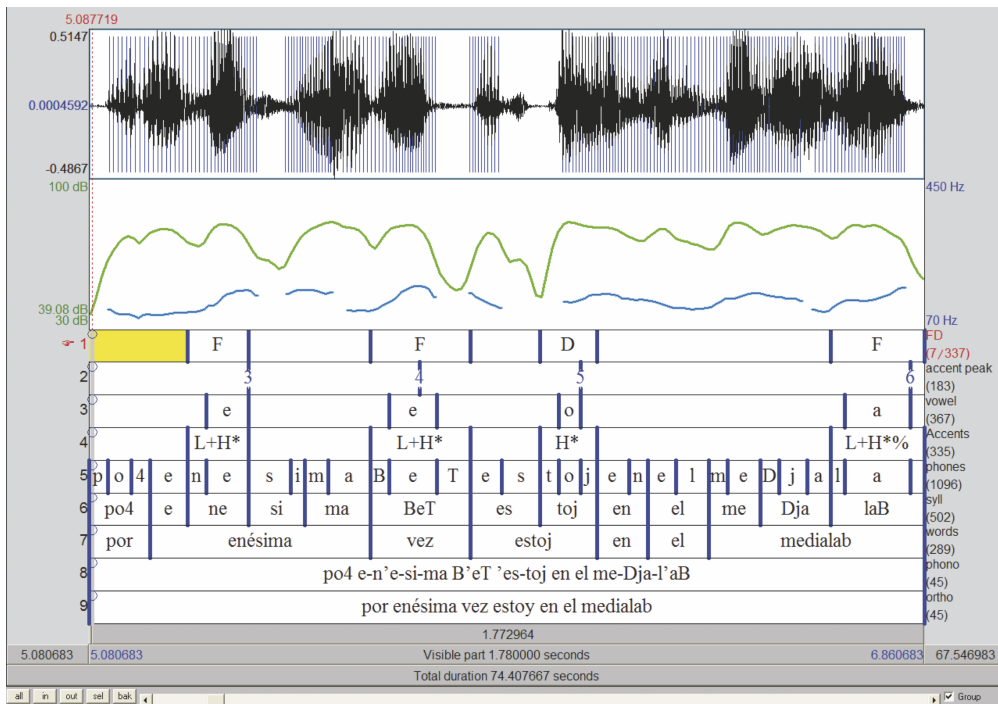


Figure 8.1: Example of the entire annotation of a particular audio fragment. The FD annotation tier contains the tags from the perceptual test. Sentence translation: "For the umpteenth time I am at the Medialab."

information about stress in Spanish [Ortega-Llebaria and Prieto, 2011].

The perceptual test to evaluate how well these variables can discriminate between weak and strong pitch accents was carried out by two experts on audio annotation separately. They were asked to annotate each pitch accent as weak or strong, according to their perception. Later, a third expert annotator made the final decision in the case of disagreement. The annotators based their classification on context, as it is important for the perception of prominence [Tamburini and Wagner, 2007].

Once the annotations were finished, we were able to check if the evidence variables (EV and EV2) were well suited to discriminate between weak and strong pitch accents. It is worth pointing out that our objective in this work is not a classification task by itself, but these classifications will help us to identify, if possible, what was perceived acoustically. MCC results for the variables can be observed in Table 8.1. The conclusion is that EV2 performs considerably better than the original EV.

In spite of this test, we validated the numeric values that did not fit its correspondent strength tag. We noticed that, in some situations when the speaker spoke loudly, the mean pitch was very high but the pitch contour was flat (pitch range close to zero and, therefore, EV2 close to zero). So, with the aim of improving the obtained results, we included the mean pitch in the pitch term (see Equation 8.1) in the EV2 equation [Abete et al., 2010]. We assigned these weights to the pitch parameters to maintain the variable between 0 and 1 and we defined this new variable as Prosody

Variable	MCC
PSI	0.671
EV2 [Abete et al., 2010]	0.630
EV [Silipo and Greenberg, 1999]	0.575

Table 8.1: MCC results for all variables.

Classifier	MCC
SPegasos	0.662
Logistic regression	0.652
SMO	0.640
Fisher discriminant	0.636

Table 8.2: MCC results for all the pitch accent classifications.

Strength Indicator (PSI):

$$PSI = D \cdot E \cdot (0.5PR + 0.5MP) \quad (8.1)$$

where  $D$  is duration of the syllable nucleus,  $E$  is mean energy of the syllable nucleus,  $PR$  is pitch range of the syllable nucleus (with the applied penalty for H+L\* pitch accents as explained before), and  $MP$  is mean pitch of the syllable nucleus. As can be observed in Table 8.1, the MCC value of PSI is higher than the one obtained by EV2. We can conclude that the use of mean pitch improves the discrimination between weak and strong pitch accents as perceived in the perceptual test.

Finally, we validated the capability of PSI to distinguish between weak and strong pitch accents using the same linear classification methods as in Section 4.4.2. None of these methods outperformed the MCC achieved by PSI (see Tables 8.1 and 8.2). Therefore, we selected PSI for the rest of the analysis.

Another interesting conclusion after finishing all the gestures and speech analysis is that it is easier to classify speech than gestures (according to their respective perceived strengths), as the MCC value of the speech classification is considerably higher. Finally, we also conclude that quantifying speech prominence is not an easy task as perceived and computed differences exist. This has also been stated for other languages like German [Tamburini and Wagner, 2007].

## 8.2 Automatic pitch accents detection

Regarding pitch accent detection, we have developed a straightforward algorithm inspired by [Maeran et al., 1997]. By pitch accent detection we mean detection of prominences in the speech stream. These prominences are potential candidates to be synchronized with gestures.

Taking a speech file as an input, we extract all the signal cycles with their associated information (amplitude, position, etc.). After selecting principal cycles, we extract voiced and unvoiced regions.



Then, we extract and normalize pitch and intensity from voiced region nucleus (defined as maximum energy cycle inside the region) and compute the strength indicator as a sum of both parameters (see Figure 8.2). Finally, we have also detected pauses and we have rewarded voiced regions preceding a pause with extra strength indicator, as we observed that prosody tends to decrease in these situations causing undetected pitch accents.

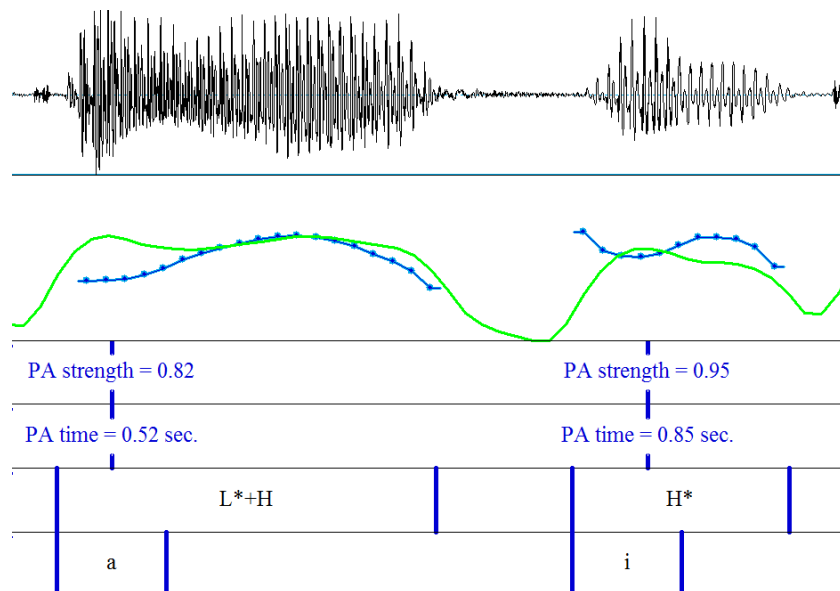


Figure 8.2: Pitch accents detection. On top, there is the speech signal. Below, intensity (green) and pitch (blue) curves are displayed. In addition, PA strength and time are shown. At the bottom, the intonation is represented (using the ToBI system [Silverman et al., 1992a]) with the affected vowel. The image was created thanks to the Praat software [Boersma and Weenink, 2011].

