José Serra

Intelligent facial animation: Creating emphatic characters with stimuli based animation

Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto
April 2017

José Serra

Intelligent facial animation: Creating emphatic characters with stimuli based animation

Thesis submitted to Faculdade de Ciências da Universidade do Porto
in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

Orientador: Verónica Orvalho
Co-orientador: Miguel Sales Dias

Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto
April 2017

Acknowledgements

This has been a long ride... Enlightening, pleasant, hard and painful are some words that describe the journey. I doubt, however, that it would have been possible to survive, or even finish, this thesis, without the support and friendship of many different people. The list is long, but I'll do my best to remember everyone and thank them with all my heart.

And, there is no-one better to start than with my parents. They constantly pushed me forward, encouraged me both in the good and the bad moments. Their unconditional love and support helped me from beginning to the end of this journey. I just can't thank them enough. My family was also very supportive, but I can't help to dedicate a few words to my adoptive grandmother Adelaide, "Lai", another pillar of my life and this Ph.D. All her love and home-made food gave me the energy to continue.

Then I need to express my gratitude to my supervisor, Verónica Orvalho. For believing in me, and setting me on this path. Procedural animation is an amazing research field, which I still very much love to work on. I couldn't have had a better suggestion for the research field, and it is due to her insights that I ended up pursuing the field. I also would like to thank my co-supervisor, Miguel Sales Dias, who, together with Verónica provided invaluable help in getting the FCT scholarship. I deeply thank them for their help.

I'm also grateful for all the support of the members of PIC. Just listening to me ranting about the problems of my work deserves quite some recognition. But I have, additionally, been helped with a wide range of tasks that vary from reviewing papers, getting 3D models, tools to deform them, improving the rendering quality and much more. From this group, I especially thank Pedro Mendes, Ozan Cetinaslan, Teresa Vieira and Catarina Runa, although everyone in PIC helped to make my work, and the many after-hours, much more enjoyable and fun. Within the Faculdade de Ciências, I too need to thank Alexandra Ferreira for all the advice on how to navigate in the Ph.D. bureaucracy.

Within the Ph.D. I have collaborated with amazing people, from which I must, first and foremost, thank Darren Cosker and his group at University of Bath. It was per his suggestion that I first looked into motion graphs and that lead to the main research thread of this thesis. He provided invaluable feedback on the work and helped to shape it, for which I'm deeply thankful. Also from his group, I thank Martin Parsons and Oscar De Mello for being so understanding and supportive when I took the time out to work on my Ph.D., even though I had just started as a studio engineer at CAMERA. Finally, I need to acknowledge Shridhar Ravikumar for the many discussions on facial animation topics and helping to compare our motion graph method with his motion capture approach.

Last but not the least, I thank my friends that encouraged me throughout the whole process and reminded me that there is life aside from the Ph.D. They helped me to relax under the stress of achieving results. I've met great people throughout the Ph.D., at conferences, meetings, uni... but it was my family and friends that held me up whenever I was falling. And from here, I need to thank Nádia Baptista and Rui Mendes for all the delicious meals and philosophic discussions accompanied by Port Wine. Pedro Lopes for always listening to me rant. Pedro Rodrigues, Tiago Pinho, Carlos Rodrigues, Rui Martins, Hugo Pereira are all invaluable friends that just by being there provided incredible support, but they did so much more. Laughing, listening, joking and, for some reason, constantly invoking "Tá Quente!" and livening up my life. And, of course, their natural counterparts: Ana Ferreira, Filipa Gouveia, Rita Beles and Carla Preto, who always bring more fun and energy to our moments together. Gustavo Augusto, Pedro Nogueira and Edgard Neto for sharing so many interests with me, and the many amazing and random conversations that would start with Ph.D. and end with asian culture... All of which very much fundamental, if not for my research, at least for my mental health.

There are not enough words to show my gratitude for all the people here. They were, and still are, a fundamental part of my life. I will, however, gladly take anyone here out for a Francesinha. As a small display of my feelings, because feelings never filled anyone's stomach. In the end, and after having used way too many words, I just want to say: *I would not have made it here without the support of my family, friends and the many people listed here*.

Thank you

Abstract

Facial animation plays a crucial role in creating virtual characters, be them part of a videogame, film or other productions. It is a key component required to convey the character's feelings and connect with the audience. However, creating facial animations and playing them at the right time are complex tasks. Animation is, traditionally, either manually key-framed or captured from someone's performance. The former is a slow process that requires vast artistic experience. The latter is faster but needs more complex equipment, techniques and an actor. Independently of the technique, animation is, by itself, just motion of specific parts of the face. What makes it interesting is that a specific motion can have deeply different meanings depending on when it occurs. These moments show insights on the character's mind and are a fundamental part of its behaviour. The choice of when to play the animations can be direct, such as in a film, or based on scripting or decision trees, which is the case of interactive applications. Scripting and trees require extensive discrimination of all behaviours and their relation with stimuli, hence being very cumbersome to produce.

Due to high costs and slowness of current approaches, the creation of varied animations and complex behaviours is only possible for hero characters, which are the main focus of the audience's attention. Thus, leaving a multitude of secondary characters, present in many productions, limited to a small array of motions and behaviours constantly repeated. This becomes a problem after a small number of interactions with the viewer/player, as the repetitions are easily and quickly noticed. Such can, in turn, affect the immersion of the viewer in the medium.

This thesis presents a *novel procedural animation method that combines unique, on-the-fly synthesis with intuitive behaviour control* to produce more believable secondary characters. Our contributions are threefold: a motion graph-based approach capable generating new facial animation with minimal input; a generic mind-map behaviour controller that endows characters of varied reactions to stimuli; and a behavioural animation method that combines and explores synergies between both approaches.

Our animation method uses motion graphs to encode a facial motion database with a low memory footprint. We introduce a threshold optimization technique that significantly sim-

plifies the process of configuring motion graphs. Motion synthesis occurs by traversing the graph between automatically labelled nodes. The graph structure is additionally explored to introduce coherent noise and generate idle motion, both obtained for free from the structure. Our method allows authoring from a low number of input parameters and synthesis of different animations on each generation, near real-time.

On the behaviour side, we show how mind maps can be used to define and control a character's decisions. This is done via a hierarchical graph that encodes both the stimulus and reactions. Whenever a stimulus occurs, the character's mood is updated and used to trigger an action. Different personality traits add more emotional complexity to the chosen behaviour. Characters thus react differently to the same stimulus, without the need to specify all pairs of stimulus-reaction. Our behaviour method is abstract and can be used with both body and facial actions. We, nevertheless, further intertwine it with facial motions to create insight (mood decaying) animations, i.e. small movements synthesized as the mood decays to a base emotional state. These animations provide insights into the characters' mind, enabling them to do more than just react to the world. We present an end-to-end facial animation system that accounts for both behaviour and animation. It enables more autonomous and emphatic characters, created at a significantly lower cost and production time than existing approaches.

We have validated the general purpose of our animation method by testing it with two databases and by comparing it against facial performance capture and another motion graph-based approach. We also extensively validated the synthesised motion by matching it with the original samples, for various compression levels, and analysing quantitatively how different the motions are. The behaviour method has been implemented and empirically tested with several proofs-of-concept. A conceptual mind map approach, i.e. no animations, was also validated within the national project LIFEisGAME - Learning of Facial Emotions using Serious Games.

The field of procedural facial animation has remained largely under-explored. Reasons for this include lack of good quality motion databases, being a multidisciplinary research topic or complexity of facial movements, which are also highly subjective. Additionally, very few methods are capable of generating both facial timing and pose details. We have explored this field and presented methods capable of filling the need for more expressive secondary character animation, with less effort. Such is becoming even more critical due to the ever-growing demand of characters that populate crowds or play small roles in video-games and virtual reality worlds. Furthermore, by combining behaviour with motion synthesis, we shift the task of animating a character to one of defining its behaviour. On the long run, this change could even lead to the democratization of character facial animation.

Resumo

Animação facial tem um papel crucial na criação de personagens virtuais, sejam estas parte de um videojogo, de um filme ou de outras produções. É um elemento chave para transmitir os sentimentos da personagem e criar laços com a audiência. No entanto, criar animações faciais e escolher o momento certo para as mostrar são tarefas complexas. Tradicionalmente, a animação é criada de forma manual, i.e. *key-framing*, ou capturada a partir da performance de alguém. A primeira abordagem é um processo lento, que requer vasta experiência artística. Por seu turno, capturar os movimentos é mais rápido, no entanto precisa de técnicas, e equipamento mais complexo, bem como de um actor. Independentemente do método, animação é, por si só, apenas o movimento de partes especificas da cara. O que a torna interessante é que um movimento concreto pode ter significados profundamente diferentes consoante a altura em que ocorrem. Estes momentos fornecem vislumbres da mente da personagem e são uma parte fundamental do seu comportamento. A escolha de qual animação mostrar num momento pode ser directa, tal como em filmes, ou baseada em scripting ou árvores de decisão, o que acontece em aplicações interactivas. Árvores e scripting requerem uma discriminação extensiva de todos os comportamentos e a sua relação com estímulos, sendo, consequentemente, a sua criação muito trabalhosa.

Devido aos custos elevados, tanto em termos monetariários como temporais, a abordagens actuais só permitem a criação de animações variadas e comportamentos complexos a ”personagens herói”, que são o foco de atenção da audiência. Isto leva a que personagens secundárias, presentes em muitas produções, estejam limitadas a um pequeno conjunto de movimentos e comportamentos constantemente repetidos, o que se torna problemático após um pequeno número de interacções com o jogador/espectador, pois as repetições são detectadas fácil e rapidamente. Ora is pode, por sua vez, afectar a imersão da pessoa no meio.

Esta tese apresenta um novo método procedimental que combina síntese de animação original, no momento, com controlo de comportamento intuitivo para produzir personagens secundárias credíveis. As nossas contribuições incluem: uma abordagem baseada em *motion graphs* capaz de gerar animação facial original com input mínimo; um controlador

de comportamento genérico baseado em *mind maps* que dota as personagens de reacções variadas a estímulos; e um método de animação comportamental que combina e explora sinergias entre as duas abordagens anteriores.

O nosso método de animação usa *motion graphs* para codificar uma base de dados de movimentos faciais e com um baixo consumo de memória. Ao mesmo tempo, introduzimos uma nova técnica de optimização de *thresholds* que simplifica significativamente o processo de configurar *motion graphs*. Os movimentos são síntetizados atravessando o grafo entre nós automaticamente rotulados. O grafo é também explorado para introduzir ruído coerente e gerar movimentos ociosos, sendo ambos os casos obtidos, sem esforço adicional, a partir da estrutura e para o utilizador final. O nosso método é controlado com um número reduzido de parâmetros e permite a síntese de animações variadas a cada geração, quase em tempo real.

Do lado do comportamento da personagem, mostramos como usar *mind maps* para definir e controlar as decisões de uma personagem. Isto é conseguido através de um grafo hierárquico que codifica tanto os estímulos da personagem como as suas reacções. Sempre que um estímulo ocorre, o humor da personagem é actualizado e usado para despoletar uma acção. Diferentes traços da personalidade acrescentam uma maior complexidade emocional ao comportamento escolhido. Permitindo assim que as personagens reajam diferentemente a um mesmo estímulo sem a necessidade de especificar todos os pares de acção-reacção. O nosso método de simulação de comportamento é abstracto e pode ser usado tanto para reacções corporais como faciais. Adicionalmente, interligámos este método com movimentos faciais para criar animações de decaimento do humor, i.e. pequenos movimentos sintetizados à medida que o humor decai para um estado emocional base. Estas animações deixam vislumbrar o que está por trás da mente da personagem, e, ao mesmo tempo, permitem que estas façam mais do que apenas reagir ao mundo. Apresentamos assim uma solução completa para animação facial que lida tanto com o comportamento como com a própria animação. O nosso método permite a criação de personagens mais autónomas e empáticas, de forma mais rápida e barata que abordagens existentes.

Validamos o carácter geral do nosso método de animação testando-o com duas bases de dados e comparando os resultados com equivalentes gerados usando captura da performance e outra abordagem baseada em *motion graphs*. Também validamos extensivamente os movimentos sintetizados comparando-os com as amostras originais, para vários níveis de compressão e analisando quantitativamente as diferenças nos movimentos. O método de simulação de comportamento foi implementado e empiricamente testado com várias provas de conceito. Um versão conceptual, i.e. sem animações, foi também validada dentro do projecto nacional LIFEisGAME - Learning of Facial Emotions using Serious Games.

O campo de animação procedimental facial tem permanecido amplamente sub-explorado. Razões para tal incluem a falta de bases de dados de movimentos com boa qualidade, ser um tópico de pesquisa multidisciplinar ou complexidade dos movimentos faciais, que também são altamente subjectivos. Para além disso, são escassos os métodos que são capazes de produzir tanto detalhes temporais e poses ao mesmo tempo. Nós exploramos este campo e apresentamos métodos capazes de preencher a necessidade de personagens secundárias mais expressivas, e com menos esforço. Isto está a tornar-se ainda mais crítico devido à procura crescente de personagens que circulam nas multidões ou desempenham pequenos papéis em videojogos e mundos virtuais. Ademais, através da combinação de comportamento com síntese de movimentos, trocamos a tarefa de animar uma personagem pela definição de como se comporta. A longo prazo, esta mudança poderia levar à democratização da tarefa de animar facialmente uma personagem.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Facial behaviours represent a significant part of human communication, hence being fundamental to endow a character with the ability to express itself. People are experts in facial motion, be it of real characters or the virtual kind, and thus it is crucial to get facial animation absolutely right (despite the concept of right being highly subjective). However, producing high-quality facial animation is slow and expensive, which limits its use to hero characters. Secondary characters are, thus, reduced to a small array of animations/expressions that are constantly played, which is easily noticeable by the viewer. The issue is further increased by the process of choosing the animation to play after an event occurs, or a player provides an input. Characters with strong emotional complexity traditionally have a very vexing behaviour definition procedure. Secondary characters again suffer from this process, as they are traditionally limited to constantly choosing the same actions/animations after a stimulus. The entertainment industry needs alternatives to the traditional techniques used to create and control facial animation, capable of filling the need introduced by characters present in crowds or playing small roles in video-games and virtual reality worlds. With these issues in mind, we present a novel procedural facial animation method that enables the creation of autonomous characters, capable of reacting to stimulus with non-repetitive facial behaviours. We propose different methods that can be used separately, to generate animation or control behaviour, or combined, to realise an independent reactive character. The author can direct the character via either stimulus or expression labels, with both timing and poses automatically calculated. The present chapter provides more details on the motivation and challenges that brought this Ph.D. to life. It then highlights our solution and its scientific contributions. Finally, it provides examples of several fields that benefit from our method.

Figure 1.1: Unique animations generated automatically using our procedural method for the same sequence of input expression labels, namely *fear, surprise* and *happy* from left to right. (Model property *Faceware*©)

1.1 Motivation

The thesis here presented started as a challenge to look into procedural methods for facial animation, as a way of meeting the ever-increasing need of facial animation from interactive applications. To understand why this is a challenge, we first considered the task of endowing a character of facial movements. In an offline case, e.g. a film, there is extensive planning of the expressions that can and should be displayed by a character. Animation is then created for each specific scene, showing exactly the desired intensity and feelings. Doing so can require up to an 1 hour to create 1 second of high quality animation [LA10]. Now consider the case of a crowd in a film, with hundreds of characters. How complex would it be to create unique animations for all the characters? Real-time applications, e.g. videogames, face similar challenges, with the multitude of characters required. This is, however, made worse by allowing the player to just interact with a character directly. Thus, introducing the unpredictability of the person in charge. From which direction will the player interact with the character? Which actions will be chosen? In which sequence? How many times? Unless the player is severely constrained, it is nearly impossible to configure a character to react appropriately to all stimuli and equip it with the subtle and varied motions required to do so.

Having a better knowledge of what happens whenever a character shows an expression allowed us to identify two main components of any facial animation. There is, obviously, the actual motion, which is associated with speech and/or non-verbal communication. But behind such display, there is the process of deciding how the character reacts to what is happening around it or its expressions when doing a specific task. It is the context around

an event and how the character behaves that enable the audience to emphasise with it. So, aside from the animation component, there is the behaviour side. Both parts cannot exist without each other, at least for interactive applications. The importance of behaviour in offline productions is significantly lower, because the scenes are defined à priori, and thus only the details of the animation matter.

On the motion side, facial animation is traditionally produced either manually, by an artist, or driven by the performance of an actor. In the first, the animator configures key poses, when they occur and how their in-betweens vary. This is the ultimate form of control and allows the creation of any type of animation, in any character. On the other hand, it requires significant experience and time to be achieved properly. Performance-driven animation appears as an alternative, where an actor is recorded and his/her motion is mapped to a character. Performance-driven techniques speed-up the creation process at the cost of a complex method to extract and map the motion, special equipment and the actor's performance. Fine-tuning is also common practice to fix any animation errors. High quality results produced with either approach are created on a per-scene basis, which makes it too laborious and costly to create for all the characters required in most productions. What happens is that hero characters are the main focus of animation efforts, leaving a huge number of secondary characters limited to a small array of facial movements. If the viewers see these characters more than once or have more complex interactions with them, the limitations of animations become clear.

Repetitions in what the person sees are, however, not only caused on the animation side. If the behaviour controller always chooses the same animation, it would not really matter if all the subtle and varied motions are available to be played. In videogames, behaviour is traditionally controlled using scripting or behaviour trees. The former relies on rules coded via a language that associates events with actions. The latter describes the rules in the form of a tree, providing a visual structure that decomposes the actions into smaller parts. While behaviour trees tend to provide simpler and more reusable control, both approaches still require extensive discrimination of actions, stimulus and their relationships. More complex behaviour will thus impact the cost and time required to define a character, even more so if it needs to account for the emotional and mind state. As a result, characters that interact with a player constantly will have more complex behaviours, while secondary characters will be limited to a shallower complexity.

These reasons explain why it is so difficult to create facial animation for massive numbers of interactive characters. The same reasons also justify why procedural animation is so interesting. Procedural methods rely in an algorithm to generate the animation from abstract input parameters. They are capable of generating animation much faster and easier than

other methods, without any additional equipment or someone's performance. They have, however, been primarily explored for body animation, with the face still not achieving "good enough" results. This is, by itself, a great opportunity, which becomes even more compelling due to previous research on combining behaviour simulation and animation synthesis. Such decreases the time to create animation and the associated level of expertise, achieving fully autonomous characters whose behaviour is defined à priori and is embedded in a procedural method. The synergies between the two topics also leads to a shift in the control and animation of characters, as the author no longer deals with the specific poses and timing, but instead directs the character, just as the director leads the actor. It is unlikely that procedural methods will drive hero characters any time soon, at least in productions where high quality is required. These methods can, however, significantly reduce the time required to create animation for secondary characters, while also increasing their animation quality.

With this in mind, we set our goal to research and implement novel procedural methods to:

automatically synthesise non-repetitive animation,
on the fly, driven by stimuli.

1.1.1 Challenges

Achieving the goal of procedurally generating animation, driven by stimuli, has required overcoming several challenges:

- *Generation of animation from a reduced number of input parameters*: This is the core goal and challenge of any procedural method, as both poses and their respective variation in time need to be generated. However, the face can produce an extremely wide range of facial behaviours that contain different poses and timing. Techniques should, ideally, be able to represent these motions in a compact form, and allow authoring in a predictable way without the need to specify a huge number of input parameters, nor directly controlling the model deformations;

- *Non-repetitive synthesis of animation from same input*: Which contrasts with the need to be able to predict the animation from a certain input. Still, this is crucial to create believable animation, as a person does not react always in the same exact way. Facial behaviours are very complex and subject to different subtleties that vary with the person and situation. There are studies that focus on specific factors influencing facial

motions, e.g. effects of context in the expression of universal emotions [Mat90]. However, there is, to the best of our knowledge, no study or validated motion DataBase (DB) that contains a wide range of motions (including non-emotion) and respective variations with regard to different people/contexts, which makes research on the subject even more complex. Overcoming this challenge requires an understanding of how one animation is different from another, while still being similar enough as to be generated from the same input;

- *Stimuli-Reaction association*: Without the need to define all pairs of stimulus-reaction, which again leads to the issue of predictability. This requires a fuzzy definition of behaviour that gifts the character with enough emotional complexity to give the illusion of a ”thinking brain”, but still produces reactions that match the role it was designed for. Additionally, human behaviour is far from being completely understood, with facial reactions being only one of many components;

1.2 Solution Overview

To generate stimuli-driven facial animation on-the-fly, we propose a solution that combines the power of motion graphs, for automatic synthesis of short clips of animation, with the simplicity of mind maps, to control a character's behaviour. The method starts (left Fig. 1.2) with a stimulus that is analysed to choose the reaction, i.e. expression. The analysis follows a manually configured mind map that acts as the character's brain. Motion graphs, automatically created from the analysis of a DB, then deal with the actual choice of the animation's poses and timing properties (centre Fig. 1.2). The output of our method is a sequence of landmarks that can then be mapped to different facial model (right Fig. 1.2). An overview of the pieces of this puzzle are shown in Fig. 1.2. We chose motion graphs as the core method due to its configuration process requiring almost no human intervention, which is a common drawback of other procedural facial animation methods. Additionally, its ability to produce results from a reduced amount of training data allows it to cope with another issue of procedural facial motion research, i.e. lack of varied facial movements databases. Our procedural method only generates sequences of 2D or 3D facial landmarks poses (depending on the DB with which it was trained). Mind maps were chosen due to their intuitiveness and lack of exploration within behaviour simulation. These were applied in a way that configuring behaviour does not require extensive discrimination of all possible interactions, while still producing varied behaviour that accounts for the character's mood.

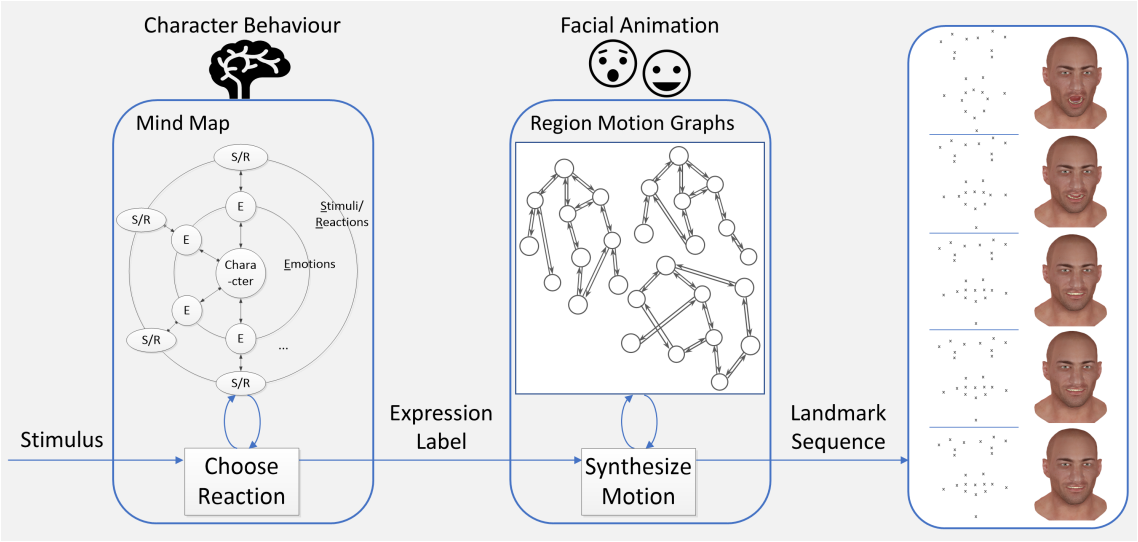

Figure 1.2: Overview of our procedural method in action. As a stimulus occurs, the character's mind map (previously manually configured) determines how to update the character's internal state. Based on this new state, a reaction is chosen and its expression label is given as input for our animation synthesis method. The different region motion graphs (previously automatically created from the analysis of a motion DB) are then traversed to synthesise a short animation, i.e. sequence of landmarks positions, that starts with the character's current expression. The example shows an initial pose of *surprise* and the input expression label of *happy*. While not the goal of this thesis, we show what the landmarks configuration could look like in a 3D model (Model property *Faceware*©).

1.2.1 Motion graph based facial animation

We propose a solution that is inspired by the motion graphs of Kovar et al. [KGP02]. Everything starts with the analysis of a labelled and landmarked facial motion DB that is processed to create an independent motion graph for different facial regions. Each graph is self-contained to avoid leakage, i.e. movement of one region influencing another. The graphs are created by finding, using a Euclidean-based metric, and merging similar poses in DB. The author controls the amount of merging indirectly, by specifying how much the DB data should be compacted, i.e. compression ratio. This considerably eases the process of configuring the motion graph and choosing the amount of information that should be kept, while also enabling a compact representation of the data. Motion is synthesised by using Dijkstra's algorithm [Dij59] to find the path that minimises the similarity between a source and sink/target node. Expression labels, associated to certain database poses, allow finding the relevant target nodes, providing an intuitive control interface with a reduced number of parameters. Information previously stored in each node allows recovering the final animation, which is further smoothed to avoid jittery motion. Synthesis of animation occurs near real-time. We additionally explore the graph structure to generate idle motion,

by using the neighbours of the node associated to the ”currently” displayed expression. This does not require any training data, and avoids moments where the character's face is completely static, which is easily labelled as unnatural. Uniqueness in animation is achieved via a combination of noise in the path and independently calculated region paths. Effectively allowing different animations to be created from the same input, which not only makes the characters more realistic, but also facilitates the use of our system.

1.2.2 Behavioural facial animation via Mind Maps

We have explored the concept of mind maps [Dav11] both as an abstract controller for virtual characters, that works with any kind of body and facial behaviours, and as facial behaviour controller tightly coupled with our motion graph based method. We define a mind map as a hierarchical graph that contains a layer of emotions and a layer of actions, further divided into performed (reaction) or received (stimulus) (as seen in fig. 1.2). The author manually configures the way stimuli and reactions are connected to the emotions, effectively encoding the behaviour of the character. Whenever a stimulus occurs, the character's mood is updated according to the mind map's connections. This update is subject to additional influences, e.g. personality and affection towards other characters. Each time the mood is updated, a reaction is triggered. Mind maps provide an intuitive user interface to add emotional/mental states to the decision process, which in turn leads to stimuli not producing always the same reactions. Mind maps connect with motion graphs by including expression labels in the reaction nodes. This is, nonetheless, an integration that would work with any other procedural facial animation method. Therefore, we further introduce the concept of *insight motions*, which occur as the mood is decaying to a base personality state, after a stimulus happens. This relies on manually specified nodes to trigger small, less intense animations, obtained from subsets of paths normally used in motion synthesis. Combining mind maps with motion graphs enables fully autonomous characters, capable of reacting to different situations in a realistic and unique way. All without having to extensively define pairs of stimulus/reaction. This leads to more believable and engaging characters, whose creation is substantially simplified.

1.3 Contributions

Despite the conceptual advantages of procedural facial animation methods, their use is still limited. They tend to either require significant manual configuration effort or large amounts

of facial motion data. Methods with a heavy configuration process, such as hierarchy based approaches where the user chooses the content and relations of each layer, typically rely strongly on the experience of the author. This results in highly subjective specifications. The lowest level of the hierarchy is also commonly associated with short animations obtained via key-framing or performance-driven techniques, which adds to the complexity and costs. As a result, these approach can have confusing implementations and/or usability. Methods requiring facial motion data, such as statistical-based, are hindered by the lack of high-quality data that relates the actual motion with relevant behaviour descriptors, which could in turn be used as input for the procedural system. So, while hierarchy or rule based methods can lead to the question of why go through all the trouble instead of just creating the animation directly, the statistical or motion-graph methods have barely been explored in the form of easily controllable techniques, mainly due to the lack of available data. Procedural methods have long been influenced by behavioural constraints, however, these tend to be inserted as part of the manual configuration efforts, further increasing the expertise and time to define the system.

Our solution is unique in the sense that we combine behaviour definition with animation synthesis, both reducing the efforts to configure and control the method. Motion graphs have barely been explored for facial animation. Their use, however, allows encoding timing and pose properties with a minimum configuration efforts, hence significantly reducing the set-up time when compared with most constraint or rule-based approaches. Authoring of animation is done directly, via a low number of parameters such as emotion labels. The interaction between animation and behaviour shifts the control, from the specific motions, to the real context of the character and the stimuli it is facing. Hence, we achieve the goal set at the motivation. Behaviour definition does introduce a layer of input; however, it does so in an intuitive way without requiring the definition of all existing interactions. The combination of methods allows us to present present an end-to-end facial animation solution that realises more reactive characters, not limited to always play the same animations. Effectively empowering the author, be him/her a player in a game, an inexperienced animator, a director in a film or even a psychology researcher to drive and affect the character.

The major contributions of this dissertation to the field of computer graphics and entertainment industries are:

- A **motion graphs based approach specifically tuned for facial animation**, which enables synthesis from a reduced amount of input and represents the original motion in a compact format.
- A **novel optimization based approach to create motion graphs** that simplifies the

graph configuration process. Other methods rely on a trial and error approach, where the user specifies an heuristic value to decide when two poses are similar enough to be merged. Our approach optimizes these values as to control the information kept in the graph. This considerably simplifies the fine tuning of a graph, while also leading it to better represent the training data.

- A **new coherent noise method for motion graphs** capable of introducing small variations in motion. This method explores the graph structure to reduce repetitions and synthesise widely varied animations, even with a small number of training samples;

- An **abstract behaviour controller method based on mind maps** that provides an intuitive user interface for behaviour definition, without requiring the specification of all pairs of stimulus-reaction. This method is further explored in combination with the motion graphs to create a **novel behaviour facial animation system** that includes the advantages of both methods in an end-to-end solution;

The work of this thesis has been presented and validated in the following venues:

- Tiago Fernandes, **José Serra**, Juan Õrdoñez and Verónica Orvalho, Mind Maps as Behavior Controllers for Virtual Character, ACM SIGCHI Conference on Human Factors in Computing Systems-CHI, Austin, United States, May 2012.

- **José Serra**, Intelligent Facial Animation, proceedings of the Doctoral Symposium of Foundations of Digital Games, Crete, Greece, May 2013.

- **José Serra** and Pedro Nogueira, Personality Simulation in Interactive Agents Through Emotional Biases, proceedings 27th European Conference on Modelling and Simulation, *Á*lesund, Norway, May 2013.

- **José Serra**, Verónica Orvalho and Darren Cosker, Behavioural facial animation using motion graphs and mind maps, proceedings of the 9th ACM International Conference on Motion in Games, San Francisco, USA, October 2016.

- **José Serra**, Ozan Cetinaslan, Shridhar Ravikumar, Verónica Orvalho and Darren Cosker, Easy Generation of Facial Animation using Motion Graphs, Computer Graphics Forum, 2017.

1.4 Application Domain

The proposed methods have a wide range of applications, being potentially useful whenever a virtual character, that shows its face, is employed. This includes, but is not limited to, entertainment, healthcare, education, enterprise... Entertainment, and more specifically **films** and **videogames** are, nevertheless, our primary target. In these mediums, characters either compensate for a complete absence of non-verbal facial behaviour with more expressive and exaggerated body motion, or are limited to a small number of facial motions constantly repeated (fig. 1.3). Here, we identify two primordial use cases of our methods: direct drive of a character's facial expressions and as the basis for animation, which is then massaged by the animator. The first serves the growing need of secondary characters required to create more believable and thriving worlds, even more so with the advent of virtual reality, where the user extensively interacts with the characters. This need is, however, nearly impossible to address using traditional high-quality facial animation techniques, due to the hundreds or even thousands of characters required and the associated production costs. Our solution pushes the state of art, creating reactive and expressive characters that are obtained at a fraction of the cost and time of existing approaches. However, it yet does not reach the quality expected of leading characters, hence, the second use case. Our animation can provide the basis for blocking, as it already generates the poses and timing, being then used by experienced artists, as a reference to produce the final animation. Again, cutting the production cost, with more or less work involved depending on the desired quality. Our methods can also be used to achieve more compelling and complete previsualizations, as quick and "dirty" animations are easily generated, on the spot, either via stimuli or direct input. Empowering the director with a more in-depth look at what the scene might become.

While the quality was a fundamental criterion in the previous examples, there are application cases which prioritise other aspects, such as cost, time or control ease. Our method is ideal for inexperienced artists that only solve more obvious issues or even use the animation directly, and by **short budgeted productions**, such as independent films, shorts, indie videogames. **Broadcasting** of e.g. news or weather forecast, where the virtual character, driven by a text-to-speech approach, can be enhanced with non-verbal behaviour triggered from what is spoken. In fact, due to the connection between speech and non-verbal facial behaviours, many procedural methods are built around visual speech. However, they are typically so deeply interconnected that it is not possible to use them for non-verbal behaviour synthesis without speech. Our solution could be extended to work on top of speech with the timing of the synthesised short clips massaged to match the words. Thus, providing more believable characters that communicate better with the viewer. The same reasons make the system useful for dialogue-based systems, used e.g. by **enterprise**s to help

customers and lower the workload at e.g. help desk, customer services or online shopping departments. Similarly, virtual assistants are becoming omnipresent and leading the way to a new interaction paradigm. It is a matter of time until they gain a face, at least in devices with an appropriate display. After configured, our method has virtually no costs, empowering users with no background knowledge in computer graphics to synthesise animation and control behaviour of characters. All without asking them to learn how to act in front of a camera. Other application fields include: **psychology**, where easily synthesizing facial behaviours leads to more emotional and believable virtual characters that can help a person to overcome certain phobias, e.g. afraid of walking/present in public, or traumas caused by war; **education**, where it allows creating virtual professors that will relate better with their students in online courses; **medicine** by simulating the patients to train future doctors; or **training** a firefighter to handle a crowd in panic. Such simulations can happen with a static face or only text-to-speech technologies, but empathy and emotions are crucial when creating a realistic environment with other people. The presented solution leads to interactions, training and research experiments that feel more real and credible, helping the subject to become more engaged with the task at hands.

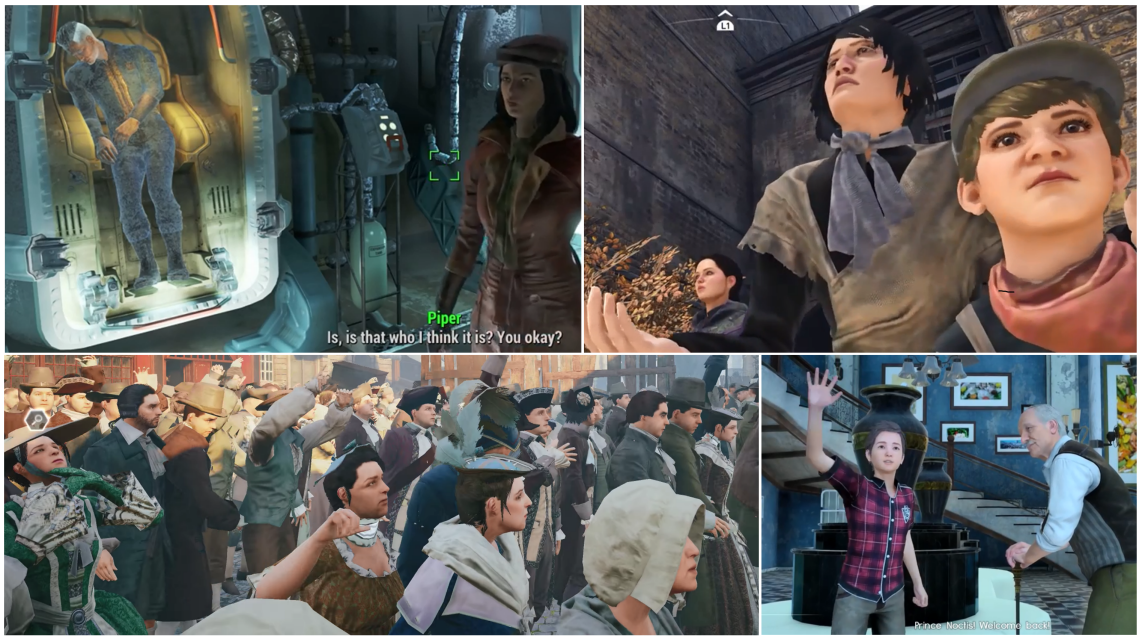

Figure 1.3: Examples of secondary characters in videogames: (top-left) character expressing worry/sadness towards the player after loss (Fallout 4™- 2015); (top-right) mother and son begging for money (Assassin's Creed Syndicate™- 2015); (bottom-left) crowd complaining/protesting (Assassin's Creed Unity™- 2014); (bottom-right) child happy to see player (Final Fantasy XV™- 2016)

1.5 Outline

The remaining chapters of the dissertation are organised as follows:

Chapter 2 starts with an overview of current facial animation techniques: key-framing, performance-driven and procedural methods. Procedural body animation methods are then explored as a source of inspiration and to understand the reason for the disparity between procedural research on body and face. Finally, procedural facial animation methods are grouped into constraint/behaviour, statistical and motion graphs based approaches, being then described in detail. This allows understanding the key challenges present in this field, and provide insights on how to overcome them. The chapter ends with the reasoning as to why we have chosen motion graphs and combined them with behaviour simulation, as a way of achieving the proposed goal.

Chapter 3 describes our solution for synthesis of facial animation using motion graphs. This includes creating the actual graph and using it to synthesise animation. It also presents our efforts on exploring the graph structure to introduce noise and idle movements. We additionally show the state of the art in facial motion databases and how the proposed method was validated quantitatively, qualitatively and with multiple databases.

Chapter 4 presents our behaviour controller based on mind-maps. First, as an abstract method that can be used with any kind of behaviours. It then shows how we take advantage of both the mind map and motion graph structures to create *insight animations* and achieve autonomous virtual characters that react uniquely and on-the-fly to the different stimuli.

Chapter 5 discusses the pros and cons of our work, how it can be improved and its implications to the future of facial animation.

Appendix shows the techniques used to drive the 3D facial model from the landmarks. It then presents unsuccessful attempts to introduce coherent noise on our motion graph based method. Finally, it describes additional work done in the context of behaviour simulation, which has not been combined with facial animation.

Chapter 2

State of the Art

Facial animation plays a crucial role when giving life to characters. The face allows conveying the underlying emotional and mind state of the character, even without verbal or body motion cues, and despite it being just a massive number of polygons. There are three main ways to animate a face model: manually, by an artist; via the performance of an actor; and using an algorithm to synthesise motion from input parameters. This thesis focuses on the last, whose use is somewhat underwhelming, given the significant time reduction that it allows. To understand why such happens, we analyse and compare different techniques within the three animation approaches, which allows us to establish their main pros and cons. While doing so, we explore the fundamental challenges still present in procedural facial animation research and analyse why body methods have not, and cannot at least directly, be applied to facial behaviours. After reading this chapter, you should understand the current techniques used to produce facial animation, the benefits of further researching procedural methods and the many opportunities that still lie ahead.

2.1 Facial Animation Background

The face plays a crucial role in human communication. Facial behaviours are inherent to speech and can compensate the loss of audible information, e.g. in a noisy environment [MM76]. The face is also, arguably, the most visible medium of non-verbal communication, as opposed to body language. Mehrabian [Meh07] found that, when communicating feelings and attitudes, 55% of the message was conveyed via facial expressions, 38% with vocal cues and 7% via verbal cues. Argyle [Arg73] states that the face is the area most closely observed whenever a person is interacting with another. Due to so much influence

in the communication, and while not the sole requirement, facial behaviours contribute significantly to create realistic and believable virtual humans [VGS+06, SSB09, JKK+11, OBP+12]. Non-verbal facial behaviours are controlled consciously and unconsciously by the brain [Leo10], and are subject to a vast array of influences. From a more psychological perspective, the factors include, but are not limited to: culture [VGS+06, Mat06, Lak06] and media [VM06]; personality, emotions and mood [VGS+06]; context, which might include the people present, their rapport, being in a public or private location [VGS+06, Nol06, Lak06]; goals and expectations [Lak06]; sex [Hal06] and age [RSF06]; environment and location [Lak06]. One example of such influences is presented by Frey et al. [FGG+99] that found that males show larger movements compared to females, and that people varying from 60-70 display larger movements than younger ones. As if such factors were not complex enough, the perception also affects in the interpretation of motions [Lak06], with culture influencing the intensity of the perceived expression [Mat06]. A smile can have so many meanings, depending on whom it is addressed, the context, if the person fakes it or not, etc. From a more anatomical perspective, facial behaviours are influenced by the facial structure i.e. skull, muscles, skin, eyes and neck (for more details we forward the reader to [PW08, RAB+14]). These can additionally vary according to age or the person's health. Both, anatomical and psychological, influences lead to an extremely large range of face motions. Birdwhistell [Bir70] estimates that the number of facial behaviours is *20,000*, while Paradiso [Par02] increased it to as vast as *50,000*. The conjugation of all these factors lead to each person having its own set of facial motions and patterns, also known as *idiosyncratic movements* [And12]. A realistic virtual character needs these variations, at least to some extent, otherwise, it will look and feel incomplete.

Non-verbal facial behaviours can be divided into: expressions, gaze and head movements [LWHW12]. All crucial to create believable facial animation, e.g. Tromp et al. [TBS+98] found that a lack of expressive behaviour can hinder the performance and communication with other people in virtual environments. The same occurred in [GSBS01, GSV+03] where a lack of expressive gaze was found to be just as problematic. The lack of any such components can lead to the Uncanny Valley problem [Mor70, HOP+05]. This is a commonly referred issue when dealing with virtual characters that, for *some* reason, do not fit the viewer's expectations. Thus, breaking the person's suspension of disbelief [Slo15]. The character is therefore labelled as unnatural, and can even induce a feeling of disgust. There are many reasons for this, from rendering of the skin, that somehow looks plastic in an hyper-realist context, to modelling, where the proportions of the face might not be correct. On the animation side, issues can arise on the proportions of the actual expressions, stiffness of motion or incoherences such as lack of, or contradicting, signals in different facial

regions, such as mouth or eyebrows. The style (hyper-realistic, stylised...) also influences the motion and the character's behaviours [OBP+12]. Incoherencies, such as having an hyper-realistic character with stylised motions, can also lead to the uncanny valley. In the end, what is clear is that people are experts when dealing with facial expressions [OBP+12], where the smallest anomaly is detected even if the actual point of failure is hard to identify.

Animation is the act of changing certain properties of the object through time, in an attempt to give the viewer, the perception of motion [Par07]. Facial animation is traditionally approached via one, or combination, of three flavours: *key-framing*, also referred as manual animation; *performance-driven*, otherwise known as data-driven or motion capture; and *procedural* also known as behaviour-based [OBP+12]. In *key-framing*, the author first defines the poses and the time when they occur, i.e. key-frames, and then controls what happens between these. Thus, the author discretises the motion and controls all of its the details. Performance-driven approaches start as a continuous input, usually from the performance of an actor, that is analysed and mapped to the motion of the character. This input can be video, audio or originate from alternative interfaces, such a keyboard [PW08]. Instead of an actor, it is possible to use an animation created for one character and map, or alter it, to another. Input for performance-driven methods can be discretised as a way to reduce errors and provide additional control to the artist. Performance-driven animation is more common in visual effects films, where the characters interact with actors and backgrounds, while key-framing tends to be more used in feature films with stylized characters [LAR+14]. In procedural based animation, the motion is generated by an algorithm. The author controls the animation via the method parameters such as walk direction or emotion to be displayed. It is important to note that these parameters do not control specific aspects of the model as occurs in performance-driven method. These flavours of facial animation are explored in more detail in sections: 2.1.2 for key-framing, 2.1.3 for performance-driven and 2.1.4 and 2.2 for procedural animation. While there are alternative, and occasionally overlapping, classifications for facial animation, such as [Par07, PW08, DN08, LAR+14], we believe these do not make a clear distinction between rig techniques and the fact that an animation approach *needs* to provide a temporal variation to the model. Before moving to more details in the different animation techniques, we need to explore the rigging stage of the production pipeline, as it is deeply connected to animation approaches.

2.1.1 Rig

Independently of the animation technique, the actual movements of the facial model are usually controlled via an intermediate layer that sits between the motion source and the

mesh, called *rig*. Creating this structure, i.e. rigging, is commonly compared to the creation of the strings that control a puppet [OBP+12]. The rig allows controlling the model without directly manipulating its vertices, which is particularly important, since controlling each vertex quickly becomes impractical as the models' complexity increases [OBP+12]. While the rig is not the focus of this thesis, we briefly describe it, due to both its importance and connection to research presented in the remaining sections. The rig has already been thoroughly studied and presented in [DN08, PW08, OBP+12, LAR+14], in different flavours: [DN08, PW08] detail various types of rigs; Orvalho et al. [OBP+12] attempt to formally define a rig, providing an overview of the field from rig creation to use and describing its different approaches. Lewis et al. [LAR+14] focuses mainly on the theory and applications of blendshapes. We forward the reader to these surveys for more details on rigging.

The most commonly used rigs in facial animation are blendshapes and bones [OBP+12] (first and second models in Fig. 2.1). In the *blendshapes* approach, several shapes, or targets, are sculpted following a common topology. Each shape represents the offsets of several vertices in regard to a base mesh, usually a neutral pose. New poses are then generated as a linear combination of all shapes [LAR+14], with the animator controlling the weights of each shape. Shapes can affect the whole face or be localised, hence allowing a more detailed control over the final pose. Assuming the model is correctly blendshaped, i.e. there are no boundary issues when activating the different shapes, the main challenge, from a manipulation perspective, is the sheer number of parameters required to control a pose. One of the most widely known examples is Gollum, from Lord of the Rings: The Two Towers (Weta, 2002), that had 675 blendshapes [For03]. This makes animation via blendshapes very time-consuming. Lewis and Anjyo [LA10] refer that, as a ballpark figure, 1 second of animation can require up to 1 hour to create. In the same work [LA10], the authors propose a direct manipulation method, where the animator pins vertices in the model and drags them. The blendshape weights are then minimised to force the mesh vertex, associated to the pin, to be as close as possible from the position it was manually dragged to. This approach has been improved in [ATL12]. Bones follow a hierarchical structure akin to the human skeleton. The rigger binds the skeleton's bones to the vertices of the mesh in a process called skinning. This process is crucial to produce good animations [OBP+12], as each bone will have its distribution map defining how its motion will influence the bounded vertices. Hence, this type of rig is known as *bone*-based, or skeleton-based. As multiple bones can have different influences on the same vertex, a bone-based rig needs to be carefully planned and executed. Nowadays, bone-based rigs in video-games can have between 250 [EMH14] (Ryse: Son of Rome (Crytek, 2013)) and 500 [Pep] bones (Uncharted 4: A Thief's End, (Naughty Dog, 2016)).