Final pitch accents are detected according to the extracted strength indicators of the voiced region nucleus. Specifically, they are chosen depending on two tunable constraint parameters: strength indicator threshold and time difference threshold. Basically, the strength indicator threshold represents what percentage of the nucleus are pitch accents candidates (taking as a reference maximum strength indicator), and the time difference threshold defines how close pitch accents can be. If two pitch accent candidates are too close according to this parameter, we keep the one with the greater strength indicator. Finally, the pitch accents strength indicator is expressed in a  $[0,1]$  scale, taking as 1 the maximum strength indicator of the serie.

Furthermore, speech is analyzed in order to extract the phoneme transcription. So, we obtain a sequence of phonemes with its type definition and timestamps. For each phoneme, initial time and final time are detected.



# Bibliography

- [Abbott et al., 1999] Abbott, P. B. I., Redpath, S. D., Llection, D. B., and Wood, D. R. (1999). User interaction with intelligent virtual objects, avatars, which interact with other avatars controlled by different users. US Patent 5,884,029.
- [Abete et al., 2010] Abete, G., Cutugno, F., Ludusan, B., and Origlia, A. (2010). Pitch behavior detection for automatic prominence recognition. *Proceedings of Speech Prosody*.
- [Albó-Canals et al., 2015] Albó-Canals, J., Fernández-Baena, A., Boldú, R., Barco, A., Navarro, J., Miralles, D., Raya, C., and Angulo, C. (2015). Enhancing long-term children to robot interaction engagement through cloud connectivity. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts, HRI'15 Extended Abstracts*, pages 105–106, New York, NY, USA. ACM.
- [Aldebaran, 2014] Aldebaran (2014). Nao robot: Intelligent and friendly companion. <http://www.aldebaran.com/en/humanoid-robot/nao-robot>.
- [Amores et al., 2014] Amores, J., Benavides, X., Comín, M., Fusté, A., Pla, P., and Miralles, D. (2014). Smart avatars. In *IUI 2014 Workshop on Interacting with Smart Objects*, Haifa, Israel.
- [Angulo et al., 2012] Angulo, C., Garriga-Berga, C., Luaces, C., Pérez-Payarols, J., Albó-Canals, J., and Díaz, M. (2012). Pain and anxiety treatment based on social robot interaction with children to improve patient experience. ongoing research. *JARCA 2012*.
- [Antle, 2007] Antle, A. (2007). Tangibles: five properties to consider for children. In *Workshop on Tangibles at Conference on Human Factors in Computing Systems (CHI 2007)*, pages 1–10.
- [Antonijoan, 2012] Antonijoan, M. (2012). Gesture-prosody correlations analysis. Master's thesis, La Salle Campus Barcelona - Universitat Ramon Llull, Barcelona, Catalonia, Spain.
- [Arikan and Forsyth, 2002] Arikan, O. and Forsyth, D. A. (2002). Interactive motion generation from examples. *ACM Trans. Graph.*, 21:483–490.
- [Arikan and Ikemoto, 2006] Arikan, O. and Ikemoto, L. (2006). *Computational Studies of Human Motion: Tracking and Motion Synthesis*. Now Publishers Inc.
- [Atari, 2015] Atari (2015). Ghostbusters: Sanctum of Slime. <https://www.atari.com/buy-games/action-adventure/ghostbusters-sanctum-slime>.
- [Atkinson et al., 2004] Atkinson, A. P., Dittrich, W. H., Gemmell, A. J., Young, A. W., et al. (2004). Emotion perception from dynamic and static body expressions in point-light and full-light displays. *Perception-London*, 33(6):717–746.
- [Atkinson et al., 2007] Atkinson, A. P., Tunstall, M. L., and Dittrich, W. H. (2007). Evidence for distinct contributions of form and motion information to the recognition of emotions from body gestures. *Cognition*, 104(1):59–72.

- [Autodesk, 2015a] Autodesk (2015a). 3D Studio Max. <http://www.autodesk.es/products/3ds-max/overview>.
- [Autodesk, 2015b] Autodesk (2015b). Maya. <http://www.autodesk.es/products/maya/overview>.
- [Autodesk, 2015c] Autodesk (2015c). Motionbuilder. <http://www.autodesk.es/products/motionbuilder/overview>.
- [Bainbridge et al., 2011] Bainbridge, W. A., Hart, J. W., Kim, E. S., and Scassellati, B. (2011). The benefits of interactions with physically present robots over video-displayed agents. *International Journal of Social Robotics*, 3(1):41–52.
- [Barakonyi et al., 2004] Barakonyi, I., Psik, T., and Schmalstieg, D. (2004). Agents that talk and hit back: animated agents in augmented reality. In *Mixed and Augmented Reality, 2004. ISMAR 2004. Third IEEE and ACM International Symposium on*, pages 141–150. IEEE.
- [Barakonyi et al., 2005] Barakonyi, I., Weilguny, M., Psik, T., and Schmalstieg, D. (2005). Monkeybridge: autonomous agents in augmented reality games. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 172–175. ACM.
- [Bates, 1994] Bates, J. (1994). The role of emotion in believable agents. *Communications of the ACM*, 37(7):122–125.
- [Beaudoin et al., 2008] Beaudoin, P., Coros, S., van de Panne, M., and Poulin, P. (2008). Motion-motif graphs. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, pages 117–126, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.
- [Beckman et al., 2002] Beckman, M., Díaz-Campos, M., McGory, J. T., and Morgan, T. A. (2002). Intonation across spanish in the tones and break indices framework. *Probus*, 14:9–36.
- [behance, 2015] behance (2015). Mimicking Human Body Movement and Facial Expressions. <https://www.behance.net/gallery/9082913/Mimicking-Human-Body-Movement-and-Facial-Expressions>.
- [Bimber and Raskar, 2006] Bimber, O. and Raskar, R. (2006). Modern approaches to augmented reality. In *ACM SIGGRAPH 2006 Courses*, page 1. ACM.
- [Blackler et al., 2005] Blackler, A. L., Popovic, V., and Mahar, D. P. (2005). Intuitive interaction applied to interface design.
- [Blake and Shiffrar, 2007] Blake, R. and Shiffrar, M. (2007). Perception of human motion. *Annu. Rev. Psychol.*, 58:47–73.
- [Bodenheimer et al., 1997] Bodenheimer, B., Rose, C., Rosenthal, S., and Pella, J. (1997). *The process of motion capture: Dealing with the data*. Springer.
- [Boersma and Weenink, 2011] Boersma, P. and Weenink, D. (2011). Praat: doing phonetics by computer [computer program]. (v.5.2.29). retrieved 12 July 2011 from <http://www.praat.org/>.
- [Bolinger, 1986] Bolinger, D. L. M. (1986). *Intonation and Its Parts: Melody in Spoken English*. Stanford University Press, Stanford, CA.
- [Breazeal, 2004] Breazeal, C. L. (2004). *Designing sociable robots*. MIT press.

- [Brenton et al., 2005] Brenton, H., Gillies, M., Ballin, D., and Chatting, D. (2005). The uncanny valley: does it exist. In *Proceedings of conference of human computer interaction, workshop on human animated character interaction*. Citeseer.
- [Bridson et al., 2002] Bridson, R., Fedkiw, R., and Anderson, J. (2002). Robust treatment of collisions, contact and friction for cloth animation. In *ACM Transactions on Graphics (ToG)*, volume 21, pages 594–603. ACM.
- [Bruderlin and Williams, 1995] Bruderlin, A. and Williams, L. (1995). Motion signal processing. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, SIGGRAPH '95*, pages 97–104, New York, NY, USA. ACM.
- [Bryson, 2005] Bryson, S. (2005). An introduction to animation and motion blending. Technical Report 98082365, University of Technology, Sidney.
- [Bulut et al., 2007] Bulut, M., Lee, S., and Narayanan, S. (2007). A statistical approach for modeling prosody features using pos tags for emotional speech synthesis. In *ICASSP 2007*, volume 4, pages 1237–1240.
- [Busso et al., 2005] Busso, C., Deng, Z., Neumann, U., and Narayanan, S. (2005). Natural head motion synthesis driven by acoustic prosodic features: Virtual humans and social agents. *Comput. Animat. Virtual Worlds*, 16(3-4):283–290.
- [Carnegie Mellon University Graphics Lab, 2004] Carnegie Mellon University Graphics Lab (2004). CMU Graphics Lab Motion Capture Database.
- [Cassell et al., 1994] Cassell, J., Pelachaud, C., Badler, N., Steedman, M., Achorn, B., Becket, T., Douville, B., Prevost, S., and Stone, M. (1994). Animated conversation: rule-based generation of facial expression, gesture & spoken intonation for multiple conversational agents. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques, SIGGRAPH '94*, pages 413–420, New York, NY, USA. ACM.
- [Cassell et al., 2000] Cassell, J., Sullivan, J., Prevost, S., and Churchill, E. F. (2000). *Embodied conversational agents*. MIT press.
- [Cassell et al., 2001] Cassell, J., Vilhjálmsón, H. H., and Bickmore, T. (2001). BEAT: The behavior expression animation toolkit. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '01*, pages 477–486, New York, NY, USA. ACM.
- [Cavazza et al., 2002] Cavazza, M., Charles, F., and Mead, S. J. (2002). Planning characters' behaviour in interactive storytelling. *The Journal of Visualization and Computer Animation*, 13(2):121–131.
- [Cavazza et al., 2004] Cavazza, M., Charles, F., Mead, S. J., Martin, O., Marichal, X., and Nandi, A. (2004). Multimodal acting in mixed reality interactive storytelling. *MultiMedia, IEEE*, 11(3):30–39.
- [Chaturvedi et al., 2011] Chaturvedi, A. R., Dolk, D. R., and Drnevich, P. L. (2011). Design principles for virtual worlds. *MIS Quarterly*, 35(3):673–684.
- [Chiu and Marsella, 2011] Chiu, C.-C. and Marsella, S. (2011). How to train your avatar: a data driven approach to gesture generation. In *Proceedings of the 10th international conference on Intelligent virtual agents, IVA'11*, pages 127–140, Berlin, Heidelberg. Springer-Verlag.
- [Chiu and Marsella, 2014] Chiu, C.-C. and Marsella, S. (2014). Gesture generation with low-dimensional embeddings. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 781–788. International Foundation for Autonomous Agents and Multiagent Systems.

- [Condon, 1976] Condon, W. S. (1976). An analysis of behavioral organization. *Sign Language Studies*, (13):285–318.
- [Curtis et al., 2011] Curtis, A., Shim, J., Gargas, E., Srinivasan, A., and Howard, A. M. (2011). Dance dance Pleo: developing a low-cost learning robotic dance therapy aid. In *Proceedings of the 10th International Conference on Interaction Design and Children*, pages 149–152. ACM.
- [Cvejic et al., 2010] Cvejic, E., Kim, J., and Davis, C. (2010). It's all the same to me: Prosodic discrimination across speakers and face areas. In *Speech Prosody 2010-Fifth International Conference*.
- [de Jong et al., 1993] de Jong, K., Beckman, M., and Edwards, J. (1993). The interplay between prosodic structure and coarticulation. *Lang Speech*, 36 ( Pt 2-3).
- [Díaz Boladeras et al., 2011] Díaz Boladeras, M., Nuño Bermudez, N., Sàez Pons, J., Pardo Ayala, D. E., Angulo Bahón, C., and Andrés, A. (2011). Building up child-robot relationship: from initial attraction towards long-term social engagement.
- [Dimas et al., 2010] Dimas, J., Leite, I., Pereira, A., Cuba, P., Prada, R., and Paiva, A. (2010). Pervasive Pleo: long-term attachment with artificial pets. In *Mobile HCI*.
- [Dreamworks Animation Llc., 2015] Dreamworks Animation Llc. (2015). Dreamworks Studios. IMDb: <http://www.dreamworksstudios.com/>.
- [Dreamworks Studios, 2001] Dreamworks Studios (2001). Shrek. IMDb: [http://www.imdb.com/title/tt0126029/?ref\\_=nv\\_sr\\_1](http://www.imdb.com/title/tt0126029/?ref_=nv_sr_1).
- [Dreamworks Studios, 2004] Dreamworks Studios (2004). Shrek 2. IMDb: [http://www.imdb.com/title/tt0126029/?ref\\_=nv\\_sr\\_2](http://www.imdb.com/title/tt0126029/?ref_=nv_sr_2).
- [Dreamworks Studios, 2007] Dreamworks Studios (2007). Shrek the Third. IMDb: [http://www.imdb.com/title/tt0126029/?ref\\_=nv\\_sr\\_3](http://www.imdb.com/title/tt0126029/?ref_=nv_sr_3).
- [Dreamworks Studios, 2008] Dreamworks Studios (2008). Kung Fu Panda. IMDb: [http://www.imdb.com/title/tt1302011/?ref\\_=nv\\_sr\\_1](http://www.imdb.com/title/tt1302011/?ref_=nv_sr_1).
- [Dreamworks Studios, 2011] Dreamworks Studios (2011). Kung Fu Panda 2. IMDb: [http://www.imdb.com/title/tt1302011/?ref\\_=nv\\_sr\\_2](http://www.imdb.com/title/tt1302011/?ref_=nv_sr_2).
- [Duda et al., 2012] Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification*. John Wiley & Sons.
- [EA, 2015] EA (2015). The Sims. <http://www.thesims.com/>.
- [Escudero-Mancebo and Aguilar, 2010] Escudero-Mancebo, D. and Aguilar, L. (2010). Procedure for assessing the reliability of prosodic judgements using Sp-TOBI labelling system. In *Proceedings of International Conference on Speech Prosody*.
- [Etemad et al., 2015] Etemad, S. A., Arya, A., Parush, A., and DiPaola, S. (2015). Perceptual validity in animation of human motion. *Computer Animation and Virtual Worlds*.
- [Feng et al., 2012] Feng, A. W., Huang, Y., Kallmann, M., and Shapiro, A. (2012). An analysis of motion blending techniques. In Kallmann, M. and Bekris, K. E., editors, *MIG*, volume 7660 of *Lecture Notes in Computer Science*, pages 232–243. Springer.
- [Fernaesus et al., 2010] Fernaeus, Y., Håkansson, M., Jacobsson, M., and Ljungblad, S. (2010). How do you play with a robotic toy animal?: a long-term study of Pleo. In *Proceedings of the 9th international Conference on interaction Design and Children*, pages 39–48. ACM.