The two approaches have different trade-offs. Bone-based rigs have a longer preparation time, but they allow generation of any possible expression. On the other hand, the final pose in blendshaped rig is always limited to the combination of the existing shapes. Blendshapes allow perfect control of the poses, as the shapes are created by hand. Such is much harder to control with bone-based rigs, where each vertex can be influenced by the motion of several bones [OBP+12]. These aspects have led blendshape-based rigs to become predominant in the animation of realistic humanoid characters in the film industry [LAR+14]. However, nothing prevents combining the two types of rigs. For example, in Ryse: Son of Rome (Crytek, 2013) [EMH14], if the character is very close to the player, the rig switches from being purely bone-based to one that has 260 joints and 230 blendshapes. Bones handle the main motions and the shapes correct any issues that arise from bone conflicts, while also increasing realism. This approach achieves the flexibility and smoothness of a bone-based rig with the expressiveness of blendshapes [LCF00, OBP+12].

Both blendshape and bone-based rigs can be enhanced with deformers to provide fine details such as wrinkles. Creating these with the referred rigging techniques would either be too time-consuming, or not possible at all. The deformers can be divided in two groups [OBP+12]: free-form and physically-based methods. Free-form deformation (FFD) was first introduced in character animation in [CHP89] and it can be described with an analogy, i.e. the mesh is inserted in a block of plastic (a lattice); as the plastic is bent or twisted, so is the model [PW08]. The concept of FFDs was extended by Kalra et al. [KMTT92], where each control point can have a specific weight of how it influences the model. Physically based rigs aim to simulate the interactions and visco-elastic properties of the facial components, which include the skin, muscles, fatty tissue and bones, thus mimicking the real behaviour of the human face. Approaches can be grouped into mass-springs and finite elements, and be used per-se or together [OBP+12]. Mass-spring methods appeared in Blader and Platt [PB81], where they were used to model skin and muscles. The poses were created by applying forces to the muscles, that, in turn, would then deform the face mesh. This concept was extended by Waters [Wat87] with different types of muscles, i.e. linear, sheet and sphincter, and by Terzopoulos and Waters [TW90] with a three-layered structure that corresponds to skin, fatty tissue and muscles, all connected with springs. Finite elements methods rely on the approximation of a continuous function, or face, by dividing it into regions whose deformation can be calculated using local functions, usually polynomials. These regions are formulated in a way that combining the individual results approximates the deformation of the whole surface. Examples can be seen in Choe and Ko [CK05], with a 2D linear quasi-static finite elements and in Sifakis et al. [SNF05, SSRMF06] with non-linear finite elements (third model in Fig. 2.1). Muscles are triggered

by sparse facial markers that, in turn, activate the motion of the skin. As placing muscles is a slow and complex manual process, more recent techniques aim to automate their placement [AZ10, CBE+15].

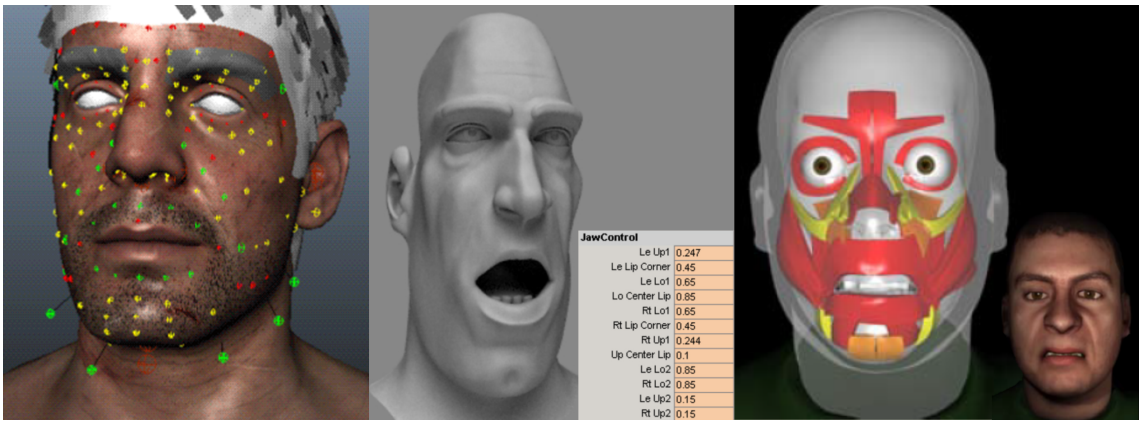

Figure 2.1: Different approaches on rigging. From left to right: Bone-based model [EMH14] with the control joints shown in different (green, yellow, red) colours; blendshaped model and respective shape weights [CPSMS72, OBP+12]; and Finite Element-based muscles model [SNF05].

Controlling a rig does not need to be done directly via shape weights or by moving the bones' joints. Alternatives include: sketching [MAO+11], window based UIs [BSK+07] or via 3D viewport UI [OBP+12], where the manipulation of 3D objects placed in the scene triggers changes in the rig parameters. An example of the last is seen in Alexander et al. [ARL+09] that uses NURBS to control muscle contraction, and whose shape helps hints the direction of muscles contraction. Another approach to control the rig, or influence its conception, is through its parameterization. Parameterizations associate specific configurations of the rig to ad-hoc parameters, which can be controlled or not, via a range of values. They present a high-level interface that can be either used by animators or in performance-driven techniques, e.g. in the film Monster House [Hav06]. Parameterizations are rig and model agnostic, which does not prevent the rig from having to be created in accordance to the parameterization specification. Examples of parameterizations include e.g. Parke [Par74], whose controls were based on empirical studies of the face. Parke [Par82] also proposed to divide the facial parameters into conformation parameters, consisting in the overall face proportions, such as nose length and eye size; and expression parameters, related to the pose itself, e.g. the eyelid opening and mouth expression. Nevertheless, the most common parameterizations are the FACS [EF78] and MPEG-FA [PF03].

The Facial Action Coding System (FACS) [EF78, EFH02] was created to allow the recognition and description of facial expressions. The basic actions are called Action Units (AUs), and each AU cannot be broken into smaller visible parts [EFH02]. FACS includes 46 AUs

(leftmost image of Fig. 2.2), of which 9 are Actions Descriptors (AD), i.e. AUs without an associated muscular action; 9 Gross behaviour codes (GBC), denoting possibly relevant information for facial actions, such as swallow, chewing, shoulder shrug and others; 5 visibility codes, which describe the visibility of certain facial areas like eyes or brows; 14 codes pertaining the head, e.g. head up tilt or forward and finally 11 codes for eyes positions: e.g. eyes up, right or cross-eye, with some having an associated movement, e.g. M69 is associated with the motion of looking to another person. AUs can also be rated in terms of intensity. The FACS additionally allow specifying the time course, which can be split into: onset, i.e. time it takes from the moment the AU is triggered until it reaches the intensity's peak; apex, i.e. the time it stays at its maximum intensity; and lastly, the offset, i.e. the time required for the AUs' muscles to completely relax. FACS provides one of the largest and most accurate reference on facial behaviours [OBP+12]. Due to this, the system has been extensively used in facial rigs (examples include [Wat87, ARL+09, Osi10], with more seen in Orvalho et al. [OBP+12]). Nevertheless, the FACS was not created for animation, e.g. it does not describe all visible and reliably distinguishable actions of the lower part of the face [PW08]. Jaw and lips provide an almost limitless number of actions, which is particularly true for visual speech. Also, not all action units can be combined, as some deal with opposing motions [PW08]. And, while the FACS accounts for different timings in facial motion, they do not provide any information on what those timings should be.

The MPEG-4 Facial Animation (MPEG-FA) described in [PF03] is the first standardised facial parameterization [OBP+12]. Its parameters are divided into facial definition parameters (FDP), defining the shape of the face, and facial animation parameters (FAP), used to create the poses. There are 66 FAPs that control the configurations of the 84 feature points (central and rightmost image of Fig. 2.2) defined in the FDPs. The FAPs are divided in low-level, directly mapped in a face, and high-level, composed by several low-level parameters. The high-level FAPs are the basic expressions and visemes (visual phonemes). The standard also includes the FAPs interpolation table (FIT), which allows determining FAP values that might not have been encoded. The main issues with parameterizations, as pointed in [DN08], is that they often have noticeable motion boundaries. This results from the quest of defining parameters that do not conflict with each other, but in doing so, causes them to affect only very specific facial regions. Another aspect is that, while parameterizations aim to be model and rig agnostic, they often make assumptions on the certain aspects of the mesh topology, and as a result, a completely generic parameterization is nearly impossible.

Creating a rig, be it based on a parameterization or not, is not an easy task. Amongst the several challenges of this process, Orvalho et al. [OBP+12] identifies the following:

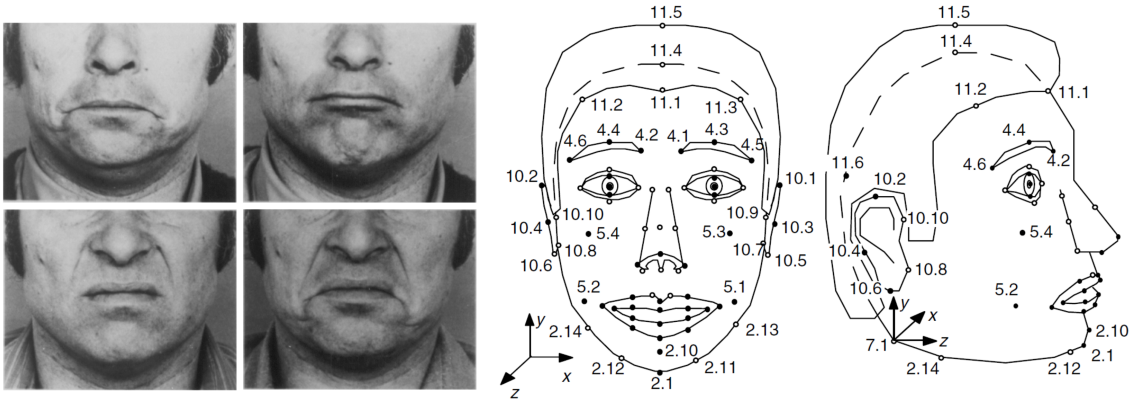

Figure 2.2: Examples of AUs 10, 15 e 17 and their combinations, as displayed in [EFH02] (left). Examples of feature points of MPEG-4 FA, as displayed in [PF03].

complexity of the face that can vary from hyper-realistic to a lamp that talks; people's sensitivity to facial behaviours in realistic characters, where any inconsistency is easily detected; lack of standard on how and which facial rigs should be applied to the different characters; and finally, the ever-increasing complexity of the rig techniques used to achieve more believable results.

Finally, it should be noted that the rig by itself does not produce animation, it is but a tool used to drive the mesh. However, as far as tools go, it is one that is highly connected, and heavily influences, the quality of the produced animation [OBP+12]. A pose not accounted for when creating the rig might be impossible to achieve when producing the animation. Unrealistic and unbelievable expressions can be easily created if the different controls of the rig influence each other in unpredictable ways, e.g. not being addictive when affecting the same part of the face. This, in turn, increases the complexity of manipulating the rig. Not only that, but it might lead to incorrect in-betweens of two poses, defined either manually or via performance capture/procedural animation. Although some of these issues can be addressed with a careful manipulation of the rig, others might not. As a result, as much the animator needs to understand the rig he/she is using, the rigger also needs to understand the kind of motions and behaviours that will be produced.

2.1.2 Key-Framing

Key-framing has its origin in traditional 2D animation, and just like in this medium, it is based on different key poses, or key-frames, that are created and assembled in the desired order by the artist. In fact, rules that make traditional animation more believable and engaging, can and should be applied when animating 3D objects [Las87]. After the poses are placed, the in-betweens need to be specified. This is commonly achieved using interpolation. Here,

a coefficient α, varying from 0 to 1, is used to calculate the current values of the rig, be it the blendshape weights or bone positions. If α is 0 then the model is in pose 1, if it is 1, then the model is in pose 2, otherwise, it is a combination of both. This process is represented by the formula:

$$basicUnitPos = \alpha(basicUnitPos1) + (1-\alpha)basicUnitPos2 \tag{2.1}$$

Parke first demonstrated how to use this kind of interpolation to create facial animation [Par72], where the basic units were the mesh's vertices. He used function 2.1, which defines the *linear* interpolation between the two poses. An example of such interpolation can be seen in Fig. 2.3 However, motion is generally subject to acceleration and deceleration near two extreme poses [Las87]. While it is possible to model such characteristics with linear interpolation, it demands an increase of key-poses. As a result, other interpolation functions have been explored, e.g. cosine functions [PW08]. Kochanek and Bartels [KB84] introduced the use of cubic splines that allow the animator to control the weights' variation using the tension, continuity and bias of the splines. Later, Reeves et al. [ROL90] presented an environment with editable splines to control individual animation parameters, per time unit (e.g. frame). This is still widely used today [OBP+12].

Figure 2.3: Linear Interpolation between poses [Par74].

Interpolation is also not limited to just two poses. Parke [Par74] showed a bilinear interpolation between 4 poses, which required 2 coefficients, and is extensible to more poses. An alternative to directly controlling the interpolation is to leave the in-betweens to the rig being used, such as a muscle system [Wat87]. This will calculate the pose variation for each in-between frame, based on the muscle properties. Key-frame animation has been the technique of choice when all details must be controlled. It has remained fundamentally unchanged since its inception [OBP+12]. While extremely easy to understand and produce animations, this technique depends heavily on the artist's skills. If there are not enough

key-frames, the resulting animation can easily look broken and rough, while also leading to unpredictable and often wrong in-betweens [Las87].

2.1.3 Performance-Driven

Performance-driven facial animation has become almost mandatory nowadays, driven by the need of both digital doubles of actors and characters closely resembling humans. It focuses on capturing and reproducing the actor's motion idiosyncrasies [OBP+12]. In this section, we explore primarily techniques that record the actor's face and analyse it to generate the animation, as opposed to e.g. techniques that use an audio signal [HR10, LYCS11, TMTM12]. Performance-driven techniques can roughly be divided into *image tracking* [OBP+12, LAR+14], where a model is trained and used on 2D video [EBDP96, PSS99, DeC00, CXH03, RHKK11, CWLZ13, GVWT13, SWTC14, FJA+14, CHZ14, CBZB15] and geometric acquisition, which starts with 3D surface data [BPL+03, ZSCS04, DCFN06, BBA+07, LD08, LAGP09, WLVGP09, BHPS10, HCTW11, BHB+11, WBLP11, BWP13, LYYB13, BGY+13, LTO+15, LXC+15, HMYL15]. These approaches can alternatively be grouped according to the input device used to capture the data: monocular camera [EBDP96, PSS99, DeC00, CXH03, RHKK11, CWLZ13, GVWT13, SWTC14, FJA+14, CHZ14, CBZB15]; stereo camera [VWB+12]; camera arrays [BPL+03, BHPS10, BHB+11]; RGB-D sensors such as Kinect [WBLP11, BWP13, LYYB13, HMYL15], which can additionally be combined with a facial strain membrane detector [LTO+15] or with audio [LXC+15]; and structured light setups [ZSCS04, WLVGP09], which can also be combined with stereo cameras [LAGP09]. While these are markerless, the use of markers is also common as it simplifies the acquisition of the facial points that drive the animation [DCFN06, BBA+07, LD08, HCTW11, BGY+13]. The latter will be followed in this section. We additionally forward the reader to the survey of Ye et al. [YZW+13]. This work focuses on motion analysis and, despite not being exclusive to facial animation, provides a very good starting point.

Monocular camera approaches provide the easiest place to start with performance facial motion capture due to the simplicity of the setup. Essa et al. [EBDP96] used dense optical flow, where the pixels' motion controls the muscles of a physically-based facial model. Pighin et al. [PSS99] fit a linear face model to each frame via Levenberg-Marquardt optimisation. DeCarlo and Metaxas [DeC00] added face edges information to optical flow, as constraints, and did the optimisation with a FACS coded face model. This model encodes both motion and shape. Optical flow constraints are also relaxed using a Kalman filter. Chai et al. [CXH03] tracked a small number of facial features, which are converted into

high-quality animation parameters via analysis of a motion capture database and localised regression. Rhee et al. [RHKK11] (top images of Fig. 2.4) separated the face into upper and lower regions in the tracking phase. They then create an example base mesh, where the deformation relies on Radial Basis functions to provide smooth results. They connect these two in an optimisation done in the PCA space. They define their own facial expression space, akin to a parametric approach based on the FACS. Cao et al. [CWLZ13] also used regression, but to train a model that converts 2D positions of landmarks into 3D points, using a webcam as input. In an à priori stage, the user records a set of standard facial expressions, which are used to create a user-specific blendshape model. This model is then combined with the training poses, creating the shape regression model. In Cao et al. [CHZ14], they removed the need for manual calibration by creating a generic regressor from public image datasets. This serves as a starting step, since the regressor is then adapted during the tracking, improving the results as the person uses the system. They optimised for *Displaced Dynamic Expression* (DDE), which incorporates the 2D facial landmarks into the linear model, instead of the blendshape model. Garrido et al. [GVWT13] also created a user-specific blendshape model, which relies on sparse facial features as the animation drivers. However, they extend previous work by reconstructing fine-scale facial detail, such as wrinkles, which is extracted by estimating the unknown lighting and exploiting shading from shape refinement. Adding detail to the blendshape model from motion capture is also approached in [SWTC14, CBZB15]. Shi et al. [SWTC14] used a multilinear face model, instead of a blendshape one, which accounts for changes in expression and in shape. They also remove any manual work from [GVWT13], where the manual correspondence between the face scan and the generic blendshape model needs to be established. Cao et al. [CBZB15] followed another approach to introduce fine detail, using a localised regressor to estimate the mesh changes associated to UV variations that occur in small patches (bottom images of Fig. 2.4). Fyffe et al. [FJA$^+$14] present a general approach that works with any number of cameras and relies heavily on region based dense optical flow fitted to a set of high-quality facial scans. They also abstract the concept of using distinct frames to reset error accumulation, such as the anchor frames [BHB$^+$11]. This is done by creating a performance flow graph, where any frame can be connected to any other, and thus serve to minimise the drift.

Facial performance capture gained considerably with consumer level RGB-D(epth) cameras, such as Microsoft Kinect™or Intel RealSense™. Weise et al. [WBLP11] proposed a method that combines 3D geometry and 2D texture registration with pre-recorded animation priors, in a single optimisation based on maximum à posteriori estimation (MAP). The user is required to perform a small set of expressions to adapt a generic blendshape model.

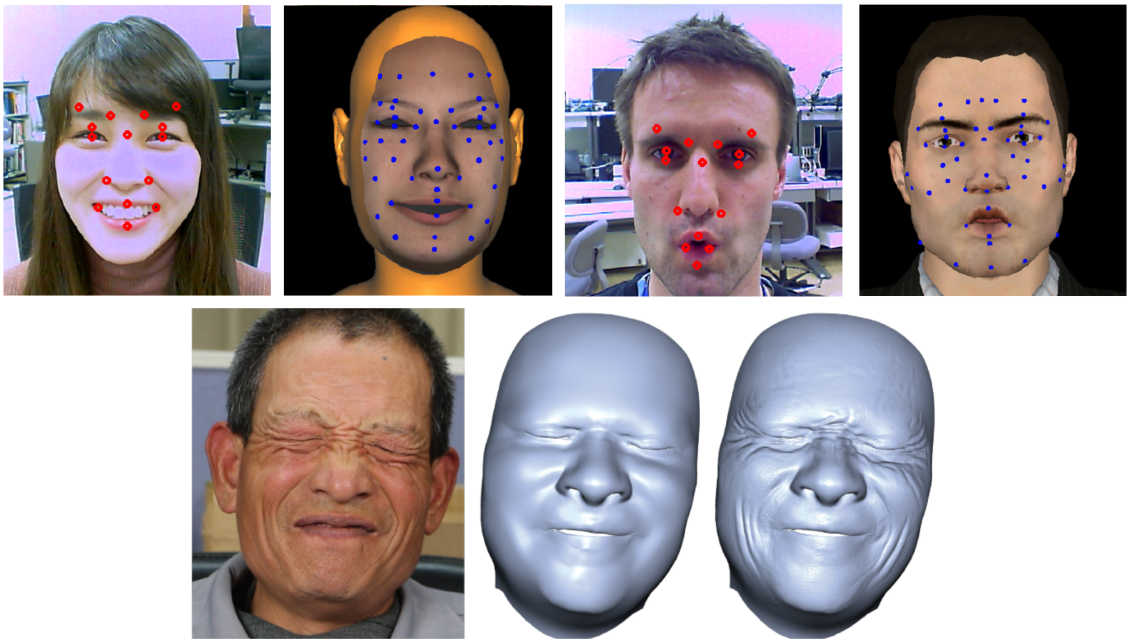

Figure 2.4: Examples of results obtained from monocular cameras. The top shows results from [RHKK11], while the bottom shows the results of [CBZB15].

Results of this method can be seen in Fig. 2.5. Bouaziz et al. [BWP13] improve this method by removing any manual calibration and introducing an adaptive model that combines the blendshape template, an identity PCA model and a parametrised deformation model. These are optimised in runtime, thus adapting to the facial features of the user. Similarly, Li et al. [LYYB13] only require a 3D scan of the subject in the neutral pose, which is then fitted to a blendshape based template. They then track the subject and extract the motion using an adaptive PCA model that contains anchor shapes, used to prevent the model from reaching a bad pose, and corrective shapes, unique to the user, learnt while tracking. Hseih et al. [HMYL15] focus on handling occlusions common in uncontrolled environments, such as the hair falling in front of the face or hand gestures. They achieve this by explicitly segmenting the face into regions and detecting which parts of the face are occluded. They then synthesise the occluded texture. Li et al. [LTO+15] deal with a permanent partial occlusion caused by the use of a Virtual Reality headset. They complement the RGB-D information with a thin flexible membrane placed in the foam of an Oculus Rift™headset that measures surface strain signals from the upper face expressions (bottom images of Fig. 2.5). By combining multiple sources of information, they are able to predict the full facial expression. Liu et al. [LXC+15] combine RGB-D with speech audio, thus being able to better handle occlusions. They train a speaker independent deep neural network acoustic model to extract phoneme state posterior probabilities (PSPP). After which a lip motion regressor refines the 3D mouth shape using the PSPP and expression weights of the 3D

mouth shapes, in a per-frame fit of a multilinear face model.

Figure 2.5: Examples of results obtained from RGB-D cameras. The top shows results from [WBLP11], while the bottom shows the results of [LTO+15]. The latter additionally combines a thin membrane on the VR headset that measures the surface strain signals.

Other researchers have approached the problem with more exquisite setups, mostly out of reach of non-expert consumers. Structured light is one such example, in which case light patterns are projected onto a surface, and the way they deform allows estimating properties of the subject. Zhang et al. [ZSCS04] presented an approach to calculate a single time-varying mesh from synchronised cameras and structured light projectors (top image of Fig. 2.6). They start by fitting a template mesh to the depth maps, optimizing its vertex motion to be consistent with optical flow. Li et al. [LAGP09] require only a rough initial template, from where all high-frequency details are removed. They then align this template to the depth maps of each frame and add the fine details in a separate pass. As this method has an initial template acquisition phase, it works not only with faces but with any other object. Weise et al. [WLVGP09] use a template created from several expressions of the actor. This model is then simplified with PCA and the created parameters are used to register the model to the depth information, on a per frame basis. By simplifying the model, they achieve real-time performance.

Alternatives to structured lights include stereo cameras [VWB+12], camera arrays [BPL+03, BHPS10, BHB+11] and working directly with markers placed in the face of the actor [DCFN06, LD08] whose motion is captured using professional grade motion capture systems like Vicon™. Valgaerts et al. [VWB+12] proposed a system that works under uncon-

trolled and time-varying lighting, using as input only a stereo camera (bottom left image of Fig. 2.6). They track a coarse-detail face template via a scene flow method that uses a Laplacian deformation model as the regularizer of the moving geometry. They then add fine detail, e.g. wrinkles, from estimating the albedo map and other shading information. This is also the precursor work of [GVWT13]. Borshukov et al. [BPL+03] proposed the Universal Capture system that deforms a laser-scanned 3D facial model via optical flow fields extracted using a five-camera array system. Bradley et al. [BHPS10] uses 14 cameras grouped in binocular stereo pairs (bottom right image of Fig. 2.6), to create a time-varying mesh and respective high-resolution texture maps. Each pair of cameras focuses only on small surface patches, which allows the cameras to capture pore level details, used in the reconstruction. Time-coherent meshes are obtained by choosing one to act as a reference and computing the mapping to all frame meshes via optical flow. Drift is detected in the texture domain and corrected in the geometry. Beeler et al. [BHB+11] used an array of 7 cameras to reconstruct a single mesh that exhibits visually realistic pore-level geometry. They introduce the concept of *anchor frames* as a way of automatically dividing a lengthy sequence into smaller parts, where the registration method is applied. Thus, reducing the accumulation of errors associated to registering consecutive meshes.

Figure 2.6: Examples of structured lights [ZSCS04] (top), stereo pair [VWB+12] (bottom left) and 14-camera array [BHPS10] (bottom right).

Until now all papers worked in a *markerless* fashion, i.e. only the natural features of the face are used. This is very complex due to the nature of the skin that is highly deformable and, in most regions, devoid of high contrast features. This leads to an initial set of errors that arise from not accurately identifying, labelling or tracking the skin features. A common practice is then, at least in high-quality productions, to place markers on the actor's face. Independently of being physical objects or directly painted, the markers' goal is to increase the contrast of the tracked points, thus allowing for an easier and more accurate tracking

of the facial movements. The main issues of markers are the tedious step-up and the obtrusiveness to the actor performance. Additionally, if the actor's skin texture is required, a post-processing step is required to recover the skin under the markers.

Deng et al. [DCFN06] and Li et al. [LD08] use a Vicon™system to recover the 3D positions of the markers. After that, Deng et al. [DCFN06] uses radial basis function regression to recover the weights of a blendshap e model, while Li et al. [LD08] analyse a facial motion DB to create an orthogonal blendshape space, using PCA, for different facial regions and respective markers. The sequence is animated by converting the captured landmarks into this space. Bickel et al. [BBA+07] combined an array of cameras to track the painted markers on the person's face (top image of Fig. 2.7) and two extra cameras for tracking medium scale wrinkles, painted previously with a diffuse colour. The landmarks deform a scanned model via a linearized thin shell model and the wrinkles are added to the mesh by minimising a non-linear energy function. Huang et al. [HCTW11] also use a Vicon™system (bottom image of Fig. 2.7), in which case the tracked performance is analysed to obtain a minimal set of poses. Facial scans are captured for these poses and registered to create a high-fidelity blendshape model, with pore-level details. The motion is generated as an optimisation problem with the markers' positions as input. Commercial software exists in the form of markerless and marker-based systems. Examples of markerless include Faceware Analyser and Retargeter [Fac16b], DynamiXYZ grabber/analyzer/bridge [Dyn16] or Faceshift studio [Fac16a]. Marker-based examples include Cara system from Vicon [Vic16] and Expression from Optitrack [Opt16]. An alternative based on paint is provided by Mova Contour [Mov14]. A special phosphorescent makeup is applied directly to the skin in a random pattern. The pattern, only seen with fluorescent fixtures, is then used to triangulate the model and to control the mesh.

Yet, manual tweaking is still required, at least in productions where high-quality motion data is a must. Reasons for this include: accumulation of error in both tracking side and in generated motion parameters, which can lead to jitter; lack of detail in the mouth region, arising from the difficulty of tracking the inner contour of the lips; eyes and eyelids movements also tend to either not be properly tracked or completely ignored, due to their fast nature, i.e. can vary between 20-200ms [PW08] (a notable exception is [BGY+13]); Template used in fitting may lack details associated to certain facial movements, most common case is the lack of a full range of mouth shapes in blendshape models; The actual performance of the actor might require fixing as to make it more expressive [LAR+14]. As stated in Orvalho et al. [OBP+12], it is also important that the actor's face closely resembles the model being animated. Using someone's motion to animate a model with a different topology and morphology can cause the animation to look strange, not convey-

Figure 2.7: Examples of painted markers [BBA+07] (top) and a marker-based (Vicon) set-up [HCTW11] (bottom).

ing the intended behaviours. Whenever such is not fulfilled, the animation needs to be retargeted. Retargeting methods focus on the issue of transferring/retargeting the motion of a source facial model to a target face model. Retargeting is crucial to reduce the work required to drive a 3D model that does not look like the source actor, as seen in films like King Kong or Avatar. It is also relevant to reduce the cost of creating animation in games or other interactive applications, where one animation can be transferred to dozens of characters. Retargeting animations is a field on its own right, with examples of research in [NN01, OZS08, SLS+12, XCLT14]. It is also commonly part of the performance-driven animation pipeline [CXH03, WLVGP09, BGY+13, LYYB13], since it is its most usual application. Amongst retargeting approaches, Noh and Neumann [NN01] coined the term *Expression Cloning* for animation retargeting. Their approach requires a small set of manual correspondences, used to find a dense correspondence via volume morphing and cylindrical projection. This system is optimised with face specific heuristics that help both in the correspondence and in transferring the motion. Orvalho et al. [OZS08] instead of focusing on transferring the vertex motion, directly transferred the rig. Thus, making retargeting a task of applying the same animation controls of the source in the target model. Seol et al. [SLS+12] switched the focus of doing per-frame retargeting into the temporal domain. They formulated the problem as a minimization of Poisson equations constrained by the target face expressions and user-input, such as middle poses. Xu et al. [XCLT14] divides facial performances into large-scale facial deformation and fine-scale motion details.

They decompose the performance into these two aspects and transfer them individually. Additionally, they allow fine detail control by painting the desired region in the model and then controlling the scale of the Laplacian coating. The changes are automatically propagated to the neighbour frames. We have only briefly delved into the field of retargeting, which was necessary given its connection with facial animation. For a more in-depth review of the field, we forward the reader to these papers and to Lewis et al. [LAR+14].

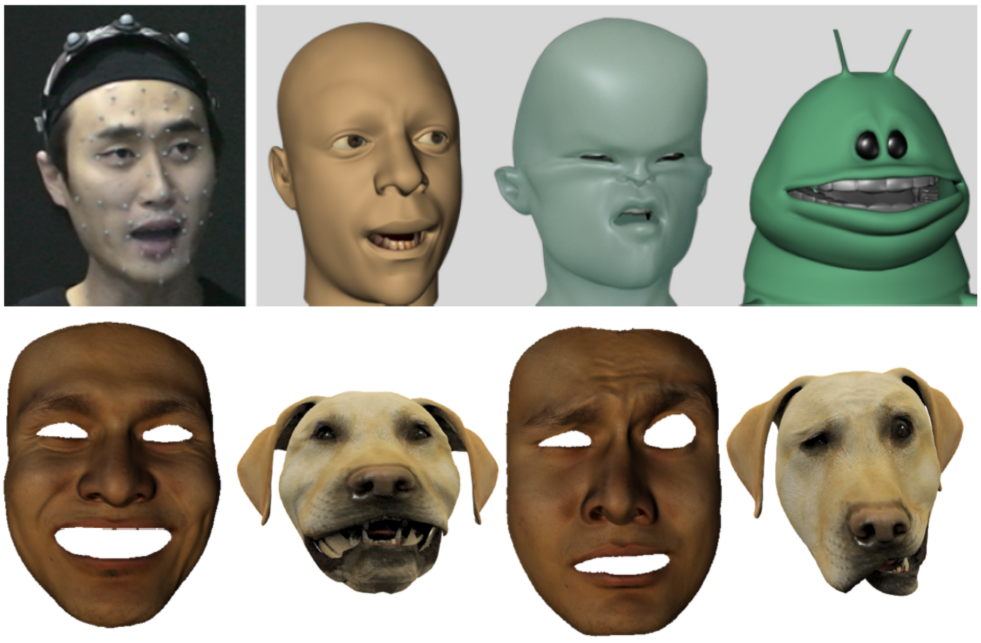

Figure 2.8: Examples of retargeting [SLS+12] (top) and [XCLT14] (bottom).

The quality of facial performance capture has been evolving at a fast pace, and this has, without question, been one of the main drivers of facial animation research. There are, however, other threads in this field. One such thread is the democratisation of facial capture, with more focus on consumer-level hardware. Hence, allowing anyone, outside of costly production environments, to record a performance and apply it to a model. Real-time animation and capture in uncontrolled environments are also gaining traction, with the former allowing instantaneous visualisation of the final animation, therefore, being relevant not only to non-expert users, but also to the director of a film that wants to previsualise what the actor will look like. Coping better with lighting variations and occlusions allows for a wider range of performances, while reducing the constraints on the place where these are captured. Facial behaviours are also composed not only of large-scale deformations but also medium details and fine details. Large scale deformations, such as emotional expressions, determine the overall shape of the face and can already be captured, producing good results. Medium details, such as wrinkles, and fine details, that result from stretching and compressing the skin [HCTW11], have been explored to a far less extent, and only recently

became a more active field of research. Extracting pore-level skin motion is also currently limited to expensive setups. Even in this context, there are still significant challenges when capturing subtle or fast expressions. The same can be said to specific facial regions such as cheeks or the inner lips motions. As a result, consumer-level hardware only magnifies such challenges, which, in turn, means there are still many research opportunities waiting to be explored in the field of facial performance capture.

2.1.4 Procedural Animation

Procedural animation is another word for automatic synthesis of animation. Automatic in the sense that it relies on an algorithm to generate the animation, i.e. a variation in time of certain properties of the object. When dealing with character motion, it is different from the previous approaches because it does not rely on someone's performance, be it captured with cameras, markers, audio or any other form, and it does not require an animator to directly control the key-frames. The algorithm can have mathematical or physical basis, or it can rely on any algorithm in the computer science sense, e.g. artificial intelligence or rule-based. Such algorithm is controlled using a set of abstract parameters, as opposed to directly controlling the transformation/deformation of the model. As Andy Bean refers [Bea12], it is the software that creates the animation; the animator has nothing to do with the motion, he just controls the parameters. The animation author defines an initial set of parameters and then provides discrete or continuous input that will drive the animation. The former can include a list of events and respective parameters/timing, and the latter can be, for example, a curve the character needs to follow. What makes procedural animation unique is that the complexity of controlling the parameters is, and should be, considerably lower than controlling the motion of the character directly. While also being capable of generating a continuous variation of motion. The aforementioned definition is but a more detailed version of the definitions available at [Kit, Par07, Ker09, GS10, VWVBE$^+$10]. Nevertheless, and despite attempting to be a strict definition, it is sometimes hard to group approaches. Examples include a very complex rig with an interface that abstracts low-level rig parameters, or maybe using a base animation that is altered by specifying additional constraints [VWVBE$^+$10]. Would these fall within manual animation? Performance-based animation? Or maybe procedural? We believe the best approach revolves around analysing the input of the method (Fig. 2.9). If the input is directly in the form of rig parameters, requiring both the specification of pose and the timing when such pose occurs, then in all likeliness this is some form of key-framing or manual animation. More or less input can be required depending on the complexity of the rig. If the input is a performance that is

mapped to a character, independently of it starting as an animation, video or audio, then it is performance-driven. Otherwise, it is procedurally-based animation. Even so, very complex rigs can somewhat overlap with procedural methods. The field of procedural animation includes not only body and facial animation, but also particle systems or flock animation [Kit, Ker09].

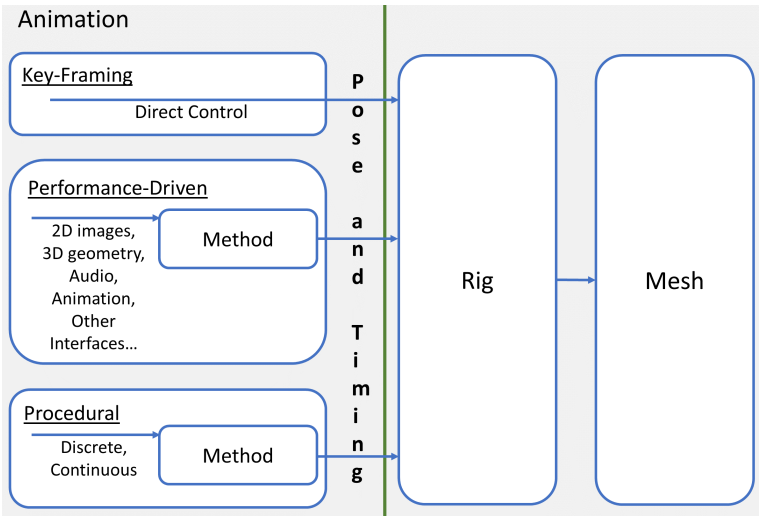

Figure 2.9: Separation of character animation techniques based on the possible input used.

The core field of this thesis is procedural facial animation, which is explored in detail in the following sections. However, to better understand how procedural methods fare against manual and performance-driven techniques we now present an overview of the core ideas behind automatic facial motion synthesis. Most methods fall within motion blending (Sec. D, 2.2.1, 2.2.2), where two or more animations are combined to create a new one. Within these, we can identify *constraint/rule* based approaches (Sec. 2.2.1) and *motion graphs* (Sec. 2.2.2). The first requires the author to define a series of rules that characterise the input and control how the animations are blended, e.g. in scripting form. The new motion, triggered from the input, results from blending previously created motions. Blending can refer to either holistic or region-based interpolation of motions, or to the concatenation of animations. Motion graphs explore the use of graphs capable of encoding motion data in the nodes. Synthesis happens by navigating the graph, with each node contributing, in some part, to the final motion. The information contained on a node ranges from a single pose to full motions, concatenated in the optimal merge point. Another group of approaches relies on *statistical* methods (Sec. 2.2.2) to learn the underlying model of the motion and use it to synthesise new motion. The main difference between these categories lies in their set-up. Constraint/rule-based systems rely heavily on manual input, while motion graphs and statistical-based approaches trade this work by data, required to respectively create the

graph or the model. The configuration is much faster, but obtaining the data comes with a cost (more details in Sec. 2.2.4).

Procedural animation biggest issue is the quality of the results, as it still requires considerable tweaking after the animation is generated. Animations often look too stiff and repetitive, with a robotic feel, which transports them directly into uncanny valley. While such might be acceptable with secondary characters, as the ones seen in a crowd, it poses too much of a constraint in main characters. Causes for this include the method or the data being used. The control parameters can also originate some problems, arising from being too unpredictable or restrictive. It is important to note that parameters are associated to behaviours and not specific details of the motion, which makes them very complex to define. A universal facial behaviour characterization would significantly ease the definition of parameters, however such is yet to be created. This, in turn, makes procedural systems highly subjective and can lead to unintuitive experiences.

2.1.5 Comparing animation approaches

Manual and performance-driven methods are the *de facto* approaches used to animate faces, especially in the entertainment industry. Manual animation or ***key-framing*** is the most *time-consuming* of the three. It provides the ultimate form of control, where the animator controls all the bits and bobs of the character's motion. The results are highly influenced by the expertise of the person and time invested. This technique can be used in any character and produce amazing animations that truly transpire the character's being and mind. However, if not properly done the results easily fall short. ***Performance-driven*** animation provides the best results when dealing with realistic, "person-like" characters, due to the nature of the motions' source. Different issues arise here as the current methods still do not perfectly capture all the idiosyncrasies of the actor's performance. In turn, this leads to cleaning and tweaking afterwards. Secondly, characters that do not resemble the actor ask for retargeting, adding a layer of post-processing. While it is true that tweaking is less than ideal, it also allows the performance of the actor to be improved or changed, without having to record it again from scratch. When animating person-like avatars, performance-driven methods is preferred over manual animation as it considerably shortcuts the production time. Reduction in time does come with an associated cost of additional equipment, research and an actor's performance. Key-framing is still preferred over performance-driven animation when dealing with characters that have a morphology completely different from a human, or when dealing with non-realistic, more cartoonish characters. ***Procedural*** facial animation falls between key-framing and performance-based methods in terms of production time.

These are capable of producing animation in the shortest time, however, as the quality of the results pales behind performance-driven techniques, more time is spent on fine-tuning them. Amongst the three, it also provides the easiest way of generating animation, with the author specifying only the control parameters, and avoiding the need for special equipment or human performance. This also makes it suitable for generation of animation on-the-fly, such as in video-games. Procedural methods rely on an interesting proposition, as any generated animation is directly connected to the one it was configured with. What this means is that, if it is trained with data similar to the one used in performance capture methods, it will face the same retargeting challenge. However, it can also be configured to work with non-realistic motion, hence potentially generating animation that would not be possible with performance-based methods. This data would, however, have to be manually animated, which clashes against manual animation, leading to the question of why create animation for a new method instead of just creating the desired motion directly. Another issue is that the data used, e.g. in statistical-based procedural methods, is more complex to obtain than the one used in performance capture, due to its need for both motion and the connection between motion and control parameters.

Procedural facial animation research shifts the way animation is created to directing the character, akin to directing the person in performance-based methods. It offers, arguably, the best relationship between cost, ease of use and quality. The latter is, however, a core desire when choosing the approach to animate characters and, in this sense, manual animation and performance-driven methods still provide better results. There are many reasons for this, ranging from difficulty in adapting the methods and configuring them, availability of facial data and the need to have a multidisciplinary approach. We have already introduced some of these aspects, but will continue to explore them in the following sections and chapters. The reality is that research has focused primarily on performance-based methods leaving procedural animation seldom explored. When this field reaches the same level of maturity as performance-based animation, it will allow creating animation in a significantly shorter time than any other approaches, while also easing the process of adding facial animation to different mediums.

2.1.6 Importance of Procedural Body Animation

Body and face are many times two sides of the same coin. Body animation has long influenced facial animation [OBP+12], and thus it is important to study existing procedural body motion methods, as they can, and did, serve as a basis for methods researched in this thesis. We show our efforts in App. D, which first provides an overview of the

field and then focuses on the domains we believed could provide more hints and lay the groundwork required advance the field of procedural facial animation. There are, however, several reasons as to why the methods cannot be ”ported” directly to the face domain:

- *Facial movements are more complex than body movements* [OBP+12]: Both are temporally non-linear, however, facial movements are also non-linear in terms of shape/proportions. This is similar to having a body motion whose sizes of arms and legs constantly change. Body motion has an underlying structure, the face has not, which results from body keeping its rigid proportions. Nevertheless, both share the issues of rigid alignment: the face due to head movements, and the body due to the different orientations in which the motions can occur;
- *Basic body motions provide more ”useful” animations than the equivalent facial motions*: We refer to basic motions as the motions whose influences of the current emotional and mind state are very subtle and hard to identify, even with a trained eye. Examples of basic body motion include locomotion or grabbing an object. Although these may be enhanced with emotional cues e.g. having lighter steps in a happy walk, they also exist without almost any such cues. These tend to be associated with physical and physiological actions. Equivalent facial motion includes for example: idle motion, resulting from small neck variations and breathing changes, eating, speech, blinking or eyes' motion. Again, these can be emotionally enhanced, but they exist purely. Speech has long been studied, however, we consider it fits within performance-driven animation, and as such is not explored here. The main difference between basic facial and body motions, in terms of ”usefulness”, is that the latter encompasses a considerable amount of the required interactions used in a game or film. Basic facial behaviours are essential to create realistic characters, and their absence is noticeable, however, if a character only has these it quickly becomes dull. Thus, while basic body motions are sufficient to animate a considerable chunk of a character's motions, basic facial motions are not;
- *Basic body motions are well understood and can be described/controlled with objective functions*: Body motion synthesis is fundamentally based on different kinds of optimisation functions and controllers. Most functions minimise errors determined by: following a path in the ground, body orientations, end effector positions, or other inputs that are mapped to aspects in the environment. Additionally, constraints, provided by the feet contacts and type of motion, allow for long motions to be generated. However, these functions can seldom be used for generation of facial movements (aside for example from smoothness or pose constraints). The same happens with the definition of controller, where input needs to be mapped to the different functions.

Basic body animation controllers have long been explored, which is not the case for facial animation. On the other hand, non-basic facial and body animation controllers have been scarcely explored, but it just so happens that for facial movements, they are an extensive requirement. While for body, they can be avoided to a certain extent;

- *Cyclic motions are more common in body than in face*: Some body motions are composed of small cycles constantly repeated, which allow creating long sequences after dealing with the respective transitions. Non-verbal facial behaviours and, for that matter, non-basic body behaviours are usually short animations that occur as a result of an internal or external event. These events depend on the situation and are subject to conscious and unconscious decisions. They, however, do not necessarily fit in a cycle, and cannot be repeated too much at the risk of becoming dull and leading to the uncanny valley problem;

- *Environment physical interactions are crucial for body synthesis*: The face has consistently presented harder challenges until now, however, facial physical interactions only represent a small part of all the facial behaviours. They occur as the result of some impact, or activities like chewing, and tend to be handled with a physically based rig, Sec. 2.1.1. Due to this, there has been hardly any study on physical facial animation from a procedural perspective. On the other hand, environment interactions are crucial for body animation and must be accounted for. As an example, most methods account for some kind of feet constraints to prevent sliding. Nonetheless, more complex and unexpected interactions are still quite an unexplored area;

These reasons make it hard for many techniques to cross the boundary and reach the facial animation. Some approaches have successfully achieved this, however, they are scarce (see following section). Thus, researching how they can be adapted and extended is, in its own right, an amazing opportunity. Based on the previous list, we believe the approaches that have less environmental constraints are more likely to applicable in a ”direct-er” form. The same applies for techniques whose controllers require input in the form of semantic properties or pose constraints. It then becomes a matter of choosing the level of input asked of the author.

2.2 Procedural Facial Animation

When exploring techniques that generate facial animation, we first need to choose where to focus. Fully featured facial motion, or behaviours, include the actual expression, which can

be coupled, or not, with speech, eye movements/gaze and head motion. Within this thesis, we have focused on animation of *non-verbal expressions*. Gaze, and associated head motion, have been thoroughly studied and already achieve realistic results (as seen in [RAB+14]). Visual speech animation falls within performance-driven animation, with most techniques relying on the continuous analysis of video or audio. As a result, speech synthesis is not explored here. There are, nevertheless, many procedural animation techniques that revolve around making speech more expressive and emotional, which arises from the deep interconnection between both. Inevitably, we also focus on these as their underlying concepts might be, with more or less effort, extrapolated to standalone non-verbal facial animation [PBS91, CPB+94, CVB01, KMT02, PB03]. Similarly, techniques that generate only poses but lack timing information [BHPN01, RNTH03, APM07, XMLD07, SAHM08] have also been studied. These represent an initial step towards complete facial behaviours and, thus are considered relevant.