- [Fernández-Baena et al., 2013] Fernández-Baena, A., Antonijoan, M., Montaña, R., Fusté, A., and Amores, J. (2013). Bodyspeech: A configurable gesture and facial animation system for speaking avatars. *Computer Graphics and Virtual Reality 2013 (CGVR -2013), Las Vegas, 2013*.
- [Fernández-Baena et al., 2015] Fernández-Baena, A., Boldú, R., Albó-Canals, J., and Miralles, D. (2015). Interaction between Vleo and Pleo, a virtual social character and a social robot. In *RO-MAN (to be issued), 2015 IEEE*. IEEE.
- [Fernández-Baena and Miralles, 2011] Fernández-Baena, A. and Miralles, D. (2011). Progressive transitions using body part motion graphs. In *SIGGRAPH Asia 2011 Posters, SA '11*, pages 3:1–3:2, New York, NY, USA. ACM.
- [Fernández-Baena and Miralles, 2014] Fernández-Baena, A. and Miralles, D. (2014). AvatAR: Tangible interaction and augmented reality in character animation. In *IUI 2014 Workshop on Interacting with Smart Objects*.
- [Fernández-Baena et al., 2014] Fernández-Baena, A., Montaña, R., Antonijoan, M., Roversi, A., Miralles, D., and Alías, F. (2014). Gesture synthesis adapted to speech emphasis. *Speech Communication*, 57:331 – 350.
- [Fernández-Baena and Miralles, 2012] Fernández-Baena, A. and Miralles, D. (2012). Fast response and quick progressive transitions using body part motion graphs. pages 77–80, Cagliari, Sardinia, Italy. Eurographics Association.
- [fxguide, 2015] fxguide (2015). Xbox One: Developers ready — fxguide. <http://www.fxguide.com/featured/xbox-one-developers-ready/>.
- [Gamer Limit, 2015] Gamer Limit (2015). Gamer Limit Review: Assassin's Creed PC: Director's Cut Edition. <http://gamerlimit.com/2009/07/gamer-limit-review-assassins-creed-pc/>.
- [Geijtenbeek and Pronost, 2012] Geijtenbeek, T. and Pronost, N. (2012). Interactive character animation using simulated physics: A state-of-the-art review. In *Computer Graphics Forum*, volume 31, pages 2492–2515. Wiley Online Library.
- [Geller, 2008] Geller, T. (2008). Overcoming the uncanny valley. *IEEE Computer Graphics and Applications*, 28(4):11–17.
- [Gibbons, 1985] Gibbons, A. (1985). *Algorithmic graph theory*. Cambridge University Press.
- [Goldman, 2011] Goldman, J.-P. (2011). Easyalign: an automatic phonetic alignment tool under praat. *INTERSPEECH-2011*, pages 3233–3236.
- [Gomes et al., 2011] Gomes, P. F., Segura, E. M., Cramer, H., Paiva, T., Paiva, A., and Holmquist, L. E. (2011). ViPleo and PhyPleo: Artificial pet with two embodiments. In *Proceedings of the 8th International Conference on Advances in Computer Entertainment Technology*, page 3. ACM.
- [Grassia, 1998] Grassia, F. S. (1998). Practical parameterization of rotations using the exponential map. *J. Graph. Tools*, 3:29–48.
- [Guo et al., 2015] Guo, S., Southern, R., Chang, J., Greer, D., and Zhang, J. J. (2015). Adaptive motion synthesis for virtual characters: a survey. *The Visual Computer*, 31(5):497–512.
- [Hall et al., 2009] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explorations*, 11.
- [Heck and Gleicher, 2007] Heck, R. and Gleicher, M. (2007). Parametric motion graphs. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games, I3D '07*, pages 129–136, New York, NY, USA. ACM.

- [Heck et al., 2006] Heck, R., Kovar, L., and Gleicher, M. (2006). Splicing upper-body actions with locomotion. *Computer Graphics Forum*, 25(3):459–466.
- [Heerink et al., 2012] Heerink, M., Díaz, M., Albo-Canals, J., Angulo, C., Barco, A., Casacuberta, J., and Garriga, C. (2012). A field study with primary school children on perception of social presence and interactive behavior with a pet robot. In *RO-MAN, 2012 IEEE*, pages 1045–1050. IEEE.
- [Held et al., 2012] Held, R., Gupta, A., Curless, B., and Agrawala, M. (2012). 3d puppetry: A kinect-based interface for 3d animation. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 423–434, New York, NY, USA. ACM.
- [Heun et al., 2013] Heun, V., Kasahara, S., and Maes, P. (2013). Smarter objects: Using ar technology to program physical objects and their interactions. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 961–966, New York, NY, USA. ACM.
- [Hu et al., 2011] Hu, X., Zhao, Q., and Liang, X. (2011). Fast computation of transition points for motion graph. In *Proceedings of the 2011 Workshop on Digital Media and Digital Content Management*, DMDCM '11, pages 150–153, Washington, DC, USA. IEEE Computer Society.
- [Inno Labs, 2014] Inno Labs (2014). Pleo World. <http://www.pleoworld.com>.
- [Ishii and Ullmer, 1997] Ishii, H. and Ullmer, B. (1997). Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, CHI '97, pages 234–241, New York, NY, USA. ACM.
- [ITU-T, 1996] ITU-T (1996). P.800 - methods for subjective determination of transmission quality.
- [Jacobsson, 2009] Jacobsson, M. (2009). Play, belief and stories about robots: A case study of a Pleo blogging community. In *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*, pages 232–237. IEEE.
- [James and Twigg, 2005] James, D. L. and Twigg, C. D. (2005). Skinning mesh animations. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 399–407. ACM.
- [Johansson, 1973] Johansson, G. (1973). Visual perception of biological motion and a model for its analysis. *Perception & psychophysics*, 14(2):201–211.
- [Johansson, 1976] Johansson, G. (1976). Spatio-temporal differentiation and integration in visual motion perception. *Psychological research*, 38(4):379–393.
- [Jörg et al., 2012] Jörg, S., Hodgins, J., and Safonova, A. (2012). Data-driven finger motion synthesis for gesturing characters. *ACM Trans. Graph.*, 31(6):189:1–189:7.
- [JRC Gamecentrum, 2015] JRC Gamecentrum (2015). The Sims 3 University Life [http://www.jrc.cz/hra\\_pc\\_dd\\_the\\_sims\\_3\\_university\\_life](http://www.jrc.cz/hra_pc_dd_the_sims_3_university_life).
- [Kalinli and Narayanan, 2009] Kalinli, O. and Narayanan, S. S. (2009). Prominence detection using auditory attention cues and task-dependent high level information. *IEEE Transactions on Audio, Speech & Language Processing*, 17(5):1009–1024.
- [Kalra et al., 1998] Kalra, P., Magnenat-Thalmann, N., Moccozet, L., Sannier, G., Aubel, A., and Thalmann, D. (1998). Real-time animation of realistic virtual humans. *Computer Graphics and Applications, IEEE*, 18(5):42–56.
- [Kendon, 1980] Kendon, A. (1980). Gesture and speech: two aspects of the process utterances. *Nonverbal Communication and Language*, pages 207–227.



- [Kenneth Turan, 2007a] Kenneth Turan (2007a). 'Beowulf' isn't poetry in motion capture. <http://www.latimes.com/entertainment/la-et-beowulf16nov16-story.html>.
- [Kenneth Turan, 2007b] Kenneth Turan (2007b). 'Beowulf' Sexes Up, Dumbs Down an Epic. <http://www.npr.org/templates/story/story.php?storyId=16334109>.
- [Kidd and Breazeal, 2008] Kidd, C. D. and Breazeal, C. (2008). Robots at home: Understanding long-term human-robot interaction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3230–3235. IEEE.
- [Kim et al., 2009] Kim, E. S., Leyzberg, D., Tsui, K. M., and Scassellati, B. (2009). How people talk when teaching a robot. In *Human-Robot Interaction (HRI), 2009 4th ACM/IEEE International Conference on*, pages 23–30. IEEE.
- [Kipp, 2001] Kipp, M. (2001). Anvil: A generic annotation tool for multimodal dialogue. In *Proceedings of the 7th European Conference on Speech Communication and Technology*, pages 1367–1370, Aalborg. EN.
- [Kipp, 2004] Kipp, M. (2004). *Gesture Generation by Imitation - From Human Behavior to Computer Character Animation*. PhD thesis, Saarland University, Saarbrücken, Germany.
- [Kipp and Martin, 2009] Kipp, M. and Martin, J. C. (2009). Gesture and emotion: Can basic gestural form features discriminate emotions? *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII-09)*, IEEE Press.
- [Kipp et al., 2007] Kipp, M., Neff, M., Kipp, K. H., and Albrecht, I. (2007). Towards natural gesture synthesis: Evaluating gesture units in a data-driven approach to gesture synthesis. In *Proceedings of the 7th international conference on Intelligent Virtual Agents, IVA '07*, pages 15–28, Berlin, Heidelberg. Springer-Verlag.
- [Kita et al., 1998] Kita, S., van Gijn, I., and van der Hulst, H. (1998). Movement phases in signs and co-speech gestures, and their transcription by human coders. In Wachsmuth, I. and Fröhlich, M., editors, *Gesture and Sign Language in Human-Computer Interaction*, volume 1371 of *Lecture Notes in Computer Science*, pages 23–35. Springer Berlin / Heidelberg.
- [Klein and Murray, 2007] Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE.
- [Kopp et al., 2008] Kopp, S., Bergmann, K., and Wachsmuth, I. (2008). Multimodal communication from multimodal thinking - towards an integrated model of speech and gesture production. *Int. J. Semantic Computing*, 2(1):115–136.
- [Kovar and Gleicher, 2004] Kovar, L. and Gleicher, M. (2004). Automated extraction and parameterization of motions in large data sets. *ACM Trans. Graph.*, 23:559–568.
- [Kovar et al., 2002] Kovar, L., Gleicher, M., and Pighin, F. (2002). Motion graphs. *ACM Trans. Graph.*, 21:473–482.
- [Kranz et al., 2005] Kranz, M., Schmidt, D., Holleis, P., and Schmidt, A. (2005). A display cube as a tangible user interface. In *UbiComp 2005, the Seventh International Conference on Ubiquitous Computing*.
- [La Salle Campus Barcelona - Universitat Ramon Llull., 2012] La Salle Campus Barcelona - Universitat Ramon Llull. (2012). MediaLab. Motion Capture, RV + RA, Animation, Videogames and CAD. <http://www.salleurl.edu/medialab>.