Most procedural facial animation approaches fall within parametric motion blending (more details on the term in App. D and D.3.2), although statistical-based methods and motion graphs have been explored to some extent. Parametric motion blending techniques, also referred to as constraint and rule-based due to those being the most common core concept, are presented in Sec. 2.2.1. We have additionally identified a sub-group of these methods that focus on modelling cognitive/emotional processes. These, in turn, drive the choice and characteristics of facial behaviours, henceforth referred as *behavioural*-based methods, Sec. 2.2.1.1. We then explore *statistical* and *motion graph*-based approaches in Sec. 2.2.2. These are considerably more abundant when dealing with body animation synthesis, hence we forward the reader to App. D.3.3 and D.3.1 for more examples. *Hybrid* techniques have also been studied (Sec. 2.2.3). We end the section with a comparison of the pros and cons associated to each group of approaches (Sec. 2.2.4).

Existing literature reviews on this field are also scarce, and in most cases, procedural facial animation is only briefly referred [PW08, OBP+12]. Nevertheless, two recent surveys [HNMP10, ONP14] have appeared, and go into more detail in constraint and behaviour-based approaches. Both focus on the process of adding expressive capabilities to embodied conversational agents (ECA), however, the first is more general and includes research on body animation, while the second deals only with the face. We forward the reader to these surveys for a more detailed review on these approaches.

2.2.1 Parametric Motion: Constraint/Rule-based

Constraint-based approaches explore empirical and practical knowledge that results from experience, often based on the needs of the application. Cognitive sciences like psychology, linguistics or even neuroscience are also common sources of knowledge when designing such approaches. This group of approaches is, to the best of our knowledge, the ones first explored for synthesis of facial animation. We have chosen to separate the methods depending on them being speech [PBS91, CPB+94, CVB01, KMT02, PB03] or non-speech related [BHPN01, RNTH03, NBPZ05, GRVT+06, APM07, XMLD07, MBXL08, NHP09, QCM10, RTR+11, DiP13]. Alternative classifications follow other characteristics, e.g. the use or extension of scripting languages to control [CVB01, KMT02, PB03, NBPZ05, GRVT+06, XMLD07, MBXL08, QCM10] or configure behaviours [CVB01, GRVT+06, NHP09, DiP13]. Other approaches instead rely purely on ad-hoc input formats, such as labels [PBS91, CPB+94, BHPN01, NHP09, RTR+11, DiP13] or points in some abstract space [RNTH03, APM07]. Another alternative is based on how the system goes from the input to facial behaviours: is it based on rules that analyse a certain input and choose the pose or animation [PBS91] to play, which allow even multiple rules to activate at the same time, or is it the result of directly specifying the input for a function or hierarchy that calculates the final pose [BHPN01, KMT02, PB03, RNTH03, NBPZ05, GRVT+06, APM07, XMLD07, MBXL08, RTR+11, DiP13]. These different aspects are not mutually exclusive [CPB+94, CVB01, NHP09, QCM10].

One sub-group of techniques has emerged while reviewing constraint-based approaches [KMT02, GRVT+06, XMLD07, MBXL08]. It focuses on modelling cognitive/emotional processes that, in turn, drive facial motion. The main reasoning for placing an approach in this sub-group, as opposed to the more general one, is that the latter has a more direct, or empirically-based, association between the input and facial motion. While the former includes additional parameters related to the character's internal state. An example of the first is the approach of Bui et al. [BHPN01]. The authors control an emotional intensity vector used as input for a fuzzy rule-based system that determines the facial expression and respective intensity. An example of *behaviour*-based approaches is Kshirsagar et al. [KMT02] that model the personality, mood and the emotion felt after a stimulus, to choose the output expression weights. In this sense, the first has a more "hard-wired" association between input and expressions, while the second adds a layer, with origins on psychological studies, into procedural facial animation.

Amongst the approaches that enhance visual speech, we forward the reader to:

- Pelachaud et al. [PBS91] present a two layered approach that connects the expres-

sions, of the 6 basic emotions (expression layer), to a muscle-based facial model, parametrised with the AUs (physical layer). They first determine how each emotional expression triggers the AUs on both upper and lower regions of the face. Several conversational facial behaviours are then described in terms of AUs, and grouped into several categories, e.g. punctuators, i.e. movements that occur in speech pauses and emotional emblems such as wrinkling nose when talking about something disgusting. Finally, several rules for triggering these behaviours are created and associated to intonation properties, such as pitch or tempo. Audio analysis provides the timing and place for triggering behaviours. Emotional expressions and speech are combined by adding the AUs following the rules. To avoid conflicts, the rules are subject to different priorities, established via a rule hierarchy. Time variation of weights is subject to rules to avoid abrupt changes;

- Cassel et al. [CPB+94] extends this approach with a dictionary that associates lexicographical content with different gestures, allowing the character to choose the utterances based on his goals. The behaviour rules are partially implemented using Parallel Transition Networks (PaT-Nets), i.e. finite state automata that encode coordination rules. Different types of motion, such as head or gaze, are encoded in individual PaT-Nets, and interact with each other to avoid conflicts. The changes between the states can be deterministic or probabilistic, based on the configuration. Facial expressions are driven by intonation rules, while conversational behaviours are manually specified in the text associated to speech;

- Later on, Cassel et al. [CVB01] presented a speech-driven body and face animation framework, i.e. BEAT. It uses an extensible knowledge DB written in an XML-based language. This DB contains rules based on non-verbal human research, which associate contextual and linguistic cues to facial and body gestures. The user writes the text, and the framework will automatically suggest several behaviours for different parts. These suggestions are defined in a tree-like structure (left image of Fig. 2.10), which includes details such as priority (to sort conflicts) and the degrees of freedom. Alternatively, it is possible to associate behaviour labels to parts of the text, with the system taking control afterwards. Speech provides the timing when the poses should be interpolated, with the duration being based on rules;

- Pelachaud and Bilvi [PB03] proposed the use of the Affective Presentation Markup Language (APML) [CCP02] to control the generation of speech and facial animation. This scripting-based approach requires specifying the text and accompanying high-level tags (behaviour related). Speech is synchronised at word level with the facial expressions, which follow previously defined onset, apex and offset values. Expression

conflicts are solved using a manually configured belief network that takes the tags as input, and calculates the probabilities of facial parts being activated. The framework outputs animation in the MPEG-4 FA format, following a previously created association between the tags and FAPs. This relation is established in two stages: FAPs are associated to Facial Basis, akin to facial parts motion, e.g. raise_eyebrow_left, and these are combined to create Facial Displays, akin to expressions.

Speech input helps to solve timing and duration issues as the expressions will be coordinated around the utterances. It also allows extracting more non-verbal information, e.g. a faster, highly pitched speech can be associated to more intense expressions. However, the use of speech gives birth to conflicts between full facial expressions, e.g. happy, and the visual counterpart of speech. Synchronisation between both is crucial to achieve believable animation. The common approach to solve these is to identify possible issues and enforce rules that prevent or resolve their appearance.

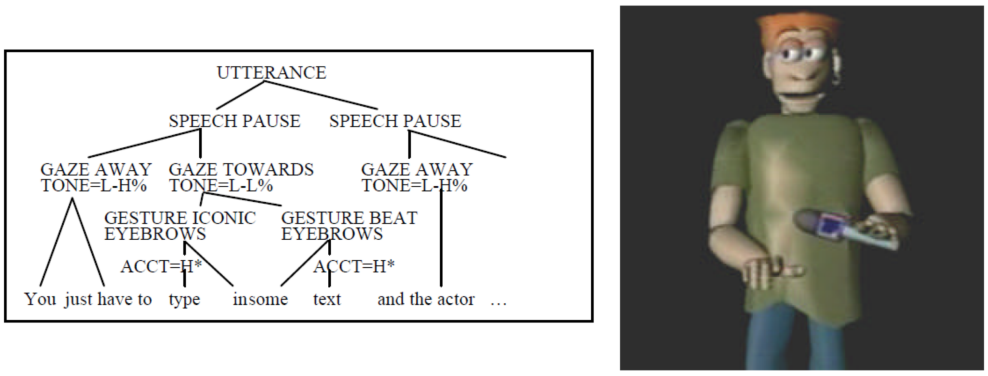

Figure 2.10: Example output of the tree structure used to control the motion of the model described in [CVB01], synced with the text below (left). Pose from the animation associated with the tree (right).

Amongst the approaches not revolving around speech, we highlight the following:

- Bui et al. [BHPN01] present a system that uses an emotion intensity vector and fuzzy rules to determine the activation intensities of a muscle-based system. Fuzzy rules are defined for single and ”blends” facial expressions. Single expression rules affect the whole face, while blend rules can lead to different intensities in separate facial regions. The implemented rules are based on Ekman’s description of universal emotions, animation principles (described in an early version of [PW08]) and on facial behaviour observation. Both the emotion and the muscle contractions are described using 5 fuzzy sets (Very low/small, low/small, medium, high/big, very high/big respectively);

- Ruttkay et al. [RNTH03] place the 6 basic expressions in a circle (left image in Fig. 2.11) with the neutral expression in the centre. New expressions are calculated by selecting a point and using bilinear interpolation based on closest expressions;
- Not et al. [NBPZ05] propose an approach where the author specifies the active communicative channels, e.g. face, eyes or speech, and their respective performing abilities, e.g. expressions or speech animation. This makes the approach independent of the model. For example, it allows the case where, if the model does not speak then, there is no need include, implement or create speech related animation. To this end, they propose a scripting language entitled SMIL-AGENT (an extension of the SMIL language [W3Cb]). It relies on specific tags to describe what happens on each channel. The author can either control the poses/animation at a given moment (high-level) or the individual pose components/intensities relevant to each channel (lower level). The tags have been previously, and manually, associated to animations, speech and other channel components;
- Arya et al. [APM07] first determine the expression units, i.e. the AUs that appear in Ekman's 6 basic expressions. They then associate the units to points at Russel's wheel [Rus91]. The author provides a point in this 2D space, which creates a new expression by blending the closest expression units;
- Niewiadomski et al. [NHP09] proposed a scripting language to describe behavioural and temporal constraints. The former controls properties such as duration, probability of a behaviour happening or repeatability. Temporal constraints characterise the relationships between behaviours, e.g. a behaviour can only occur after another, or not, or which signals can co-occur. They analysed several motions to extract behaviours and temporal constraints, while also noting the respective AUs. The animation is controlled by picking a label(s) from the DB and a total duration. Facial behaviours are then chosen based on the rules, probabilities of occurrence, and then combined following temporal rules (right sequence of images in Fig. 2.11);
- Queiroz et al. [QCM10] proposed the Face Description Language to control expressions, synchronised speech and eye's behaviours. Their framework relies, at the lowest level, on the use of MPEG-4 FA to define the poses. Each facial behaviour component has its own engine that generates FAPs. Rules are employed to solve conflicts arising from different engines that deal with the same facial regions. Facial expressions use predefined poses that are directly controlled in the scripting language, which are also coupled with the timing;
- DiPaola [DiP13] propose an abstract and hierarchical framework based on the association of numerical values, or range, to specific aspects of facial behaviours. Examples

of low-level parameters include forehead height or jaw width, and high-level parameters include exaggeration or emotional expressions. This hierarchical structure allows describing identity, pose and timing properties. The relationship between high-level and low-level parameters is manually established, and can be one-to-one or one-to-many. Authoring can be done at the different levels, thus being friendly for both novice and experienced users.

Finally, a simple, yet relevant approach is presented by Roesch et al. [RTR+11]. This is based on a direct mapping between AUs and a FaceGen [Inv] generated model. To generate the animation, the author provides the activation curves for the desired AUs. Despite its simplicity, this is an excellent example of where a technique from computer graphics can be, and is, useful in other fields such as psychology.

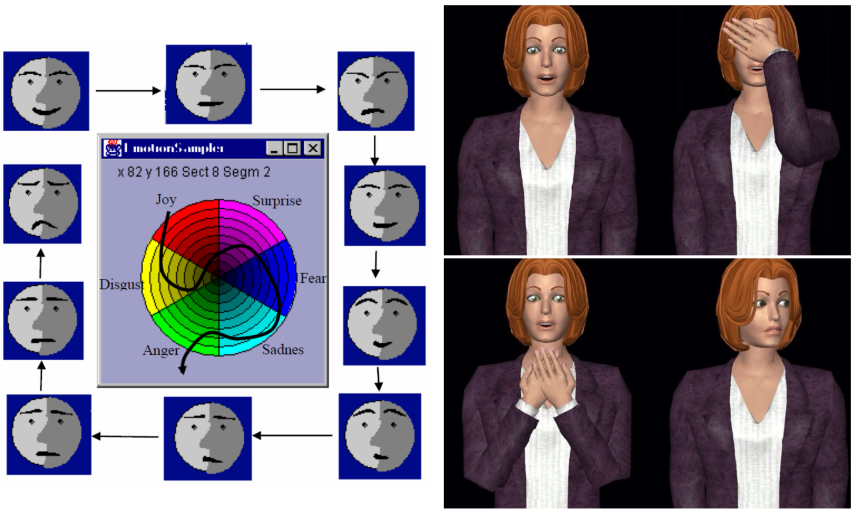

Figure 2.11: Emotion disk for a cartoon face [RNTH03], where selecting a point produces a new face that results from a bilinear interpolation of close, previously placed faces (left). Multimodal expression synthesis based on the annotation of panic fear [NHP09] (right).

2.2.1.1 Behaviour based

Behaviour based approaches leverage the knowledge from cognitive sciences and apply it in a practical form to drive facial, and some times body, animation. Traditionally, these use some kind hierarchy [XMLD07, MBXL08], where the author can provide input on several levels or just the highest one [GRVT+06]. Some approaches also store internal variables that represent the current emotional or cognitive state [KMT02], influencing the choice of behaviour. In the present section, we focus only on pure behaviour-based approaches, leaving hybrid techniques to Sec. 2.2.3:

- Kshirsagar and Magnenat-Thalmann [KMT02] propose an approach that accounts for personality, mood and emotions when choosing facial behaviours related to speech. Personality is modelled using a Bayesian Belief Network (BBN), whose weights are manually introduced based on ”intuition” and guidelines from the Five Factor Model [MJ92]. This layer models the uncertainty usually associated with human behaviour. The BBN receives as input the current mood, multiple emotional tags and respective probabilities. The last two are manually associated with utterances via scripting. The BBN outputs a sequence of emotion tags and respective probabilities, which are used to update the mood and decide the facial expression. Moods and emotions are based on the OCC model [OCC88], while the scripting language extends the Artificial Intelligence Markup Language (AIML) [AIM17] (XML-based). They only map the output to the 6 basic emotional expressions, whose choice follows rules over the different output probabilities. If the top values are close, the final expression will be a blend. Facial behaviours are synchronised on an utterance level, and include à priori manually defined onsets, apexes and offsets;

- Garcia-Rojas et al. [GRVT$^+$06] propose an ontology created using Ontology Web Language (OWL) [W3Ca] that associates the semantics of facial behaviours to MPEG-4 FA parameters. Their approach is based on 25 archetypal expressions, i.e. variations of the 6 basic emotions, which have been described in terms of FAPs and respective range of motions. These archetypal expressions are represented as points at Whissel's [Whi] activation-evaluation 2D space. Intermediate expressions are then created based on the proximity of a point to the archetypal expressions and on rules on merging FAPs and respective motion ranges. The author controls expressions by querying the ontology e.g. the FAPs associated to certain expressions;

- Xue and colleagues [XMLD07, MBXL08] present a different hierarchical approach where all layers contribute to the final expression. This approach models behaviour using social, emotional and physiological layers (Fig. 2.12), with each having a separate level of influence in the final expression. Input is provided for each layer, e.g. emotion felt and intensity in one and pain level in another. Fuzzy rules are applied on the emotion layer, which is based on OCC [OCC88]. An emotion can be associated to several expressions in the same way that an expression can be associated to several emotions. The emotional input vector is analysed to choose both the resulting emotional expression and its intensity. The relationships between layers are manually configured à priori, e.g. the social layer parameters can lead to an intensification, or hiding, of an expression obtained from the emotional layer;

```
<seq>
  <Social priority="0.1">
    <rest intensity="0"/>
  </Social>
  <Emotion priority="0.5" mood="negative" negative_weight="1">
    <anger intensity="0.5"/>
  </Emotion>
  <Physiological priority="0">
  </Physiological>
</seq>
```

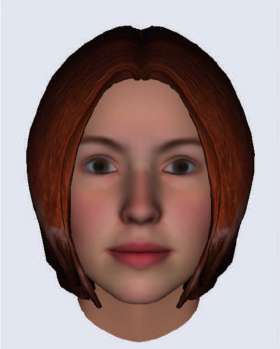

Figure 2.12: Example of layered input required by [XMLD07, MBXL08] to generate new expressions.

2.2.2 Statistical & Motion Graphs based Approaches

Motion graphs and statistical-based approaches shift the effort of manually configuring and implementing the system, to the choice and fine-tune of the method used to analyse motion data. However, the lack of facial motion data has hindered, and still does, research on these approaches. This topic is explored in more detail in the Sec. 3.3.1. Purely statistical-based [KMMT01, RNTH03, SAHM08] and motion graph [ZSCS04] techniques are presented in this section, with hybrid approaches left to Sec. 2.2.3.

Kshirsagar et al. [KMMT01] start by capturing speech, visemes and 6 basic expressions, and using them to create a PCA space (axis of this space show in left image of Fig. 2.13). Speech animation occurs by getting the viseme coordinates and interpolating between them using a cubic spline. Blending emotional expressions with visemes is done by adding the PCA vectors associated with both components. Similar to previous approaches, this method is mainly driven by speech. However, representing visemes/emotional expressions in the PCA space and blending them there can be explored within procedural facial animation. As for the input, the timing and emotional labels are manually specified in the speech animation sequence. Ruttkay et al. [RNTH03] also rely on PCA to define two squares, i.e. emotional squares, whose axis are associated to the first 4 PCA vectors. These were obtained from 59 artistically created facial expressions that were mapped to 15 FAPs. Choosing a point in each square leads to the creation of a new expression. Susskind et al. [SAHM08] present an approach for generating static images of facial expressions using a deep belief network (DBN) composed of 3 RBM layers [HOT06] (right image of Fig. 2.13). They train the DBN with data from [KTC00], which includes images of facial expressions, respective activated AUs and identity labels. The user only specifies the last two to generate a facial expression. Despite the method not providing input, at least directly, to a facial model, it could very well be used to drive the poses on such a model. This approach, however, only produces

low-resolution images and does not include any timing information.

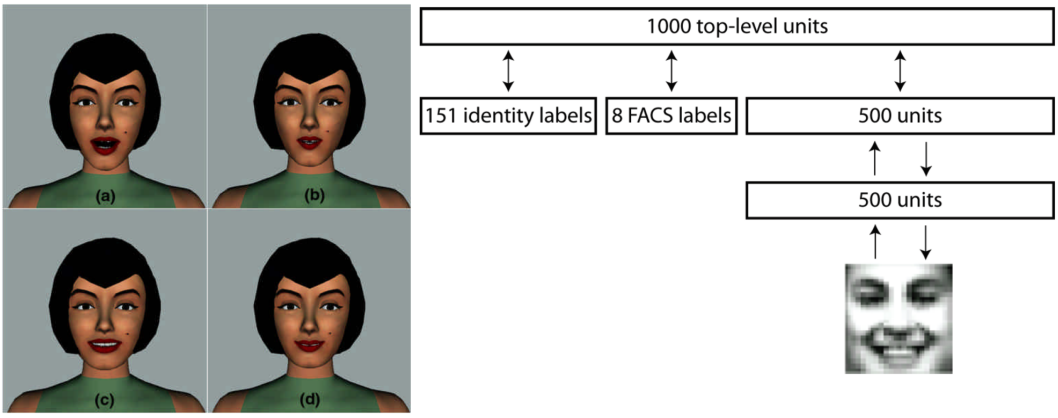

Figure 2.13: Effects of 4 principal components in the facial expression [KMMT01] (left). DBN architecture [SAHM08], which takes an identity image and one of 8 AUs as input to generate a new expression (right).

To the best of our knowledge, Zhang et al. [ZSCS04] present the only approach that uses motion graphs to learn, represent and synthesise facial animation. Amongst the revised literature, this method is also the closest to the core contributions presented in this thesis, having been effectively used as an inspiration. Zhang et al. [ZSCS04] create a motion graph, referred to as face graph, by connecting all poses in a training sequence. Each node contains a facial mesh and each edge, connecting two nodes i, j, contains the respective similarity value. This value is calculated using the L2-norm, i.e. Euclidean distance, as shown in the following equation. It uses information from both i, j and $i+1, j$, where $i+1$ is the next node in the original sequence.

$$w(i,j) = dist\left(M_{i+1}, M_j\right) + \lambda dist\left(M_i, M_j\right) \tag{2.2}$$

The user specifies the source and destination nodes, and the optimal in-betweens are found using Dijkstra, thus minimising similarity metric. In this sense, the user needs to have an in-depth knowledge of the graph. They additionally explore the creation of multiple graphs for different facial regions, with a separate optimisation occurring in each region.

2.2.3 Hybrid Techniques

Several researchers have combined aspects of different categories to achieve more realistic, easily configurable or controllable systems. These approaches are now presented in the following flavours: constraint/rule-based combined with behaviour [PG96, Per97, ONPS05, AD07, ZDA08, BSG10], behaviour with statistical-based methods [EKMT03], and finally

only one approach that combines all elements [SBR+14]. A particularly good read is presented by Pelachaud [Pel09], which provides an overview of many years of research on the topic, from which we have explored a small part [CPB+94, PB03, NHP09]. Despite reviewing approaches from multiple categories, the core of Pelachaud's work falls within pure constraint or behaviour-based methods, and on combining aspects of both.

- Perlin and Goldberg [PG96] (Constraint + Behaviour) present a method that supports behaviour definition and includes a hierarchical animation engine. This method was explored into more detail for the face in [Per97], in what has remained one of the most complete procedural facial animation methods. The layered approach is based on associating degrees of freedom (DoF) to labels, e.g. they can control the range of a joint's DoF (low-level) or control the activation of a smile (high-level), by varying input value between 0 and 1. DoFs' variation can follow different mathematical functions, such as sine, or rely on coherent noise [Per85] to introduce small variations in motions, such as is required for the neck or gaze motion. As a new pose is determined, interpolation occurs between the current and the next poses, hence creating a smooth transition. Blending is subject to rules, such as having a specific intermediate pose or how the weights vary. Conflicts are solved via direct rules and by defining actions that can coexist or need to fade-out/in when blending. The latter is done by placing actions in different groups, whose relationships are created. Behaviour combines scripting with a rule-based system based on personality, mood, current goals and character's intentions. The author creates small scripts that contain deterministic and non-deterministic sequences of actions/animations, and specifies the probabilities of changing between them. There is always one script running, with the scripts changing whenever an event is triggered. Such also leads to updates in the character's emotional and/or physiological state. Fuzzy logic is incorporated when choosing the motion, thus taking into account environment properties and internal state. Input for this system ranges from defining the behaviour, and providing the events, to directly controlling the animation in the hierarchical engine;

- Ochs et al. [ONPS05] (Constraint + Behaviour) distinguish two levels of emotion: felt/elicited and expressed. The former is purely based on the event, while the latter is strongly related to the sociocultural context. They then use the Artimis engine [SBP97] to control the interactions with the user (BDI-based [RRG95]), and keep track of the character's mental state. Hence, controlling the felt and expressed emotions. Conflicts between these types of emotions are identified from the OCC model [OCC88], which allows determining when emotions should be suppressed, masked or intensified. Fuzzy rules associate expressions with emotions, controlling the way

poses are blended and the respective intensities. Separate facial regions, i.e. upper and lower part of the face, are used when blending multiple emotions. The choice of which emotion goes to each region follows behavioural studies;

- Arya and DiPaola [AD07] (Constraint + Behaviour) present a multispace-based approach, also explored in [ZDA08]. It relies on 4 independent spaces: knowledge, personality, mood and geometry. The first is controlled via an XML-based language compatible with MPEG-4 that supports decision making and high-level timing control, the second and third have behavioural psychology and user study bases, relating personality traits and emotional states to facial actions, while the last relies on a hierarchical description of facial expressions. The association between personality and facial actions is coded in a Wiggins' circumplex model [WTP88] and was obtained via questionnaires. The personality profile also defines the timing for activating visual cues, which can be either random, periodic or based on speech energy. Emotions and moods are mapped to a 2D space, i.e Russel's circumplex model [Rus91]. These have associated facial actions, which are activated according to the current mood. Conflicts between the spaces are avoided two ways: by having different motions in each, e.g. personality can trigger the motion of nodding, while the mood space can deal with eyebrows specification, and by having the spaces triggering motions in different timings;

- Bidarra et al. [BSG10] (Constraint + Behaviour) model the character's internal state by mapping emotions, mood and personality to points in the PAD space [MR76]. Emotions are associated to the PAD space based on [Geb05] that maps OCC emotions [OCC88] to this space (top left image of Fig. 2.14). Mood is directly defined within the same space and personality is based in the five-factor model [MJ92], which has also been mapped to the PAD space in [Meh97]. The mood is updated using a push and pull approach triggered by the emotions, which have been associated to stimuli. They additionally allow the specification of external factors or physiological aspects, such as drugs or hunger respectively, and how these lead to a variation of the mood over a time. On the animation side, previously created short facial animations are placed in the PAD space, which are then chosen and blended based on proximity to the current mood (bottom images of Fig. 2.14). Noise is added based on arousal values, with gaze and head movements controlled by rules over the PAD space;

- Egges et al. [EKMT03] (Behaviour + Statistical) model personality, emotion and mood via independent vectors. Their approach uses OCC's emotional categories [OCC88] and Five Factor model [MJ92] to define the personality. While these models are the basis for their approach, as the vectors are independent, it is possible to

easily replace or extend the approach to account for other aspects. When a stimulus occurs, the mood and emotional states are updated based on a series of weight-based equations that also account for the mood history. This then outputs an emotional tag that serves as input for the approach of Kshirsagar et al. [KMMT01]. Onset, apex and offset of the motions are influenced by the speech animation and values defined à priori;

- Sagar et al. [SBR+14] (All) present arguably the most advanced method for procedural generation of facial animation (top right image of Fig. 2.14). Instead of creating a pure facial animation system, they attempt to fully model the human brain, and with it, all the learning and communicative processes. Rules, statistical and behavioural models are combined to create a generative model of facial expressions. They simulate, using e.g. recurrent neural network models, the different neurobiological systems, such as the Hippocampus and Facial Nucleus, which trigger the motor circuits, thus driving facial muscles. Animations are generated by activating pre-computed biomechanically simulated deformations, which are associated to a physically based facial model. Due to sheer amount of used techniques, they have developed a scripting language called Brain Language (BL) that both connects the different models, and allows changing their parameters;

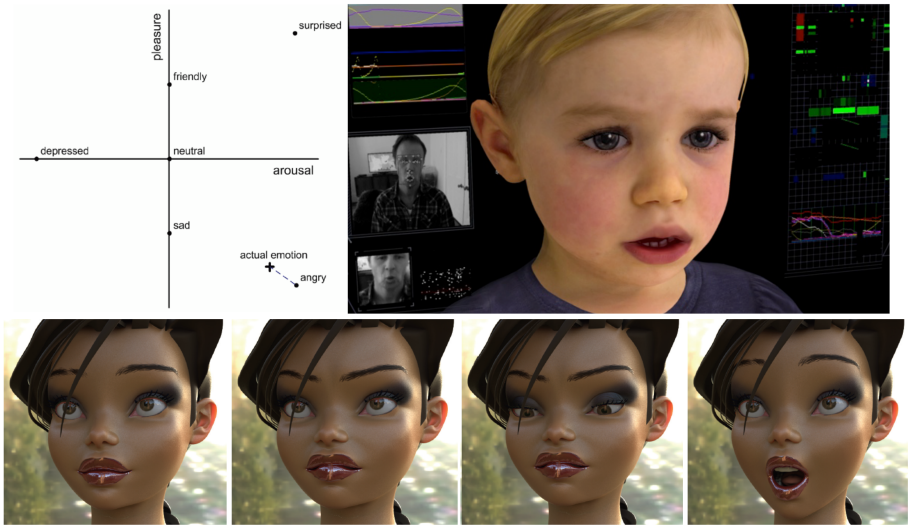

Figure 2.14: Multiple inputs and outputs of the model presented at [SBR+14], ranging camera input, activity of specific neurons and the actual output (top right). Example of current emotion in the PAD space [MR76], which also shows the positions of several emotions (top left). This approach is used by [BSG10] to drive the model shown at the bottom.

2.2.4 Comparing approaches

Reviewing constraint/rule, statistical and motion graph-based methods allows us to establish the general pros and cons of each group:

Constraint-based approaches are the most widely explored and rely on the expertise of the person(s) creating the hierarchy or defining the script/rules. This individual(s) *needs* to be highly knowledgeable of both the facial poses and motion timing, as they tend to be manually defined, while also having the end user in mind. Not only this but creating such a system can become very complex to design due to the number of layers, flow of data and relationships between the different components. This leads to practical implementations that are initially tailor-made to a task and afterwards extended for more general-purpose uses. Extending the system to characters different from the original target can be as simple as using the same rig, require creating the base animation for the other character or, in the limit, rebuilding the whole system. Constraint-based approaches do allow a novice author to jump into the definition process, however, this will normally impact either the quality of the results or the usability of the system. Usability itself can range from being extremely simple to having a flood of parameters that need to be controlled, leading to difficulties in predicting the outcome animation. Nevertheless, if done properly, controlling the motions is easy and even allows different levels of control, suitable for a wide range of expertise. Other advantages include ease of understanding and explaining the core concept of the approach, which in turn facilitates the overall use of the system. Constraint-based approaches also do not require much external data other than the poses/small animations in the model(s) being controlled. All these aspects lead to the most subjective category within automatic generation of facial animation.

Statistical methods work by mapping a facial motion database to alternative spaces, where the user then navigates, or by learning a model that converts certain input values to facial properties. These methods are very complex, both in terms of describing their inner-workings and predicting the results accurately, making their use generally a black box. Statistical approaches also need a DB to be trained on. The number of samples required ranges from only a few to hundreds or thousands, depending on the underlying technique. And, as we will see in Sec. 3.3.1, available DBs are quite scarce for facial motion. Some methods are able to generalise training data to some extent, however, more varied data will lead to better results. The control provided by these approaches is also deeply connected to the training data and underlying model, with the input parameters needing to either be defined in the samples or embedded in the method, e.g. a specific optimisation function. Adding a new parameter can require updating all samples in a DB, or even capturing new

samples. Configuring this system is done by the method expert and not the final user. Nevertheless, after the implementation is in place, the author only needs to control the input or retrain the method with more data. In this sense, it is considerably more objective than constraint-based methods. Different users are limited to the rules available to control the black box, which also means that the configuration time for the author is considerably lower, as long as he/she does not need to create the training samples. These methods can also generate both facial poses and timing without specifically defining them à priori.

Motion graph-based methods work by creating a graph that represents the training data and whose exploration allows the synthesis of behaviours. It occupies a place between constraint and statistical-based approaches in easiness of understanding the method's inner-workings. The base concept is simple to describe and understand, however, optimisations, e.g. on synthesis, are more akin to statistical methods. Hence, placing in the middle of previous categories regarding the capacity to predict the resulting animation, for a given input. Control, however, can require a deep knowledge of the training data or motion graph, which does not make it user-friendly. In terms of training data, motion graphs start working with a reduced number of samples, but the results can be limited. More data leads to more realistic motions and helps to fill for the fact that motion graphs seldom generalise the training data. As with statistical-based approaches, it requires an expert to implement and extend, however, the moment such is done, the user only needs to provide control input or more data to retrain the graph. Therefore, making it more objective than constraint-based methods. Also, similar to statistical-based approaches, these methods can generate timing and pose information without explicitly defining it à priori.

2.2.4.1 Table Comparison

As procedural facial animation is the centre of the thesis, we additionally present the approaches of sections 2.2.1, 2.2.2 and 2.2.3 in Tab. 2.1. Thus, matching them according to the core method, control and capacity to generate animations:

- **Core concept of the method** and respective group within the division followed in the previous sections. The underlying concepts are: *Rules* on how to analyse a certain input that triggers some facial behaviour(s), with multiple rules possibly activating for the same input; *Direct Association* of keywords to facial motions; *Scripting*, i.e. a formal definition on how the input should be provided or how facial behaviours should be created (more details in the third bullet); *Hierarchy* that contains at the lowest level motion information, and then builds on top of the base info with successive layers

of parameters; *2D/3D space*, whose coordinates are associated to facial expressions; *PCA* vectors used to define the space (similar to 2D/3D spaces); *Fuzzy Logic* relies on a soft mapping between input and the choice of pose or its weight, as opposed to a hard mapping, where an input always triggers the same pose; *Ontology* is a formal definition of the knowledge space via types and their relationships; *Motion graphs*, *Deep Belief Networks* and *Neural Networks* are the names of the core approaches used;

- **Pose Control** deals with the specific details of how the method is forced to generate new poses/animation, and includes but is not limited to: Speech intonation, which means that either keywords are provided with an accompanying text or an audio stream is analysed and the intonation extracted; Scripting parameters, i.e. the author will follow the language definition to control the characters, which usually means using keywords/labels and respective parameters to describe the facial behaviour at a specific moment; Behaviour definition & Scripting means that facial behaviours are controlled via a language, as described in the previous case, but the keywords are then analysed according to the character's behaviour;

- **Scripting in control and/or definition** deals with the place in the method where scripting is used. Scripting is traditionally associated to controlling the input that then triggers the animation. However, several approaches have also formally defined a language to configure the system. An example of such is in a hierarchy based approach that has rules on how each layer's parameters need to be defined and how to establish relations with other layers. We believe it is important to make the distinction between the two scripting types, as these are not mutually inclusive. Scripting itself can also include details on both poses and/or timing;

- **Timing in control and/or definition** deals with the full duration of the short animation and how the poses vary within that duration. Not all approaches account for this aspect and deal only with static poses. An animation needs timing, but before that it also needs poses. So these approaches, while incomplete, are still relevant in this field. Approaches that account for timing can be based on parameters: defined when configuring the system, (à priori), provided as control input or both. Despite rarer, some approaches do not require explicit definition of timing.

Paper	Methods				Pose control	Scripting in control	Scripting in definition	Deals with timing	Specified on input	Specified *à* priori
	Constraint-based	Behaviour-based	Statistical	Motion Graphs						
[PBS91]	Rules	-	-	-	Intonation in speech	-	-	Yes	Speech based	*
[CPB+94]	Rules & Direct Association	-	-	-	Some expressions from intonation, others manually placed	-	-	Yes	Speech based, scripting	*
[BHPN01]	Fuzzy Logic	-	-	-	Emotion intensity vector	-	-	No	-	-
[CVB01]	Rules & Scripting	-	-	-	Context and linguistic based, scripting keywords	(Opt) *	*	Yes	(Opt) Speech based, scripting	*
[PB03]	Scripting	-	-	-	Scripting keywords	*	-	Yes	Speech based, scripting	*
[RNTH03]	2D space	-	-	-	Point in 2D space defined with basic expressions	-	-	No	-	-
[NBPZ05]	Scripting	-	-	-	Communicative channels, performing abilities, scripting keywords	*	-	Yes	Scripting	*
[APM07]	2D space	-	-	-	Point in Russel wheel	-	-	No	-	-
[NHP09]	Rules, Scripting	-	-	-	Scripting keywords	-	*	Yes	Duration of whole sequence	-
[QCM10]	Rules, Scripting	-	-	-	Scripting keywords	*	-	Yes	*	-
[RTR+11]	Direct Association	-	-	-	AUs	-	-	No	Activation curve for each AU	-
[DiP13]	Hierarchy	-	-	-	Layer Parameters	-	*	Yes	-	*
[KMT02]	-	Scripting	-	-	Scripting keywords, probabilities	*	-	Yes	Speech based, scripting	*
[GRVT+06]	-	Ontology	-	-	Query using labels and properties	*	*	No	-	-
[MBXL08, XMLD07]	-	Hierarchy	-	-	Layer parameters	*	-	No	-	-
[KMMT01]	-	-	PCA	-	Emotional expression	-	-	Yes	Speech based	-
[RNTH03]	-	-	PCA	-	Points in 2D spaces formed from PCA vectors	-	-	No	-	-
[SAHM08]	-	-	Deep Belief Network	-	Identity labels, AUs	-	-	No	-	-
[ZSCS04]	-	-	-	Motion Graph	Nodes	-	-	Yes	-	-
[PG96, Per97]	Rules, Hierarchy	Rules, Scripting	-	-	Behaviour definitions, events or layer parameters	(Opt) *	*	Yes	(Opt) Hierarchy input	
[ONPS05]	Fuzzy Logic	Rules, Scripting	-	-	Behaviour definition, emotions intensities	-	*	Yes	-	*
[AD07, ZDA08]	Scripting, Hierarchy	2D space	-	-	Behaviour definition, scripting keywords	*	-	Yes	*	*
[BSG10]	Rules	3D space	-	-	Behaviour definition, Events in PAD space	-	-	Yes	-	*
[EKMT03]	-	Rules, State Machine	PCA	-	Events	-	-	Yes	Speech based	*
[SBR+14]	Rules, Scripting	Rules	Neural Networks, ...	-	Audio, visual or text input as natural language	-	*	Yes	-	-

Table 2.1: Comparison between procedural facial animation techniques in terms of: category of core approach; type of input to trigger animation generation; presence of a scripting language; ability to generate timing and source of information required for timing.

2.3 Discussion

Procedural facial animation is the dark horse within the three main techniques for animating faces. Manual animation, or key-framing, is the most flexible, while also achieving the best quality. However, this comes at the cost of being very time-consuming to create. Performance-driven animation significantly reduces the time when dealing with humanoid characters, but it still requires fine-tuning, additional equipment and an actor's performance. Procedural methods create animation in even less time, without any additional requirement, but make up with the worse quality results that need more cleaning and fine-tune, hence falling between the previous approaches in terms of production time. Manual and performance-driven animation are the most widely used both in video-games and films. However, procedural techniques are good alternatives when trading quality for ease of generation is acceptable, such as in secondary characters of video-games or crowd simulation. This trade-off comes at the cost of either a time-consuming configuration, that is case-specific and requires manually created animation, or the need of data more complex than the one used with performance-driven methods.

Procedural body animation (reviewed in App. D) is not on the same foot as its' facial counterpart. It has been widely explored, and this makes it the perfect field to obtain insights on how to tackle the issue at hands. Amongst the many techniques reviewed, we have identified example-based approaches as the most relevant and within these focused on: motion graphs, parametric-based motion, and statistic-based approaches. We have followed this convention to group facial animation methods in Sec. 2.2. These explore existing motion samples to generate new animation that meets the desires of the author. Alternative approaches demand a physical or mathematical understanding of facial motions, which either already exists, in the form of physically-based rigs, or is still an open question. We have also analysed the reasons for the disparity between body and facial research, which include: facial movements being more complex than body, due to lack of rigid constraints; basic and cyclic body motions are extremely useful, while the same cannot be said about the basic facial motions; a significant chunk of research on body needs to focus on physical interactions with the environment, which the face is rarely subject of. These reasons also prevent body animation methods from being directly "ported" into the facial domain.

Despite the scarcity of procedural facial animation techniques, we have identified three main groups of approaches: *constrain/rule*-based methods, where empirical and practical knowledge are explored to define a mapping between input and the animation; *statistical*-based methods aim to learn a model that generalises a facial motion DB and allows synthesis; and finally, *motion graphs* that focus on the use of a graph structure to represent motion data and

synthesise animation by traversing the created structure. Amongst the many differences, the largest is the trade-off between effort and knowledge required to directly create and configure constraint/rule-based systems, and the expertise required to train statistical or motion graph-based methods and time to collect the required data. In terms of using the system after it is configured, methods have the potential to be similar, however, that is not yet the case. While the constraint-based approaches allow configuring any type of input that is then used to generate motion, other methods require appropriate facial data. It is exactly the lack of this element that severely hinders research on these fields. When that is solved, we believe these approaches will flourish. Hybrid approaches have mostly focused on combining behaviour definition with either constraint-based or statistical methods, which adds a layer of information that allows the choice of more believable motions. No methods were found that combine behaviour with motion graphs. Behaviour modelling can also be used as an abstraction to reduce the control input, as by defining how the character should behave, it allows animation to be controlled via stimulus or commands, which effectively turns the author into the character's director.

2.3.1 Challenges

The holy grail of procedural facial animation is to have a method that is not only predictable and easily controllable, but also generates good quality animation and is easy to configure. Real-time also appears to be a consensus, being mainly driven by interactive applications. However, several challenges need be better understood and/or overcome before such is achieved:

- Complexity of facial behaviours that vary with factors such as person's anatomy, cognitive and emotional state, background context, history ...;
- Multidisciplinary research that involves techniques from computer graphics, machine learning, psychology, linguistics, neuroscience ...;
- Highly subjective control input that varies with the application, the person configuring the system and the experience of the final animation author. Directing a character is significantly different than directly controlling its facial pose;
- Lack of high-quality facial motion data, which needs to have the appropriate parameters related to the control input, and difficulty to acquire it;
- Lack of methods optimised facial synthesis that take into account face specific constraints;

With all these challenges at hand, we can further ask where is the field heading and what still lacks significant exploration. Despite research having started almost 3 decades ago, this field is still in its infancy. Approaches using manually defined hierarchies or rules have already been widely explored, and while there are always different variations possible, there is only so much that can be done following a pure manual configuration approach. Statistical and motion graphs-based approaches still have a wide ocean waiting to be explored, and thus present significant opportunities. The field is also evolving in the direction of hybrid approaches, specifically ones revolving around behaviour. We particularly point to its use on top of other methods, enabling the author to direct the character. An important aspect lacking in most approaches is complex behaviour. Most methods remain shielded to basic emotions, with other behaviours largely under-explored.

Our goal with this chapter has been to explore existing approaches and determine the main issues within procedural facial animation, and, while doing so, identifying the opportunities for research. To this end, we have explored performance-based animation techniques, procedural body animation (App. D), more specifically example-based approaches, and, as expected, other procedural facial animation methods. Our choice ended up on exploring motion graphs for facial animation, and then combining them with behaviour definition through the use of mind maps. Other paths were clearly possible, however, motion graphs already showed promising results with only the tip of iceberg having been explored. The use of mind maps further allowed pursuing the quest of controlling the character via stimulus, while avoiding the need to describe all possible stimulus-reaction of a character.

Procedural animation techniques have been scarcely explored for faces, however, these are, in our view, the most likely to shape the future of facial animation. By identifying both relevant approaches and issues with the field, we hope to propel further research that will result in easily controlled and better quality animation. Thus, enabling procedural facial animation to be used in films, video-games or other productions to animate main and secondary characters alike.

Chapter 3

Facial Animation Using Motion Graphs

Facial animation is a time consuming and cumbersome task that requires years of experience and/or a complex and expensive set-up. It is nearly impossible to achieve high quality and unique animation for all, and every, character present in mediums ranging from films to interactive applications. All with very different constraints and goals. We address this issue within the context of secondary characters, such as the ones present in a crowd, with a novel method that uses motion graphs to learn the motion relations present in a facial behaviour database, and synthesize similar, but unique, animations. Short motion clips are generated using only a behaviour label, e.g. emotion, as input. It has a simple set-up process and works near real-time, using a compact representation of the data. Hence, significantly reducing the animation creation time and enabling more expressive characters at a lower cost. We have tested our method with different databases and validated the motion quantitatively by comparing the synthesized animation against the original samples. Additionally, the results were qualitatively compared with other state of the art techniques. By the end of this chapter, you should have a full understanding of the inner-works of our procedural technique. You should also confirm its capacity to learn and synthesize motions from a DB, both uniquely and intuitively. Thus, being ideal to generate the multitude of secondary character facial animation required nowadays.

3.1 Problem Statement

Animating faces is currently done using either key-framing (Sec. 2.1.2) or perfomance-driven animation (Sec. 2.1.3). The first relies on the knowledge and time of an animator that controls the key-poses of a model, when they occur and what happens in their in-betweens.

This is a very cumbersome and slow process, which as a ballpark figure can require up to 1 hour to produce 1 second of high quality animation [LA10]. Facial performance capture significantly reduces the animation creation time, however, it comes at the cost of expensive equipment, complex set-up and the need of an actor. Depending on the quality needed, further cleaning might be required. Due to both cost and time constraints, it becomes infeasible to create the same high quality animation for the dozens or even hundreds of characters that fill the streets of a city, stores or react to an event collectively... Inevitably, these constraints play a stronger influence on characters that have less screen time, limiting them to a small group of animations that are constantly replayed. The problem, however, happens when these repetitions become noticeable, which can lead the viewer to break his/her suspension of disbelief. This issue is particularly visible in interactive applications, where the user can roam and have multiple interactions with the same characters. Offline applications, such as films or series, have more means to avoid this issue, by controlling the time, distance and speed with which secondary characters appear. Still, this issue appears in productions whose budget is more constrained, with crowds that fill locations in the world, e.g. a stadium. This thesis deals with the increasing need of facial animation to create more emphatic and compelling secondary characters, that display complex and varied behaviours when prompted to do so, be it by a player or direct authoring. Uniqueness resulting from a same command is particularly crucial in the context of video-games, as the character can interact with the player exactly in the same context and task multiple times. Our technique works near real-time and is capable of producing slightly different motions, even for the same input. Thus, simplifying the process of controlling which motion to generate, and being a perfect candidate to animate e.g. an angry crowd or a storekeeper that always receives the client happily.

On a lower level, facial animation consists in the change of poses through time. In key-framing, the animator controls all the details manually, while in performance-driven it is the actor's movements, captured continuously (at the highest possible frame-rate), that drive the model's poses. All done by analysing each a frame of information, which can be an image or markers positions captured using e.g. Vicon's Cara. Procedural methods, on the other hand, need to generate both facial poses and timing, i.e. when poses occur and what happens between them, from scratch. Procedural methods that actually generate short animations are already scarce, and they additionally tend to be connected to speech. As seen in sections 2.2,2.2.4, most methods require significant manual efforts, with the author providing timing and pose details that are either used directly, or combined to create the animation. Our method reduces these efforts by analysing a facial motion DB and representing it with motion graphs. These encode all the poses/durations in the nodes and the transitions as

edges. We have additionally proposed a compression method that enables an easy and intuitive configuration step, and avoids keeping redundant information originally present in the DB.