- [Lafferty et al., 2001] Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- [Larriba et al., 2015] Larriba, F., Raya, C., Angulo, C., Albo-Canals, J., Díaz, M., and Boldú, R. (2015). Externalising moods and psychological states to smooth pet-robot/child interaction through bluetooth communication. In *Bioinformatics and Biomedical Engineering*, pages 683–693. Springer.
- [Lawrence, 2005] Lawrence, N. (2005). Probabilistic non-linear principal component analysis with gaussian process latent variable models. *The Journal of Machine Learning Research*, 6:1783–1816.
- [Le et al., 2012] Le, B. H., Ma, X., and Deng, Z. (2012). Live speech driven head-and-eye motion generators. *Visualization and Computer Graphics, IEEE Transactions on*, 18(11):1902–1914.
- [Lee et al., 2002] Lee, J., Chai, J., Reitsma, P. S. A., Hodgins, J. K., and Pollard, N. S. (2002). Interactive control of avatars animated with human motion data. *ACM Trans. Graph.*, 21:491–500.
- [Lee et al., 2010] Lee, Y., Wampler, K., Bernstein, G., Popović, J., and Popović, Z. (2010). Motion fields for interactive character locomotion. *ACM Trans. Graph.*, 29:138:1–138:8.
- [Leonard and Cummins, 2011] Leonard, T. and Cummins, F. (2011). The temporal relation between beat gestures and speech. *Language and Cognitive Processes*, 26(10):1457–1471.
- [Levine et al., 2010] Levine, S., Krähenbühl, P., Thrun, S., and Koltun, V. (2010). Gesture controllers. *ACM Trans. Graph.*, 29(4):124:1–124:11.
- [Levine et al., 2009] Levine, S., Theobalt, C., and Koltun, V. (2009). Real-time prosody-driven synthesis of body language. *ACM Trans. Graph.*, 28(5):172:1–172:10.
- [Lhommet and Marsella, 2014] Lhommet, M. and Marsella, S. (2014). Metaphoric gestures: Towards grounded mental spaces. In *Intelligent Virtual Agents*, pages 264–274. Springer.
- [Lockwood and Singh, 2012] Lockwood, N. and Singh, K. (2012). Finger walking: Motion editing with contact-based hand performance. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '12*, pages 43–52, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.
- [Loehr, 2004] Loehr, D. (2004). *Gesture and Intonation*. PhD thesis, Georgetown University, Washington, D.C., United States of America.
- [Luo et al., 2009] Luo, P., Kipp, M., and M., N. (2009). Augmenting gesture animation with motion capture data to provide full-body engagement. In *Intelligent Virtual Agents*, 5773, pages 405–417. Springer Berlin Heidelberg.
- [Luo and Neff, 2012] Luo, P. and Neff, M. (2012). A perceptual study of the relationship between posture and gesture for virtual characters. In *Motion In Games*, pages 254–265. Springer.
- [Maeran et al., 1997] Maeran, O., Piuri, V., and Storti Gajani, G. (1997). Speech recognition through phoneme segmentation and neural classification. In *Instrumentation and Measurement Technology Conference, 1997. IMTC/97. Proceedings. Sensing, Processing, Networking., IEEE*, volume 2, pages 1215–1220 vol.2.
- [Maes, 1995] Maes, P. (1995). Artificial life meets entertainment: lifelike autonomous agents. *Communications of the ACM*, 38(11):108–114.

- [Magenat-Thalmann and Thalmann, 2005] Magenat-Thalmann, N. and Thalmann, D. (2005). *Handbook of virtual humans*. John Wiley & Sons.
- [Mahmudi and Kallmann, 2013] Mahmudi, M. and Kallmann, M. (2013). Analyzing locomotion synthesis with feature-based motion graphs. *IEEE Transactions on Visualization and Computer Graphics*, 19(5):774–786.
- [Marsella et al., 2013] Marsella, S., Xu, Y., Lhommet, M., Feng, A., Scherer, S., and Shapiro, A. (2013). Virtual character performance from speech. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 25–35. ACM.
- [Marshall et al., 2002] Marshall, P., Rogers, Y., and Scaife, M. (2002). Puppet: a virtual environment for children to act and direct interactive narratives. In *2nd international workshop on narrative and interactive learning environments*, pages 8–15.
- [Matthews, 1975] Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta*, 405(2):442–451.
- [Mazalek and Nitsche, 2007] Mazalek, A. and Nitsche, M. (2007). Tangible interfaces for real-time 3d virtual environments. In *Proceedings of the international conference on Advances in computer entertainment technology*, pages 155–162. ACM.
- [McCann and Pollard, 2007] McCann, J. and Pollard, N. (2007). Responsive characters from motion fragments. *ACM Trans. Graph.*, 26.
- [McNeill, 1985] McNeill, D. (1985). So you think gestures are nonverbal? *Psychological Review*, 92(3):350–371.
- [McNeill, 1992] McNeill, D. (1992). *Hand and Mind: What Gestures Reveal about Thought*. University of Chicago Press, Chicago.
- [McNeill, 2008] McNeill, D. (2008). *Gesture and Thought*. Phoenix Poets Series. University of Chicago Press.
- [Menache, 1999] Menache, A. (1999). *Understanding Motion Capture for Computer Animation and Video Games*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition.
- [Meredith and Maddock, 2001] Meredith, M. and Maddock, S. (2001). Motion capture file formats explained. *Department of Computer Science, University of Sheffield*, 211:241–244.
- [Microsoft, 2013] Microsoft (2013). Microsoft Speech API. <http://www.microsoft.com/en-us/download/details.aspx?id=10121>.
- [Microsoft, 2015a] Microsoft (2015a). HoloLens <https://www.microsoft.com/microsoft-hololens/en-us>.
- [Microsoft, 2015b] Microsoft (2015b). Kinect Xbox One. <http://www.xbox.com/es-ES/xbox-one/accessories/kinect-for-xbox-one>.
- [Microsoft, 2015c] Microsoft (2015c). Windows - Microsoft <http://www.microsoft.com/en-us/windows>.
- [Min and Chai, 2012] Min, J. and Chai, J. (2012). Motion graphs++: a compact generative model for semantic motion analysis and synthesis. *ACM Trans. Graph.*, 31(6):153:1–153:12.
- [Moeslund and Granum, 2001] Moeslund, T. B. and Granum, E. (2001). A survey of computer vision-based human motion capture. *Comput. Vis. Image Underst.*, 81:231–268.