3.2 Solution Overview

We propose a procedural facial animation solution that, at its core, relies on the use of motion graphs. This was inspired in the work of Kovar et al. [KGP02], and, similar to other graph based approaches, has two main steps: graph creation, from the analysis of a motion DB, and motion synthesis, by traversing the graph. This process, outlined in Fig. 3.1, is now described. Before even creating a graph, it is important to find an appropriate DB (Sec. 3.3). With the labelled and landmarked facial movement DB chosen (Sec. 3.3.1), the *motion graph creation* process (Sec. 3.4.1) starts by comparing all poses present in each DB sample and creating a temporary sample graph. This graph is produced by identifying different poses that are similar "enough", using the metric described in Sec. 3.4.2, and then merging them. The author indirectly controls when the poses should stop merging by specifying how much the original data should be compressed, i.e. compression ratio. This information is used to automatically optimize the sample graph creation threshold (Sec. 3.4.3). After all the samples graphs have the desired compression, they are combined to create the final region graphs, in a process that follows the same compare-and-merge approach. An optimization over the merging threshold takes place to force the final graph to achieve a desired merge compression. *Motion synthesis* occurs by traversing the graph and determining which nodes should contribute to the generated animation clip. The author provides a behaviour label that is used to find the motion's start and end nodes, with the path obtained by minimizing the edges' similarity value (Sec. 3.5). The final motion is then obtained by analysing information previously stored in the path nodes (Sec. 3.5.1). The graph structure is additionally explored to coherently introduce noise in the synthesized motions (Sec. 3.5.2.1) and generate idling behaviours (Sec. 3.5.2.2).

For clarity, we now present a list of the most commonly used terms in our motion graph-based technique.

- *Region Graphs*: Motion graphs created for each facial area;
- *Sample/Sequence*: Sequence of poses/frames in the original DB $< P_1 ... P_n >$;
- *Peak or Apex sample poses*: Most intense/expressive poses in a sample. Traditionally associated to moments where all the relevant muscles are activated. Akin to the key

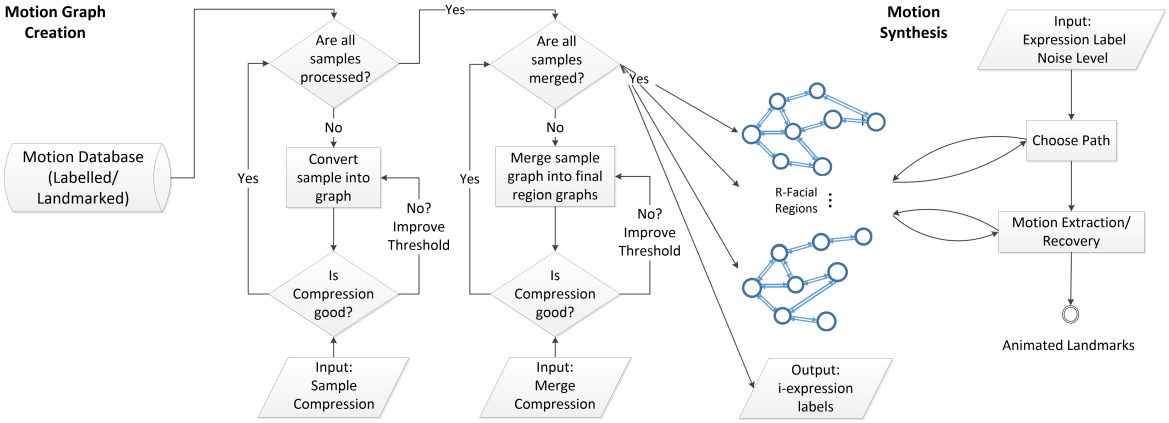

Figure 3.1: Overview of the procedural animation pipeline. The DB is analysed to create a graph per facial region. Samples are processed to create sample graphs and then merged. Both steps rely on matching the obtained and desired compressions, and, if the values differ, repeating the process with a different threshold. Motion synthesis occurs by using the input expression label to find a path in the graph that allows recovering the final motion.

poses used in manual animation;

- *Sample Graph*: Group of region graphs created from one sample, $g_{samp} =< g_{region_1} \cdots g_{region_n} >$;
- *Final Graph*: Group of region graphs that results from merging all sample graphs, $G_f =< G_{region_1} ... G_{region_n} >$;
- *Expression labels*: Labels associated to peak poses in the DB samples;
- *Landmark displacement*: Difference between current landmarks coordinates and the base pose landmark coordinates. The base pose is the average of first frame/pose of all samples;
- *Node*: A graph node contains the landmarks displacement and additional data (Sec. 3.4) used to recover the motion (Sec. 3.5.1);
- *Similarity*: is a heuristic based value calculated by comparing the landmarks of two poses, and indicates how much they are similar (Sec. 3.4.2);
- *Source Node*: First node in a motion generation;
- *Destination/Sink Node*: Last node in a motion generation;
- *Compression*: % calculated based on the difference of data that would need to be stored to represent a graph containing a node per sample pose and the current final graph.

3.2.1 Solution Features

Our system provides an end-to-end facial animation solution with many features useful to the creation of more expressive characters. Features are separated from technical contributions (next section) as a way of distinguishing the advantages of building the whole system, and the advances in the state of the art. Our solution's features are then:

- Complete non-linear motion generation, with both timing and poses automatically calculated;
- Compact model;
- Intuitive control on the training phase via specification of the desired compression;
- Can be trained with any 2D or 3D sparse landmarked motion database;
- Capable of representing any facial motion containing well defined peak poses;
- Supports extension to more motions without retraining the whole model (scalable);
- Fast generation of animations, on-the-fly and near real-time;
- Intuitive authoring of animation via behaviour labels;
- Unique animations in each generation;
- Idle animation for any training motion effortlessly;
- Authoring of animation with a low number of input parameters;

3.2.2 Solution Technical Advances

We advance the state of the art by introducing:

- A motion graphs based approach designed to cope with the variety of facial movements and determine the common poses across multiple behaviours. This is achieved through a new node structure and graph creation process that leads a compact representation of the training data. The same structure also enables intuitive control of animation synthesis through a low number of input parameters;
- A novel approach for threshold(s) choice in motion graphs that eases the graph configuration process. Most methods leave this choice to the user, which is slow and challenging as the threshold(s) are heuristic based values that depend on the similarity metric. Varying the threshold(s) also tends to produce predictable results only

with large changes. We optimize the thresholds for the information to keep in the graph, which considerably simplifies the fine tuning of a graph. Additionally, global threshold(s) affect merging of samples equally, which is far from ideal given that some motions are more subtle than others. Thus, being too aggressive in one sequence and too soft in another. Individually optimized thresholds create graphs that represent the DB motions better and lose less information than with a global threshold(s);

- A new coherent noise method for motion graphs capable of introducing small variations in motion. This method explores the graph structure around a chosen animation path to reduce repetitions. Together with the underlying nature of using region graphs, to separately encode and generate motion on different facial parts, our solution produces widely varied animations, even with a small training database;

- A new method to create idle motions that explores the nodes around the current expression node. This method does not require any training data, with its behaviour being controlled with minimum input. It is capable of generating idling movements for any pose present in the DB;

3.3 Motion Database

The creation of the region motion graphs starts with a dynamic 2D/3D sparse DB, whose samples need to be aligned to remove influences of identity and head movements. Such influences would introduce noise into the comparison of poses. Any sample can then be incorporated into a graph as long as it: has sparse landmarks, starts with the same equivalent pose (P_{eq}) as all other samples, has the transitions between peak poses (P_{peak_pose}) identified and has these peaks associated to expression labels. All these bits of information will later be used to create the control mechanism. A sample, defined as $< P_1 ... P_n >$, can then be further divided into $< P_{eq} ... P_{peak\_pose\_1} ... P_{peak\_pose\_2} ... P_n >$. Each pose is composed by m number of landmarks $< L_1 ... L_m >$. Marker sparsity has been chosen over density due to being more lightweight and reducing the leakage of motion from one facial region to another, despite the latter containing more information. All samples need to start with the same equivalent pose, however, the remainder of the sample is not relevant. This pose is the common denominator between all samples, allowing the motion of landmarks to be converted to displacement space. Our approach explores the structure of the graph to introduce coherent variations, however, a motion graph should, ideally, be trained with multiple examples of the same label. Hence, allowing it to learn different variations of the same expression label and to combine the region motion present in multiple samples.

Our approach for pose alignment revolves around two steps. First, and for each sample, all poses are rigidly aligned with their equivalent pose. If the sample does not include any markers from where the rigid motion can be directly extracted, we use Procrustes analysis (uniform scaling not allowed) to extract the transformation matrix from a manually chosen subset of markers. These markers should be ”as rigid as possible”. Secondly, for each sample and each region, the first pose is aligned with the average of all equivalent poses, from all samples', using again Procrustes analysis. This transformation is then applied to all other sample poses, resulting in *”non-rigid”* alignment, where the regions are individually aligned. Thus, reducing the effect of different proportions. Fig. 3.2 shows an example of a pose landmarks not aligned and after alignment.

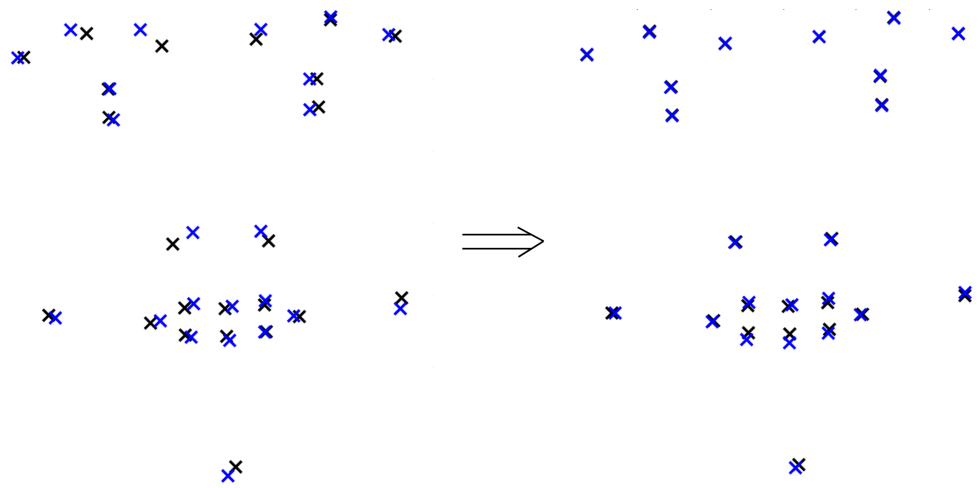

Figure 3.2: Effect of pose alignment. On both images the black crosses show the average of all equivalent poses. The blue landmarks show the non-aligned first pose of a sequence (on left) and its aligned results (on right).

3.3.1 Facial Animation Databases

There are, unfortunately, not many databases that satisfy our requirements for the creation of motion graphs. We have surveyed existing databases, with the results seen at tab. 3.1, 3.2 and 3.3. In this context labelled and landmarked facial motion DBs are very scarce, with only the CK/CK+ [KTC00, LCK$^+$10], BU-4DFE [YCS$^+$08], USTC-NVIE [WLL$^+$10, WLW$^+$13] and BP4D-Spontaneous [Zha14] matching all the requirements. [MMKJ99, BBBB$^+$03, OHS$^+$05, PVRM05, RRÉM$^+$06, BLF$^+$06, GP06, GKN08, VP10, BS10, MVCP10, VDSHFD11, DGLG11, DGLG12, BMS12, SMMH12, KCBW12, RSSL13] do not have landmarked sequences, which is unfortunate given the quality and the wide range of motions present in some of these DBs [OHS$^+$05, PVRM05, GP06, MVCP10, VP10, BS10, VDSHFD11, DGLG11, DGLG12, KCBW12, BMS12, SMMH12]. Marks et

al. [MHM10] present a motion DB that is very limited in terms of samples, and thus does not include enough expression variation. A similar issue occurs with [TLJ07, LCP+11]. Mavadati et al. [MMB+13] and Cosker et al. [CKH11] only label the sequences in terms of AUs, lacking descriptors of facial behaviours as a whole.

When comparing the CK/CK+ [KTC00, LCK+10], BU-4DFE [YCS+08], USTC-NVIE [WLL+10, WLW+13] and BP4D-Spontaneous [Zha14], the last three have the upper-hand regarding the motion quality. BU-4DFE and BP4D-Spontaneous are, however, paid, and we were unable to obtain the landmarks of USTC-NVIE. As a result, and in spite of not being originally intended for motion synthesis, we have selected the Cohn-Kanade (CK and CK+) data set [KTC00, LCK+10]. It fits our labelling requirements, while also having psychologically valid facial expressions. This DB has also been widely used in both psychology and computer science, hence being a good candidate for validating our method. CK/CK+ contains expression labels associated to the emotions of: *happiness, sadness, disgust, surprise, anger, fear, contempt* plus *neutral*. All samples follow the structure: neutral-to-peak expression, with each pose containing 68 landmarks. These landmarks have been reduced to 23 as a way of lowering the overhead of graph creation, motion synthesis and errors in similarity calculation. Fig. 3.3 shows the full and reduced sets of landmarks, as well as the ones used for non-rigid alignment. Rigid alignment of the CK/Ck+ relies on the tip of the nose and eyes' corner landmarks.

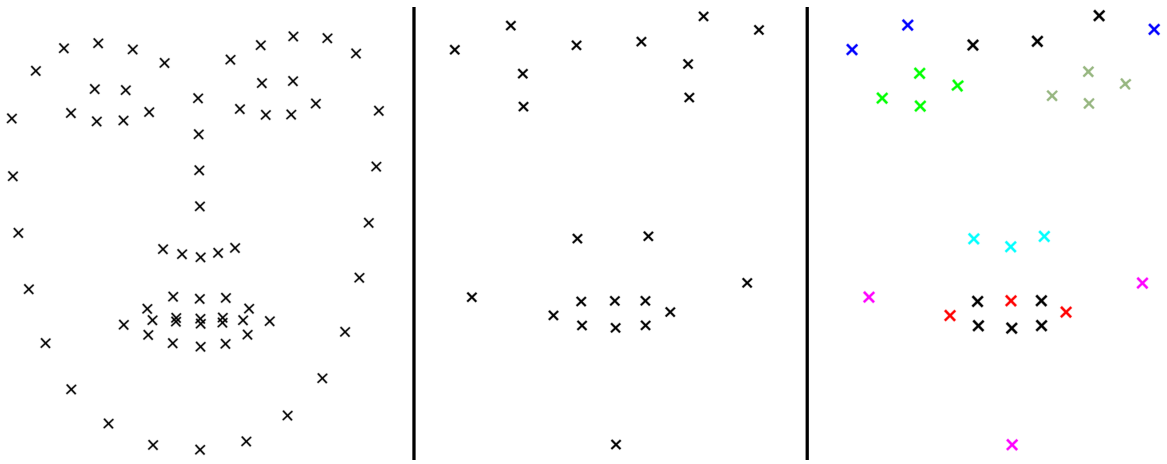

Figure 3.3: CK/CK+ landmarks: from left to right, are the following: original 68 landmarks; reduced set used to create the graph; markers used for "non-rigid" alignment of the face regions, grouped by colours: blue for eyebrows; different greens for eyes, cyan for nose, red for mouth and magenta for jaw and cheeks.

Papers/Links	DB Name	Year	How to Obtain	Format	Quality of Format	Metadata / Motion Labels	Is Landmarked?	Number of Subjects	Contains Head/Gaze
[BBBB+03]	Banca DB	2003	Paid	2D Video	720x576	Neutral, smile, anger, scream	No	208	No
[KTC00, LCK+10]	Cohn-Kanade (CK and CK+) database	2000, 2010	Free, direct online request	2D video	640x490	Happy, surprise, anger, disgust, fear, sadness, contempt and neutral; FACS coded; Posed/Non-Posed	Yes	210	No
[YCS+08]	BU-4DFE (3D + time): A 3D Dynamic Facial Expression Database	2008	Paid	2D video, 3D dynamic	3D mesh ~35,000 vertices; Texture video has a resolution of 1040x1329	Angry, disgust, fear, happiness, sadness, and surprise. Each sequence contains neutral expressions in the beginning and the end. Meshes are registered	Yes, obtained from mesh	101	No
[Zha14]	BP4D-Spontaneous: Binghamton-Pittsburgh 3D Dynamic Spontaneous Facial Expression Database	2013/2014	Paid	2D video, 3D dynamic	3D mesh ~30000-50000 vertices; Texture video has a resolution of 1040x1329	Happy, Sad, surprise, embarrassment, fear, pain, anger/embarrassment, disgust. FACS coded	Yes, 2D and 3D	41	Head pose tracked
[MHM10]	IR Marks video data set	2010	Free	2D videos	Multi-camera video (4 visible, 3 near-infrared) 640x480	Poses identified: neutral, closed-mouth smile, happy, sad, angry, disgusted, afraid, surprised, neutral	Yes, 2D and 3D	1	No
[BLF+06]	RU-FACS	2006	Free, Download link not working	2D video	Multi-camera video (4) 640x480	FACS	No	100	No
[MMB+13]	DISFA	2013	Free	2D video	1024x768	Intensity of 12 AUs manually annotated on a six-point ordinal scale (0 and 5 increasing levels of intensity). The AUs chosen were amongst those most common in emotion expression and social interaction	Yes	27	Yes
[CKH11]	D3DFACS	2011	Free, Request required	3D dynamic	3D mesh ~30000 vertices; Texture video has a resolution of 1024x1280	FACS coded on the the peak frame of each sequence	Yes, from mesh	10	No
[BS10, BMS12]	GEMEP Corpus	2010,2012	Free, Request required	2D video	-	18 emotions played by actors in different conditions (normal, low intensity, masked): panic fear and anxiety; hot and cold anger (rage and irritation); depressed sadness, despair, pleasure, elated joy, pride, amusement, relief, and interest, admiration, tenderness, surprise, digust, contempt and shame	No	10	Yes

Table 3.1: Survey of existing facial motion databases that were matched against: format and quality of information captured, metadata/motion labels present, being landmarked or not, size and if they contain head/gaze movements. (Part I)

Papers/Links	DB Name	Year	How to Obtain	Format	Quality of Format	Metadata / Motion Labels	Is Landmarked?	Number of Subjects	Contains Head/Gaze
[SMMH12]	BINED: The Belfast Induced Natural Emotion Database	2012	Free, Request required	2D Video and audio	Set 1 & 2: 720x576; Set 3: 1920x1080	Spontaneous disgust, surprise, fear, amusement, anger, sadness, frustration. Additional data include self reports on felt emotions, intensity and continuous rating of the felt valence. 3 sets of procedures to elicit emotions.	No	Set 1: 114; Set 2: 90; Set 3: 60	Yes
[PVRM05, VP10]	MMI facial expression database	2005,2010	Free, Request required	2D static and 2D dynamic	720x576	FACS annotated (partially at frame level). Includes AUs temporal dynamics (onset, apex, offset). Subjects display 79 expressions ranging from single AUs to the emotions: happy, surprise, anger, disgust, fear, sad. Spontaneous disgust, happy and surprise added later.	No	Initially 75; Extended with 25 for spontaneous expressions	Yes
[MMKJ99]	XM2VTSDB multi-modal face database	1999	Paid	2D static, 2D dynamic, static 3D model and audio	3D mesh ~8,000 vertices; 720x576	Short clips of dialogue	No	295	Yes
[RSSL13]	RECOLA: Multimodal Corpus of Remote Collaborative and Affective Interactions	2013	Free, Request required	2D Dynamic, Audio, electrocardiogram (ECG) and electrodermal activity (EDA)	1080x720	All modalities are synchronized. Emotion variation continuously measured on two dimensions: arousal and valence. Social behaviour is rated once per clip, according to the dimensions: agreement, dominance, engagement, performance and rapport.	No	46	Yes
[GKN08]	VAM - Vera am Mittag German audio-visual emotional speech database	2008	Free, Request required	2D static, 2D dynamic and Audio	352x288	Some clips are continuous-valued for: valence, activation and dominance. Happy, anger, sad, disgust, fear, surprise and neutral are also typically identified.	No	104 speakers from which 20 have expression labels	Yes
[DGLG11, DGLG12]	SFEW - Static Facial Expressions in the Wild & AFEW - Acted Facial Expressions In The Wild	2011, 2012	Free, Request required	2D Static, 2D dynamic	-	Happy, surprise, anger, disgust, fear, sadness, contempt and neutral (loosely classified, e.g. smile and cheer behaviours are considered as happy). Some images/videos have multiple subjects	No	Based on 54 films	Yes
[WLL+10, WLW+13]	USTC-NVIE - Natural Visible and Infrared facial Expression Database	2012,2013	Free, Request required	2D dynamic	704x480	Posed and spontaneous DBs. Subjects display happy, angry, sad, fear, disgust and surprise under multiple illuminations. Spontaneous poses have onset and apex frames identified, with the later having the valence described in a three-point scale. Landmarks calculated by [JW12]	Yes (Could not find)	103 spontaneous, 107 Posed	Yes

Table 3.2: Survey of existing facial motion databases that were matched against: format and quality of information captured, metadata/motion labels present, being landmarked or not, size and if they contain head/gaze movements. (Part II)

Papers/Links	DB Name	Year	How to Obtain	Format	Quality of Format	Metadata / Motion Labels	Is Landmarked?	Number of Subjects	Contains Head/Gaze
[LCP+11]	UNBC-McMaster Shoulder Pain Expression Archive Database	2011	Free, direct online request	2D dynamic	320x240	Spontaneous pain expressions. FACS coded. Pain frame-by-frame scores, self-report and observer measures	Yes	25	Yes
[VDSHFD11]	ADFES - Amsterdam Dynamic Facial Expression Set	2011	Free, Request required	2D static, 2D dynamic	720x576	Anger, contempt, disgust, embarrassment, fear, joy, pride, sadness, surprise, neutral. These expressions were displayed to target the prototypical expressions which have the AUs defined. All expressions were displayed in frontal view and with the head rotated 45°.	No	22	No
[GP06]	FABO: Bi-modal Face and Body Gesture Database	2006	Free, direct online request	2D dynamic	1024x768	Posed neutral, anger, uncertainty, surprise, fear, anxiety, happiness, disgust, boredom and sadness.	No	23	Yes
[MVCP10]	SEMAINE	2010	Free, Request required	2D dynamic, audio	580x780	All displays continuously rated for: valence, activation, power, anticipation/expectation and overall emotional intensity. Displays also partially annotated for basic emotions: anger, disgust, amusement, happy, sad, contempt; 6 epistemic states (e.g. interested (not)); 10 interaction states (e.g. show solidarity); and 4 validity labels (e.g. social concealment).	No	20	Yes
[KCBW12]	The large MPI Facial Expression Database	2012	Direct Download	2D dynamic	768x676	55 acted facial behaviours	No	19	Yes
[RRÉM+06]	STOIC: A database of dynamic and static faces expressing highly recognizable emotions	2006	Direct Download	2D dynamic	256x256	Fear, happy, surprise, anger, sadness, disgust, pain and neutral. All faces were aligned based on 3 facial landmarks (pupil centers and tip of the nose)	No	34	No
[TLJ07]	RPI ISL Facial Expression Databases	2007	Free, Request required	2D dynamic	320x240	First DB contains the activation of 14 AUs; Second DB contains 34 landmarks, gaze tracking and activation of 15 AUs for the emotions of happy and surprise	Yes	First DB: 10, Second DB: 8	Yes
[OHS+05]	A Video Database of Moving Faces and People	2005	Free, via request; requires sending hard drive where the data is placed and sent back	2D Video and audio	720x480	Speech with facial expressions and gaze, 5-second clips displaying happiness, sadness, fear, disgust, anger, puzzlement, laughter, surprise, boredom, or disbelief, ”blank stare”.	No	284	Yes

Table 3.3: Survey of existing facial motion databases that were matched against: format and quality of information captured, metadata/motion labels present, being landmarked or not, size and if they contain head/gaze movements. (Part III)

The Cohn-Kanade database provides the core training data to test and validate our method. However, its samples are very limited, only containing neutral-to-peak motions. While this helps to validate the capacity of our method to generate transitions between peak poses (details Sec. 3.6.1), it is not enough to verify the capacity of the technique to encode more complex motions such as peak-to-peak movements. As a result, we have created a small database with more demanding samples. We have used *Faceware*©'s Analyser to extract the landmarks of: 6 videos with motions in the format $< P_{neutral} ... P_{peak\_expression} ... P_{neutral} >$ and 3 in the format $< P_{neutral} ... P_{peak\_expression\_1} ... P_{peak\_expression\_n} ... P_{neutral} >$. For coherency, we asked the subject to do the same emotional expressions as the ones present in the CK/CK+: *happiness, sadness, disgust, surprise, anger, fear*. The DB contains 62 landmarks that were reduced to 23, as shown in fig. 3.4. The use of this DB to train a motion graph provides a more realistic and generic use case of our method (more details in Sec. 3.6.1), while also allowing the analysis of the impact of complex motions in both the graph structure and the generated animation.

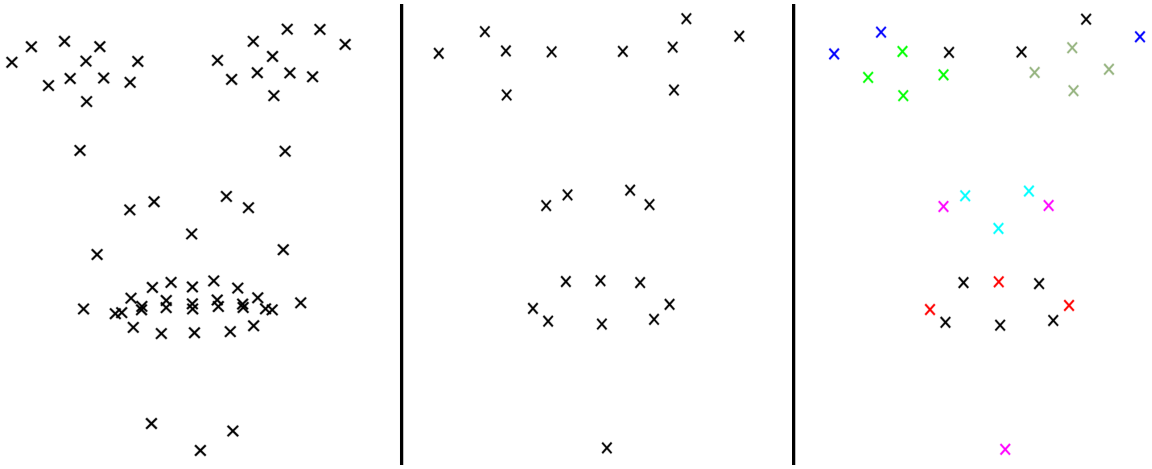

Figure 3.4: Small Database landmarks: from left to right, are the following: original 62 landmarks; reduced set used to create the graph; markers used for "non-rigid" alignment of the face regions, grouped by colours: blue for eyebrows; different greens for eyes, cyan for nose, red for mouth and magenta for jaw and cheeks.

3.3.2 Processing Samples

The CK/CK+ additionally required pre-processing to reduce errors/jitter. We have cleaned the data manually, and then fit a sigmoid to each landmark's displacement, per sample. For coherency, the same sigmoid fitting procedure was applied to the smaller DB. This process smooths the motion while keeping its individual variations and timings. The sigmoid function (defined in Eq. 3.1) was chosen as it incorporates the acceleration variations of human motion. The sigmoid is controlled by: b_1, curve's maximum value; b_2, steepness; and b_3

translation. Least squares (Eq. 3.2) is employed to find the optimal parameters for each sample data, with y_i being the landmark coordinates, and m being the number of points in the curve.

$$g_{b_1,b_2,b_3}(x) = \frac{b_1}{1+e^{-b_2(x-b_3)}} \tag{3.1}$$

$$\min_{b_1,b_2,b_3} \sum_{i=1}^{m} \| y_i - g_{b_1,b_2,b_3}(i) \|^2 \tag{3.2}$$

3.4 Motion Graph Generation

The core structure of our approach is a directed graph, where each node contains the average landmarks' displacement of all poses that were merged in it, and each edge has a similarity value (Sec. 3.4.2). Displacement was chosen as opposed to absolute landmarks positions to mitigate errors in alignment and effects of each person's facial proportions. This displacement is calculated from an *equivalent* pose in all the samples, i.e. the same facial expression, which can be e.g. a neutral face. This common pose enables expressions to have distinct global coordinates but the same displacement values, e.g. this same neutral pose will have different landmarks positions in different people, and even in separate samples of the same person, but the same displacement. The first node created in the final graph, additionally, contains the average landmarks of all DB common poses, $\overline{P_{eq}}$. This represents the base pose, which originates from the first pose, present in all samples, always having a displacement of 0 (on all landmarks). The base pose landmarks are added to the displacement of any given node, to create the actual pose represented in it. Each graph node also contains the expression labels of the sequences whose nodes were merged in it. We store additional information in the nodes to allow for better motion recovery (Sec. 3.5.1). This information is calculated from all the poses merged in each node, and consists of:

- Average number of consecutive merged nodes and their respective landmarks velocities. Consecutive refers to, when merging a sample graph into the final graph, a node from the latter might absorb consecutive n sample graph nodes. Both n and the respective landmarks velocities are stored. After the graph is created, these values are averaged;

- For destination nodes, i.e. nodes containing at least one peak expression, we store the average number transition's frames from neutral-to-peak or peak-to-peak expressions, and the respective standard deviation;

The proposed method relies on region graphs instead of the holistic approach, with only one graph. This allows more accurate similarity values, where the motion of one region does not affect the similarity value of another. It also leads to more compression, as each graph only stores its most significant motions. Following a holistic approach would, likewise, make the thresholds considerably harder to control, which would translate into the need of a finer grained optimization (Sec. 3.4.3). The regions chosen for this paper are: eyes, eyebrows, nose, mouth, with the cheeks and jaw grouped into another. This choice is based on observation of the samples' movements, although other configurations are possible.

3.4.1 Graph Creation

Graph creation is the same irrespective of the region. It first converts each sequence/sample, $< P_1 ... P_n >$, into a sample graph, g_{samp}. The connections between adjacent nodes/poses are established here, hence storing the temporal dynamics. The order in which the poses occur becomes part of the sample graph in the form of edges. Afterwards, the sample graphs are merged to create the final graphs, G_f (as shown in Fig. 3.5). In this step, similar nodes are identified and serve as transition points, effectively bridging one sample to another. As both stages are similar, we now describe the process of creating a sample graph (algorithm 1), with differences highlighted afterwards. Calculating the similarity between two poses/nodes and finding the optimal thresholds are described in the following sections.

A sample motion graph is then generated following these steps for each pose P_i in the sample (see algorithm 1): **1)** calculate the displacement, D, between current pose landmarks, P_i, and the sample's first pose P_1; **2)** add D to $\overline{P_{eq}}$ to create P_{i_new}; This converts the sample's displacement/landmarks into their counterparts in the final graph. The P_{i_new} is used to, **3)**, calculate the similarity value to all the current sample-graph nodes, $< SN_1 ... SN_n >$, and choose the lowest valued node, SN_{low}; if **4)** the similarity associated to SN_{low} is below a threshold (determined by Sec. 3.4.3), merge P_{i_new} into SN_{low}; Otherwise, **5)** create a new node, SN_{new}, and append it to the sample graph. In both cases, a connection is established to the previous iteration node.

Creating the final graph is an iterative process seeded with a random sample graph. Merging a sample graph with the final graph follows the same comparing-and-merge approach. However, when the nodes are merged, we use Procrustes Analysis to align the sample graph node with the absorbing final graph node, following the same groups of fig. 3.3 or fig. 3.4 depending on the DB. Uniform scaling is not considered to reduce changes in the shape. We additionally apply this transformation to all following sample nodes, since it will generally improve the similarity computation when merging them into the final

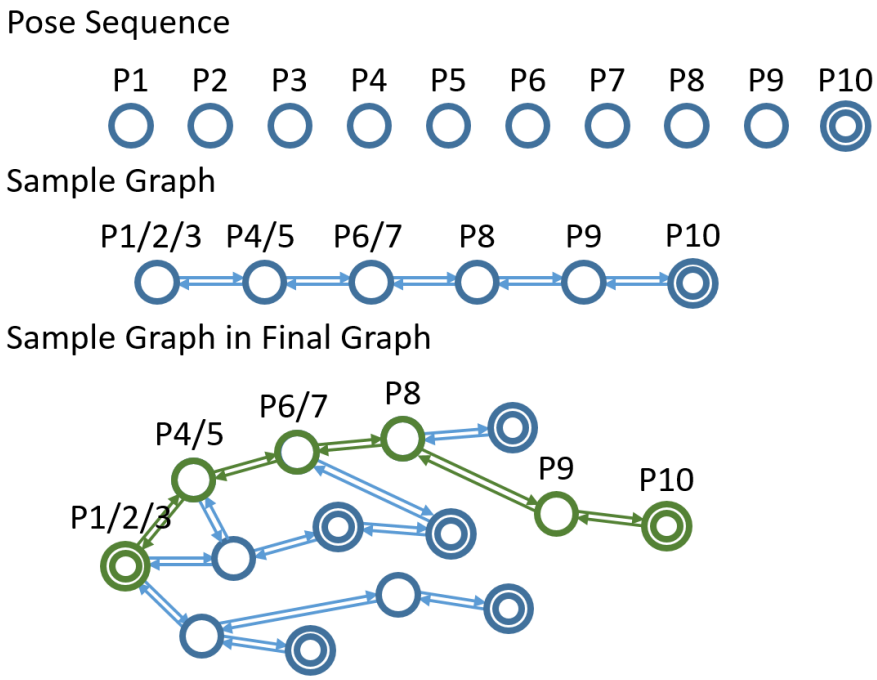

Figure 3.5: Example of what happens with a neutral-to-peak sample, through the whole graph creation process (only shown for one region graph). The sequence is first converted to a sample graph and is then merged into the final graph. Sample nodes containing poses P1 to P8 are all merged into existing final graph nodes. Double circle nodes represent destination nodes, and contain peak expressions.

graph. As the alignment is applied to the sample, there is no need to store or compute the transformations when synthesising motion.

3.4.2 Similarity Metric

The chosen similarity metric, Eq. 3.3, takes into account both spatial location of the landmarks and their instantaneous velocity, with lower values representing higher similarity. When comparing two different poses a and b, we define that each pose/frame P_a is composed by m number of landmarks $< La_1 ... La_m >$. Thereby, the distance between two poses for the same landmark i is given by $dist(La_i, Lb_i) = \sqrt{\sum_{j=1}^{dimensions}(La_{i,j} - Lb_{i,j})^2}$. The instantaneous velocity of a landmark i for the P_a is given by $\vec{V}(La_i) = La_i - Pa_{prev}La_i$, where $Pa_{prev}La_i$ is the position of La_i in the pose/frame immediately before P_a. On top of this, we calculate the velocity's influence, v_{infl}, which is a scalar that represents how similar are the velocities of a landmark i in two poses. This is given by $v_{infl}(La_i, Lb_i) = 1 - |\vec{V}(La_i) \cdot \vec{V}(Lb_i)|$. v_{infl} varies between $[0, .., 1]$: 0, if the two vectors are the same, independently of the direction; 1, if they form $+/-90°$. If both vectors are close to opposite, we argue they represent onset or offset phases, thus the influence should be the same. This is achieved using the absolute

Algorithm 1 Sample Motion Graph Creation

```
1:  procedure CREATESEQUENCESUBGRAPH
2:      Avg all equivalent sample poses >> P_eq (bar)
3:      for each P_i in < P_1...P_n > do
4:          P_i − P_1 >> D
5:          D + P_eq (bar) >> P_i_new
6:          CalcSimilarity(P_i_new, g_samp) >> < Sim_1...Sim_n >
7:          ChooseLowestSimNode(< Sim_1...Sim_n >, g_samp) >>
8:      Sim_low, SN_low
9:          if Sim_low < threshold then
10:             Merge P_i_new with SN_low
11:         else
12:             Create node with P_i_new and D >> SN_new
13:             Add SN_new to g_samp ¿¿ < SN_1...SN_n, SN_new >
14:         Connect SN_low/SN_new to prev. analysed node
```

value. The final similarity value is obtained via:

$$sim(P_a, P_b) = \sum_{i=1}^{m} dist(La_i, Lb_i)(1 + \lambda v_{infl}(La_i, Lb_i)) \tag{3.3}$$

λ controls the influence of the velocity (smoothness) in the result. We have kept the λ as 1 on all the obtained results. When this metric is computed between two graph nodes, the instantaneous velocity is calculated for each incoming edge, and the lowest similarity is chosen as the representative value. The similarity metric is used both when identifying nodes to merge and to set the graph edge weights. These weights are only obtained after the graph is finalised, as changing the connections to a node can change such values.

3.4.3 Optimizing for Compression

The author governs the graph by specifying the desired compressions, C_{samp}, C_{merg}, and the tolerances, t_{samp}, t_{merg}, for which it is acceptable to deviate from the respective compressions. Sample graph creation and sample graph merging have, thus, a separate set of arguments for finer control of where more information is merged. These values are the key attributes through which the author to controls the trade-off between motion quality (smoothness) and flexibility of the graph (compactness). Lower compression values lead to more information being kept since fewer nodes are merged. The thresholds, referred in Sec. 3.4.1, are independently optimized for each region graph and stage of integrating a sequence

into the final graph. Each optimization starts with an initial/default threshold, for which the sample graph or merging is simulated. The compression is then calculated and if it falls within $[C_{samp} - t_{samp}...C_{samp} + t_{samp}]$ (given the sample graph creation stage as example) the optimization finishes. Otherwise, the threshold is updated and the process repeated until the desired compression is achieved.

Compression is calculated differently for each stage. For the sample graph creation, this value is based on comparing the number of nodes, n_g, with the original number of sequence poses, n_p, by doing n_g/n_p. When merging, the compression is obtained by comparing the increase of nodes in a graph, n_{G_inc}, after being merged with a sample graph containing n_g nodes. $n_{G\_inc} = n_{G\_after\_merge} - n_{G\_before\_merge}$. The compression value is thus obtained via n_{G_inc}/n_g.

On top of the compression goal, the author needs to provide an initial threshold, thr, and step, st. The main question is, thus, how to update these values. This process is based on the current compression, $comp$, as shown in Eq. 3.4. If it is lower than desired, the threshold value is increased by the st. A higher threshold means more nodes will be merged, which increases the compression. If the compression is higher than the desired, then the number of nodes has been reduced too much. Hence, the threshold is decreased by st.

$$thr = \begin{cases} thr + st, & \text{if } comp < C_{samp/merg} - t_{samp/merg} \\ thr - st, & \text{if } comp > C_{samp/merg} + t_{samp/merg} \end{cases} \tag{3.4}$$

st remains the same while the addition or subtraction operation is repeated. On all thr updates after the operation is inverted for the first time, the step becomes $st = st/2$. This will continue until the desired compression is obtained or the step reaches a minimum value. We have set this 1, although this value can vary with the range of landmarks used. The downside of this approach is that a very small tolerance may lead the compression to not be met. Nevertheless, as each optimization approximates the defined goals, the final compression will generally approximate the target values.

Our method locally optimizes the thresholds for each stage/sequence, with the goal of achieving a desired global compression. This prevents the case where a threshold is too aggressive for one sequence but too lenient in another, which is the case of global threshold methods. Still, it is possible to configure our method to work as if it had global threshold, by setting the compression to 50% and tolerance to 50%. This means the initial threshold is used and any comparison between the number of nodes is valid.

3.5 Motion Synthesis

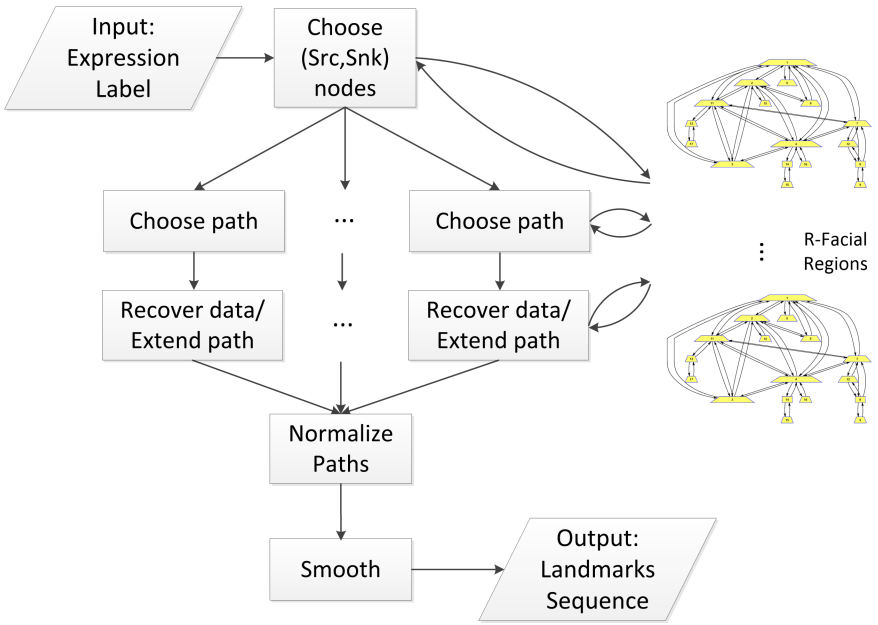

Figure 3.6: Overview of the synthesis process. Expression labels allow finding the desired sinks in each graph, with motions generated by path finding. Each node's information is used to recover the landmarks positions. After, all the region motions are assembled and smoothed to obtain the final animation.

Generating new motions (as shown in Fig. 3.6) now becomes a task of traversing the graph and choosing the nodes relevant to the desired facial behaviour. The author can directly choose the sources and sinks/destinations in all graphs, which provides very fine control. However, this approach requires a deep knowledge of the graph. As an alternative, we assume the nodes containing peak/apex expressions are the most desirable to animate a character. These nodes have been flagged and associated to the respective expression label when the graph was created. The author specifies the label, which is used to select the next sink via random sampling of a valid peak node. In a long animation, the current sink node becomes the source for the next iteration. The first source of the sequence is manually specified, which is easily done if the pose is a peak or neutral node. The path is chosen using Dijkstra's algorithm [Dij59], which minimizes the similarity values between the source and sink nodes. This is done for all region graphs to create a full facial behaviour. An example of this process can be seen at Fig. 3.7. While the path is the basis for motion, the actual landmark movements still needs to be extracted (Sec. 3.5.1). The path cannot be used directly as the temporal dynamics were partially lost, e.g. a sequence with 10 poses can be represented with only 3 nodes. We additionally explore the graph structure to introduce the uniqueness required to animate multiple characters (Sec. 3.5.2.1) that have repeated interactions with the viewer, and to generate idle motions, required when characters are on

stand-by or between other facial behaviour (Sec. 3.5.2.2).

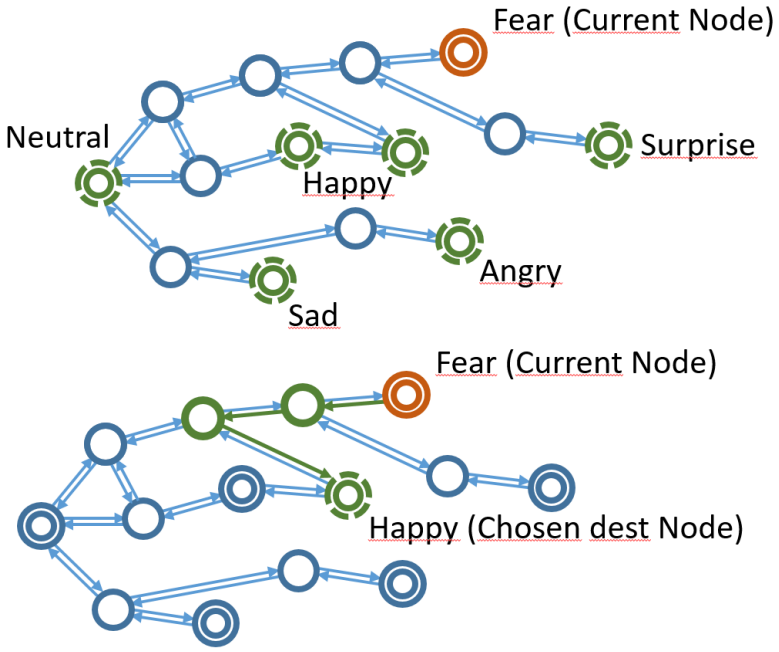

Figure 3.7: With a character currently displaying fear, the author just needs to choose the next expression label, e.g. happy. This is used to randomly select a sink node from all possible destination nodes (top graph). Dijkstra's algorithm [Dij59] is then used to find the path that connects the source and sink nodes.

3.5.1 Reconstructing motion

The data contained in each node (Sec. 3.4) allows approximating the information lost when the poses were merged. This process starts with Dijkstra's path nodes, from where the core poses and dynamics are determined (step 1). Afterwards, these are extended to achieve a more realistic motion (steps 2 and 3) and, finally, smoothing removes any existing jitter (step 4). These steps, detailed next, rely primarily on basic operations that require a small amount of data, keeping the stored data to a minimum.

1. Determine the number of poses generated by a node, using the average of consecutive merged nodes, and extract the poses' displacement using the average landmarks' velocity. This is crucial to adequately recover non-linear aspects of motion. Applying velocity depends on the number of poses to synthesise. Only one pose means no velocity is applied, as the pose is represented by the node's average displacement (Node 4 Fig. 3.8). With more poses, the node's displacement is used as the central point to which either the velocity, or its inverse, is added (Nodes 1,2,3 Fig. 3.8). This

happens as poses are respectively created in the direction of the next or the previous node. Poses created this way are clamped to the poses contained in the previous and next nodes, e.g. the middle node, of a neutral-to-smile motion, cannot create a smile that is wider than the smile contained in the middle following node. Nevertheless, this approximation will, in the limit, produce linear motions, which is the case when the desired compression is too high. Thus, leading the graph to no longer represent the dynamics of facial behaviours.

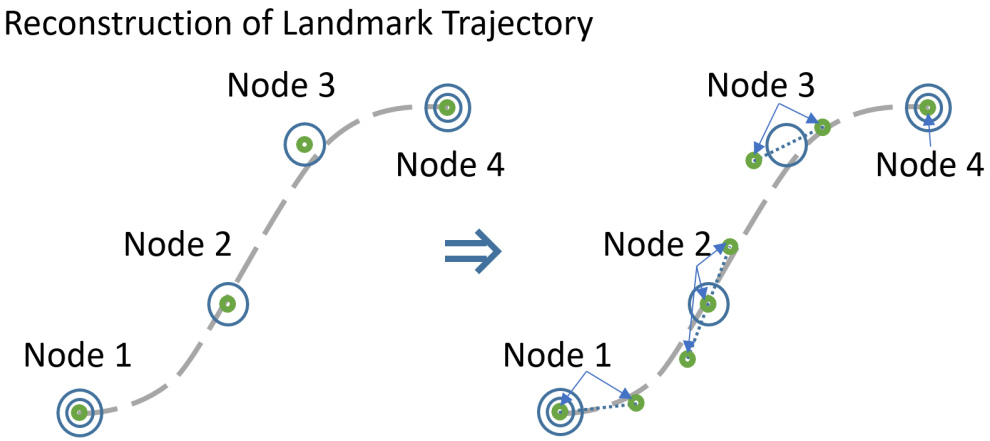

Figure 3.8: Example of the first reconstruction step in a landmark's motion, from a set of 4 nodes. The green circles show the average landmark position stored on each of the nodes (left) and the result of reconstructing its' positions using the data stored in each node (right). Different nodes generate different number of landmarks depending on the contained information. The sigmoid that results of all the steps is seen in both images, and serves as a reference of final landmark's motion.