- [Moeslund et al., 2006] Moeslund, T. B., Hilton, A., and Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Comput. Vis. Image Underst.*, 104:90–126.
- [Mori et al., 2012] Mori, M., MacDorman, K. F., and Kageki, N. (2012). The uncanny valley [from the field]. *Robotics & Automation Magazine, IEEE*, 19(2):98–100.
- [Müller, 2007] Müller, M. (2007). Dynamic time warping. *Information retrieval for music and motion*, pages 69–84.
- [Müller et al., 2007] Müller, M., Röder, T., Clausen, M., Eberhardt, B., Krüger, B., and Weber, A. (2007). Documentation mocap database hdm05. Technical Report CG-2007-2, Universität Bonn.
- [Navarro et al., 2013] Navarro, J., Sancho-Asensio, A., Garriga, C., Albó-Canals, J., Ortiz-Villajos Maroto, J., Raya Giner, C., Angulo Bahón, C., and Miralles, D. (2013). A cloud robotics architecture to foster individual child partnership in medical facilities. In *Cloud Robotics Workshop in 26th IEEE/RSJ International Conference on Intelligent Robots and Systems*.
- [Neff et al., 2008] Neff, M., Kipp, M., Albrecht, I., and Seidel, H.-P. (2008). Gesture modeling and animation based on a probabilistic re-creation of speaker style. *ACM Trans. Graph.*, 27(1):5:1–5:24.
- [Ng et al., 2010] Ng, W. W. L., Choy, C. S. T., Lun, D. P. K., and Chau, L.-P. (2010). Synchronized partial-body motion graphs. In *ACM SIGGRAPH ASIA 2010 Sketches*, SA '10, pages 28:1–28:2, New York, NY, USA. ACM.
- [Nguyen, 2007] Nguyen, H. (2007). *Gpu gems 3*. Addison-Wesley Professional.
- [Nintendo, 2015a] Nintendo (2015a). Amiibo by Nintendo - Amiibo for Wii U and New Nintendo 3DS XL <http://www.nintendo.com/amiibo>.
- [Nintendo, 2015b] Nintendo (2015b). Nintendo 3DS - AR Cards. <http://www.nintendo.com/3ds/ar-cards>.
- [Nobe, 1996] Nobe, S. (1996). *Representational Gestures, Cognitive Rhythms, and Acoustic Aspects of Speech: A Network/threshold Model of Gesture Production*. University of Chicago, Department of Psychology.
- [Numaguchi et al., 2011] Numaguchi, N., Nakazawa, A., Shiratori, T., and Hodgins, J. K. (2011). A puppet interface for retrieval of motion capture data. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, pages 157–166, New York, NY, USA. ACM.
- [Olfati-Saber, 2006] Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems: Algorithms and theory. *Automatic Control, IEEE Transactions on*, 51(3):401–420.
- [Onuma et al., 2008] Onuma, K., Faloutsos, C., and Hodgins, J. K. (2008). FMDistance: A fast and effective distance function for motion capture data. In *Short Papers Proceedings of EUROGRAPHICS*.
- [Ortega-Llebaria and Prieto, 2011] Ortega-Llebaria, M. and Prieto, P. (2011). Acoustic correlates of stress in central catalan and castilian spanish. *Lang Speech*, 54(1):73–97.
- [Ortiz-Lira, 1999] Ortiz-Lira, H. (1999). La aplicación de tobi a un corpus del español de chile. *Onomázein*, 4:429–442.
- [Oshita and Masaoka, 2011] Oshita, M. and Masaoka, N. (2011). Generating avoidance motion using motion graph. In *Proceedings of the 4th international conference on Motion in Games*, MIG'11, pages 120–131, Berlin, Heidelberg. Springer-Verlag.

- [Oshita et al., 2014] Oshita, M., Oshima, H., Senju, Y., and Morishige, S. (2014). Character motion control by hands and principal component analysis. In *Proceedings of the 13th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry*, pages 171–179. ACM.
- [Oshita et al., 2013] Oshita, M., Senju, Y., and Morishige, S. (2013). Character motion control interface with hand manipulation inspired by puppet mechanism. In *Proceedings of the 12th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI '13, pages 131–138, New York, NY, USA. ACM.
- [Paepcke and Takayama, 2010] Paepcke, S. and Takayama, L. (2010). Judging a bot by its cover: an experiment on expectation setting for personal robots. In *Human-Robot Interaction (HRI), 2010 5th ACM/IEEE International Conference on*, pages 45–52. IEEE.
- [Pejsa and Pandzic, 2010] Pejsa, T. and Pandzic, I. (2010). State of the art in example-based motion synthesis for virtual characters in interactive applications. *Computer Graphics Forum*, 29(1):202–226.
- [Pelachaud, 2005] Pelachaud, C. (2005). Multimodal expressive embodied conversational agents. In *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTIMEDIA '05, pages 683–689, New York, NY, USA. ACM.
- [Pierrehumbert, 1980] Pierrehumbert, J. (1980). *The Phonology and Phonetics of English Intonation*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, United States of America.
- [Pitsch and Koch, 2010] Pitsch, K. and Koch, B. (2010). How infants perceive the toy robot Pleo. an exploratory case study on infant-robot-interaction. Second International Symposium on New Frontiers in Human-Robot-Interaction (AISB). The Society for the Study of Artificial Intelligence and the Simulation of Behaviour.
- [Pixar, 1995] Pixar (1995). Toy Story. IMDb: [http://www.imdb.com/title/tt0114709/?ref\\_=nv\\_sr\\_1](http://www.imdb.com/title/tt0114709/?ref_=nv_sr_1).
- [Pixar, 2003] Pixar (2003). Finding Nemo. IMDb: [http://www.imdb.com/title/tt0266543/?ref\\_=fn\\_al\\_tt\\_1](http://www.imdb.com/title/tt0266543/?ref_=fn_al_tt_1).
- [Pixar, 2007] Pixar (2007). Ratatouille. IMDb: [http://www.imdb.com/title/tt0382932/?ref\\_=nv\\_sr\\_1](http://www.imdb.com/title/tt0382932/?ref_=nv_sr_1).
- [Pixar, 2008] Pixar (2008). The incredibles. IMDb: [http://www.imdb.com/title/tt0317705/?ref\\_=nv\\_sr\\_1](http://www.imdb.com/title/tt0317705/?ref_=nv_sr_1).
- [Pixar, 2009] Pixar (2009). Up. IMDb: [http://www.imdb.com/title/tt1049413/?ref\\_=nv\\_sr\\_6](http://www.imdb.com/title/tt1049413/?ref_=nv_sr_6).
- [Pixar, 2012] Pixar (2012). Brave. IMDb: [http://www.imdb.com/title/tt1217209/?ref\\_=ttfc\\_fc\\_tt](http://www.imdb.com/title/tt1217209/?ref_=ttfc_fc_tt).
- [Pixar, 2015] Pixar (2015). Pixar. IMDb: <http://www.pixar.com/>.
- [Planet et al., 2008] Planet, S., Iriondo, I., Martinez, E., and Montero, J. A. (2008). True: an online testing platform for multimedia evaluation. In *Proceedings of the Second International Workshop on EMOTION: Corpora for Research on Emotion and Affect at the 6th Conference on Language Resources & Evaluation*, LREC '08.

- [Platt, 1999] Platt, J. C. (1999). *Fast training of support vector machines using sequential minimal optimization*, pages 185–208. MIT Press, Cambridge, MA, USA.
- [Poslad, 2011] Poslad, S. (2011). *Ubiquitous computing: smart devices, environments and interactions*. John Wiley & Sons.
- [PROFITguide.com, 2015] PROFITguide.com (2015). Great Ideas: The Presentation Secrets of Steve Jobs <http://www.profitguide.com/manage-grow/leadership/great-ideas-the-presentation-secrets-of-steve-jobs-29644>.
- [Pullen and Bregler, 2002] Pullen, K. and Bregler, C. (2002). Motion capture assisted animation: texturing and synthesis. *ACM Trans. Graph.*, 21:501–508.
- [Qin et al., 2010] Qin, Y., Zhang, X., and Ying, H. (2010). A hmm-based fuzzy affective model for emotional speech synthesis. In *Signal Processing Systems (ICSPS), 2010 2nd International Conference on*, volume 3, pages V3–525. IEEE.
- [Qualcomm Connected Experiences, Inc., 2014] Qualcomm Connected Experiences, Inc. (2014). Unity Extension - Vuforia v2.8. <https://developer.vuforia.com/resources/sdk/unity>.
- [Quintilian. and Butler, 1920] Quintilian. and Butler, H. E. (1920). *Institutio oratoria / Quintilian; with an English translation by H.E. Butler*. Heinemann, London.
- [Rabiner and Juang, 1986] Rabiner, L. R. and Juang, B.-H. (1986). An introduction to hidden markov models. *ASSP Magazine, IEEE*, 3(1):4–16.
- [Rahim et al., 2009] Rahim, R., Suaib, N., and Bade, A. (2009). Motion graph for character animation: Design considerations. In *Computer Technology and Development, 2009. ICCTD '09. International Conference on*, volume 2, pages 435–439.
- [Reitsma and Pollard, 2007] Reitsma, P. S. A. and Pollard, N. S. (2007). Evaluating motion graphs for character animation. *ACM Trans. Graph.*, 26.
- [Ren et al., 2010] Ren, C., Zhao, L., and Safonova, A. (2010). Human motion synthesis with optimization-based graphs. *Computer Graphics Forum*, 29(2):545–554.
- [Renwick et al., 2004] Renwick, M., Yasinnik, Y., and Shattuck-Hufnagel, S. (2004). The timing of speech-accompanying gestures with respect to prosody. *From Sound to Sense: 50+ Years of Discoveries in Speech Communication*, pages 97–102.
- [Riedl and Young, 2010] Riedl, M. O. and Young, R. M. (2010). Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research*, 39(1):217–268.
- [Robert Zemeckis, 2004] Robert Zemeckis (2004). Polar Express. IMDb: [http://www.imdb.com/title/tt0338348/?ref\\_=nv\\_sr\\_1](http://www.imdb.com/title/tt0338348/?ref_=nv_sr_1).
- [Robert Zemeckis, 2007] Robert Zemeckis (2007). Beowulf. IMDb: [http://www.imdb.com/title/tt0442933/?ref\\_=nv\\_sr\\_1](http://www.imdb.com/title/tt0442933/?ref_=nv_sr_1).
- [Rodrigo Pegorari, 2013] Rodrigo Pegorari (2013). Tagarela - Open source lip sync system for Unity. <http://rodrigopegorari.net/blog/?p=241>.
- [Roekhaut et al., 2010] Roekhaut, S., Goldman, J., and Simon, A. (2010). A model for varying speaking style in tts systems. *Fifth International Conference on Speech Prosody*.
- [Rose et al., 1998] Rose, C., Cohen, M. F., and Bodenheimer, B. (1998). Verbs and adverbs: Multidimensional motion interpolation. *IEEE Comput. Graph. Appl.*, 18:32–40.

- [Rose et al., 1996] Rose, C., Guenter, B., Bodenheimer, B., and Cohen, M. F. (1996). Efficient generation of motion transitions using spacetime constraints. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 147–154, New York, NY, USA. ACM.
- [Russell, 1980] Russell, J. (1980). A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161–1178.
- [Ryokai et al., 2009] Ryokai, K., Lee, M. J., and Breitbart, J. M. (2009). Children's storytelling and programming with robotic characters. In *Proceedings of the seventh ACM conference on Creativity and cognition*, pages 19–28. ACM.
- [Sadoughi et al., 2014] Sadoughi, N., Liu, Y., and Busso, C. (2014). Speech-driven animation constrained by appropriate discourse functions. In *Proceedings of the 16th International Conference on Multimodal Interaction*, pages 148–155. ACM.
- [Safonova and Hodgins, 2007] Safonova, A. and Hodgins, J. K. (2007). Construction and optimal search of interpolated motion graphs. *ACM Trans. Graph.*, 26.
- [Samsung Electronics Co., Ltd., 2015] Samsung Electronics Co., Ltd. (2015). AR EdiBear. <http://apps.samsung.com/earth/topApps/>.
- [Segura et al., 2012] Segura, E. M., Cramer, H., Gomes, P. F., Nylander, S., and Paiva, A. (2012). Revive!: reactions to migration between different embodiments when playing with robotic pets. In *Proceedings of the 11th International Conference on Interaction Design and Children*, pages 88–97. ACM.
- [Seol et al., 2013] Seol, Y., O'Sullivan, C., and Lee, J. (2013). Creature features: online motion puppetry for non-human characters. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 213–221. ACM.
- [Shalev-Shwartz et al., 2007] Shalev-Shwartz, S., Singer, Y., and Srebro, N. (2007). Pegasos: Primal estimated sub-gradient solver for svm. In *24th International Conference on Machine Learning*, pages 807–814.
- [Shattuck-Hufnagel et al., 2007] Shattuck-Hufnagel, S., Yasinnik, Y., Veilleux, N., and Renwick, M. (2007). A method for studying the time alignment of gestures and prosody in american english: 'hits' and pitch accents in academic-lecture-style speech. In Anna Esposito, Maja Bratanić, Eric Keller and Maria Marinaro, editor, *Fundamentals of Verbal and Nonverbal Communication and the Biometric Issue*, volume 18 of *NATO Publishing Sub-Series E: Human and Societal Dynamics*. Washington, DC.
- [Shin and Oh, 2006] Shin, H. J. and Oh, H. S. (2006). Fat graphs: constructing an interactive character with continuous controls. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '06, pages 291–298, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.
- [Shin et al., 2005] Shin, M., Kim, B.-s., and Park, J. (2005). Ar storyboard: an augmented reality based interactive storyboard authoring tool. In *Proceedings of the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 198–199. IEEE Computer Society.
- [Shiratori, 2014] Shiratori, T. (2014). User interfaces for character animation and character interaction. In *The Impact of Applications on Mathematics*, pages 335–347. Springer.
- [Shiratori et al., 2011] Shiratori, T., Park, H. S., Sigal, L., Sheikh, Y., and Hodgins, J. K. (2011). Motion capture from body-mounted cameras. *ACM Transactions on Graphics (TOG)*, 30(4):31.

- [Silaghi et al., 1998] Silaghi, M.-C., Plänklers, R., Boulic, R., Fua, P., and Thalmann, D. (1998). Local and global skeleton fitting techniques for optical motion capture. In *Modelling and Motion Capture Techniques for Virtual Environments*, pages 26–40. Springer.
- [Silipo and Greenberg, 1999] Silipo, R. and Greenberg, S. (1999). Automatic transcription of prosodic stress for spontaneous English discourse. In Olds, J. J., Hasegawa, Y., Ohala, M., and Bailey, A. C., editors, *Proceedings of the XIVth International Congress of Phonetic Sciences (ICPhS99)*, pages 2351–2354. The Regents of the University of California.
- [Silverman et al., 1992a] Silverman, K., Beckman, M., Pierrehumbert, J., Ostendorf, M., Wightman, C., and Hirschberg, J. (1992a). TOBI: A standard scheme for labeling prosody. In *Proceedings of ICSLP-92*, pages 867–879, Banff.
- [Silverman et al., 1992b] Silverman, K., Beckman, M. E., Pitrelli, J. F., Ostendorf, M., Wightman, C. W., Price, P., Pierrehumbert, J. B., and Hirschberg, J. (1992b). Tobi: a standard for labeling english prosody. In *ICSLP*. ISCA.
- [Sony, 2014] Sony (2014). Aibo. <http://www.sony-aibo.co.uk/>.
- [Stone et al., 2004] Stone, M., DeCarlo, D., Oh, I., Rodriguez, C., Stere, A., Lees, A., and Bregler, C. (2004). Speaking with hands: creating animated conversational characters from recordings of human performance. *ACM Trans. Graph.*, 23(3):506–513.
- [Streefkerk et al., 1999] Streefkerk, B. M., Pols, L. C. W., and ten Bosch, L. (1999). Acoustical features as predictors for prominence in read aloud dutch sentences used in ann's. In *EUROSPEECH*. ISCA.
- [Syrdal et al., 2001] Syrdal, A. K., Hirschberg, J., McGory, J., and Beckman, M. (2001). Automatic tobi prediction and alignment to speed manual labeling of prosody. *Speech Commun.*, 33(1-2):135–151.
- [Tamburini and Wagner, 2007] Tamburini, F. and Wagner, P. (2007). On automatic prominence detection for german. In *INTERSPEECH*, pages 1809–1812. ISCA.
- [Tanco and Hilton, 2000] Tanco, L. M. and Hilton, A. (2000). Realistic synthesis of novel human movements from a database of motion capture examples. In *Proceedings of the Workshop on Human Motion (HUMO'00)*, HUMO '00, pages 137–, Washington, DC, USA. IEEE Computer Society.
- [Tarjan, 1972] Tarjan, R. (1972). Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160.
- [Teran et al., 2005] Teran, J., Sifakis, E., Blemker, S. S., Ng-Thow-Hing, V., Lau, C., and Fedkiw, R. (2005). Creating and simulating skeletal muscle from the visible human data set. *Visualization and Computer Graphics, IEEE Transactions on*, 11(3):317–328.
- [Terken, 1991] Terken, J. (1991). Fundamental frequency and perceived prominence of accented syllables. *The Journal of the Acoustical Society of America*, 89(4):1768–1776.
- [Thalmann, 2007] Thalmann, D. (2007). *Crowd simulation*. Wiley Online Library.
- [The Capture Lab, 2015] The Capture Lab (2015). Motion Capture - The Capture Lab <http://www.capturelab.com/motion-capture/>.
- [Unity Technologies, 2014a] Unity Technologies (2014a). Mecanim: Simple and powerful animation technology. <http://unity3d.com/unity/animation>.