2. Use the peak nodes average duration as the sequence length. The obtained sequence of displacements is stretched or shrunk accordingly, using linear interpolation. While the previous step recovers the shape of the temporal dynamics, the final duration is usually too short, as the nodes contain information from several samples, which has diluted the correct duration value. Using the peaks' durations allows more realistic synthesis. Graphs not trained with peak-to-peak transitions have an additional estimate exactly for this type of transitions. Such is done using the path node closest to the node containing the common/equivalent pose. The number of nodes between this node and both, the source and sink, serve as weighting factors of the respective peak durations. For example, with a path of 10 nodes, if the closest node is the third starting from the source, the weights of the respective peak durations will be 0.3 and 0.7.

3. Normalize the displacements of each facial region. Each path is created independently, hence, the durations of the region paths tend to be different. Motions are normalized by finding the longest path and stretching the others to match its size,

using linear interpolation. Linear interpolation is used here and in the previous step, however, only for the in-betweens of the curve formed in step 1. As a result, the motion as a whole does not become linear, and keeps its core motion properties.

4. Smooth the sequences using the Savitzky-Golay window-based filter [Orf96] and sigmoid fitting (Sec. 3.3.2). These approaches complement each other, as the first removes drastic motions, preventing the sigmoid from being too sharp, and the second removes any left zigzag. This tends to occur when generating sequences not in the DB or when path noise is introduced (Sec. 3.5.2.1). Sigmoid curves are associated to motions with slow in and out, which is the case of most movements of facial regions. However, they can approximate linear motions, such as blinking, as long as the compression does not make the blinking nodes merge with other labels nodes;

3.5.2 Exploring the Graph

Finding the source and sink nodes in a graph, and exploring them to generate motion, is the most obvious use for the available data structure. However, this can be explored in more unique ways, making full use of not only the information contained on each node, but also the relations between the poses, represented as edges. We now describe the properties of the structure that enable the generation of unique motions and how, on top of these, we further explored the graph to increase the differences between the generations, and to create idle motions. All without the need for additional training data.

3.5.2.1 Coherent Noise

Synthesizing animations that share a common label, but are slightly different, is a crucial requirement for our method. It allows creating a crowd that reacts, e.g. to a joke, with the same core behaviour but slightly different and idiosyncratic motions. Our approach inherently introduces noise, as each facial region has its own motion graph and respective path. Thus, a new sequence is composed by elements that originate from different samples. We force additional variations in the sequences' length and the chosen path. The first, relies on the standard deviation, stored per peak node, to define a normal distribution. We randomly sample this space and stretch/shrink the displacement accordingly. The path variations build on top of Dijkstra's path (Sec. 3.5), by randomly replacing path nodes with their neighbours. These connect to the original path nodes and, as such, contain poses similar to the ones being replaced. To reduce the chance of selecting a completely different pose, we only consider neighbours that share at least one label with the current source or

sink nodes. Noise is controlled by the percentage of the original path that can be changed, and the number of neighbour hops from which a node can be selected. Thus, coherent noise is added without disrupting the whole sequence. Nevertheless, depending on the graph, the resemblance of nodes might be broken with more than 2 hops, making the noise random. With these two approaches, enough samples in the DB, and the inherent variation of the graph, the number of unique sequences generated from minimum input is almost limitless.

3.5.2.2 Idle movements

Idle movements are crucial to create more realistic characters, as a complete lack of motion is unnatural [EMT05]. We generate this motions by finding the neighbours of the node associated to the currently displayed expression, and randomly hop between these. We limit the hops to neighbours that share the expression label of the current expression. This way, the graph structure is explored to find poses similar to the present one, generating coherent idle motions independently of the expression. These movements are learned purely from other behaviours and the similarity of expressions, which means there is no information on the temporal dynamics, as no references exist in the DB. Therefore, the author needs to manually provide the hops occurrence interval and the time remaining in each hop. The author can also control the intensity of the neighbours, i.e. if the pose of the character should completely change to that of the neighbour, or only be varied by a smaller percentage (obtained e.g. with linear interpolation). Thus, controlling how much of the neighbour displacement is applied. This is particularly relevant in graphs that have a high compression ratio, as even direct neighbours can represent a significant pose change, hence, not being directly usable for idle movements.

3.6 Results and Validation

We validated extensively our approach both quantitatively and qualitatively. The former, by matching the synthesized sequences against their original counterparts, for different compression levels. The latter by verifying the correct behaviour of specific aspects of the proposed method and empirically comparing the results against other state of the art techniques, i.e. face graphs of [ZSCS04] (fig. 3.15) and performance capture (app. A.2, fig. 3.16). The aspects of the method validated empirically include: difference between generated motions and original samples (fig. 3.9), capacity to generate non-linear motions (fig. 3.10), impact of path variation technique in coherent noise (fig. 3.11) and idle motions (fig. 3.12), impact of compression into the motion (fig. 3.13) and generalization capacity (fig. 3.17). Our procedural approach was implemented in *Matlab*© and tested on a laptop

with an i7-4720HQ and a NVidia GTX 970M.

Motion graphs were created and tested with 70 sequences from ~20 subjects of the CK/CK+ [KTC00, LCK$^+$10] DB and with a smaller DB containing 9 video sequences from 1 subject (Sec. 3.3.1). This DB allowed testing our method with more complex samples, and verify the impact of creating a graph with(out) the transitions between peak expressions, which are not present in the CK/CK+. It is, however, too small for a more in-depth validation of our method, and this is where the CK/CK+ comes in. We use it to compare the training samples, i.e. baseline, against the synthesized equivalent, for different compression ratios. In this context, we refer to compression ratio as the memory footprint gains that would occur if we compared our graph, with a graph containing a node per pose of the DB. Given the 70 CK/CK+ samples, this base graph has 1792 nodes. The results can be seen in Table 3.4, whose columns are: 1) motion graph creation time; 2) and 3) contain the different inputs used in the compression optimization respectively for the stages of sample graph creation and merging; 4) and 5) show the compression ratios in regard to number of nodes and in terms of data, i.e. landmarks and numerical values associated to motion recovery; 6) duration difference and respective deviation, between each new sequence and its counterpart in the DB; 7) landmark distances, and deviation, between each sample pose and respective synthesized pose; 8) angle difference, and respective deviation, between instantaneous velocities calculated for the same poses as the previous column. The results are obtained using the dot product and converted to % for easier interpretation. A perfect match corresponds to 0%, and a 90° between the velocities to 100% (i.e. perpendicular vectors); 9) synthesis times, and their deviation, required to generate a new sequence.

Aside from the quantitative validation, it is also important to verify how much the results translate into differences in actual, visible motions. To do this, we have generated and visually analysed the differences between several **synthesised motions and their original DB counterparts**. This matching was achieved by storing, on each graph node, an identifier (ID) for all the samples that have poses merged into that node. This way, it is possible to identify the peak node that contains the information of a specific DB sample, and trigger synthesis from the graph source (neutral pose) to that node. No noise is introduced in the synthesized sequences. Figure 3.9 presents two examples of this comparison for a motion graph with a compression ratio of ~85%. The upper sequence shows almost no deviation, while the lower displays visible differences, primarily in the animation curve.

Test No	Generation Time (seconds)	Input Compress. Seq. Creation (%)	Input Compress. Seq. Merge (%)	Nodes Compress. (%)	Data Compress. (%)	Seq. Duration +/- Deviation (fps)	Landmark Diff. +/- Deviation (pixels)	Angle Vel. Diff. +/- Deviation (%)	Synth. Times +/- Deviation (seconds)
1	3411,1	0,1+/-0,1	0,2+/-0,2	-54.07	22.74	0+/-0	1,43+/-1,96	14,49+/-13,49	0,39+/-0,45
2	5560,3	0,1+/-0,1	0,3+/-0,2	-10,27	44,22	0+/-0	1,67+/-1,967	14,85+/-13,77	0,49+/-0,60
3	4973.2	0,1+/-0,1	0,4+/-0,2	8,37	53,02	0+/-0	1,84+/-2,17	15,06+/-13,96	0,55+/-0,57
4	2859,6	0,2+/-0,1	0,5+/-0,3	38	66,8	0+/-0	2,09+/-2,40	15,64+/-14,37	0,72+/-0,57
5	1575	0,2+/-0,1	0,6+/-0,3	56,36	74,36	0,03+/-0,17	2,33+/-2,5	16,75+/-15,24	0,89+/-0,70
6	1280,4	0,2+/-0,1	0,7+/-0,3	58,76	75,66	0,029+/-0,17	2,39+/-2,62	16,83+/-15,25	0,90+/-0,74
7	1892,1	0,2+/-0,1	0,7+/-0,2	57,59	75,54	0,014+/-0,11	2,37+/-2,62	16,61+/-15,04	0,88+/-0,75
8	2253,8	0,2+/-0,1	0,8+/-0,2	60,94	77,66	0,043+/-0,21	2,46+/-2,61	16,89+/-15,37	0,91+/-0,73
9	1227,5	0,3+/-0,1	0,6+/-0,3	57,92	75,5	0,029+/-0,168	2,33+/-2,41	17,07+/-15,62	0,87+/-0,80
10	1038,2	0,4+/-0,1	0,6+/-0,3	60,38	77	0,057+/-0,293	2,30+/-2,58	16,40+/-15,14	0,84+/-0,71
11	887,5	0,5+/-0,1	0,6+/-0,3	62,5	78,2	0,043+/-0,27	2,12+/-2,29	16,54+/-15,4	0,88+/-0,76
12	1576,6	0,5+/-0,1	0,8+/-0,1	65	80,3	0.057+/-0,294	2,26+/-2,66	16,63+/-15,17	0,96+/-0,85
13	951,4	0,6+/-0,1	0,7+/-0,2	67,5	81,6	0,014+/-0,119	2,07+/-2,13	16,25+/-14,91	0,98+/-0,82
14	814,7	0,7+/-0,1	0,7+/-0,2	73,8	85,3	0.014+/-0,119	1,91+/-1,82	16,16+/-14,4	0,96+/-0,72
15	905,1	0,7+/-0,1	0,8+/-0,1	75,1	85,99	0,11+/-0,54	2+/-1,82	16,42+/-14,61	1,02+/-0,79

Table 3.4: Comparison between multiple graphs and quality of the sequences generated

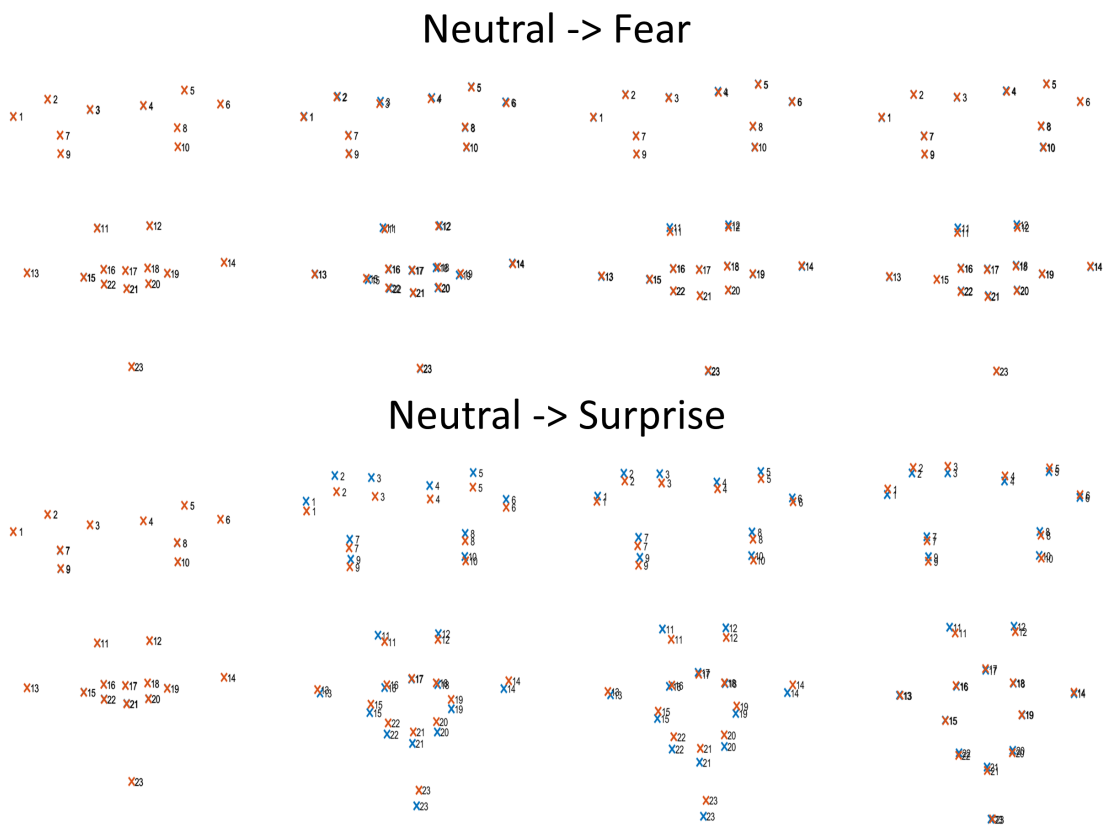

Figure 3.9: Example frames from the equivalent sequence in the original database (blue) and the synthesized version using our motion graph approach (red). The motion graph has a compression ratio of ~85%. The upper example shows almost a perfect match, while the lower already shows some differences.

Facial movements are fundamentally non-linear, and the proposed method has steps that specifically account for this (Sec. 3.5.1). The main problem here is the compression that can lead to so much information merged that the graph loses its capacity to represent non-linear motion. As a result, we have compared **synthesized motions with their linear equivalents** to verify how the method behaves. This test was done with a high compression graph, ~85%. It relies on generating a new motion and comparing it with a linearised version of the original DB sequence. This allows for a more accurate comparison, as the linearised version is closer to the original sample. The same ID technique, described previously, is used to determine the ID of the original sample. No noise is introduced in the synthesized sequences. Fig. 3.10 shows the two examples of this comparison, with the upper sequence showing more visible differences, while with the lower only displaying very subtle changes.

Tests have also been done to verify the correct behaviour of different aspects of our method and include: **impact of path variation in the coherent noise** (fig. 3.11), **idle motions** (fig. 3.12), **impact of compression** on synthesised motion (fig. 3.13) and **uniqueness of synthesis** (fig. 3.14). Path variation is analysed using the same source and sink nodes to

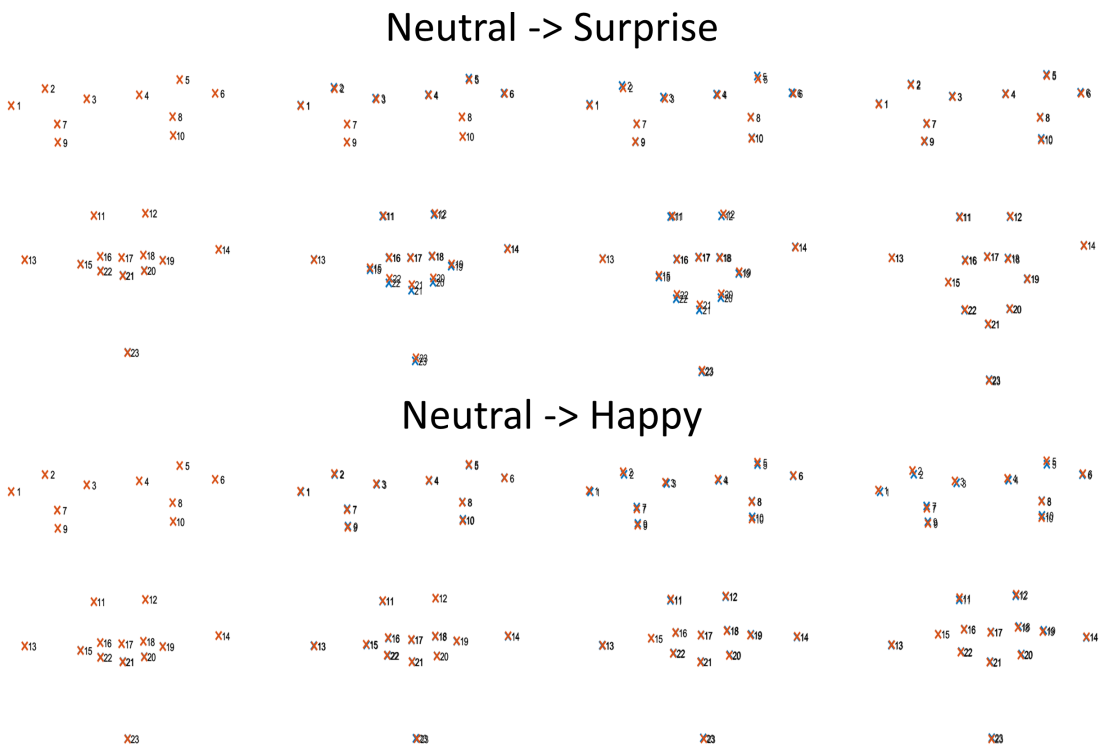

Figure 3.10: Example frames from the equivalent sequence obtained using linear interpolation from poses/timing of database samples (blue) and the synthesized version using our motion graph approach (red). The motion graph has a compression ratio of ~85%. The upper example shows a clear distinction between linear motion and the result of our method, while the lower shows very subtle differences.

generate two sequences, one that follows the pure Dijkstra path and another that additionally follows the process described in Sec. 3.5.2.1, with the limit of one hop and 50% of nodes changes allowed. Idle motions are tested by synthesizing a sequence and, between the peak expressions, introducing these small motions. This sequence is compared against one that contains static expressions on these intervals. The hops occurrence interval is set to 0.5 seconds, which means that each half a second the expression will change to another hop, neighbour of the current pose node. The duration of this transition is set to 0.4 seconds and the intensity of neighbour is 30%. These hops are limited to one neighbour. These values were found empirically. To understand the impact of compression in the motion quality, we have generated equivalent sequences in two graphs with ~85% and ~44% compression values. The sequences are matched again using the original samples IDs and without introducing any noise. Finally, we have verified the behaviour of our method in its natural and expected use case, i.e. having one motion graph generating multiple and unique instances of animation for the same input. This is the case of a storekeeper that interacts with the player multiple times in the same context, or the crowd that has multiple characters reacting differently to the same stimulus. We have used a graph with a compression ratio of

~85% with synthesis noise limited to one hop and 50% nodes' changes.

Figure 3.11: Example frames from the sequences generated from the same source and sink nodes but that do (red) or do not (blue) employ our path variation method, part of the coherent noise process (Sec. 3.5.2.1). The motion graph has a compression ratio of ~85%. The upper example shows a clear effect of using the path variation method, while the lower shows very subtle differences.

Aside from testing the behaviour of the method on its own, we have **compared it with the face graphs** of Zhang et al. [ZSCS04]. This approach was implemented in the same environment as our solution and trained with the same samples, i.e. CK/CK+. Effectively, creating a common ground that allows comparing the motion produced by both approaches. In their paper, the graph was trained with a single, long sequence of 3D face meshes that has the transitions between many expressions. The CK/CK+ has a different format (Sec. 3.3.1). Our approach for this was creating a single sequence from all the CK/CK samples, in the format of $< P_{neutral} \ldots P_{peak\_sample\_1} \ldots P_{neutral} \ldots P_{peak\_sample\_2} \ldots P_{neutral} \ldots >$. This is possible by converting all the samples into displacement space and then back to global coordinates using the average of all the equivalent poses. We matched sequences generated using the face graph technique with a motion graph that has a compression ratio of ~85%, following the previously described original sample ID technique. No noise was introduced. Figure 3.15 shows the results of these comparison.

Our approach is fundamentally different from traditional facial animation methods. Nevertheless, the final goal is the same. In this context, we have compared our **synthesised**

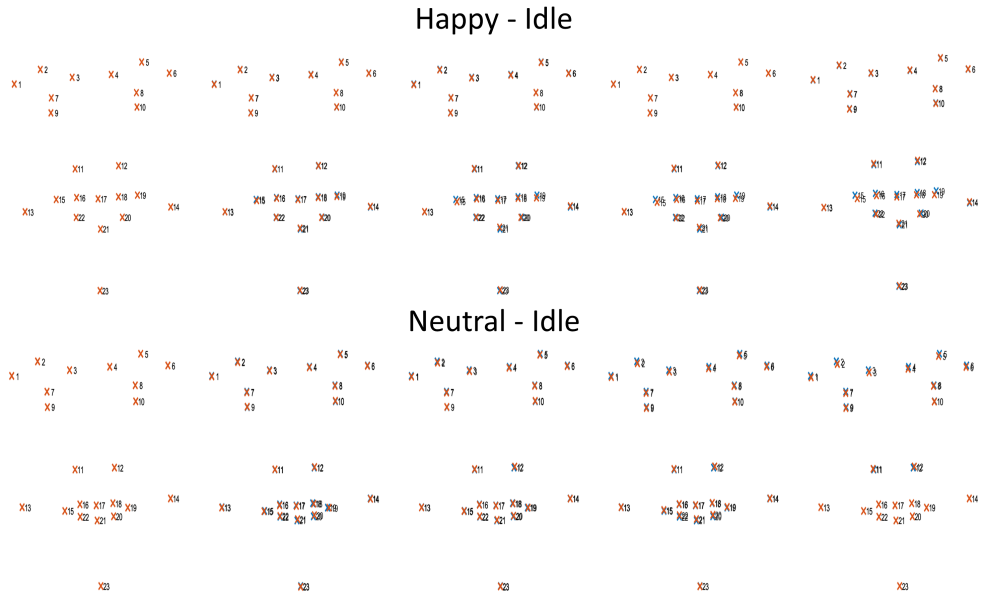

Figure 3.12: Example frames showing the differences between using (red) or not (blue) our idle motion technique (Sec. 3.5.2.2). In the latter case, the pose remains static. The motion graph has a compression ratio of ~85%.

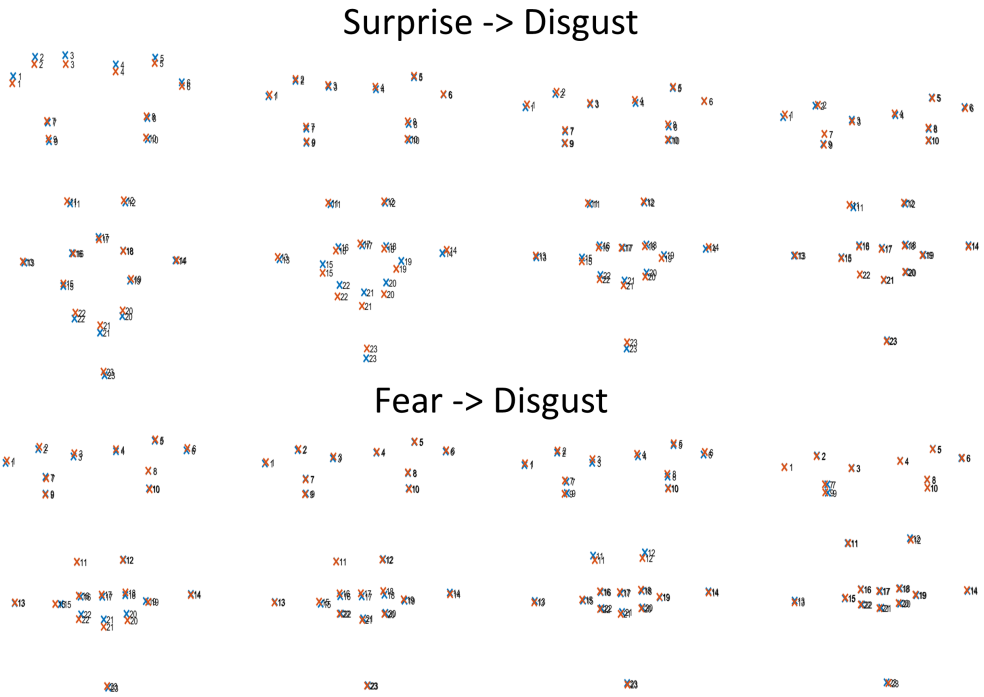

Figure 3.13: Example frames from the equivalent sequences obtained from a graph with a compression of 85% (blue) and one with a compression of 44% (red).

results with a markerless facial animation technique. The performance of a person is captured while executing the same order of expression labels as the equivalent synthesised sequence. Since it is nearly impossible for the subject to match the timing of our generated

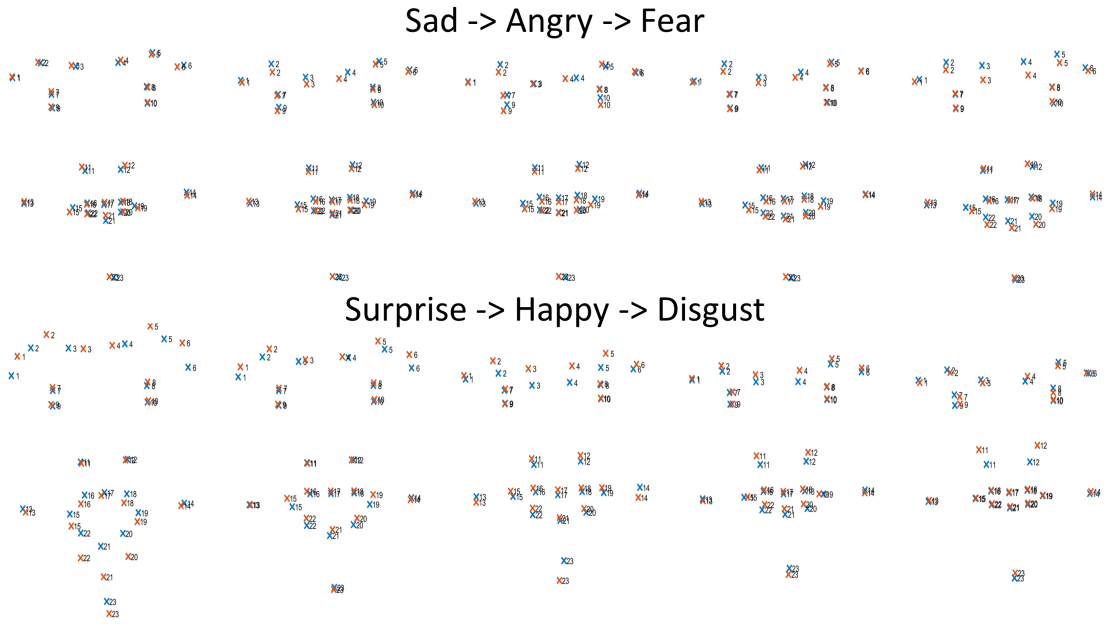

Figure 3.14: Example frames from the sequences generated using the same input labels from the same motion graph, with a compression of ~85%.

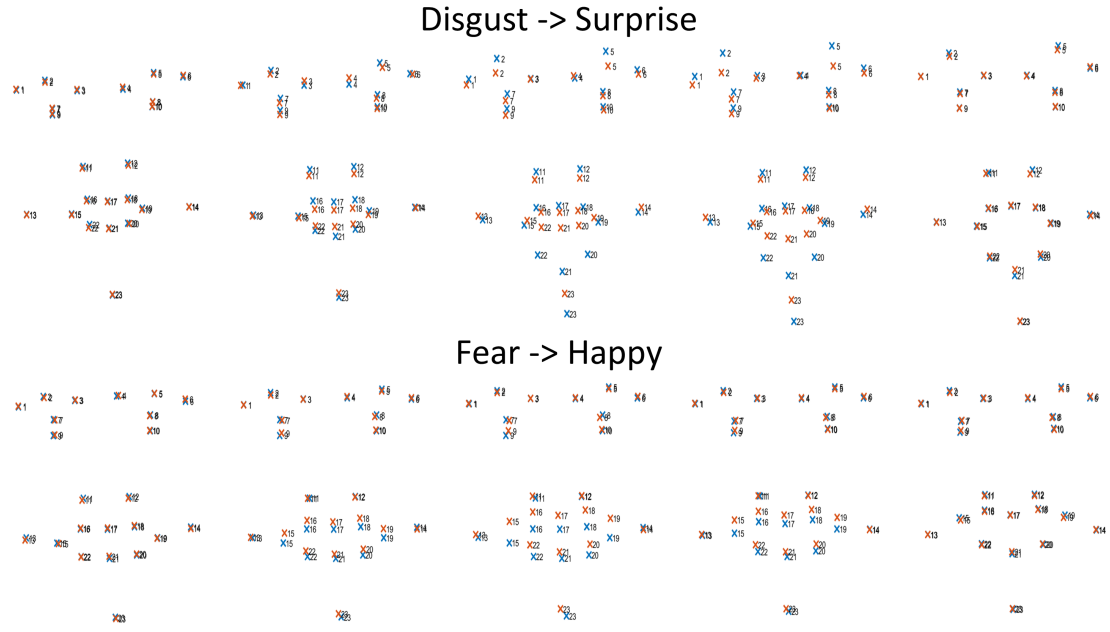

Figure 3.15: Example frames from the equivalent sequences obtained using our method (blue) and the Face Graphs of Zhang et al. [ZSCS04] (red). Both methods were trained using CK/CK+. The motion graph has a compression ratio of ~85%. The upper example shows an example where the Face Graphs introduce an intermediate pose. On the lower example, despite having different timing, the results are similar.

sequence, we have, additionally, manually aligned the frames so the motions start at the same time. The markerless technique is described in App. A.2, and combines the information from tracked landmarks with a personalized blendshaped model, as an optimization for

the blendshape weights. Our procedural approach only generates 2D landmarks positions, and, as a result, is not directly comparable. Hence, we use the direct manipulation approach [LA10], as described in App. A.1, to drive a 3D model from the landmarks. The sequences are imported in *Autodesk Maya 2011*© as locators that are associated to vertices in the surface. The blendshape weights are again determined by an optimization for the vertex motions. We have used this technique to generate 3D facial animation for 2 models, i.e. old man and victor from *Faceware*© [Fac15] (fig. 3.18), for all the qualitative test sequences. This allowed comparing markerless performance capture with our method, and have a rough approximation of the actual animation results produced by our method in the other scenarios.

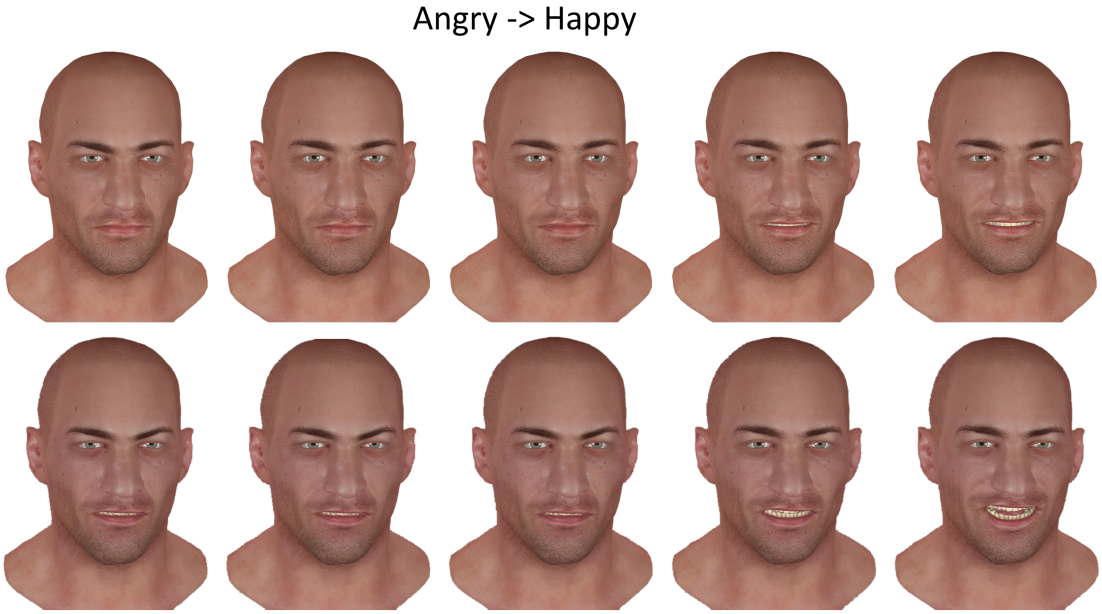

Figure 3.16: Example frames from matching our method's output (top) with a facial performance (bottom). We used a markerless technique (App. A.2) to capture the motion of the subject and aligned the timing for when the motions start. The motion graph has a compression ratio of ~85% (Model property *Faceware*©).

Finally, our method was **trained with the small database** (Sec. 3.3.1) containing neutral-to-peak and peak-to-peak transitions. This has both different motions and a slightly different set of landmarks, which allows validating the general character of our method. It, additionally, enables verifying the impact of creating a motion graph from a DB with, and without, the transitions between peak expressions, which are not present in the CK/CK+. Results can be seen at fig. 3.17, obtained from two graph with a compression of 80% and 82%, respectively. Matching sequences are again generated following the previously referred sample ID technique. No noise is introduced in the synthesized sequences.

Figure 3.17: Example frames from the equivalent sequences obtained using from two motions graphs trained with our DB (Sec. 3.3.1). The blue sequence was, however, trained only neutral-to-peak motions, while the red sequence contains additionally peak-to-peak motions. The motion graphs have, respectively, the compression ratios of 80% and 82%.

3.6.1 Analysis

Our goal with the previous results was twofold: verify that the proposed motion graph based method works as intended and is, indeed, a valid solution for the lack of animation in secondary characters, with all the different solution components contributing positively to this cause. We have thoroughly analysed all the results and now discuss what they mean in practical terms. The limitations of our technique are presented in the following section. We start with the quantitative evaluation present at Table 3.4, which provides insights on how different compression values affect motion graph creation and synthesis of animation. Regarding graph creation, data compression ratio falls typically within the boundaries defined by the user input. However, it is difficult to predict the exact values, since this is highly dependent on the content of the DB. The nodes' compression value provides a pure measure in terms of the number of nodes, which is good as light-hearted insight on the size of the graph. However, this is a "misleading" measure, as it accounts for the nodes of all the region graphs, hence, for lower compression values, the number of nodes easily surpasses the number of nodes of the baseline. Consider that, on the baseline, a full pose is represented by a node, which translates into 5 region nodes in our approach. Nevertheless, even with more nodes, similar landmarks have been merged into the same regional node, and as a result data compression already exists. Additionally, the structure representing the connections, which traditionally is a matrix, grows exponentially with the

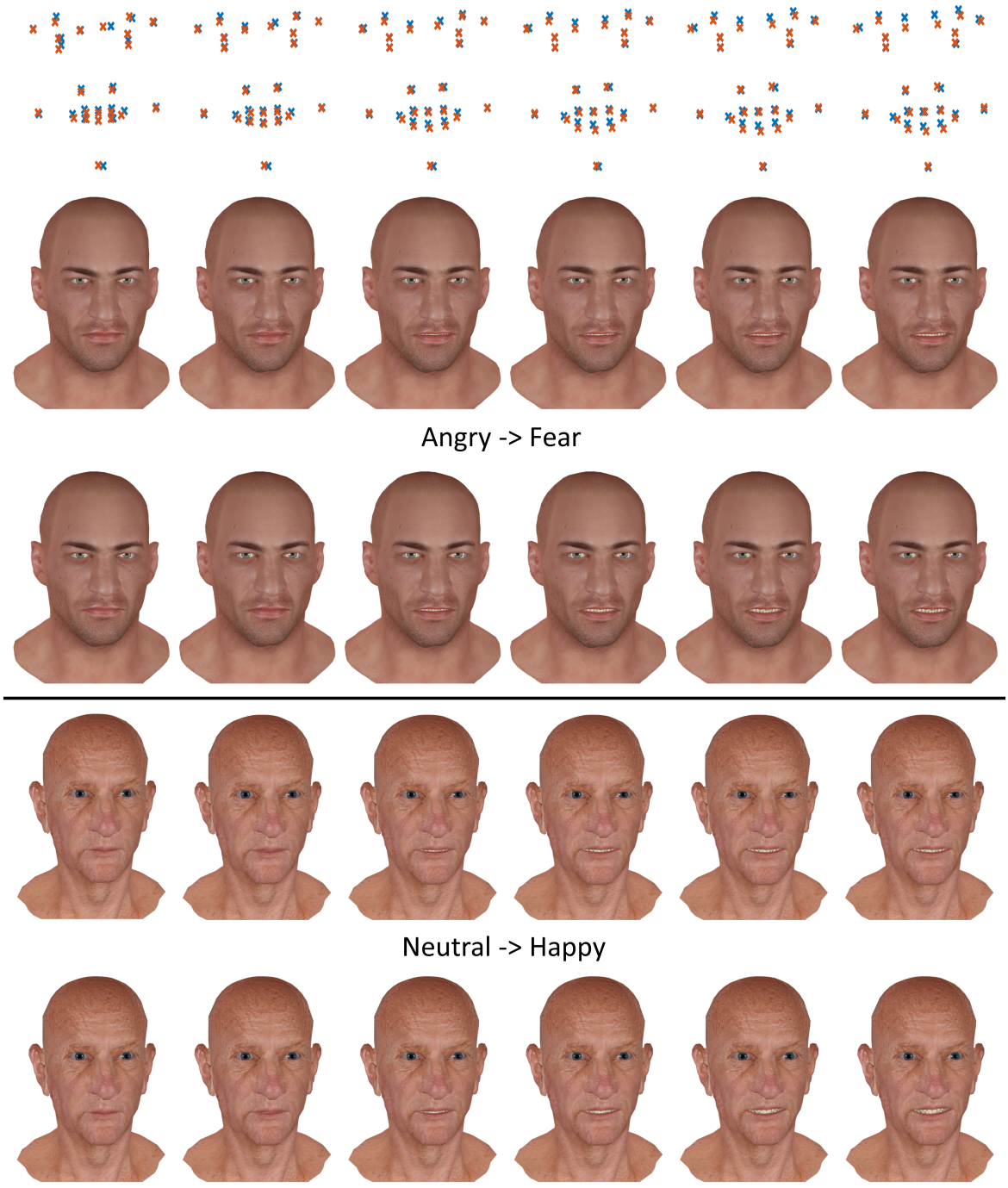

Figure 3.18: Example sequences generated from applying the synthesised landmarks onto a 3D facial model. In the first row it is possible to observe the actual landmarks associated to the following two model sequences. All the poses were extracted from the accompanying videos. The motion graph has a compression ratio of ~85% (Model property *Faceware*©).

number of nodes of the graph. Thus, significant savings occur by creating a smaller matrix for each region graph, instead of a matrix that connects all DB poses. The processes of

creating and using the motion graphs are also subject to local minima, whose impact can be seen across almost in all columns. An example is the case of the data compression of 85.3%, that recovers motions more similar to those in the DB than other lower compression cases. Here, the structure of the graph represents the core poses better, which leads to better motion recovery when compared with other structures that have lower compression. Nevertheless, a lower data compression will generally represent the motion better and require more time to be created. This behaviour will be similar in larger DBs, i.e. more samples will lead to longer generation times, with the size of the graph following the desired compression.

Regarding the differences between original and synthesised sequences, the duration shows almost no loss of information, which is explained by the choice of storing the peak duration. Differences in the landmarks position have an error ~2-4 pixels, which is acceptable given the range of landmarks positions, ~270 pixels in the x-axis, and ~380 pixels in the y-axis. Finally, the velocity error ranges between 10 to 30%, which we consider acceptable given the achieved compression. An interesting consequence of our method is that for the lowest compression tests, data recovery is still not complete. This results from some information being lost in merged poses, which is, possibly, only recoverable at extremely low compression values. In which case, the extra data (Sec. 3.4) would lead to more space being used than the original data. These results confirm the method significantly reduces the DB size, while keeping the lost information to a minimum. Also relevant is the synthesis time that decreases as the graph size increases. This occurs due to bigger graphs requiring less time to recover information, due to faster convergence of the sigmoid fitting method. Additionally, the path finding time for this type of graph and size is negligible. In larger graphs, that originate from more comprehensive DBs, the path finding will start to impact the synthesis time. However, we expect this to still be acceptable, as Dijkstra's algorithm [Dij59] performs well in sparse graphs with a complexity of $\mathcal{O}(Elog(N))$, where E is the number of edges and N the number of nodes [CLRS09]. This is the case of the created graphs, as facial poses are connected and followed only by similar poses. The results of these tests allow us to confirm that, even for high compression ratios, the proposed technique is capable of encoding the training DB and synthesising motion similar to the original data. Thus, providing a solid base for new motions generated using this same structure. Additionally, synthesis is performed near real-time, which leads us to believe that porting this technique to a game engine would easily surpass the real-time constraint of interactive applications.

As referred previously, most facial motion is non-linear. As a result, it was important to verify that our method is, indeed, capable of generating motions that have non-linear characteristics. We have confirmed that this effectively occurs in almost all the generated

sequences, as seen in the top motion of fig. 3.10. However, the differences between these two types of motions are generally very subtle, specially when the face transitions between more similar poses. This makes even more sense given that the duration used for the linear interpolation is obtained from the original samples, and our method is capable of recovering this information very accurately, so they will almost always match. As a result, there is a limit to the difference that exists between linear and a non-linear facial motions. This subtle different is shown e.g. in fig. 3.19 extracted from Schmidt et al. [SACR06], which shows the displacement of the lip corner during a smile. If this motion was linearised using the same start (frame 2) and the same end (frame 20), the visual differences would not be significant, specially when seeing all the face landmarks in motion. Our method will, however, sometimes generate linearised motion (previously referred at Sec. 3.5.1 - step 1). This will occur in high compression graphs, for some regions and motions. In this case, extreme merging has occurred and whole motions are encoded in a very small number of nodes. We believe this limitation is not severe, and will rarely occur for sensible chosen compression ratios.

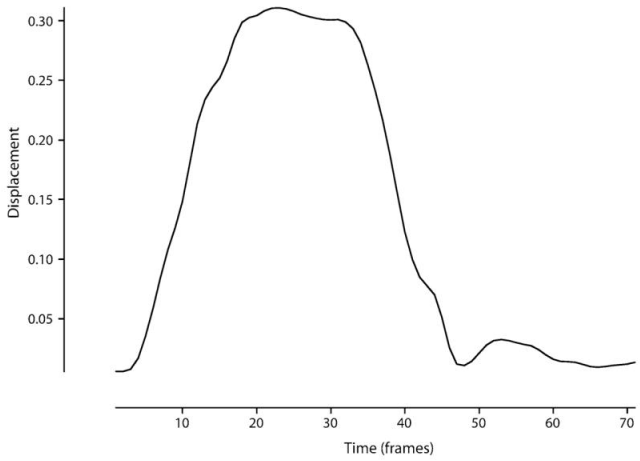

Figure 3.19: Lip corner displacement during smile, extracted from Schmidt et al. [SACR06].

The capacity to generate new and unique animations was evaluated by generating results with and without the path variation process (fig. 3.11), and by providing a graph with the same input twice (fig. 3.14). Despite the small role played by path variation, within the noise process, it is still important to analyse its real impact. Our results (fig. 3.11) show that this method mostly influences the shape of the animation curve. This is expected given that both the start/end poses and the duration of motion are unaltered by this process. These influences are particularly visible in the top row of fig. 3.11. Some steps in this process are based on randomness, which means that not all paths will be altered enough to produce visible changes into the final motion. This is accentuated if only a small fraction ($<25\%$) of the nodes are replaced with their neighbours. Despite small, the changes in motion in-

troduced by path variation will accumulate with the other aspects of our method, effectively contributing to the generation of unique animations. The most significant contributor for generation of unique animations is the natural combination of samples that occurs within the method. Each synthesised sequence blends the motion for each facial part that can originate from any training sample. In the limit, with 5 facial regions and three sufficiently different samples of the same behaviour, our method can produce 75 different motions. This value is obtained following the assumption that the moment a facial region changes its motion, even if all the others remain the same, the viewer will consider the final movement as a new one. However, this level of variation will generally not occur, as different displays of the same behaviour will have a certain % of the poses/motion that is similar. Hence, being merged in the same node when creating the graph. Also, our method relies on randomness in the choice of destination nodes and paths, which means that it is unavoidable to, sometimes, generate the same motion from the same input. However, this case is increasingly rare with a higher number of training samples and size of the motion graph. One issue that was also encountered was the strange combination of regional poses. It can occur when poses from different labels are merged into the same node, which might not affect evenly across all the regions used to the generated pose. While in some combinations theses conflicts are not noticeable, in other poses they can lead to strange expressions, which in turn could fall into uncanny valley. Based on our tests, this problem is rare and was significantly mitigated by considering both eyebrows as a region, and doing the same for the eyes. These tests provide the foundation for the proposition that our method can generate unique animations from the same input, which is crucial for characters that react differently to an event/player, in the same context. We believe our method provides enough variation even when a small DB is employed to train it. Still, these differences will only be further intensified with the application of motion to the different facial models, which is the traditional case with interactive applications and films.

Synthesised idle motions bring to life moments where short clips of animation are not being triggered. Although shorter, they also contribute to create more realistic apex motions. We have confirmed that our method behaves as expected in both cases. It does, however, require fine tuning to generate motions coherent with the graph. This is the case, specially, of the intensity with which the poses of the neighbours are used. A graph, with a higher compression, will have neighbours that contain poses more disparate, from the currently displayed pose node, than a graph with a lower compression. The connectivity of the graph also impacts the quality of idle motions, as low connectivity, i.e. only one or two neighbours, will limit idling to a very reduced number of animations, which in turn can lead to noticeable repetitions. Such will occur particularly with nodes that contain more extreme poses that

are not common to other training samples. This is solved by creating the graph with more training sequences. For the CK/CK+ training data, we have noticed this issue specially when idling the apex poses of wider happy and surprise behaviours. On the other hand, having the neutral pose as the first pose on all the samples leads the respective node to be highly connected. This presents itself as an advantage, given that characters will, typically, display a neutral motion when no events are happening. In which case, our implementation generates varied neutral movements.