- [Unity Technologies, 2014b] Unity Technologies (2014b). Unity - Game Engine <http://unity3d.com/>.
- [University of California, San Diego. CSE 169: Computer Animation, 2015] University of California, San Diego. CSE 169: Computer Animation (2015). Chapter 2: Skeletons. [http://graphics.ucsd.edu/courses/cse169\\_w05/2-Skeleton.htm](http://graphics.ucsd.edu/courses/cse169_w05/2-Skeleton.htm).
- [Valbonesi, 2002] Valbonesi, L. (2002). *Multimodal Signal Analysis of Prosody and Hand Motion: Temporal Correlation in Speech and Gestures*. University of Illinois at Chicago.
- [van Basten and Egges, 2009] van Basten, B. J. H. and Egges, A. (2009). Evaluating distance metrics for animation blending. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, FDG '09, pages 199–206, New York, NY, USA. ACM.
- [van der Sluis and Kraemer, 2007] van der Sluis, I. and Kraemer, E. (2007). Generating multimodal references. *Discourse Processes: A Multidisciplinary Journal*, 44(3):145–174.
- [van Rijsbergen, 1974] van Rijsbergen, C. (1974). Foundation of evaluation. *Journal of Documentation*, 30(4):365–373.
- [Van Welbergen et al., 2010] Van Welbergen, H., Van Basten, B. J. H., Egges, A., Ruttkay, Z. M., and Overmars, M. H. (2010). Real time animation of virtual humans: A trade-off between naturalness and control. *Computer Graphics Forum*, 29(8):2530–2554.
- [Vilaplana and Prieto, 2009] Vilaplana, E. and Prieto, P. (2009). La notación prosódica en español. una revisión del sp-tobi. *Estudios de Fonética Experimental XVIII*, pages 263–283.
- [Vilhjálmsón, 2003] Vilhjálmsón, H. (2003). *Avatar augmented online conversation*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, United States of America.
- [Vinayagamoorthy et al., 2006] Vinayagamoorthy, V., Gillies, M., Steed, A., Tanguy, E., Pan, X., Loscos, C., Slater, M., et al. (2006). Building expression into virtual characters.
- [Vukobratovic, 2012] Vukobratovic, M. (2012). *Introduction to robotics*. Springer Science & Business Media.
- [Wallbott, 1998] Wallbott, H. G. (1998). Bodily expression of emotion. *Eur. J. Soc. Psychol.*, 28(6):879–896.
- [Wampler et al., 2013] Wampler, K., Popović, J., and Popović, Z. (2013). Animal locomotion controllers from scratch. In *Computer Graphics Forum*, volume 32, pages 153–162. Wiley Online Library.
- [Wang and Bodenheimer, 2008] Wang, J. and Bodenheimer, B. (2008). Synthesis and evaluation of linear motion transitions. *ACM Trans. Graph.*, 27:1:1–1:15.
- [Wang et al., 2014] Wang, X., Chen, Q., and Wang, W. (2014). 3d human motion editing and synthesis: A survey. *Computational and mathematical methods in medicine*, 2014.
- [Ward et al., 2007] Ward, K., Bertails, F., Kim, T.-Y., Marschner, S. R., Cani, M.-P., and Lin, M. C. (2007). A survey on hair modeling: Styling, simulation, and rendering. *Visualization and Computer Graphics, IEEE Transactions on*, 13(2):213–234.
- [Weiser, 1993] Weiser, M. (1993). Hot topics-ubiquitous computing. *Computer*, 26(10):71–72.
- [Wheatland et al., 2015] Wheatland, N., Wang, Y., Song, H., Neff, M., Zordan, V., and Jorg, S. (2015). State of the Art in Hand and Finger Modeling and Animation. *Computer Graphics Forum*.

- [Wikipedia, 2015a] Wikipedia (2015a). HoloLens [https://upload.wikimedia.org/wikipedia/en/4/48/Microsoft\\_Windows\\_Holographic.png](https://upload.wikimedia.org/wikipedia/en/4/48/Microsoft_Windows_Holographic.png).
- [Wikipedia, 2015b] Wikipedia (2015b). Pitch accent (intonation) [https://en.wikipedia.org/wiki/Pitch\\_accent\\_\(intonation\)](https://en.wikipedia.org/wiki/Pitch_accent_(intonation)).
- [Wikipedia Commons, 2015a] Wikipedia Commons (2015a). Motion Capture. [http://en.wikipedia.org/wiki/Motion\\_capture](http://en.wikipedia.org/wiki/Motion_capture).
- [Wikipedia Commons, 2015b] Wikipedia Commons (2015b). Steve Jobs. [http://en.wikipedia.org/wiki/Steve\\_Jobs](http://en.wikipedia.org/wiki/Steve_Jobs).
- [Witkin and Popovic, 1995] Witkin, A. and Popovic, Z. (1995). Motion warping. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, SIGGRAPH '95*, pages 105–108, New York, NY, USA. ACM.
- [Xing et al., 2014] Xing, W., Wei, X., Zhang, J., Ren, C., and Lu, W. (2014). Hybrid motion graph for character motion synthesis. *Journal of Visual Languages & Computing*, 25(1):20–32.
- [Xu et al., 2011] Xu, J., Takagi, K., and Sakazawa, S. (2011). Motion synthesis for synchronizing with streaming music by segment-based search on metadata motion graphs. In *ICME*, pages 1–6. IEEE.
- [Zhao and Safonova, 2009] Zhao, L. and Safonova, A. (2009). Achieving good connectivity in motion graphs. *Graph. Models*, 71:139–152.
- [Zordan et al., 2005] Zordan, V. B., Majkowska, A., Chiu, B., and Fast, M. (2005). Dynamic response for motion capture animation. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 697–701. ACM.



Aquesta Tesi Doctoral ha estat defensada el dia \_\_\_\_ d\_\_\_\_\_ de 201\_\_  
al Centre\_\_\_\_\_

de la Universitat Ramon Llull, davant el Tribunal format pels Doctors i Doctores  
sotasignants, havent obtingut la qualificació:

President/a

\_\_\_\_\_

Vocal

\_\_\_\_\_

Vocal \*

\_\_\_\_\_

Vocal \*

\_\_\_\_\_

Secretari/ària

\_\_\_\_\_

Doctorand/a

\_\_\_\_\_

(\*): Només en el cas de tenir un tribunal de 5 membres