The use of motion graphs for facial animation is scarce, with the only known exception being the work of Zhang et al. [ZSCS04]. This method relies on a graph, i.e. face graph, created by connecting all poses in the training sequence, and each node contains a facial mesh. Our approach shares some concepts with this approach. However, we extract meaningful connections from the samples instead of connecting all nodes, which reduces the graph size considerably. Similar to [ZSCS04], we also traverse each region graph. However, we vary the chosen paths for coherent noise. Finally, expression labels facilitate specifying the nodes, thus easing the control of the animation. In terms of the quality of motions, the results are similar with different issues. Face graphs produce motions that occasionally have distinguishable phases, i.e. there is a mid-expression, usually neutral, when going from expression A to B (top row fig. 3.15). Our approach sometimes produces distinguishable region timings, which can also occur in Zhang's [ZSCS04], although less distinguishable. Both can occasionally generate an exaggerated pose jump. The proposed method's main advantage is the compression, as a face graph is created by connecting all poses with all other. We also present an intuitive mechanism to control the synthesis of motions, however, their approach could easily be extended in the same way. This indeed fully recovers the motions from the DB, however, it also leads to a longer path finding, usually taking ~1.9 seconds.

Before delving into the comparison between our method and facial capture, we need to address the process of driving a facial model using landmarks. This entails retargetting the motion so it fits adequately the model being animated. This step is, traditionally, carefully crafted by an artist and, if not done properly, can introduce errors in the resulting motions/poses. We have, nevertheless, created this mapping as best as possible, however, due to lack of experience, it is clearly sub-optimal. This means that, while some expressions, e.g. happy, will map properly to the model, some others, e.g. sad or disgust, tend to be barely distinguishable. Despite their landmarks clearly showing differences. This extra step is not a contribution of this thesis, however, we believe it provides a mean to compare the results with the more traditional approach of performance driven animation and to previsualize what the final results could look like. Hence, these should serve only as a rough approx-

imation of the final animation, with better results achievable with artistic help. Regarding the comparison with face capture, the latter tends to produce more expressive and natural results, with the actor making poses that better match the model shapes. This comes at the cost of noise that leads to strange motions. Our method produces considerably more well-defined and smooth motion. However, it also leads to more stiff and less expressive motion. Generation via motion graphs is also limited to the poses present in the DB, which does not occur with performance capture. On the other hand, our approach requires far less input to create an animation, eliminates the need for additional equipment and the actor, which considerably reduces the animation cost. Performance capture and procedural animation share some challenges, such as pose alignment, however, these are significantly different, with distinct requirements and constraints. As a result, our goal with this comparison is just to provide a reference point on how both fare.

Our solution has been tested with two different DBs. The reasons for using the CK/CK+ and manually creating a smaller DB are described in Sec. 3.3. They do, however, have different issues, which makes their selection less than ideal. The CK/CK+ was not created with facial animation in mind, the poses are sometimes too subtle/small to be applied to a model, leading to animations that are hard to recognize. The landmarks are either not enough, or properly placed, and contain some jitter even after cleaning. It only has motions from neutral-to-peak, whose direct use would lead to a graph that only generates paths to peak expressions, i.e. dead-ends. This issue is "*bypassed*" by appending to the end of a sample, its own reversed copy. Nevertheless, facial behaviours have different onsets and offsets [ACR09], thus, ideally, full behaviours should be used. Only having neutral-peak-neutral transitions proves that our method can learn the dynamics of transitions, but it does not mean these are the most correct ones. The new DB allows testing our method with a different set of landmarks and, more importantly, more complex peak-to-peak sequences. Hence, proving that our method is, indeed, general enough to be encode and represent complex facial behaviours. This additional data has also allowed us to study the impact of peak-to-peak motions in both the generated motion and in the created graph. It should be noted that the quality of the motion is still subject to the same issues of subtle/small motions and the markers positions not being optimal, e.g. no cheeks or reduced number of more rigid markers. Our results show that the impact of training a graph with/without peak-to-peak transitions is mostly visible in the curve between two poses. It is, however, hard to judge which motion is more realistic, in which case further studies would be needed. Timing changes do prove that, as peak-to-peak samples are added, new connections between existing nodes are formed. Thus, creating a more deeply connected graph, which is exactly the expected behaviour of our method. Small differences in duration also confirm that our

timing approximation (Sec. 3.5.1 - step 2), for non-existent transitions between two labels, is a reasonable one. Nonetheless, it is preferable to have some training samples with these transitions, which leads to more realistic timing. Small differences also occur between the poses generated by the two graphs. This originates from having a person display the same expression multiple times. Despite the pose being nearly the same, it is almost impossible to achieve exactly the same pose every time. These slightly different expressions tend to be merged into the same node, which translates into the final pose represented, by that node, being different in the tested cases. Such behaviour is natural given the inner-workings of our method, and confirms that, as a graph is created, similar poses are merged into the same node. What this DB falls short is, however, the number of samples and lack of interpersonal variation, capable of enabling a proper validation of the full method. Despite our goal of trying to match the facial behaviours in both DBs, in hindsight, this validation would have benefited from the capture of other types of facial behaviours. Nevertheless, and while limited to a reduced number of emotional expressions, these DBs allow us to test different aspects of our approach, and show its potential that would only increase with larger, more varied DBs.

3.6.2 Limitations

Our solution is, nevertheless, not absent of limitations, both found through empirical analysis and by the previous tests. We start with our method's heavy reliance on well-defined and distinguishable peak expressions. Facial behaviours do not always have these perfectly defined. Subtle motions, for example, are not very wide or expressive and could, potentially be merged with the in-between poses of other larger expressions. Thus, partially losing their meaning. More complex motions such as visual speech might require an extremely large graph, which is not ideal. Our approach to align and reduce the effects of individuality, via Procrustes' analysis, is also not sufficient to completely remove all the effects of different facial proportions, and can sometimes introduce peculiar motions. High compression values can also lead peak expressions, of different labels, to being merged in the same node. This translates into the character displaying the same pose for multiple labels. Using sigmoid fitting to smooth the results can also lead to excessive removal of the individual variations, hence removing a layer of uniqueness. Facial behaviours include not only expressions, but also gaze and head movements, which currently cannot be represented with our graph structure. We consider this method is fully validated on its own, however, it would have been a great complement to the presented tests, to have embedded it in a game engine. This would allow comparing how much more "believable" (or not) would the secondary characters be, against the traditional approach. This is a very demanding task in terms of

resources, specially artistic, which we were unable to supply. Finally, while not a limitation, a seed sequence is randomly selected and used as the starting graph. This makes the graph slightly different each time it is generated.

3.7 Discussion

In this chapter, we have presented, validated and discussed a novel procedural facial animation approach. It enables generation of unique animations, on-the-fly, with minimum input. Region motion graphs encode the movements present in a DB. Traditional procedural approaches have a lengthy configuration step, however, we enable the author to control directly how much the original data should be compressed via simple parameters. Our approach optimizes the merging thresholds for each sequence, which leads to a better representation of the training data, while also simplifying the set-up process. Synthesis is based on a provided expression label, which is used to find relevant nodes in the graph. The nodes between these form a path, obtained via minimization, that provides the information required to reconstruct the final motion. A new node structure enables both a compact representation and accurate synthesis.

The proposed method is capable of generating an almost limitless number of unique sequences, due to the choice of having separate region graphs, and by introducing small variations in the minimum distance path. This, together with generating idle motions from the neighbours of the current pose node, shows innovative ways of exploring the graph structure. Additionally, an existing graph is easily extensible, since a new sample can just be merged into it, as long as it follows the same landmarks configuration and has the same reference pose.

We have validated both the individual components and premises of our solution, and their combined behaviour. Our method was matched against the closest state of the art approach [ZSCS04] and other types of animation techniques. We are confident that our technique advances the state of the art in procedural facial animation, while still being easy to configure and intuitive to use. Thus, opening the doors to more realistic and expressive secondary characters.

Chapter 4

Behavioural Facial Animation using Mind Maps and Motion Graphs

Defining character behaviour is a tedious and laborious task that involves extensive discrimination of all the actions and their relation with stimuli. Even more so when dealing with the emotional processes that occur within a person and influence the choice of behaviour in different situations. This issue is only increased when dealing with facial motions, which can require a manual specification of expression intensity. We introduce a novel method that explores the use of mind maps as generic behaviour controllers. This abstract concept provides an intuitive interface to define the relations between the character's internal state and both stimuli and their reactions. It is ideal when precise control of the reactions to stimuli is not fundamental, such as the case of interactions with secondary characters present in crowds. This technique can be used with any action, be it a body or facial display. However, we further leverage our knowledge of the motion graphs to create a full end-to-end behaviour facial animation system. We have validated our control mechanisms as two separate proof-of-concepts, which have been tested for usability and performance, and the results empirically analysed. By combining behaviour with facial animation synthesis, we further simplify and reduce the time required to animate a character. After reading this chapter, you should have a full grasp of our generic controller and how it is combined with the previously described animation system. You should also understand its potential to create more reactive characters with less effort than existing techniques, which, in turn, can shift the focus of animating a character to one of directing it.

4.1 Problem Statement

A fundamental part of creating truly believable and amicable characters is the way they externalise their feelings, via motions or animations. But before playing the animation it is actually necessary to choose it and, if need be, specify its input parameters, such as intensity. Commercial applications, e.g. videogames, traditionally approach this step using scripting and/or behaviour trees (BTs) [Cha07, Ras16]. The former relies on conditions and rules that extensively relate stimuli with actions. BTs describe these same conditions and rules in the form of a tree, which decomposes the choices and actions into smaller parts. Hence, making the definition of behaviour easier and more extensible, via reuse of the parts. While different implementations of BTs are possible, a tree will generally contain: actions, which in our case are the motions; conditions that control the flow of execution, taking into account the stimuli and world states; and finally, tasks, which define sub-regions, i.e. sub-trees, that contain groups of actions, conditions or even other tasks, related to a specific group of behaviours. The use of BTs with very large behaviours can be overwhelming, as seen in fig. 4.1. In the same sense, a large number of conditions in scripting become hard to manage and extend. Finally, both approaches support coherent generation of different reactions from the same stimulus, by introducing additional conditions/variables that force variation within a certain range of choices. These variables can include an internal state of the character, however, this approach is far from transparent and can lead to an increase of the complexity on the way each approach is used. Although BTs enables simpler, more intuitive, definition of behaviour, when compared with pure scripting, both approaches still require extensive discrimination of the relation between stimuli and motions. Nothing prevents the creation of realistic and emotionally rich characters using these techniques, it just means that the complexity and maintenance costs are higher than creating a character with only basic emotional behaviours. The implication of this is, however, similar to the one described in the previous chapter, i.e. more complex behaviours becomes limited to hero characters. Especially when dealing with emotional facial motion that is usually not seen as fundamental to the unfolding of the told story. Hence, leaving secondary characters to always choose the same action to a stimulus, and even have different characters that react exactly the same way. Our solution manages to overcome these issues by introducing the use of mind maps as behaviour controllers. These provide an intuitive interface capable of choosing different behaviours from the same stimulus, without requiring the definition of all pairs of stimulus-reaction.

Behaviour choice in commercial applications generally sits on top of animation layer, just setting in motion the desired action, at the correct time. However, there is much to be

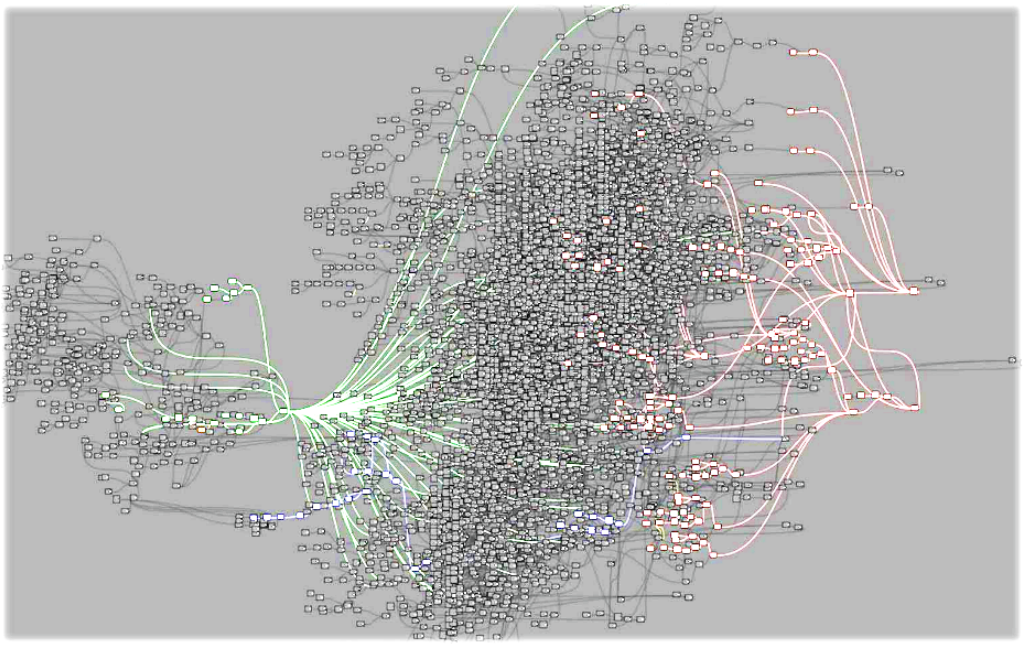

Figure 4.1: Example of behaviour tree used in Tom Clancy's The Division© [Gil16]. Different colours represent different tasks.

gained from exploring the synergies between the different layers. Research has already followed this path, with some approaches falling within hybrid procedural animation methods (Sec. 2.2.3), however, the path still remains widely unexplored. Approaching this task requires expertise from multiple areas, but it is worth the effort as creating a full end-to-end behaviour animation system can lead to significant gains in the configuration time and expressiveness of the characters. State of the art techniques achieve this by relying on some kind internal state that keeps track of what happened and enables the character to have more complex reactions to the same stimulus, given at different moments. The main limitation of these methods lies in their extensive [PG96, Per97, ONPS05, AD07, BSG10, SBR$^+$14] and complex [AD07, SBR$^+$14] setup; Our approach relies on mind maps, which provides a visual counterpart to scripting techniques [PG96, Per97, AD07, SBR$^+$14], hence making the definition of behaviour more intuitive. Similar to [ONPS05, BSG10, SBR$^+$14], we also explore synergies between the internal state and the animation being synthesized. Sagar et al. [SBR$^+$14] present, arguably, the most advanced work on facial animation generation, which comes at the cost of the complexity associated to gift the model with new behaviours. In which case, the author needs to re-train the model. Creating new behaviours on other approaches requires considerably less expertise, where the goal is to provide the characters with just enough emotional and cognitive complexity. Additionally, because the proposed approach aims to simulate the brain, unexpected behaviours might arise like in real inter-

actions, which can produce undesirable results in e.g. a videogame. Our approach, still requires some configuration efforts, however it is significantly less laborious and complex to configure than other end-to-end behavioural facial animation techniques.

4.2 Solution Overview

We propose a generic behaviour controller that uses mind maps to encode the behaviour of a character. The author first creates a hierarchical graph, with its emotional and action layers (Sec. 4.3), defining the relation between the character's internal state and both the stimulus and actions (fig. 4.1). On this stage, the author also sets the character's personality traits, such as initial mood and personality. As stimuli occur, the character's internal state is updated (Sec. 4.3.2), which, in turn, triggers a new action to be chosen (Sec. 4.3.1).

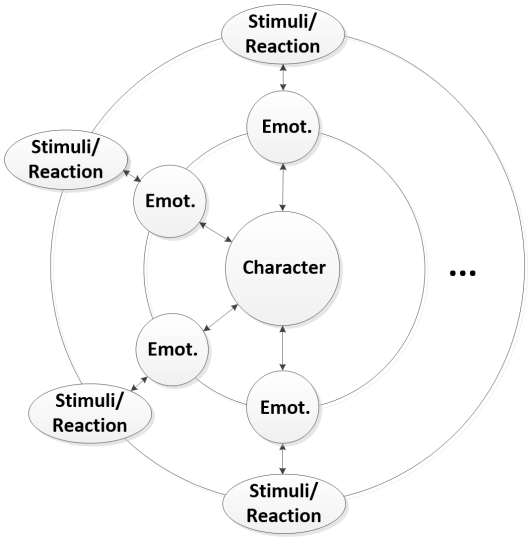

Figure 4.2: Simplified representation of a mind map. The outer ring contains the stimuli/actions that the character can be subject of or display. The middle ring contains the emotions to which the stimuli/actions are attached. Based on these connections, the character's internal state is updated and triggers behaviours.

We have then combined the generic controller with our procedural animation approach (ch. 3) to create an end-to-end behavioural facial animation system. The full system works in the following way: there is a set-up stage, where the motion graphs are created from the analysis of a DB (A) and the behaviour defined using the mind maps interface (B), and a runtime/motion synthesis stage, where the behaviour is chosen and the motion generated. This process is outlined in fig. 4.3. The creation and use of motion graphs and mind maps has already been described, the only missing part is the actual connection between them. This is achieved via two bridges (Sec. 4.4): a direct connection, with the animation labels

used as actions in the mind map (Sec. 4.4), and via the introduction of specific reaction nodes that trigger smaller, more subtle motions, using a revised version of our motion graph-based synthesis method (Sec. 4.4.1). These *insight animations* occur after a stimulus forces an update of the character's mood, and while the mood is, then, decaying into the character's personality. To achieve a more realistic character, our system also includes idle motions that are constantly introduced (Sec. 3.5.2.2, 4.4) when no other motion are displayed.

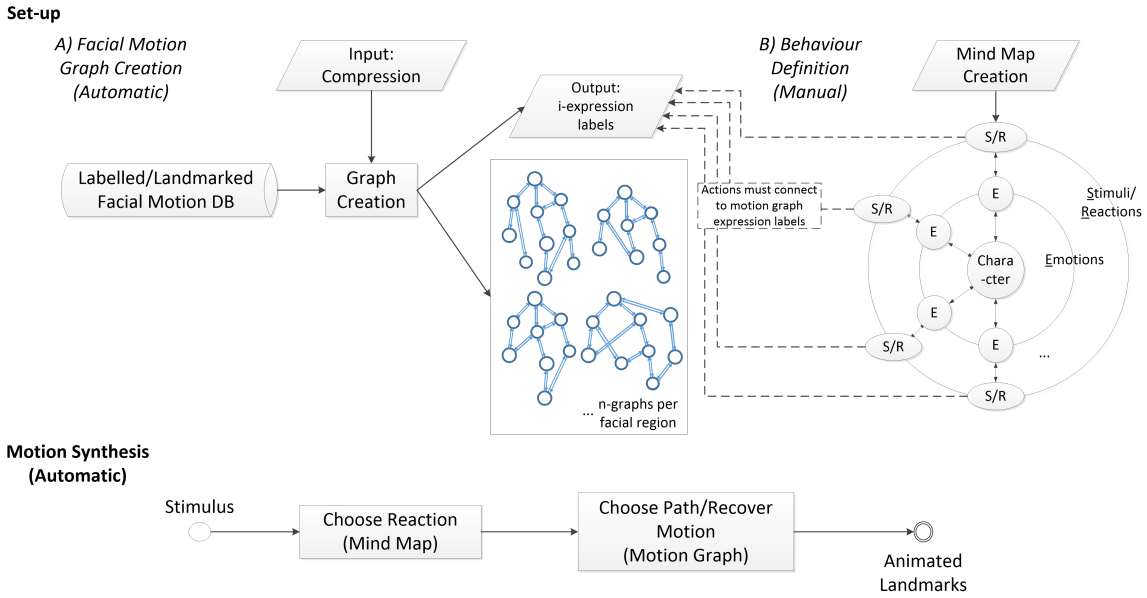

Figure 4.3: Overview of the our behavioural animation pipeline. In the set-up step, the motion graphs are created by analysing the DB and matching the user provided DB compression. The mind map is then created and the motion labels provided from the DB are used as reaction nodes. As stimuli occur, the mood is updated following the rules present in the character's mind map, which in turn trigger the appropriate reaction/animation. With the label chosen, the new motion is synthesized by traversing the graph.

4.2.1 Solution Features

Our full system for behaviour definition and facial synthesis has many advantages that make it useful to create more believable and autonomous characters. We now focus only on the features inherent to our behaviour controller and those resulting from combining it with the procedural approach of ch. 3. Nevertheless, because the presented behavioural animation solution is based on the previously described work, all of its features (Sec. 3.2.1) are also built-in into the current approach. We do, however, abstain from referring them again. Finally, and similar to section 3.2.1, we distinguish between features of the system and its technical contributions (next section). Our solution's features are then:

- Intuitive definition of behaviours using a mind maps interface;

- Varied behaviour without the need to discriminate all pairs of stimulus-actions;
- Varied behaviour when facing the same input stimulus, coherent with the mood of the character;
- Behaviour that accounts for the character's personality, location, source of stimulus;
- Reaction to stimuli not present in a character's mind map, by learning it from other characters;
- Additional layer of expressiveness with insight animations that provide a peek into the character's mood;
- Complete solution to create emotionally expressive autonomous characters.

4.2.2 Solution Technical Advances

We advance the state of the art by introducing:

- A novel mind maps based interface for defining the behaviour of a character. It works by indirectly establishing the relation between stimuli and reactions using the character's internal state as a common ground. This interface enables intuitive definition of behaviours and does not require an extensive discrimination of all combinations between stimuli and reactions. It is, additionally, generic, allowing any type of stimulus or action to be used, as long as they connect to the internal state, via an emotional layer.
- A new behaviour facial animation system that tightly integrates behaviour control with unique, on-the-fly facial motion synthesis. This approach reduces the time required to create more reactive and emotionally complex characters (when matched against other commercial and state of the art approaches). Furthermore, the combination of these two components allows the creation of insight animations, which are produced with barely any increase in the configuration time.

4.3 Mind Maps as Behaviour Interface

A mind map is a network of connected and related concepts [Dav11]. It often starts with a key, central concept, that is connected to related ideas, which in turn are connected to other similar ideas. Hence, having a network that branches out from a single core concept. This is an abstract notion that we have implemented practically as a hierarchical graph. The central

node represents the character and connects to an emotional nodes layer, which in turn, connects to an action/stimulus nodes layer. Emotional nodes can be any kind of emotion, such as happiness, or assume references to other psychological or mind states, e.g. bored or absence of emotion (neutral). Although the number of emotional nodes is not limited, as their number increases, so does the complexity of creating and predicting the behaviour of a character. Despite the possibilities, emotions are a common denominator between behaviours, being a natural bridge between actions/stimuli and the character's mood. And it was in this context that the first incarnation of this research was born [FSOnO12]. It revolved around the 6 basic emotions of Ekman and Friesen [EF71] and how to help autistic children to better identify these emotions [Cen11]. This is the reason why we have continued to call this bridge as the "emotional layer" of the mind map. Above this layer sits the action layer, whose nodes can be further divided into received, akin to stimuli, and performed, akin to reactions. Performed nodes can represent any type of behaviour, be it body and facial motions, or even actions such as run away to home (which could occur after a scary event). In the same sense, receiving nodes are also abstract and can represent any type of stimulus label. It is up to the author to decide the most coherent and suitable labels, which establish the connection between the character and the medium it is inserted in. The relation between action and emotional nodes is established via directional edges. All stimuli must be connected to at least one emotional node, although more connections are supported, with each edge having an influence weight. The weights represent the impact of the stimulus on the character's internal state. Multiple connections enable more complex influences in the mood, as opposed to only one connection, which would lead to pure "black and white" stimulus. For example, receiving a gift can at the same time be surprising and make the character happy. Performing actions have two types of edges, both in the opposite direction of stimulus nodes. There are the edges that start in an emotion node and end at a reaction node and edges that connect two performing nodes. The former also has a weight, which represents the mood required to trigger that behaviour. We refer to the later as *chain reactions*. The reasons for this edge are twofold: allowing more atomically performed actions that can be reused in different behaviours; and more complex reactions that would not be possible with only one edge, e.g. having a character that is so scared that first shows a fear motion and then starts crying.

The mind map provides the interface to define both how a stimulus affects the character's internal state, and how the state is used to choose a behaviour. As for the internal state, it is composed by the current mood and personality traits. The mood is represented by a weighted array with one entry per emotional node. The mood's values are updated after a stimulus occurs and when it is decaying towards the personality (Sec. 4.3.2). Afterwards,

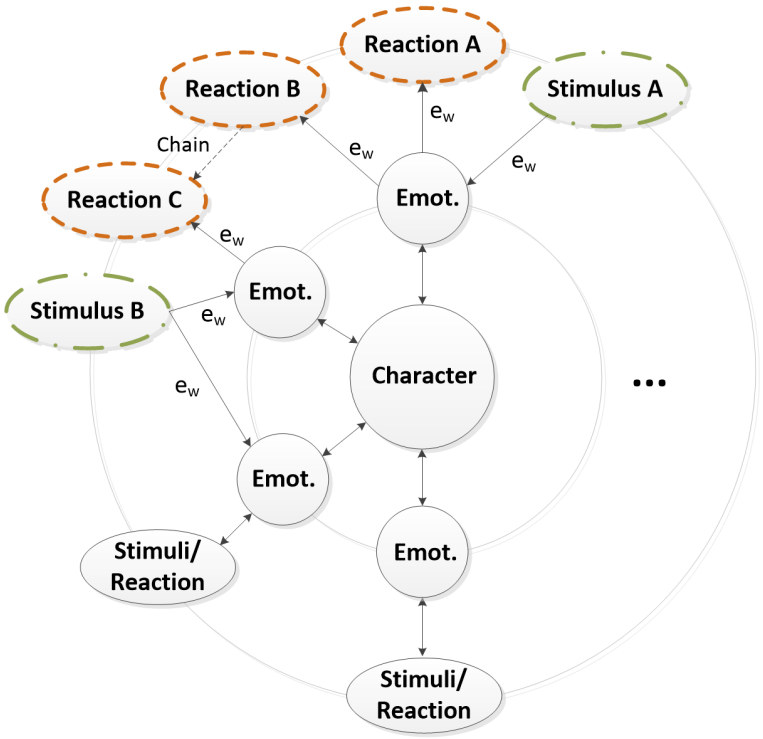

Figure 4.4: Representation of a mind map showing a possible configuration of stimuli (green) and performing nodes (orange). e_w represent the different weights on the relation between the stimuli/reaction nodes and the emotions of the middle ring. Finally, it is also possible to see a chain reaction between reaction B and C.

they are used to trigger a performing action, or chain (Sec. 4.3.1). Personality traits consist mainly of different arrays, with the same dimension of the mood, that influence different aspects of the mood update process. These include:

- Personality is an array (with the same dimension as the mood) that influences the mood update process and to which it decays after a stimulus occurs. The later requires an additional parameter that specifies the duration of the decay. The dynamics are controlled by a curve that can be e.g. linear, quadratic or sigmoid;

- Background context is an array (with the same dimension as the mood), which represents the influence of the environment in the mood. It adds another common factor into the update process and enables configuring characters that are e.g. happier in sunny environments;

- Affection matrix holds independent mood arrays towards other characters. These can be manually specified, or created automatically from the history of the interactions with other characters, e.g. using a weighted stimuli/reaction average;

While the starting mood and personality are mandatory, the background context and affection matrix are not.

4.3.1 Triggering Behaviour

A reaction occurs whenever a stimulus happens, akin to the perception-decision-action loop [DDLT02] of behaviour generation. This process, thus, starts with a stimulus that is found in the mind map and the emotion weights obtained from the edges (1); these weights are used in conjunction with the personality traits to update the current mood (Sec. 4.3.2) (2); finally, the highest value present in the mood array determines the relevant emotional node, from where the performing action is selected (3). The choice is based on edge weight closest to the mood value (fuzziness is added for similar edge values). The full overview is presented in fig. 4.5. Additionally, as characters interact, it is possible that a character faces a stimulus not present in its mind map. When this happens, the character will copy the stimulus and respective emotional weights from the mind map of the stimulus source. For world stimuli, e.g. an explosion, the particular stimulus should contain the emotional node and edge weights to be copied.

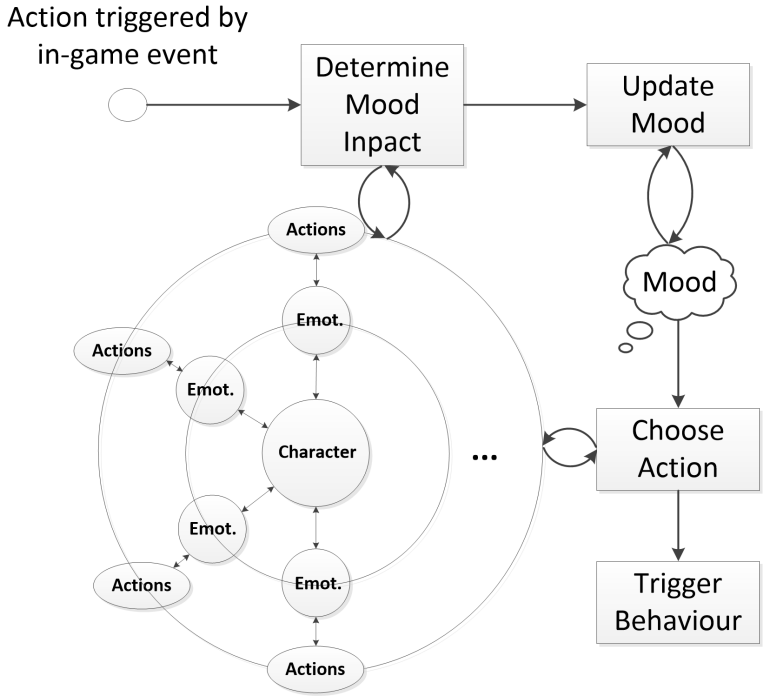

Figure 4.5: Overview of the mind map triggering process. As an action/stimulus is triggered in the game, the mind map is analysed to find how the stimulus affects the current mood. After the mood is updated, the performing action is chosen and triggered.

4.3.2 Updating Internal State

Updating the mood consists in a simple weighted sum of all properties, with fine tune possible on each component's influence. The equation 4.3 was split purely for better readability. The *mood* represents the current mood, *ActionEmot* is the array formed by the stimulus edge weights (0 for emotions without edges), with λ controlling how much the affection towards another character, i.e. $Affect$, can change the perception of the stimulus. *Personality* represents the influence of the base personality in the stimulus and *BGContext* deals with the influence of the current location. α controls the importance of current stimulus over the personality traits. Finally, β regulates how much the received stimulus can change the current mood. The resulting mood is normalised, as to maintain the sum of all values as 1. Without the normalisation, several consecutive stimuli from the same emotional node branch would make the mood too biased towards that node.

$$Stimulus = (1-\lambda)ActionEmot + \lambda Affect \tag{4.1}$$

$$PTraits = \frac{Personality + BGContext}{2} \tag{4.2}$$

$$mood = |(1-\beta)mood - \beta(\alpha Stimulus + (1-\alpha)PTraits)| \tag{4.3}$$

After the mood is updated, it starts decaying to the base personality. This endows the characters of the capacity to cope with the stimuli and not remain static in the mood after a stimulus. The author controls the decay process by specifying its duration and dynamics. The latter is achieved via an animation curve that can be linear, quadratic or sigmoid-based (or be changed to alternatives, according to the author's desires).

4.4 Combining Behaviour and Animation

Research that explores synergies between behaviour and facial movements, although rare, already exists. However, it tends to have a complex set-up and can sometimes still rely on a small group of animations, whose blend weights are controlled by the approach's behaviour controller. In this context, combining our two methods (ch. 3 & Sec. 4.3) is an obvious evolutionary step, and effectively allows achieving the proposed goal of this thesis, i.e. *explore methods to synthesise non-repetitive animation, on the fly, driven by stimuli*. The mind map acts as the character's brain and controls the input for the motion graph technique, which in turn generates singular animation near real-time each time. The bridge between approaches is two-laned via: direct association, where the author adds the expression labels that control our procedural method as reaction nodes in the mind map;

and insight animations, which are triggered as the mood decays to the personality. We focus on the former now, with the latter presented in the following section (Sec. 4.4.1). In the direct association, after a mood update occurs, the mind map will choose the appropriate expression label. This, in turn, is used to generate an animation from the character's current expression to the one associated to the label. After this animation finishes, if there are chained reactions, the mind map will trigger each reaction label after the completion of the previous. With all the reaction nodes done, the mind map triggers a "wrap-up" animation that forces the character back to a base pose, traditionally neutral. All the animations can start immediately as soon as the previous finishes, or be delayed by a t time, which represents the expression's apex. t can be statically defined or sampled from a small interval. Macro expressions, such as the ones present at the CK/CK+ DB, will typically have a total duration that varies between 0.5 and 4 seconds [Ekm03, MH11, YWL$^+$13], which includes the onset, apex and offset phases. Based on tests from previous chapter, motions (onset and offset phases) will generally have a duration of ~1 second. Therefore, we can consider the upper limit of t as 2 seconds, or determine it on-the-fly based on the duration of the motions synthesised. Yang et al. [YWL$^+$13] have determined, for micro-expression, that the onset's lower limit varies between 65ms and 260ms, while the total duration of micro expressions varies between 170ms and 500ms. This effectively means that there can be an instantaneous apex or close to it. However, this study was not done for macro-expressions. In this sense, we believe a more sensible lower limit of t would be at least 0.2 seconds, which makes peak expressions more visible. The combination of behaviour and our on-the-fly facial animation generation brings one other advantage at virtually no cost. It is not impossible for stimuli to interrupt the behaviour generated by the occurrence of previous stimuli. Whenever this occurs, the mind map only needs to send a new expression label to the motion graph, which is capable of generating an animation from any current pose. This is done by determining the node in the path associated to the current pose, and just trigger a new animation that uses that node as the new animation source.

Characters are, however, not constantly bombarded by events and stimuli. Even when the character in doing some activity that produces some stimuli or interacting with other characters, there is no guarantee that the flow of events is high enough to trigger another animation after the previous finishes. Since animation is primarily produced when a stimulus occurs, then the character's face would remain static the rest of the time. And, as Egges and Magnenat-Thalmann [EMT05] refer, a complete lack of motion is unnatural, which leads directly to the uncanny valley. Our solution, thus, introduces idle motion (Sec. 3.5.2.2) whenever the character is not reacting to a stimulus and while in the apex phase between two synthesised motions.

4.4.1 Triggering Insight Animations from Mood Decay

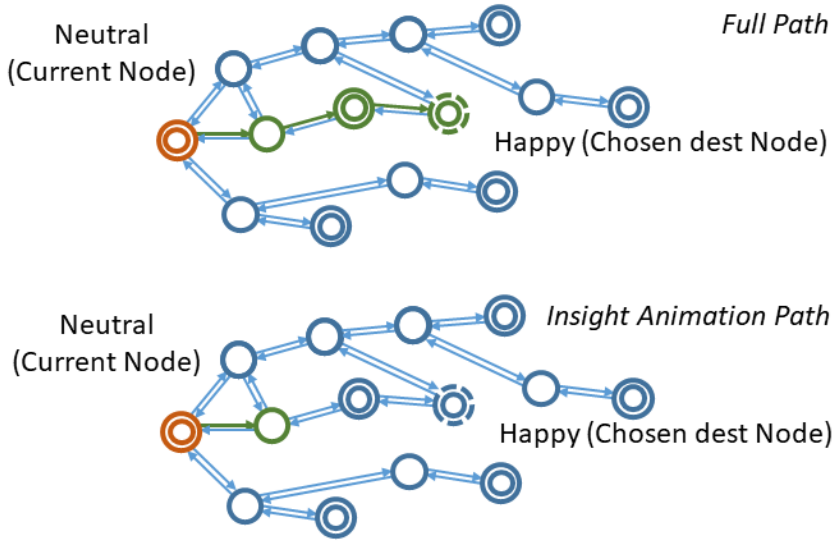

Figure 4.6: Our base approach for motion generation uses the expression label to find a sink node in the graph, which is connected to the current pose node via the minimum distance path. Insight animations explore the same principle, but instead of recovering animation from the full path are limited to its initial nodes.

The term *insight animations* originates from the goal of these motions, which aim to provide some intuition on the character's current mood. They are subtler versions of the motions generated by our procedural method that are triggered when the mood is decaying. One such application example would be a character showing occasional smirks when happy. This enables the player to better choose how to interact with the character, as he/she knows what is the underlying mood of the target of the interaction. On the behaviour side, these animations result from extending the mind map with flagged reaction nodes. The author selects/flags one performing action per emotional node that is triggered at certain intervals, t, as the mood decays (Sec. 4.3.2). We consider this action as the representative motion for when the character is predominantly in that mood. t can be based on: a static number of seconds, an interval from where t is randomly sampled or even the % of the decay duration already elapsed. The actual choice of reaction to trigger is similar to the one described in Sec. 4.3.1. In which case, the current mood is used to determine the emotion node associated to the highest mood value, from where the representative reaction is chosen. As for the motion synthesis part, our goal was not to generate full-fledged animations, only hinting the current mood. Therefore, the process (Sec. 3.5) starts the same way, with the path determined between the current expression and the sink node. However, instead of using the entire path, we focus only on the nodes closest to the source. The % of nodes in the path to use can be controlled, although the value should remain low ($< 50\%$), as the

motion should only display a fraction of its full intensity. An example of this can be seen at fig. 4.6. When the chosen emotional node does not have a representative node, there is no synthesis of insight animations.

4.5 Results & Validation

We have validated the methods described in this section by implementing two proof-of-concepts and presenting them in appropriate venues [FSOnO12, SOC16]. The first tested the use of mind maps as a generic interface, being implemented as part of the LIFEisGAME (LearnIng of Facial Emotions usIng Serious GAMEs) project [Cen11]. It was integrated into an early version of the tools developed in this framework. The latter tested the combination of motion graphs with mind maps to drive the facial behaviour of a character and was implemented in a pure research context.

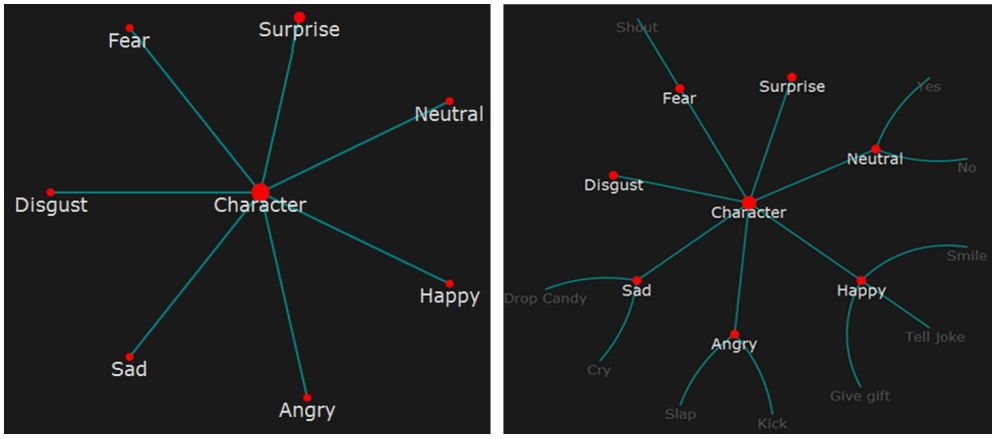

Figure 4.7: Mind Maps user interface provided to the users within the context of LIFEisGAME project.

LIFEisGAME was an international project aimed at socially and emotionally impaired individuals that have difficulties to recognise and respond to emotions displayed by others, e.g. people with autism. Within this project, mind maps enable non-experienced users, i.e. therapists, to specify and control the behaviour of a character that then interacts with the subject being helped. In this context, we have implemented a configuration tool that enables creating and exporting mind maps (fig. 4.7). The therapist assigns these to characters within a game engine. Such characters were part of the world that created the ”Live the Story” game mode, where the player’s task was to interact with the characters and lead them to a certain mood or action. A mock-up of this interaction is shown in fig. 4.8. Mind maps main objective was, thus, to provide an intuitive interface that had a reduced learning curve, and allow the therapists to quickly update the behaviour of a character. This, in turn, would

lead to a faster, more iterative and efficient therapy session. We have made initial tests with a mind map that had 7 emotional nodes: happiness, sadness, disgust, surprise, anger, fear and neutral (fig. 4.7). This implementation also limited stimuli nodes to be connected only to one emotion. The mind map interface was implemented using HTML and Javascript and the LIFEisGAME's "Live the Story" game mode was implemented using C++, QT and OpenGL 3.0 by the LIFEisGAME project team. This game mode, however, did not get a public release, nor was completed as the development shifted to other approaches. Results of using mind maps as behaviour controllers were, therefore, limited to internal tests within the research group.

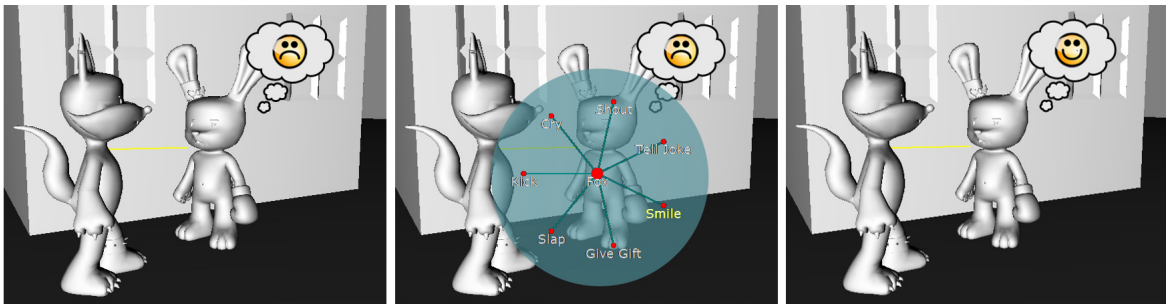

Figure 4.8: Mock-up of an interaction within LIFEisGAME's "Live the Story" game mode. All the behaviours were chosen using our mind map based approach.

For the behaviour facial animation proof-of-concept, we re-implemented the mind map based approach inside of *Matlab*©, i.e. the same environment of our procedural method. Mind maps again contained emotional nodes for the 7 basic emotions, however, we now add chain reactions and multiple connections between stimuli and emotional nodes. Due to limitations on the actual facial behaviours that can be generated by our procedural method, owed to the used DB, it is impossible to visually test how the method behaves with several reactions per emotional node. The lack of training data makes our procedural method unable to handle e.g. a happy node that has two distinct ways of expressing happy, which are shown for different happiness levels. We have thus tested and confirmed the behaviour of the mind map with several abstract reactions, e.g. $< react\_1, react\_2, ... >$. This allowed verifying the capacity of the mind map to choose "expected" reactions, in a structure more complex than the initial tests of the LIFEisGAME project. Within these tests, we have verified that the choice of reaction node takes on average ~0.05 seconds. Despite the referred constraints, we have also tested the full visual pipeline, although in a reduced scale. Two different mind maps were created and given the same stimulus. The main differences between the maps are the distinct stimuli connections and emotional weights, as well as the chain reactions. Single reactions are the same in both maps. An example of this validation is displayed in fig.

4.9, with stimuli on top and followed by the mind maps reaction output and the respectively generated peak pose. Motion graphs were trained with CK/CK+ DBs following the method described in the previous chapter. The graphs were, however, created using manually chosen thresholds, as the optimisation technique, described in Sec. 3.4.3, was yet to be born. All tests were done on a laptop with an i7-4720HQ and a NVidia GTX 970M. Finally, landmarks are applied to a 3D facial model following the same technique described in Sec. 3.6 and App. A.1.

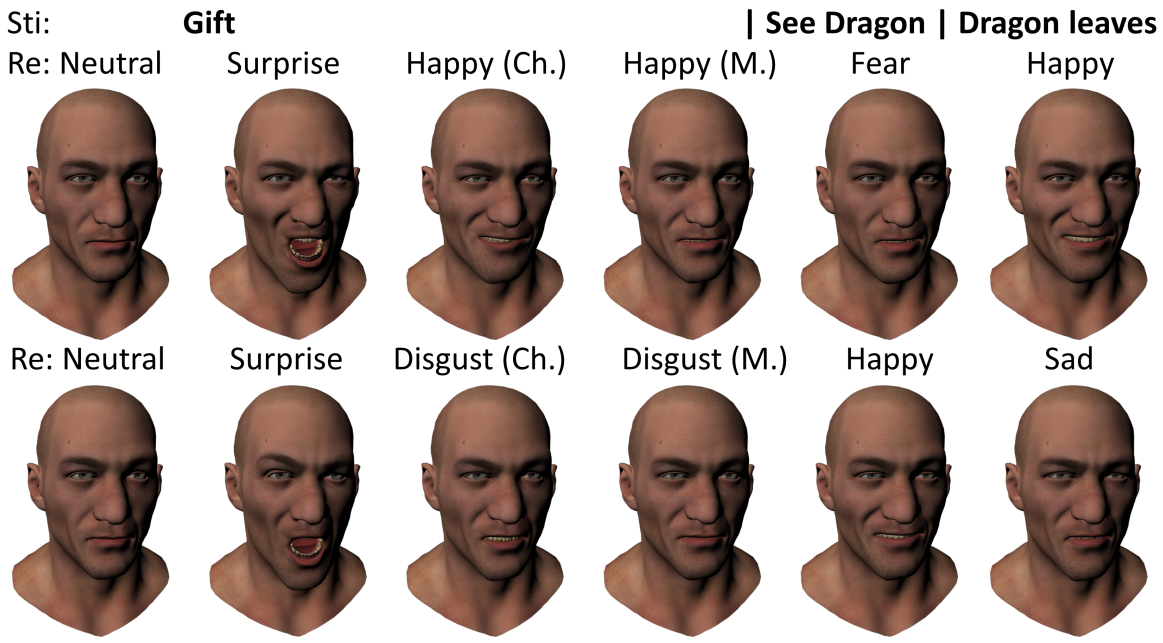

Figure 4.9: Example sequences that show the result of providing the same stimuli to two different mind maps, both bound to the same motion graph bellow. The motion graph had a compression ratio of ~82%. Above each sequence of poses, we additionally show the input chosen by the mind map. *Ch.* stands for the second pose in a chained reaction and *M.* stand for a mood decaying animation, i.e insight animation

4.5.1 Analysis

The two proofs-of-concept enabled us to verify the correct behaviour of both the mind maps, as a stand-alone method, and the full behavioural facial animation system. These tests also provided insights on the limitations of the methods. We now focus primarily on the mind map component of both concepts, since the procedural animation approach is already discussed in Sec. 3.6.1. Additionally, most usability tests were done by the authors and colleagues in the research group. The following discussion originates from the authors' observation of both the method's chosen reactions, system configuration and its general

performance.

Regarding the creation of behaviour, our approach enables a simpler, more direct and intuitive configuration, when comparing against scripting and behaviour trees. When setting up a behaviour, the author only needs to consider how the stimuli would make the character feel and how those feelings would lead to the actual reactions. Scripting and behaviour trees require a higher awareness of the context and factors that can influence the choice of an action. In the same sense, "reading" a mind map is easier than either options, because of its underlying structure constraints. This also makes our method more suitable for novice users that do not have knowledge on behaviour definition. On the other hand, mind maps are not suited to represent the complex cognitive and emotional processes of the brain (to which we forward to [SBR+14]), nor controlling the decisions to achieve a certain goal, e.g. speech-based interactions. This is on the verge of being impossible with mind maps, requiring far more planning than any other approach and much finer grained weight map, which counters the fundamental goal of mind maps. For more intricate interactions, traditional scripting, behaviour trees or even alternative scripting languages [APM07] are more suitable, as these offer detailed control on all the factors that lead to the choice of an action. Allowing the author to specify exactly when each behaviour should be played. Mind maps are, thus, more suitable to control characters whose decisions can vary freely and are not required to do something specific at a given moment and situation.

Within the chosen reactions, we have verified that mood updates resulting from stimuli and from mood decay lead to a varied choice of performing actions, even for the same stimulus given consecutively. The time when stimuli are provided likewise influences the choice, since the current mood can be decaying. Nevertheless, with the mood stabilised in the personality, the selected reaction nodes will always be the same given one stimulus (provide a stimulus; wait until mood decays to personality; provide the same stimulus). This is further noticeable if the mind map is very simple, e.g. only a reaction per emotional node. In this sense, we recommend at least 3 different behaviours choices. Mind maps also work primarily as a reactive framework. This means that if no stimulus is provided, the character will not choose any performing action, hence lacking behaviours. Animations triggered by the mood decay, i.e. insight animations, reduce this effect. However, the are not applicable for any type of body/face motion and, after the mood stabilises in the personality, no more behaviours will be triggered. Making this issue more noticeable. While testing multiple weights for the reaction nodes, we have also found an interesting bias in the reaction choice, of emotional nodes that have several performing actions. Reactions with low weight values (< 0.2) are seldom chosen. This makes sense given that before selecting a behaviour, its associated emotional node needs to represent the highest emotion value in the mood array.

The minimum mood array value, for which an emotional node can be selected, is given by $> 1/number_emotion_nodes$. And this considers that all other mood array values have a value slightly bellow the average, which is rare. Typically, an emotional node starts being selected when the respective mood array value is higher than 0.3. As afterwards, the reaction is chosen by weight proximity, low emotional weight reactions are rarely selected (given reactions, and respective emotional weights, are well distributed in the emotional node). We have only verified this with 7 emotional nodes, however, we expect the behaviour to be the same with other numbers of emotional nodes, with different weight values.

As for the combination of mind maps with procedural animation, these tests have further proved the usability of our motion graph approach. Even with a reduced number of labels, the generated animations were still different. As for the insight animations, they showed subtler versions of the full-fledged motions. Nevertheless, similar to choosing the number of nodes that vary a path (3.5.2.2), the nature of the graph influences how subtler are the insight animations. As a result, fine tuning is required to achieve the best level of expressiveness. In the same context, configuring other parameters of the behavioural facial animation system such as the rate with which insight animations are triggered, parameters for idle motions and duration of peak animations requires some experimentation. Even with this, we believe our approach is more intuitive and enables a faster configuration when compared to both industry and state of the art approaches for behaviour facial animation.

4.5.2 Limitations

Our mind map approach was designed to remove the need to define all pairs of stimuli-reactions, however, this came at the cost of the ability to precisely control the behaviour of a character. Another design choice was the use of mind maps and the current mood as the author's interface. While this enabled an intuitive definition method, it also prevented the creation of more complex social and emotional behaviours. Our method has some notion of time, which is embedded in the nature of the mood updated process, that always builds the new mood on top of the previous one, and via the mood decay procedure. The affection matrix also enables the character to evolve its perspective on other characters. These lead to more coherent, varied choices, while also adding complexity to the emotional behaviour. What we have not done is endow our method with the capacity to coordinate a series of social interactions that have a relation between them, nor to have short or long term goals. These would allow the character to cross the boundaries of being purely reactive and become independent. Another approach, supported by our method, to increase the complexity of behaviour is through the addition of emotional nodes. However, this also makes the

definition of mind maps more convoluted and time consuming and can lead to the stimulus weights to become more diluted. Therefore, making the mood behind the reaction choice less obvious. The mood update formula is currently, at its core, a weighted average of all the mind maps components, where the author can control the different weights. However, there are other paths that might consider a deeper relation between the current mood and personality, e.g. bias that makes the transition between happy to sad harder than happy to neutral. Done properly, these other paths would be capable of further reducing the author's input.

The limitations of combining behaviour with procedural facial animation encapsulate the limitations of both approaches, which have already been presented. Still, this integration requires additional parameters that are currently obtained via trial and error. To further reduce the authoring time, it would be interesting to initialize these parameters based on psychology literature or even learn them via machine learning techniques. Finally, while we have done the essential tests to validate both the use of stand-alone mind maps and paired with our procedural method, we have not done extensive quantitative nor qualitative validations. It would be highly beneficial to integrate both methods in e.g. a video-game context, where the interactions between player and character occur naturally. Here, it would be possible to measure the impact of (not) using our method. Achieving this would, however, require not only implementing the methods in a game engine and multiple behaviours, but also solving all the mapping of facial landmarks to 3D facial models issues. Similar to the issue of going from theory to practice with motion graphs, we were unable to supply all the artistic needs required to achieve this validation.

4.6 Discussion

In this chapter, we have proposed, tested and discussed a generic behaviour controller and integrated it with our procedural facial animation method. Thus, creating an end-to-end motion generation pipeline. The former relies on mind maps to provide an intuitive user interface, which has been widely explored in fields like advertising, education, finance, medicine due to their easiness to describe concepts and relationships [Dav11]. Mind maps connect stimuli and reactions via a common ground, i.e. character's mood, avoiding the traditional need to extensively discriminate all the conditions required to trigger some behaviour. This, additionally, translates into variety in the choice of reactions, even when the same stimulus is provided consecutively. We believe our mind map behaviour controller provides a viable approach to simulate non-scripted and non-verbal interactions, where the

character does not have to precisely behave in a pre-defined way. It fits our goal of endowing the character with enough emotional complexity that allows it to cope with a wide range of interactions and behave in a coherent way, but doing so in a simple way, without going to the complexity of techniques like Sagar et al. [SBR+14]. Although our base of comparison has been traditional techniques, such as scripting and behaviour trees, our technique does not aim to, nor it is capable of, replacing these for coordinated sequences of actions, dependent on the context. Instead of matching both approaches, they can actually complement the issues of each other, and further reduce the set-up time. While mind maps enrich the choice of behaviours with more emotional and cognitive complexity, scripting/behaviour trees handle the behaviour choices that absolutely must occur. They can work either on intermittently, with either assuming full control depending on the situation, or use mind maps to provide an emotional variable that is used by the other techniques.

In our quest to explore the synergies between behaviour and facial motion, we have managed to propose a combined approach that considerably reduces the set-up time on both sides of the coin, and still achieve unique, on-the-fly synthesis, even if the stimulus is repeated. The mind map acts as the brain of the character, while the motion graph handles all the pose and timing details on the animation. The bridge between approaches is established via the expression labels in the reaction nodes and insight animations. The latter enables the viewer to obtain a quick peek on the mood of the character. Insight animations occur at certain intervals when the mood is decaying, leading the character to show traces of a previously felt emotion. Existing techniques that approach the problem in similar ways have a considerably higher configuration stage and generally do not provide as much variation in the results. Our behavioural facial animation technique simplifies and reduces the manual labour required to forge a virtual character, while leading the way to achieve more autonomous and emotionally rich characters. These advances, in turn, lead to an increase of meaningful and believable interactions that occur virtual worlds and, on the long run, even shift the task of animating a character to one of directing it.

Chapter 5

Conclusion & Future Directions

This Ph.D. thesis has focused on the issue of automatically generating facial animation, as a way reducing the costs and labour to create believable characters. To this end, it presents a novel procedural method that learns how to generate varied animations from the analysis of a motion database, which is encoded in a compact structure. Short clips of animation are produced, on-the-fly and near real-time with minimal input. While the method already speeds-up the creation of facial animation, we further combine it with a behaviour controller. Reducing, even more, the efforts associated to the creation and control of virtual characters reactions. In this chapter, we summarise the main contributions of this dissertation's research and discuss how it can be improved and lead to new paths of innovation.

5.1 Conclusion

Facial animation is still a significant challenge in many productions, specially when money and time limitations are involved. This is particularly clear with many secondary characters present in video-games, virtual worlds, films... They fill the streets of the world, providing for small roles that constantly interact with the viewer/player. Thus, leading to worlds that are alive. The costs of endowing high-quality facial animation and behaviour to characters are impossible to bear for both hero and secondary characters. This leaves the latter limited to a small range of motions and choices that are constantly repeated, which is easily noticeable after few interactions. We present a procedural behavioural animation method aimed at the ever-increasing need of facial motion. It is composed of two novel components: an automatic method for motion synthesis and an intuitive behaviour controller that melds

with the animation system. Thus, creating a complete method capable of handling both the character's choices and its actual motions. Our system is capable of producing complex emotional reactions and unique animations at a fraction of the cost and time of traditional approaches.

Our facial animation method introduces a new type of motion graphs that have been optimised to compactly represent facial motion. Motion graphs approaches assume that certain segments of motion are the same, or at least very similar, across multiple movements. By identifying these segments, different motions are bridged, which in turn enables the creation of new motions, i.e. not present in the training data. This idea had previously only been followed by Zhang et al. [ZSCS04] for facial animation, however, we believe there was, and still is, much to be explored within non-verbal facial communication. This idea is thoroughly used in our method when creating the graph and when reconstructing the motion. Each movement synthesised is only based on the most relevant nodes, which were created with information from different facial behaviours. Expression labels, contained in the training DB, are used to find these nodes and provide an intuitive mechanism for the author to control the generation of new motion. It is exactly via the use of motion graphs and expression labels that we present a solution to the first challenge described in Sec. 1.1.1, i.e. *generation of animation from a reduced number of input parameters*. Our solution deals with all timing and pose properties of the generated motions, without needing any additional input. It also significantly reduces the configuration efforts, by allowing the user to control how much the original data should be compressed. Other approaches either have unintuitive input or a lengthy set-up. Our method is also capable of generating varied animations, even in the face of the same input. This is caused by our method to choose the relevant nodes from the labels and because we introduce different ways to explore the graph structure. We follow the idea that nodes close to each other contain similar poses and, thus, can be switched with each other in a motion. This is a form of coherent noise, which, in turn, introduces small variations in the motions. Such is our answer to the second challenge in Sec. 1.1.1, i.e. *non-repetitive synthesis of animation from same input*. Another way that we use the graph structure is by generating idling behaviour for any pose present, with minimal work for the user and without the need for additional data. To understand if our hypotheses hold, we have extensively tested our procedural method with different databases. This allowed us to analyse the quality of the animation synthesised, ensure the method is capable of accurately recovering the encoded motion and to generate unique animations. We have concluded that with sufficiently distinctive samples, even if in small number, our approach is capable of generating almost limitless variations of the motions present in the DB, all with minimum user effort.

Finally, we have explored the use of mind maps to define how a character reacts to different stimuli. Our approach relies on the character's internal state as the middle ground between stimuli and actions. The use of an internal state is not new in research, what is novel is the use of mind maps as the interface that connects stimuli to the internal state and then back to the reactions. Our assumption was that mind maps could be somewhat restricted and used to describe the relation between emotions and stimuli/reactions. As a stimulus occurred, the mind map would contain the information required to update the internal state, and then, based on the new state, the mind map would be consulted to trigger a reaction. Effectively providing an intuitive interface that does not require matching all the stimuli with the respective behaviours. Characters take into account their current mood and personality traits when choosing the proper action to display, reacting differently depending on the location, source of interaction and even history of events. The mind map controller is our solution for the third challenge in Sec. 1.1.1, i.e. *stimuli-reaction association*. We combine this interface with the animation synthesis method to create autonomous, emotionally aware characters that display idiosyncratic motions. The bridges between approaches are established by adding the expression labels, as reactions of the mind map, and by introducing insight animations. These enable the user to catch a glimpse of what the character is feeling, which can lead to more meaningful interactions. Behavioural animation methods tend to just overlay behaviour simulation on top of facial animation, however, our goal has been to explore a tighter integration between the methods. Our assumption was that the way an animation is displayed could be improved with information from the behaviour side. Insight animations show an initial approach that does just that. The synergies between both methods also allow shifting the task of animating a character to one of directing it. This makes our behaviour animation method particularly suited for previsualization and to enable non-experienced authors to adventure in the world of animating characters. In cases where traditional techniques still need to be used, our method can provide an additional layer of non-verbal behaviours, so commonly absent in secondary characters.

5.2 Future Directions

We have focused on providing the best quality results with minimum efforts, both on the use and on the configuration stages of our final system. However, despite validating the animation results and the ”correct” choices of the behaviour controller, we were unable to obtain hands-on feedback from the final user. Obtaining feedback on what is right or wrong, from a user perspective, would allow us to push the presented approaches forward and further favour the user.

Proposing and implementing our end-to-end behavioural facial animation solution has required understanding many different fields that range from psychology to computer animation, along the likes of machine learning and usability. All these fields contributed to the current work, but they can contribute more. We now describe some of the most interesting directions for future research that would both benefit from our work and extend it:

- **Procedural Facial Animation**: Our research has focused primarily on this field. Despite scarce, there are already some approaches capable of synthesising facial animation. However, the usability and quality of the results still pales behind more established fields, such as performance capture, especially when higher quality animation is needed. Another common issue of existing procedural techniques is the tendency to rely on ad-hoc approaches, specific to the application case, that are both difficult to extend and configure. We explored the knowledge of different fields and incorporated that in our approach, creating a method that fundamentally relies on motion data analysis, and keeps the user's input to a minimum. This shifts the focus of the method from the user's previous knowledge to the actual training motion, making our method easier to set-up and extend. In this context, we have also proven that, even with limited databases, it is possible to research facial motion synthesis and shown its potential for growth and improvement. Our method still does not reach the same quality of traditional approaches, requiring further fine-tuning. However, we believe it clearly advances the field. It is, nevertheless, just the beginning of what will unfold in the next years with more and more use of machine learning techniques;

- **Complex Facial Behaviours**: Most research nowadays focuses on a minimal number of facial expressions (our method including), which is to be expected given the limited number of good quality facial motion databases. However, facial behaviours include far more complex motions that are not limited to displays of basic emotions. These are very important, without doubt, but so are others like nodding when speaking with other people or expressing pain through a series of facial motions. Many of which are very subtle. Our method can be somewhat limited in its capacity to represent subtle motions, as these can be incorporated into larger poses, i.e. macro-expressions, which are then averaged and the fine details lost. It would be interesting to explore a finer grained motion graph capable of coping with both macro and subtle expressions.
  Another relevant aspect of facial behaviour is the relationship between expressions and head/gaze motions. These have been researched and their synthesis already produces realistic results, on their own. Even so, approaches capable of generating facial behaviour that incorporates all facial modalities are still very scarce. This issue remains an open question. Our solution focuses on facial expressions, but it would be

interesting to explore how motion graphs could represent and be used to synthesise these additional motions, without increasing their complexity significantly; Synthesising both more complex behaviour and facial modalities in unison, believably and without an extremely complex set-up still requires further research. It is, nevertheless, worth the investment, as it enables the creation of truly real virtual characters.

- **Interaction between Concurrent Behaviours - Speech**: Non-verbal facial behaviours occur many times in the presence of speech. Our technique is not aimed at this task, which traditionally falls within performance capture. Conceptually, such would be possible, however, the graph would need to be very fine grained as to accurately cope with co-articulation effects. Additionally, conflicts between the timing generated by our approach and the speech audio would need to be solved, as the synchronisation between the two modalities is crucial. A more interesting venue is, however, combining speech, generated by an existing approach, with our procedural technique, thus, producing expressive visual speech. This would increase the quality of speech present in secondary characters. Synchronisation still need be addressed, but the precision, to which this must work, is lower than with phoneme level synchrony. In the same sense, the mouth region would be controlled by the two approaches, so issues would need to be solved as conflicting motions might be synthesised;

- **Inclusion of the mind processes into facial animation**: This is not a new idea, as many procedural approaches borrow concepts from psychology and neuroscience to simulate behaviour, which then triggers animations. However, the combination of both commonly assumes the form of two distinct layers, where the upper layer, i.e. behaviour, just chooses the animation to be played. We have explored a tighter integration with insight animations that show subtler motions, as a way of peeking into the character's internal state. This, in turn, led to the creation of a new approach to synthesise motion in the graph. These insight animations would not exist if the two methods were completely independent. Sagar et al. [SBR+14] is another notable exception that also integrates mind and facial behaviours. Deeper synergies between mind and the animation system, be it procedural or performance-based, could lead to methods that are easier to control and inherently consider what the character is feeling when generating animations, hence leading to more emphatic displays of behaviours. We believe that mind processes will play a crucial role in the future of facial animation, in which case, we have only scratched the surface.

5.3 Final thoughts

Facial animation is complex and challenging to create, independently of the chosen approach. It has been evolving at an amazing pace, especially in the performance-capture domain. Each year, better and better results are achieved in uncontrolled environments, without personalised training and requiring less fine-tuning. It shows the importance of high-quality facial animation to films, video-games and other productions. What it does not tell is that it is impossible to use this technique, or manual animation for the case, to generate the motions for the hundreds of characters that compose ever-increasing worlds. In the end, both methods have a heavy reliance on either animators or actors, which makes them slow and expensive to fulfil that need of animations. Procedural facial animation methods have the potential to achieve this, but their research is very much in the beginning, despite being explored for quite some time. And, while some of the challenges are common to other fields, there are still so many opportunities just waiting to be explored. Especially, with the advent of machine learning right upon us. We believe that this area will be the future, it is just a matter of time until research changes the focus to the topic. Our method shows a possible path, and while doing so, it attempts to unlock the field and increase the pace with which research and investment occur. With our work, we lead by example and hope that it will encourage others to pursue research on procedural facial animation and come up with new ideas.

Appendix A

3D Facial Animation

All 3D facial animations of *Victor* and *Oldman* models (*Faceware*©) have been generated using the technique described either in App. A.1 or as part of the facial markerless capture approach of App. A.2. The former was implemented by Ozan Cetinaslan and is based on the work of [LA10]. We have borrowed this technique to drive a model from the output of our motion graph based technique. The performance-based method was implemented by Shridhar Ravikumar, within the context of his Ph.D., and allowed comparing how the results of our method fare against a more traditional take on facial animation. Despite these techniques not being the subject of this thesis, we believe it is important to include them as appendices, making this document more self-contained.

A.1 Direct Manipulation of Blendshapes

Our method's output is directly related with the DB it was trained with. As both DBs used, i.e. CK/CK+ [KTC00, LCK$^+$10] and the small DB obtained from *Faceware*©'s Analyser, only contain 2D points, then our motion graph method only produces 2D landmark sequences. To drive a model from these points, the blendshape direct manipulation method [LA10] was implemented. From here, we associate 14 landmarks (locators) to manipulators in *Autodesk Maya 2011*© via a parent constraint. As the size of the 3D facial model and the landmarks are not the same, we manually rescale the landmarks to keep them proportional to the model, using the neutral pose as a reference. The manipulators are associated to vertices in the surface of the model. As the manipulators move, so do the vertices, which forces an update of the blendshape weights as to match the differences in the vertices. This process starts with the blendshape model, which is the sum of the linear vectors of the defined target

shapes [LAR+14] (equation A.1):

$$F = \sum_i b_i w_i \tag{A.1}$$

where F is the final face in vector form. It includes all vector positions of the face model in an arbitrary order of *xyz*, b_i is the vector of each blendshape target, and w_i is the corresponding weight. According to the original mathematical framework of [LA10], there is a direct relationship between the blendshape weights (w) and the moved manipulators. After the manipulators move on the surface, the proposed system will use the weights to keep the resultant face model as close as possible to the moved manipulator. This relationship can be explained as $m = Bw$, where m is the moved manipulator, B is the blendshape matrix, which includes all blendshape target vectors. Therefore, by following [LA10], a regularized least square form is employed to create interactive weight updates (equation A.2),

$$\min_w \|Bw - m\|^2 + \alpha\|w\|^2 \tag{A.2}$$

where α is the regularization term, which should be small number such as 0.0001. As a result of equation A.2, our weight update equation is:

$$w = (B^T B + \alpha I)^{-1} B^T m \tag{A.3}$$

Equation A.3 updates the facial poses during the movements of each manipulator. We additionally enhance the framework by enabling interactive computation of equation A.3 and demonstration of the corresponding face movements simultaneously.

A.2 Video Driven Facial Animation

The performance face capture results were generated using a monocular blendshape based tracking approach. We first obtain the 3D mesh representing the neutral face of the actor, scanned using a 3D scanner – Artec Eva [3D17]. An existing template neutral mesh and respective topology is then deformed to match the scanned mesh using the non-rigid ICP algorithm of [ARV07]. The deformation transfer approach of [SP04] is then used to automatically generate person specific blendshapes with an existing template of blendshapes as a basis. This provides the set of blendshapes with the desired topology that we then use to solve for animation parameters.

For tracking the actor's performance, we capture the monocular video of the actor. We then track 68 distinctive landmark points on the face, on a frame by frame basis, using the dlib

library [Kin09]. From these, we choose landmarks equivalent to the ones used in the direct manipulation approach. These landmarks are then manually matched with points in the 3D mesh. Finally, we solve for the optimal blendshape weights for each frame by minimising the following energy term:

$$E = \sum_{l=1}^{N} \|\Pi_Q(M(B_0 + \sum_{i=1}^{N_B} \alpha_i B_i)^{(v_l)}) - q^{(l)}\|^2$$

where:

- N is the number of landmarks
- Π_Q is the camera projection matrix
- M is the rigid transform from object space to camera coordinates
- B_0 is the neutral expression blendshape
- α_i is the weight associated with blendshape i
- B_i corresponds to the i-th blendshape
- v_l represents the vertex corresponding to landmark l
- $q^{(l)}$ represents the l-th 2D landmark point in the image

The per-frame landmark detection can induce noise in the result, so we smooth the obtained blendshape weights over the sequence using a moving average filter to obtain our final animation weights.

Appendix B

Gaussian-based Coherent Noise - An Unsuccessful Draft

Amongst the venues followed in this thesis that did not pan out (spoiler alert), the most intriguing, and relevant, was the use of multivariate Gaussian distributions to encode the differences between all the poses merged in a node. And, in turn, sample these distributions coherently to create new poses. Hence, working together with the path variation technique (Sec. 3.5.2.1) to increase the variety of animations produced. We have explored bivariate (Sec. B.1) and multivariate (Sec. B.2) distributions, per landmark and per facial region respectively. These were calculated from the information assembled when creating both the sample graph and the final graph (Sec. 3.4, 3.4.1). Variations of the base pose contained in each node are then calculated by finding equiprobability points in all the different distributions. This enables small changes of the pose coherent with the training data, instead of having each node always use the same base, averaged pose, when reconstructing motion from a path (Sec. 3.5.1). With per landmark distributions, we have proved that, under the right circumstances, Gaussian-based noise is capable of introducing small, coherent variations in the animation (Sec. B.1.2). However, the method has its limitations which was what we have approached with region-based distributions (Sec. B.2). The later has, unfortunately, not worked in the intended way.

B.1 Individualised Gaussian Noise

The core component of our first method is a bivariate Gaussian distribution created per facial landmark, with the two dimensions corresponding to x and y coordinates. This distribution

is calculated after all the poses/nodes are merged and the final graph created. Due to lack of a definitive number of samples required to create an accurate distribution, and the reduced number of samples from the DB (CK/CK+ [KTC00, LCK$^+$10]), we have decided to not calculate a Gaussian distribution unless more than 5 poses are merged. This can, however, be changed by the author. Independently of this threshold, the $\mu = \frac{1}{m}\sum_{i=1}^{m} X^i$ is calculated for each landmark. This is, effectively, the average already present in the graph nodes (Sec. 3.4). For nodes with more than 5 samples, the $\Sigma = \frac{1}{m}\sum_{i=1}^{m}(X^i - \mu)(X^i - \mu)'$ is found. This step would, thus, add the covariance of each landmark to the data structures of the node, which affects the compression results described in Sec. 3.6. As for the noise variation process, it starts with the creation of an average covariance matrix, from all the distributions of the landmarks in each region. This distribution is then randomly sampled once, and the probability/direction, in regard to the centre of average Gaussian, of that sample are obtained. The probability and direction form the basis that enables finding an equivalent sample in the distributions of all the landmarks that form each facial region, i.e. equivalent point. The direction vector is, additionally, inverted either vertically or horizontally according to rules as to create more sensible variations. The result in a new pose that is slightly different from the average pose contained in the node, which is used as the basis for the motion recovery steps described in Sec. 3.5.1. This technique is detailed in the following section.

B.1.1 Generating Variation

Similar to previously described techniques, our method aims to abstract as much details as possible, thus easing its use. The author only needs to provide the intensity of the noise, r, which acts as a scalar of each landmark's covariance matrix, applied via $r\Sigma$. r is limited to vary between $[0,..,1]$, where 0 represents no noise introduced. Our method also relies on some rules that alter the direction of variation according to the position of the landmark in the face. We have defined these rules, which should remain the same even for other landmark configurations. The author can, however, also control the association. The noise is, itself, chosen using the mean $\overline{\Sigma}$ of all distributions in each region, centred at the origin. We then sample a point, X, following the $\overline{\Sigma}$ distribution. X provides the basis of the variation that will be extracted for all the landmarks of each region. The direction $\vec{d}$, between X and $\overline{\mu}$, is then stored, together with the point's probability, $p(X)$. The latter is obtained from the cumulative distribution function. In this context, and for different Gaussian distributions, we define a variation, X_{eq}, to be equivalent to X, if $p(X_{eq}) = p(X)$ and its normalized direction $\hat{\vec{d}}_{eq} = \hat{\vec{d}}$. Visually, this translates into the equivalent variations shown at fig. B.1. Computing

X_{eq} is the result of a simple intersection of the ellipse eq. B.2, formed at $p(X)$ for the respective Gaussian, and the line eq. B.3 associated to $\vec{d}$. Eq. B.2 requires a and b, which are respectively associated to the major and minor axis. Thus, we first calculate the eigenvectors v_i and eigenvalues λ_i from the Gaussian and then scale them with eq. B.1 [And15]. This equation scales the eigenvalues to have the dimensions expected of an ellipse on $p(X)$. χ represents the Chi-square distribution for 2 degrees of freedom. We now have a and b to plug into B.2, however this equation assumes an axis-aligned ellipse (in a Cartesian coordinate system), hence, if we intersected the current ellipse with $\vec{d}$, the results would be in a Gaussian different from the landmark's. As a result, we find the angle $\alpha = \arctan\frac{v_x}{v_y}$ that rotates the biggest eigenvector to match the Cartesian x-axis and apply that rotation to $\vec{d}$, thus obtaining $\vec{d}_{rotated}$. From this new vector, we extract the $m_{rotated}$ for eq. B.3, which is centred at the origin of the system. Finally, the intersection of eq. B.2 and B.3 is calculated, which returns two intersection points. We then only need to invert the α rotation to convert these points into their correct space. The chosen point, i.e. the variation vector X_{eq}, is the one whose signals respect are the same as $\vec{d}$.

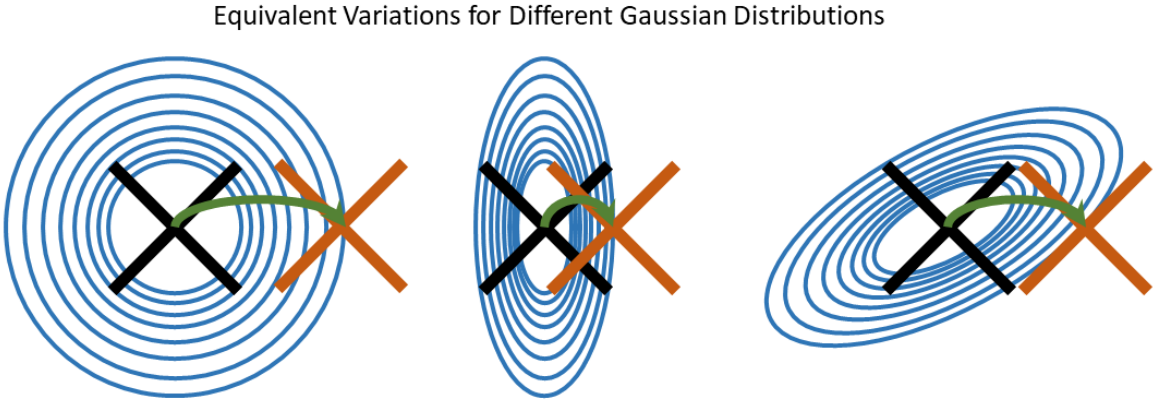

Figure B.1: Equivalent variations of the same landmark that follow different Gaussian distributions. Black X shows the landmarks' initial positions, while the orange X shows their final positions, all following the same probability ellipse in the distribution. Blue circles/ellipses show equiprobability points (the ellipses do not display an accurate variation of the probabilities in a Gaussian distribution).

$$a, b = \sqrt{\chi^2_{2(1-p(X))}}\sqrt{\lambda_i} \tag{B.1}$$

$$1 = \left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 \tag{B.2}$$

$$y = m_{rotated}x \tag{B.3}$$

We now have noise that is different between landmarks, due to separate distributions, but at the same time is coherent, as it falls within the same probability and direction. However,

applying this directly to all the landmarks would produce non-sensible variations. Consider, for example, the mouth region, if the direction of variation was upwards, then all the landmarks would move up. This would produce a shift in the mouth pose as a whole, which is something unexpected and can lead to non-feasible expressions. We solve this with a set of rules that modify each landmark's variation direction according to their position in the face. These were chosen empirically as to match the behaviours of each region, from analysis of the motions present in CK/CK+ DB [KTC00, LCK$^+$10]. We now describe the inner-workings of rules with an example of how they impact mouth region's noise. Other regions follow the same principles, which can be seen at fig. B.2. The mouth rules are based on a division of the space by a vertical, y, and horizontal, x, axis, with each landmark assigned to a quadrant. If a landmark is assigned to the left side of y-axis, X_x is inverted, becoming $-X_x$. If it is assigned to the right side, X_x will remain the same. As for the x-axis, if a landmark is above this axis, X_y continues with the same value, while if bellow, X_y is inverted, becoming $-X_y$. Special rules are added to points considered to overlap with the vertical axis (cyan points in fig. B.2). Here, X_x is dropped, i.e. the x value of the non-varied landmarks is used.

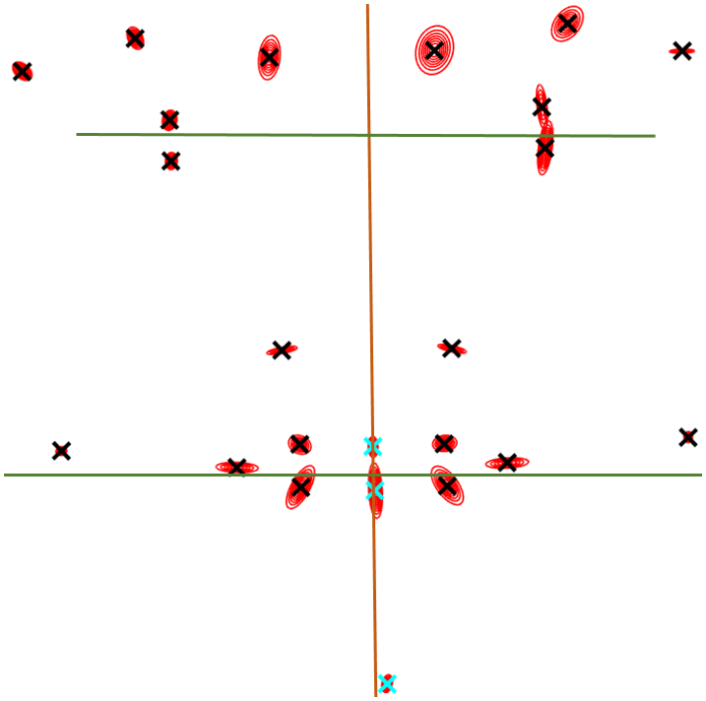

Figure B.2: Axis-rules used for each landmark and equiprobability rings/contours of the Gaussian distributions for each landmark, in red (extracted from a node in the motion graph). Rules of regions revolve around the horizontal and vertical axis, green and orange lines respectively. Mouth, eyes and jaw/cheeks region landmarks are subject to vertical and horizontal changes according to the quadrants; Nose and eyebrows are only subject to vertical rules. Landmarks in cyan will drop the variation's x value, using the one from the base landmark.

B.1.2 Results & Issues

We built this method on top of motion graphs described in Sec. 3.6, created with the CK/CK+ DB [KTC00, LCK+10]. Everything was implemented in *Matlab*© and tested on a laptop with an i7-4720HQ and a NVidia GTX 970M. We have studied the behaviour of our method by generating multiple sequences and analysing the differences between motions generated with and without Gaussian-based noise. No other form of noise was introduced from any technique described previously. Fig. B.3 shows examples of equivalent poses extracted from our tests. The landmarks' motion was transferred to a 3D model (from WrapX ©) containing a bone-based rig (adapted within Porto Interactive Center group) in *Autodesk Maya 2012*© and then exported to *Unreal Engine 4*©, from where the 3D model images were extracted. Fig. B.2 additionally shows the Gaussian distributions (scaled for better visualisation) of a node in the motion graph.

The different videos generated allowed us to conclude that our method works as intended, introducing coherent variations across the facial regions. However, they also showed that the impact of noise is barely noticeable in the animation as a whole, as shown in fig. B.3. Amongst the reasons for such, we refer to the method's nature that prioritises small variations over larger ones. Most variations will be close to the centre of the Gaussian, i.e. the base landmark position. The number of poses merged in a node, and their similarity can also contribute to this result, as the lack of significant variations means that, even when farther away from the Gaussian's centre, the final variation of the landmark might be very small. Also in this context, nodes, with more extreme expressions, have a lower number of poses merged on them, as these poses are not common to many facial behaviours. This can even lead to extreme nodes not having a Gaussian distribution, as the number of poses merged did not reach the minimum threshold. As a result, the presented method can be particularly useful for graphs with a high compression value, where merging has occurred extensively, even in nodes with more extreme poses. This advantage comes at the cost of lower compression graphs, as saving the covariance matrix, for each landmark, increases the amount of data that each node needs to store. One solution to the noise being too small would be to scale either the landmarks' covariance matrices or the size of their variations. The main problem with doing so is, however, the change of the information represented by each node. Variations generated would no longer comply with the ones present in the original data. The use of rules to mirror the effects of the variation is also not optimal, as this makes the directions of the noise symmetrical, which might be different from reality. While rules lead to more coherent and sensible results, they do not accurately represent the relations between the landmarks, in a specific facial region. Finally, our goal with this method has been coherency of the generated noise in the landmarks. Spacial coherency

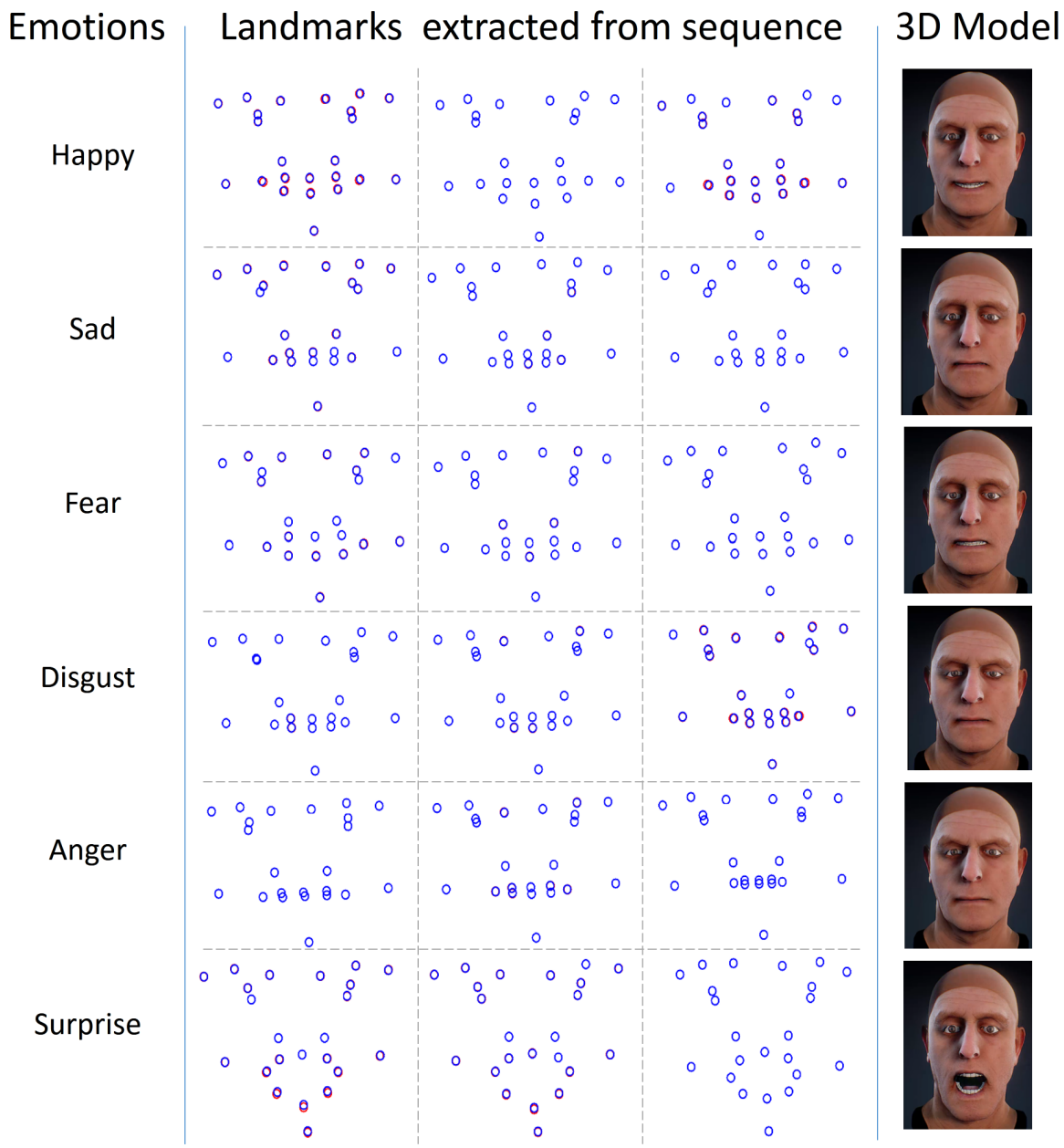

Figure B.3: Example sequences of landmarks and respective animation labels with (red) and without Gaussian noise (blue). On the right, we also show the result of applying one of the landmarks' poses to a 3D face model.

has indeed been achieved, however, the same cannot be said for temporal coherency. This is important as introducing noise in consecutive nodes can produce opposing variations, which leads to a jiggling effect in the resulting animation curve. This, in turn, can lead the noise to be completely absorbed by the final smoothing (Sec. 3.5.1), making all the additional computations irrelevant.

B.2 Temporally Coherent Region-based Gaussian Noise

Despite current results being underwhelming, they validate the core idea behind the method, i.e. the use of Gaussian distributions to model the variations of all poses merged in a node and to generate coherent variations of the landmarks. As a result, we have focused on improving the method by addressing temporal coherency and the rule-based system. To this end, we created a multivariate distribution for all the landmarks of each facial region. This removes the need for rules, as the distributions encode the relationships between landmarks. Therefore, choosing a random variation inherently changes the pose in a spatially coherent way. Our hypothesis to solve temporal coherency was to use the same method to find equivalent variations, but with distributions from different nodes of the chosen graph path. The method is fundamentally the same as the one described before, just for higher dimensions. It follows the updated eqs. B.4, B.5, B.6. n in eq. B.4 represents the number of dimensions. Eq.B.6 represents the parametric equation of line, with one entry for each dimension and $\vec{d}_{x_i}$ being the component of i-th dimension of the vector formed between the centre of the Gaussian and the chosen variation point X.

$$\begin{cases} a_i = \sqrt{\chi^2_{n(1-p(X))}}\sqrt{\lambda_i} \\ \dots \end{cases} \tag{B.4}$$

$$1 = \sum_{i=1}^{n} \frac{x_i^2}{a_i^2} \tag{B.5}$$

$$\begin{cases} x_i = \vec{d}_{x_i} t \\ \dots \end{cases} \tag{B.6}$$

The problems appeared when the method did not always converge to a solution. Results were promising for certain regions like the eyes or jaw/cheeks, however, distributions with more than 5 landmarks, i.e. 10 variables, had a high chance of at least one node, in a graph path, failing to calculate an equivalent pose. Based on our tests, we believe the problem arises from the multivariate distribution being too skew in one or more dimension(s). In which case, the respective eigenvalue(s) are ~0, which leads to the impossibility of defining an ellipsoid.

B.3 Discussion & Future Work

In this appendix, we presented and discussed another take on introducing coherent noise into procedural facial animation. Our goal was to explore multivariate Gaussian distributions to

encode and synthesise variations of the poses present in the graph nodes, thus producing new poses coherent with the training data. To this end, we proposed a technique that finds equiprobability rings in multiple distributions, at landmark and facial region levels. While this showed promising results, at least with landmark level distributions, we consider the impact of noise not worth the trade-off of increased data, stored at each node, which reduces the compression levels of our motion graph method. Other issues have been identified, such as time coherency or ad-hoc rules, which we started addressing with whole region landmark distributions, however, the method revealed unexpected scalability issues. We have, thus, chosen to follow alternative paths and, in due time, return to this issue. However, that did not turn to be the case :(given Ph.D. time constraints), and we have thus decided to present this approach in the appendix, evoking further research on the subject. As for future work, the simplest improvement is the addition of time coherency into the method described in Sec. B.1, i.e. using the same probability and direction for all the landmarks in each region and each node along the chosen path. Scaling the covariance matrix or direction to produce larger, more visible variations is an option also. The variations would still be somewhat coherent with the initial data, but further subjective tests would be needed to validate this approach. Despite unsuccessful, we still believe in the use multivariate Gaussian distributions to represent the variations on each region. In this context, reducing the dimensions of the data and finding coherent variations in the latent space would be an interesting path. This requires further research and, thus, we do not consider the path closed.

Appendix C

Personality Simulation Through Emotional Biases

We have explored alternative ways of specifying the perception-decision-action loop [DDLT02], aside from the use of mind maps as behaviour controllers. Similar to these, though, is the emphasis on avoiding the slow and tedious process of defining all pairs of stimulus-reaction through an emotional variation model. To this end, we present a novel approach that introduces the use of height-maps specified in a space defined by Russel's circumplex model of emotion [Rus91] (fig. C.1). The height-maps represent the emotional bias of a character's personality, which impact the way each stimulus is analysed and, in turn, lead to different choices in the reactions. The circumplex model provides a unifying framework, where the personality is defined and the stimulus/reactions are mapped to/from. This results in an intuitive approach to define behaviour, which provides non-linear emotional simulation and behaviour choice. Each character has a current mood, i.e. emotion at a specific time step, which is modified by the occurrence/perception of external stimuli. Stimuli have an initial emotional weight that is altered by the personality biases, i.e. areas in the space that will pull or push the stimulus in their direction. Consider a character that has a tendency towards positive emotional states, and a negative stimulus is provided, then its impact on the mood will be reduced. How much the emotional weights of the stimulus are changed depends on the height value associated with the current mood of the character. Similar to mind maps, this method enables different reactions to be chosen from the same input stimulus. We have, however, chosen to keep it within the appendix section, as the method was not combined with our procedural facial animation approach.

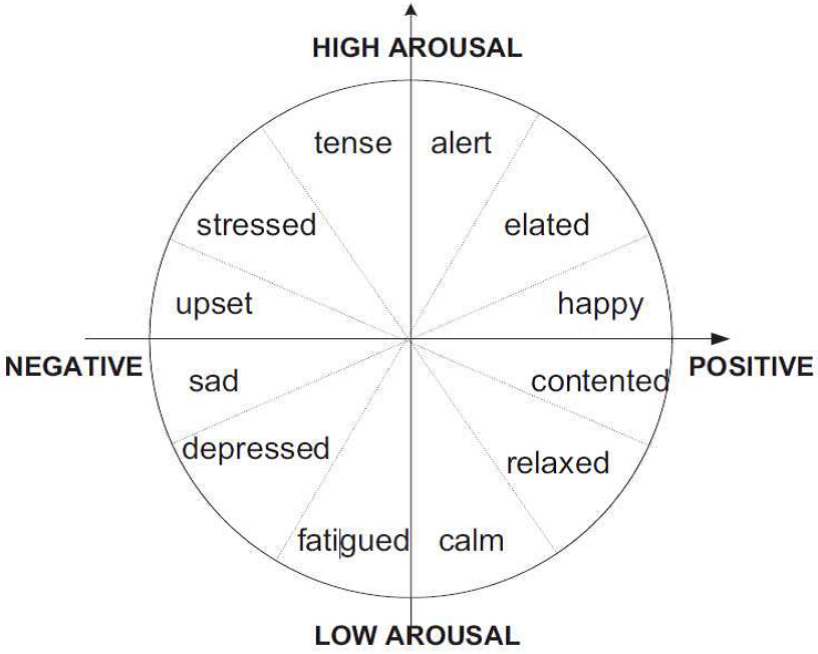

Figure C.1: Russel's bi-dimensional emotional model [Rus91] with the association of emotions to specific areas within the model.

C.1 Method

We propose a method that applies the bi-dimensional representation of emotions presented by Russel [Rus91] (fig. C.1) to the simulation of emotional processes. This space is defined by an arousal and a valence axis, with the emotional states assigned to specific areas in the circumplex model. The origin is associated to a state of neutral valence and a medium arousal and is traditionally regarded as the neutral emotion. Points closer to the limits of the space represent higher intensity emotions/affects. The circular nature of the space enables comparing any two emotions, represented as points, using a simple Euclidean distance metric. Within this space, we define: the current character's mood, as a point that is updated each time a stimulus occurs; a stimulus, as a point that represents the emotional charge of a given event; a personality, which is a height-map that represents the emotional biases of the character towards specific locations in the space; and, finally, actions of the character, whose selection is based on specific locations within this space. Our method spreads these components into three components/layers: input abstraction layer, responsible for the mapping stimuli to their emotional points/vectors; stimuli evaluation layer that keeps track of the character's current mood and alters it according to stimuli and personality; and the output abstraction layer, in charge of the action decision process. This architecture can be seen in the Fig. C.2.

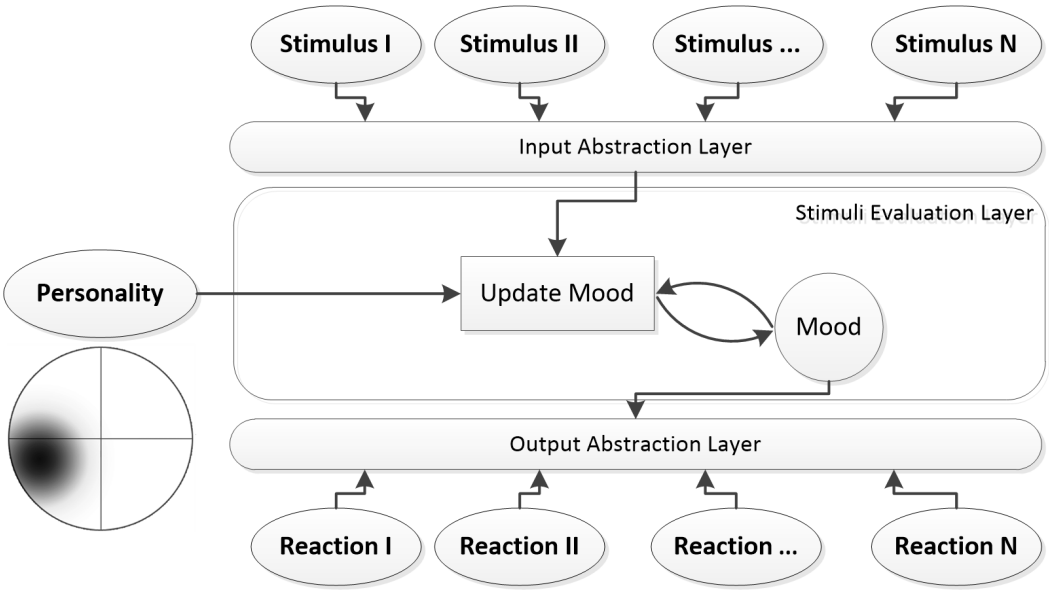

Figure C.2: Method's architecture that is composed of 3 layers: input abstraction layer that receives the stimuli as input and converts them to the circumplex space; stimuli evaluation layer that updates the mood based on the stimuli and personality; and the output abstraction layer that uses the current mood to choose an appropriate action.

The Input Abstraction Layer defines the mapping between external stimuli and their emotional weight. It is an abstract entity that can represent anything from social interactions to abstract sensory stimuli, such as an image or sound, as long as it is associated to a point in the emotional space. This layer allows multiple stimuli to occur simultaneously, which can also occur in the real-world. These can be handled sequentially, by feeding them separately into stimuli evaluation layer. In which case the order matters, as the resulting mood will not be the same. Another approach is to do additional processing in the layer, and combine multiple stimuli into one, via e.g. linear combination of the weights, which would lead to effectively only one stimulus being passed to the next layer. Alternative approaches, such as defining rules on the combination, are also a possibility.

The character's emotional response is controlled by the Stimuli Evaluation Layer. It contains the mood and personality model, both of which authored à priori. The emotional coordinate received from the input abstraction layer is converted to a vector (from the centre of the space) that is applied to the current mood (Eq. C.1). The stimulus vector is, however, first influenced by the character's personality.To define the personality, we explore the concept of attraction forces, represented in the circumplex model as height-maps (fig. C.3). We refer to these forces as the emotional biases. They behave as gravitational valleys/wells that pull the perception of the stimuli towards a specific emotion(s) coordinate(s). Height-maps are traditionally associated with computer graphics, where they represent the height of a terrain

or texture as the brightness of each pixel. In our approach, the brightness values represent the forces that each well exerts on the perceived stimuli. Darker values mean a higher is the pull towards the centre of the valley. As a result, if a stimulus tries to move the character's current mood away from a gravity well, the stimulus's strength will be reduced, which is done using countering gravitational vector. On the other hand, if the stimulus is in the same direction as the centre of the well, the stimulus strength will be increased. A personality valley is defined by its centre coordinates, height, radius and height variation between the centre and the limit of the radius. There are multiple ways to define the latter, e.g. linear interpolation on the distance or following a Gaussian distribution. Within this work, we follow the latter, although the author can opt for alternative approaches.

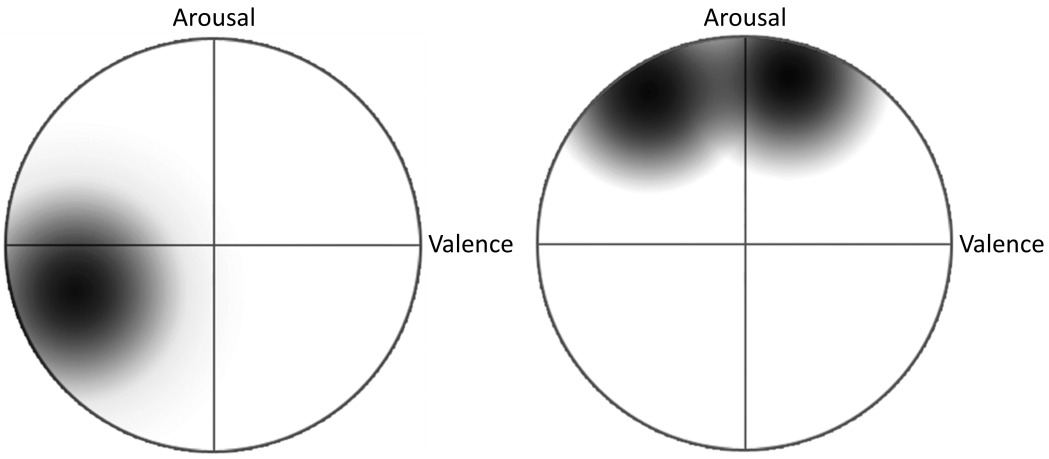

Figure C.3: Example of two height-map based personalities. The left and right images respectively show depressive and excitable/tense personalities. In both cases, the character would easily become sad or alert/tense, and then require extra stimuli to change to other affective states.

The current mood, m_c, is updated using eq. C.1, where: v_d is the vector from the current mood and the centre of the closest valley; h_c represents the influence of height on the current mood; s_w is the stimulus emotional vector. In the case of the current mood being between multiple personality valleys, a new v_d is calculated from the averaged sum of all the vectors from the current mood and the respective centres. h_c is calculated by normalizing the height to an interval between [0..1], where 1 corresponds to the maximum height. In fig. C.3, the height's influence of pure black corresponds to 1, while for white it corresponds to 0. In our method, the closer the current mood is from a valley centre, the smaller is v_d, however, the larger will h_c be. This, in turn, reduces the influence of the stimulus.

$$m_c = m_c + v_d * h_c + s_w * (1 - h_c) \tag{C.1}$$

The Stimuli Evaluation Layer outputs an updated mood each time a stimulus occurs, which

is then passed to the Output Abstraction Layer. This last layer is responsible for choosing the character's behaviour from the analysis of the current emotional state. Such can be achieved in multiple ways, traditionally relying on some kind of dictionary that translates a coordinate in the space to the desired reaction. Some possibilities include: associating actions to positions in the circumplex space and choosing one based on the distance to the current mood; or defining areas for each action and selecting the one where the current mood falls. Another option for the output abstraction layer is to use it to change the domain of the result, e.g. extract the emotion of the character from the current mood, via the space division of Russel [Rus91] (fig. C.1). Thus, effectively fitting our method as an emotional simulation component within other, larger frameworks/systems.

C.2 Results & Validation

The proposed architecture was validated as a proof-of-concept, by analysing the emotional variations that result of applying between 500 and 1000 random stimuli to 2 personality models (fig. C.3). The results are presented in the form of scatter graphics at fig. C.4. Our approach was implemented in C++ and the results generated in a laptop with a Core 2 Duo T9600 and an ATI HD4650.

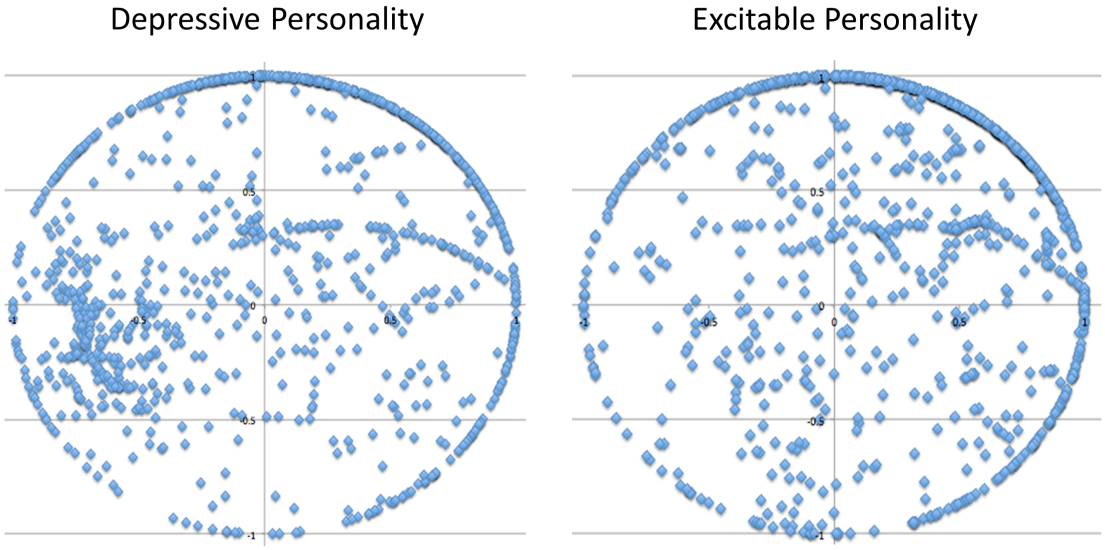

Figure C.4: Scatter graphics after 500-1000 randomly chosen stimuli for both depressive (left) and excitable (right) personalities. The left image shows a clear distribution of moods around the personality valley. In the right image, the mood points are distributed primarily on the upper boundaries of the circumplex space.

Both graphics show that the current mood converges to the personality valleys previously defined, although more distinctly on the depressive personality. It is also possible to observe

that the mood is not restricted to the valleys and will, with enough stimuli, wander away from these convergence areas. This confirms that our method behaves as intended, with the personality influencing the emotional variation process. These initial tests have, however, also revealed a limitation of our method, i.e. a high concentration of the moods in the edge of the space. The results from stimuli occurring when the current mood is near the circular limit, which can push the mood to exceed the boundaries. In this context, clipping occurs, and thus the mood will remain on the edge. This is the main reason why there are many mood points on the upper limit of the excitable personality simulation, making the convergence less visible. Such issue would be reduced with the introduction of emotional decay over time. Here, after a stimulus occurs and the mood is updated, the impact of such stimulus will start to degrade and lead the current emotion to decay to either a neutral emotional state or the personality's main valley(s). As this is not simulated and with the mood already at the edge, only an opposing stimulus vector will move the current mood away from the limit. Another approach, that comes at the cost of more user input, is the introduction of an additional layer of dynamism via rules on the current mood position. These rules would increase or reduce the stimulus weights to simulate emotional exertion, e.g. when a character is happy, receiving a happy stimulus will not have the same impact as when the character was in a neutral emotional state.

C.3 Discussion & Future Work

With this work, we have explored an alternative approach to control the emotional variation process of a character and embedded it within a new layered framework. The input and output layers provide a transparent and powerful abstraction of stimuli and their elicited reactions, enabling the author to not only use our proposed approaches but also extend them and explore alternatives. At the core of our solution, we have introduced a simple and intuitive personality definition method that explores the use of height-maps to represent the inherent bias within human behaviour.

Our method and respective proof of concept can be further explored in many ways, including combining it with facial animation (following section). The first venue identified is the introduction of the notion of time, through the emotional decay. This is crucial because people do not remain in the same affective state after the occurrence of stimuli, and will fall back to a certain mood. Hence, emotional decay would gift the character with a more realistic emotional simulation core. Height-maps are also currently limited to a linear or Gaussian height variation/shape. This is not ideal, as it constraints the freedom with which

the personality biases can be defined. In this context, it would advantageous to allow a free definition of the height-map, via e.g. a mesh, in which case the personality influence vector would be calculated from the slope. This would also lead to the definition of both valleys and peaks, which would pull or push the mood towards/from a certain direction. Combining our height-maps approach with mind maps as behaviour controllers is also possible. In this case, the latter would replace the input/output layers and the personality biases would replace the single vector used in the mind-maps approach. The validity of our method could also be further verified by having multiple people interacting with a character and ”guessing” its personality representation. This could be done after several interactions that would have different outcomes, depending on the personality bias, with the subject finally having to draw the personality biases.

Similar to mind maps, this approach does not aim to consider all possible information in a real-world scenario, or attempts to provide the most ”accurate” emotional variation process. Our method seeks to introduce more complex choices that lead to less static and black/white characters, while also being simple to define and control. We believe that, despite only being an initial proof of concept, this was achieved with a method that can easily be integrated into existing techniques, thus providing more emotionally aware and believable characters, that react in a more realistic way.

C.3.1 Triggering facial animation - Possible solutions

While purely a rhetorical exercise, we can explore alternatives to combine our personality based method with facial behaviours. For these we identify two approaches: by direct association of motion graph labels to specific points in space, which would be selected based on the distance to the current mood; and by dividing the space into areas that are again associated to labels. These are different in the sense that the latter would allow exploring the mood distance to the centre (neutral) as a scalar that controls the intensity with which animations are displayed. The latter translates into more expressive animations when the mood is closer to the limits of the space, and more subtle expressions near the origin.

Appendix D

Procedural Body Animation - Focused Literature Review

Research on procedural facial animation is scarce, however, that is not the case with body animation. As Orvalho et al. [OBP+12] refer, body animation methods have long influenced techniques used to animate faces, and thus, serve as perfect sources of inspiration for procedural facial methods. Generally, research on procedural body animation falls into either *example-based* methods [PP10], also known as motion editing [VWVBE+10], where motion capture data is used as the basis to create new animation, or *simulation-based*, where physical, mathematical or logical constraints provide the basis for the motion. Mathematical and logical constraints are also commonly referred as procedural-based. Hybrid approaches have also been researched. Nonetheless, further categorization has not been completely agreed upon [VWVBE+10, PP10, GP12, BE12, GSC+15]. Despite some overlapping, which is to be expected, each survey approaches procedural body animation differently. Welbergen et al. [VWVBE+10] provide the most comprehensive overview of the field, focusing on the trade-offs between naturalness and control over the generated motion. Pejsa and Pandzic [PP10] focus in example-based animation and, more specifically, synthesis from analysis, instead of methods to change existing animations. They approach the classification from a perspective of the core structures and methods used to analyse, represent and synthesise motion. Geijtenbeek and Pronost [GP12] focus only in physically-based simulation with particular emphasis on the different approaches for motion control design. Aside from the more common comparison of naturalness and control, they also address aspects like robustness to external perturbations and difficulty of implementation. Basten and Egges [BE12] present a smaller survey on Motion Transplantation, i.e. transplant an auxiliary motion onto a base motion, thus having separate motions, or techniques, driving

different regions of the body. Guo et al. [GSC+15] analyse how example-based, physically-based simulation and hybrid techniques vary in terms of naturalness, controllability and how they cope with: being applied to different characters, varying in terms of geometry and topology; environment changes, such as in the terrain; perturbations in the motion and user input. Since they do not completely overlap, it is possible to combine the categorizations of all approaches, which we do in fig. D.1. The base of this figure is the classification presented in [VWVBE+10], where we update some names for increased coherency between surveys, and connect to the specific details of [PP10, GP12, BE12, GSC+15]. It should be noted that not all the categories present in this figure are described here. This happens mainly due to lower ”expected” importance for the work of this thesis. While most surveys consider naturalness as one of the main goals in motion synthesis, they tend to lack a more formal definition of the concept, with the exception of [GSC+15]. They define naturalness as the subjective comparison between the synthesised motion and the equivalent organic action. We further define *equivalent* as being the intrinsic expectations of the person viewing the motion.

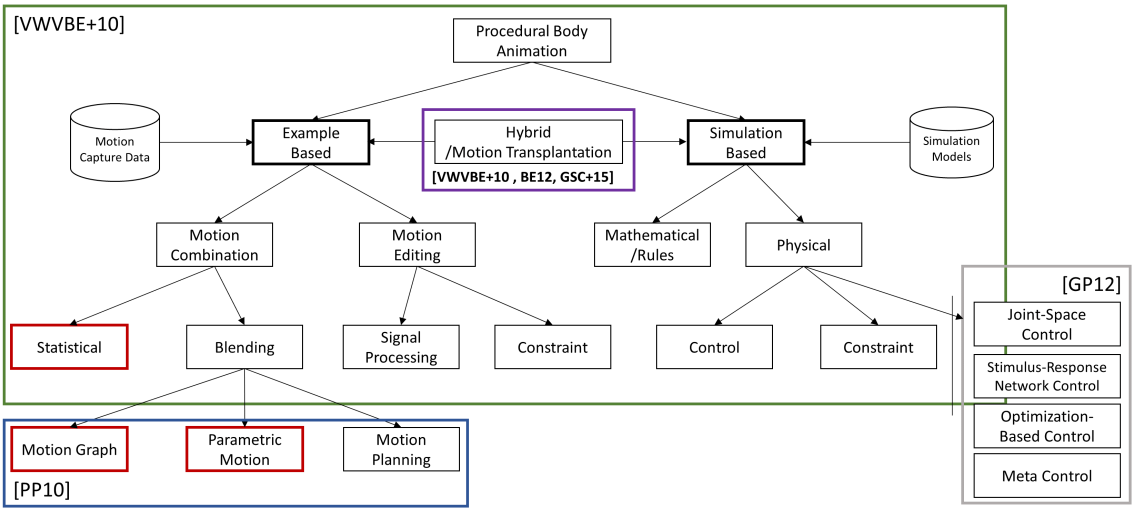

Figure D.1: Diagram that categorizes the approaches of the procedural body animation field based on the surveys [VWVBE+10, PP10, GP12, BE12, GSC+15]. The core diagrams are presented in the [VWVBE+10] from where the tree grows. [PP10] deal with example-based animation; [GP12] with physically-based animations, providing an alternative categorization to [VWVBE+10]; [BE12] focus only in motion transplantation. [GSC+15] focus on example-based, physically-based and hybrid techniques. The squares in red represent the focus of Sec. D.3.

We now present a brief overview of the different categories of [VWVBE+10], with the updated naming convention (fig. D.1). For more details on the categories described in the following text and the associated image, please refer to the appropriate surveys [VWVBE+10, PP10, GP12, BE12, GSC+15]:

- **Example-based** (previously motion editing) aim to generate new motion from recorded motion data, be it captured using motion capture, generated by artist or alternatives such as retargeting. Two subgroups appear in the form of approaches that start with an initial motion and alter it, i.e. motion editing (previously modification), and approaches that analyse different motions and create new ones by combining them, i.e. motion combination:
  - *Signal Processing* (Motion editing), where the motion is considered a signal and signal processing techniques, such as displacement mapping, are applied to change properties of the motion;
  - *Constraint-based* (Motion editing), where additional constraints, such as a new pose, are specified and the motion altered to satisfy these constraints;
  - *Blending* (Motion combination), which, in its most basic form, relies on interpolation to blend two or more animations, hence making anew. This, however, cannot be done directly as the motions need to properly fit each other, or artifacts will appear. Thus, alignment between animations, using e.g. time warping, is required before blending. Also, blending does not necessarily mean through the whole animation. It can occur only in a small part.
  - *Statistical Models* (Motion combination), which use techniques from machine learning to create models that represent the statistical variation of body motion and thus are able to synthesise it.

- **Simulation-based** uses parametrised physical or mathematical/logical (previously procedural) models to create the animation. In the former, there is a physical simulator that controls the joints while accounting for different physical properties of the character and the environment. Any animation results from the simulation of these properties, which at the lowest control level is done through the application of forces and torques. Mathematical/logical methods do not have this layer of simulation, thus controlling the character model directly at the lowest level:
  - *Control-based* (Physical) use a dynamic controller that manages the torques applied to the joints of a bone-based rig. The user specifies the desired output, and the system calculates forces that lead the model to the achieve that state. An example of such is the Proportional Derivative Control approach, where the output torques are calculated from the difference of position and velocity between the desired and current states;
  - *Constraint-based* (Physical) works similarly, but adds constraints such as a pose at a certain time or ground contact. This problem is usually handled from an

optimisation perspective, where the goal is to minimise e.g. the expended energy while fitting the desired constraints;

- *Mathematical/Rules-Based* (previously procedural) methods use parametrised mathematical formulae or a series of rules to construct the motion space, such as describing the movement path of end effectors (e.g. hands or feet) or mathematically defining the interpolation between poses;

- **Hybrid** methods attempt to combine the best of the worlds. Several ways exist to achieve this, for example: on a per frame basis, such as in [vWZR09], where the motion generated from rule-based or example-based approaches is combined with physical methods, each responsible for controlling different parts of the body. As new torques are calculated from the analysis of rules/example-based motion, the controller of the physically-based part of the body updates the respective motion; motion transplantation [BE12], such as in [ZH99], where motion capture data controlling the upper part of the body is combined with a physically-based controller that drives the lower part and keeps the character's balance; or as an alternative combine both sequentially having e.g. the example-based generated motion as the base animation and when a perturbation occurs, go into simulation mode. The main challenge here is going back to example-based motion after procedural or physical simulation is activated. An example is seen in [ZMCF05], where motion capture is combined with short periods of physically-based animation. When there is a disturbance, they estimate how the character will look like in the end. They then use this information to search all the motion capture data and determine which should be the end poses of the physical simulation. They effectively account for this in simulation, forcing the model's pose to finish close to the performance capture data.

D.1 Comparing procedural body animation categories

Each group of approaches has different trade-offs between *naturalness*, *control* and *adaptability* to environment [VWVBE+10, PP10, GP12, GSC+15]. Different methods should not be grouped under the same umbrella when comparing the category they are inserted in. Nevertheless, they will typically have the pros and cons of their category. ***Example***-based techniques provide the most natural motions, at the cost of requiring large databases. And it is not only the quantity that matters, the built-in variations and quality of the motion greatly influences the quality of the results. As a result, each time a new parameter is required, more data needs to be recorded, which can lead to the DB growing exponentially with the number of parameters. Additionally, these techniques seldom provide physical

interactions with the environment, which means they are particularly useful whenever such is not present. ***Mathematical/rule***-based methods allow precise control over all aspects of the motion via a large number of parameters, however, incorporating the subtleties and variations of the motion into the mathematical formulae is hard. Thus, they tend to look too clean, lacking on the naturalness side. Physically realistic animation is also hard to create. Finally, ***physically***-based simulation achieves physical realism and allows interactions with the environment, enabling the character to cope with external perturbations. This is extremely hard to attain realistically with other approaches. However, since the character motion is not controlled directly, it is more difficult to provide precise control (it is also particularly challenging to get characters up after they fall [GP12]). Naturalness and style are also harder to achieve when compared to example-based approaches, at least when there are no interactions involved. Finally, physically-based techniques tend to be more complex to implement [GP12]. From a practical standpoint, mathematical and physically-based methods are more easily controlled than example-based approaches due to their parametric nature, but the lack of naturalness compromises their choice. This results from the models used being typically an approximation of the actual rules that govern realistic motion [PP10]. This translates into example-based approaches being the most common in computer games and films [VWVBE$^+$10, PP10]. Physical simulation and mathematical methods are restricted to specific cases. Physical simulation is primarily used in games where interaction with the environment is primordial, and there are too many variables that invalidate recording of motion, such as in sports games. Mathematical/rules-based animation tends to be used where highly correlated and timed animation is required. There are also hybrid approaches that allow combining the naturalness of example-based techniques with the adaptation capabilities of physically-based animation, thus reducing the size of the DB required. Even though combining different techniques is more complex than implementing them separately, the results suggest that the merits of hybrid approaches exceed those of the individual ones.

Due to their naturalness, we have decided to focus primarily on example-based procedural body animation in Sec. D.3. These methods are more likely to be applicable to facial motion, with due changes, and providing a simpler adaptation when compared to mathematical or physically-based methods. Such occurs because, as referred in the beginning of Sec. 2.1, the face has almost limitless behaviours, which currently are not fully understood in terms of mathematical and physical constraints. Additionally, the face is rarely subject to direct environmental disturbances, such as impact by an external force. Despite this, it is still possible to embed physically-based animation into facial movements through physically-based rigs (Sec. 2.1.1), which can control in-betweens of manually/performance-based/procedurally

specified poses. Mathematical approaches can be used in similar fashion by controlling the motion curve that determines the in-betweens (Sec. 2.1.2). Motion of the eyes presents itself as a notable exception. The eyes are even considered easier to animate via biomechanical models [LM08], which fit in mathematical/rules based animation. We additionally exclude motion editing techniques, part of the facial retargeting field (Sec. 2.1.3), which are already well studied both individually and as part of many performance-driven techniques. Thus, we have concluded that example-based techniques, dealing with synthesis from analysis of recorded motion, provide the biggest opportunity for facial animation, and thus, their body counterpart is explored here. For more details in the different example, physically and mathematical/rule-based techniques, we forward the reader to the excellent surveys of [VWVBE+10, PP10, GP12, BE12, GSC+15].

D.2 Frameworks

Frameworks that deal with complete procedural body synthesis are scarce, as far as we are aware. On the research side, and open source, there is the Dynamic Animation and Control Environment (DANCE) [Ari12] that provides several physically-based controls, while also allowing the definition of new ones. In terms of commercial applications, Natural Motion's Euphoria™[Mot16a] and Morpheme™[Mot16b] focus specifically in body animation, while other tools such as Massive™[Mas16], Golaem™[Gol16] or Houdini™[Sid] deal with animation within the crowd simulation context. Natural Motion software combines Morpheme with Euphoria to create fully automatic and reactive characters. Morpheme is used to author the animation via physically-based methods and example-based blending [Mac14], while also being able to incorporate previously recorded animation [VWVBE+10]. Euphoria then allows specifying the behaviours of the character and when the different animations are triggered. Massive [Mas16], Golaem [Gol16] and Houdini [Sid] work similarly via a combination of physically-based animation, example-based blending and motion capture data. They additionally support the definition of group behaviours, allowing the user to control how the crowd behaves instead of authoring each character individually. Other engines, such as Havok™[Hav16] or Unreal Engine 4™[Epi16], also allow physically based animation but these tend to be limited to ragdolls, not providing any additional control.

D.3 Example-based Body Animation - Motion Combination

Within example-based animation, we focus on the categories of motion blending and statistical-based approaches [VWVBE+10, PP10]. *Motion blending* refers to techniques that rely

directly on the combination of the samples via operations like stitching/concatenation or interpolation [PP10]. Both can occur in a smaller scale, on parts of the original sequence, or as a whole, where the sequences are fully combined or concatenated. In the limit, these operations can occur at a frame level. *Statistical-based* approaches use the different samples to learn models capable of generalising the original data. This can be done by mapping the data to another space, more suitable for synthesis and where the motions are characterised, or it can rely on approaches like neural networks or reinforcement learning, to encode and learn the underlying model of motion. There can also be separate models for different behaviours or have a model that attempts to represent the behaviours altogether.

In this section, we follow the division proposed by Pejsa and Pandzic [PP10] for motion blending techniques, i.e. motion graphs (Sec. D.3.1) and parametric motion (Sec. D.3.2). Additionally, we focus on statistical-based techniques (Sec. D.3.3). [PP10] present another group of approaches referred to as motion planning. *Motion planning* methodologies incorporate the desired goal and external factors into the method's optimisation. As a result, the choice and combination of animations take into consideration the entire relevant state [PP10], instead of being purely local-based. As more information is used, the optimisation changes from locally optimal to globally optimal. This takes its toll in the performance, making most approaches pre-compute as much information as possible and then using it online, at least for methods that focus on real-time. Motion planning approaches extend motion blending and motion graph concepts to account for more environment interactions and long-term goals, which means that sometimes it is hard to split them. In this sense, we focus primarily in methods where the knowledge of the environment is localised, e.g. knowing only the destination and the contact points. It is important to note that this does not prevent the use of global optimisation methods that consider only localised information. The main advantage of motion planning approaches is that they are near guaranteed to accomplish the desired goals, in the best way possible, which does not occur in locally optimal approaches [PP10]. As referred in [PP10], motion planning techniques appear to be the next step in procedural body animation, however, their application to facial animation is fuzzy, to say the least. Mostly because facial behaviours tend to be localised and have short durations. As a result, we choose to not to focus on this group of techniques.

D.3.1 Motion Graphs

Motion graphs based approaches formally appeared in 2002 with the seminal works of [KGP02, AF02] and more loosely in [LCR+02]. Afterwards, several other approaches have relied on motion graphs to synthesise animation, e.g. [AFO05, LL13]. Kovar et al.

[KGP02] define motion graphs as a directed graph capable of encoding motion in a way that synthesising movements is done by traversing the structure. The graph can be more or less structured at the cost of more user interaction, e.g. the user can define the content of the main graph nodes. This allows easier control when traversing the graph, and reduces the computation cost. Unstructured graphs like [KGP02, AF02] greatly reduce the set up time, but they tend to require more complex approaches to synthesise motion, usually via some kind optimisation over the input parameters. In this section, we focus primarily on unstructured graphs, leaving structured approaches to the parametric motion graphs section (Sec. D.3.2). The literature on motion graphs up until 2008 is loosely organised in [Gle08].

The fundamental goal of motion graphs is to reduce the work associated with the creation and control of body motion of characters in videogames. This is traditionally done using move trees [MBC01], which also relies on a graph structure, however, it is manually created. Each node contains a short animation, with the edges carefully chosen so the animations blend smoothly. The animations, themselves, are crafted under strict specifications, either manually or via motion capture. Creating a move tree for a character, that can have dozens or even hundreds of animations [PP10], ends being a thorough and slow process, which becomes even worse when dealing with the many characters required by a game. Additionally, even minor changes in the initially planned character's motion can require reconfiguring of considerable parts of the graph. Motion graphs exploit the massive amounts of recorded animation to tackle this issue, with automatic creation of the motion structure. Thus, considerably reducing the time associated to embed animations into virtual characters.

Creating and using motion graphs can be roughly divided into *graph creation* and *motion synthesis*. In the former, several motion sequences are analysed and compared to find connection points between them, also known as transitions. These places result in edges, and are found via some distance metric that provides a quantitative value of the similarity value between individual, or groups of, frames in each sequence. Since most metrics rely on the actual joint world position, it is common to use some operation to remove effects of rigid transformations between two comparing poses. This is extremely important because, if the poses are not aligned, there will have a low similarity value, even if the poses are similar. A user-defined threshold traditionally dictates when to create the edges. Motion synthesis then occurs by traversing the graph according to a specific function that depends on the goal, e.g. moving to a certain place or grabbing an object. As the poses, or sequences, are added to animation they need to be blended to prevent discontinuity or jitter in the animation. This results from the final animation having motion pieces of different samples, which have been captured with different orientations and translations. Additionally, it is strictly impossible to capture all possible motions, hence the different motions might not

always perfectly align with each other. Solving this issue is crucial to create smooth and visual appealing animation, and can be accounted for when creating the graph, by removing all effects of discontinuity, or when traversing the graph, based on the current animation flow.

The edge creation threshold is also extremely important. It is with this value that a user controls the trade-off between good transitions/smoothness and connectivity/flexibility of the graph. A perfect transition is one where the user cannot identify where it occurs, being comparable to what a person would do. The easiest examples can be seen in the training data, e.g. walking forward and then turning left, which effectively provides the baseline for new motions. With a low threshold, the number of edges created is scarce, which means that motion synthesis will follow the original data most of the time. Edges will be established when the poses are extremely similar, not only in terms of joint values but also velocities and accelerations. This naturally leads to seamless transitions [KGP02], where the motions are smooth and without any visual discontinuity. However, with a scarce number of edges, the graph will be mostly limited to original data with small variations. Transitioning between different animations can also be an issue due to scarce connectivity, where several intermediate nodes are required to achieve the desired animation. This can lead to strange animations and a character that has delayed reaction to the user input. In this sense, good connectivity refers to how quickly and directly it is possible to transition from one pose, in a behaviour, to a pose in another behaviour [ZS09]. Good connectivity is also important because it allows more combinations between the samples. A high-connectivity will solve the previously referred issues, however, if it is achieved at the cost of bad transitions, the produced animation will easily look disconnected. Such results from different motions that have no good transitions suddenly being connected. And people, as experts in motion, easily notice this issue. Having good transitions and high-connectivity is always a goal when using motion graphs, and there is even specific research dealing only with this subject [ZS09, ZNKS09]. Reitsma and Pollard [RP07] study another important issue in this field, i.e. how to evaluate the quality of the motion graph without actually seeing a considerable amount of animations. They propose an extensive methodology that uses several metrics to evaluate the graph according to the desired task. These metrics range from navigation to performing local tasks, in specific locations. Crucial to these metrics is also the environment where the motion graphs will be used.

Regarding the *structure* of the motion graph, it is always a direct graph as the direction of the motion matters, however, cycles might [GSKJ03, HG07] or might not be allowed [KGP02]. As for the content of the graph, common approaches include: nodes containing entire sequences, with the edges established between specific frames [AF02]; edges containing

the motions and the nodes being only transition points [KGP02]; having a node per frame [LCR+02]. A node can also store several sequences [CTGH12] (more details Sec. D.3.2). Different methods might change slightly the structure, or add an additional layer with extra information [LCR+02], but typically motions graphs follow either one or a combination of these forms.

There are many approaches to **create motion graphs**. We now present the seminal works and some of the more recent approaches. Unless specified otherwise, all the approaches rely on a manual threshold to decide when the edges are created:

- Arikan and Forsyth [AF02] present an approach that compares all pairs of frames using a distance metric that accounts for position, velocity and acceleration of the joints. The torso is used as the reference to align the poses. Since multiple edges can exist between two nodes, originating from different parts of the respective sequences, they optimise the structure by clustering the edges in a hierarchical way, thus reducing the search space to synthesise the animation;
- Kovar et al. [KGP02] propose a windowed distance metric centred around a frame. As neighbour frames are also considered in the metric, the effects of velocity are automatically considered. The comparison is based on the Euclidean distance between point clouds. Alignment is done by minimizing the 2D rigid transformation using square error between the clouds. They additionally prune the graph to reduce discontinuities and failure cases, such as dead-end nodes, which would be impossible to leave once reached;
- Lee et al. [LCR+02] rely in a first order Markov process to find the probabilities of transitioning between two frames, using a Euclidean distance and velocity based metric. They prune transitions according to several rules, e.g. ground contact. When calculating the similarity, and depending on the sample's motion, they either remove the effects of translation and rotation, e.g. locomotion, or they leave them to influence the metric's value, e.g. interaction with a step stool. On top of this graph, they create a statistical layer by clustering the frames using Gaussian Mixture Models. For each frame, a tree of clusters is created based on the frame's connections of this frame in the base graph, which is then connected with other nodes' trees;
- Lee and Lee [LL04] follow the structure of [LCR+02], but they group consecutive frames with single out connection into an action edges. The non-merged nodes are referred as states. Their method revolves around the use of targets, i.e. discrete locations in the map that skeleton should go and do something like punching. So, on top of the graph structure, they pre-compute a lookup table that takes the current

state and the desired target, and outputs the edge that should be followed. This table is created using reinforcement learning via dynamic programming;

- Arikan et al. [AFO05] create a motion graph using techniques from [AF02, KGP02], however, the nodes are additionally labelled according to the motion capture data being the response to an external force or not;
- Matsunaga and Zordan [MZ07] focus only on finding how much the different body parts should influence the metric calculation. Their approach is to account for the character dynamics, by estimating the mass for each body region and using constraints associated to ground contact and friction;
- Zhao and Safonova [ZS09] improve the graph creation process by introducing well-connected motion graphs. To enforce good transitions and connectivity, they first compute a set of interpolated poses from the original samples, and use them to create the new graph. The graph structure and distance metric are the same as [KGP02]. As creating a graph with all nodes would lead to a huge graph, they prune it, by keeping only the interpolated nodes and edges that improve the paths' quality between the original nodes. Additional reduction can be achieved with sub-optimal paths, trading quality for fewer nodes;
- In [ZNKS09] they focus on creating a minimum size motion graph capable of connecting a set of user chosen sequences. They speed up the process of [KGP02] by clustering the frames, and using the clusters in the distance metric. They introduce an iterative sub-graph algorithm that revolves around connecting nodes, from the user selected sequences, via nodes of other motions, hence reducing their distance. Additional constraints, like maximum number of nodes between two extreme nodes, can be forced. In the end, nodes not relevant to connect user sequence nodes are pruned. This method, additionally, allows determining the samples relevant to represent the chosen motions, thus minimizing size of the graph;
- Ng and Fernández-Baena [NCLC10, FBM11] focused on creating motion graphs for different body parts. The main issue with doing this is the synchronisation between the parts that is lost. Ng et al. [NCLC10] use the approach of [HKG06] to split the body motion into upper and lower parts motions. They create synchronisation edges between the corresponding upper/lower parts in the original sequences, and identify additional edges, by aligning the different sequences via dynamic time warping (at least when alignment is possible). Afterwards, a motion graph is created, per part, with normal edges, and pruned according to specific rules. Fernández-Baena and Miralles [FBM11] divide the body into torso, arms and legs, and create a separate motion graph for each part, following the distance metric of [LCR$^+$02];

- Ren et al. [RZS10] instead of optimizing for the motion, optimise for the graph creation. They convert all sequences into independent graphs and then pick frames whose transition, to other sequences, will be analysed and, if possible, created. If no connection is already present (this is an iterative process so it might already exist via another node), an initial guess is synthesised by forcing C^1 continuity. This guess is then optimised for a set of physically based constraints. The user is then prompted to confirm if the transition is good, if not it is removed. They reduce the number of prompts by estimating the probability of a certain trajectory being natural. This is done by estimating the trajectories of the centre of mass and end effectors via linear interpolation and dynamic programming;
- Huang et al. [HTCH15] present a technique that extends the work of [KGP02] to 4D performance capture, named surface motion graphs. The distance metric now accounts for shape and texture changes to obtain more accurate values. And, like in Kovar's work, they also use the neighbour's poses in the distance metric. Each node of the graph contains a sequence and each edge denotes one or more possible transition points.

Pejsa and Pandzic [PP10] divide motion graph methods based on the information used in the path search algorithm, i.e. local or global search methods. Local methods use a limited amount of information based on the current and/or desired state, being preferred for real-time approaches. However, this also means the final path is usually sub-optimal and may even fail to achieve the desired outcome. Such is addressed with global search methods. Instead of using this classification, we follow the same structure of the motion graph creation section, presenting the papers in a time-line fashion. It should be noted that not all previously presented papers have a counterpart in the following list as they either use other paper implementations or just focus on the graph creation:

- Arikan and Forsyth [AF02] allow the definition of hard and soft constraints. Hard constraints need to be fulfilled in the synthesis, e.g. having a pose occur at a specific time and place. Soft constraints do not need to be met exactly, e.g. having a certain number of frames. They use a randomised search algorithm that starts at the coarsest level and chooses n paths that match the constraints. The path that best satisfies the constraints is then chosen. Finally, they smooth the animation in a windowed fashion;

- Kovar et al. [KGP02] use a branch and bound approach with a function that evaluates the error of appending an edge to the current path. They implement this function to force the character to follow a path on the ground. Blending occurs between the

end frames of an edge and the beginning frames of the following. Annotations, in particular frames, can act as constraints to prevent e.g. foot sliding, and restrictions can be added to avoid certain types of motions from being chosen in the search algorithm;

- Lee et al. [LCR+02] present several ways of traversing the structure: 1) compute all the cluster paths, extracted from analysing the cluster tree of the current node, and obtain a representative motion based on the hop probabilities. They then present several of these motions to the user that will choose what to do next; 2) compute several cluster paths and see which one has the centre of mass closer to a desired path; 3) without clusters, find the path in the graph that best fits in a ground path. Transition blending is achieved by forcing the poses and velocities between the frames to match using displacement mapping [BW95];

- Arikan et al. [AFO05] focus on generating motion that results from a push. When such is detected, they choose the path using only push captured data nodes. They train an oracle by generating animations and asking a user to evaluate their quality. The results are combined using scattered data interpolation. They then synthesize multiple animations, all starting in the current graph node, and use the oracle to evaluate the most visually appealing;

- Fernández-Baena and Miralles [FBM11] use the legs' graph to drive the other body parts' graphs, enforcing the same start and end nodes in all the graphs, and using a common search window around these two nodes. Since the graph paths have different lengths, they linearly interpolate between them to force the same base size;

- Lee et al. [LL13] approach path generation using A* and present multiple heuristics ranging from a distance heuristic to a directional one, and a combination. Additionally, they merge the transitions, and respective nodes, with the same similarity value, which effectively condenses information from several hops into one node, thus reducing the search space. When they reach a merged state that satisfies the goal, they split the nodes and choose the one that best matches the goal.

- Huang et al. [HTCH15] approached animation synthesis as a path optimisation problem where the user provides intermediate poses and/or start/end positions that act as hard constraints. Additional soft constraints include path distance and animation length. Optimisation is then addressed primarily using a linear programming solver.

D.3.2 Parametric Motion

Parametric approaches blend motion from different samples to achieve the desired animation. The term *parametric motion* originates from the involvement of the user in the control of the animation via high-level parameters that are mapped to low-level blending information [PP10]. This is different from motion graphs, where an optimisation function is the only way to control the graph search. We loosely follow the division of [PP10] into: motion retrieval, motion blending and parametric motion graphs. In *motion retrieval* methods, the original database is analysed and converted into a more compact representation that allows cheap search of the motions via a query. *Motion blending* approaches deal specifically with interpolation of 2 or more motions to create a new one. Finally, *parametric motion graphs* build on top of motion graphs, but add the effort of controlling the structure of the graph, which simplifies the choice of the relevant nodes and, thus, reduces the cost of searching the best path [HG07]. Pejsa and Pandzic's survey [PP10] presents a thorough review on this field, which cannot be fully presented here, nor is that the goal, as not all methods are directly related to our work. Also, despite having about 6 years, it is still largely updated, so, we forward the reader to it for a more in-depth review.

Motion retrieval techniques appeared due to the sheer number of motion samples found in existing databases. While, in the beginning of time, it was possible to manually organise motion databases as to allow easy retrieval of its content, the ever-increasing amount of recorded data makes such approach infeasible nowadays. Motion retrieval techniques aim to efficiently, and accurately, search the database and return sequences similar to the one provided by the user [PP10], which can then be used to create the final motion by concatenation. Amongst the many approaches for motion retrieval, we include here the following:

- Arikan et al. [AFO03] allow specifying a sequence of annotations, e.g. walk or jump, which steer the animation and are extracted directly from the DB. All samples need to be labelled, so to speed up this process they train a support vector machine (SVM) to help estimating the labels for new sequences. Motion synthesis is approached as an optimisation via dynamic programming with multiple constraints, which include time-line annotations and, for example, pose constraints that specify the final pose and orientation. Smoothness is enforced using a distance metric similar to [KGP02, AF02] computed in PCA space to reduce the cost. Additionally, since dynamic programming is too computationally costly to apply on all the DB frames, they first apply it to subsequences. Hence, achieving a rough animation. They then refine the search for consecutively smaller sequences;

- Kovar and Gleicher [KG04] introduce the concept of match webs, a structure that stores the optimal correspondences for sub-sequences of the original motions. This is constructed by temporally aligning two sequences with dynamic programming and enforcing e.g. continuity and monotonicity to the mapping. Each time a query is provided, the precomputed match webs allow finding similar motions in an iterative way, i.e. first they find closely related sequences and then use these to find more sequences and so on. With these, they create a parametrised motion space to map the blend weights to the motion parameters from user input. They then create the inverse function from scattered data interpolation;
- Müller and Röder [MR06] create motion templates (MT) using different relational binary motion features. A MT is a matrix where each row represents a feature and each column a time frame. Learning occurs by providing a set of samples for a specific motion, from where the features are extracted. One of the motions is chosen as a reference to where all the others are aligned using Dynamic Time Warping (DTW). Using this info, they stretch/compress all the matrices so their length matches, and finally the MT is created via averaging. To compare an unknown motion with a MT, they first quantize it and then match it with the sample using a variation of DTW;
- Deng et al. [DGL09] manually define a spatial hierarchy of body parts and then segment the motion via probabilistic PCA. Sharp probability differences between following frames, exceeding a certain threshold, provide the split points. The segments, associated to each body part, are then grouped via k-means clustering, and a set of basic motion patterns is created by averaging the segments of each cluster. All the patterns are then stored in a kd-tree. Retrieval is obtained by segmenting the sample using the same procedure and finding the corresponding motion patterns for each body part. Finally, they compare the sequence of selected patterns with the DB's sample patterns, via an optimised string match algorithm;
- Kapadia et al. [KCT+13] first simplify the motion by determining the extremes of each joint's motion curves. They then rely on Laban's theory of human motion [LU71] to define the properties describing each motion, which include e.g. displacement and orientation of an end effector, centre of mass or flow of motion. They extract these properties, referred as motion keys, for each sequence. They then construct a trie, i.e. a prefix-based tree structure, with the information from the motion keys. Queries can be specified using constraints in the key space;
- Wang et al. [WLP+14] proposed a technique based on the analysis of the eigen vectors and values. This changes the focus of the comparison from the dynamics to the transform domain. They extract the vectors/values from whole body and local

features of 5 body parts. Retrieval occurs at two levels, first by matching the vectors associated to whole body features, which makes an initial filtering, and then matching the remaining parts. Since the data being compared is considerably reduced, this approach easily works in real-time;

- Wang and Neff [WN15] present a technique that relies on a deep autoencoder to convert a sample to a 32-bit binary representation, which they refer to as deep signature. An autoencoder is formulated as a 4-layered Restricted Boltzmann Machine (RBM), with the first being a Gaussian RBM to convert real-values to binary input. The whole DB is converted into its binary representation, with retrieval occurring by extracting the deep signature of a sample, and matching it with all the deep signatures of the DB. This technique allows representing the DB in a much smaller form and makes the comparisons simple bitwise operations;

Motion blending deals with the process of combining two or more sequences via interpolation. As Wang and Bodenheimer [WB08] refer, a good blending has two main steps: finding the intervals of the sequences where blending will occur, and the duration of these intervals. These are the transition points between the sequences and, thus, are the place where discontinuities can occur. Thus, it is crucial to handle them properly. Approaches for blending include:

- Park et al. [PSS02] blend different motions around locomotion key-times, e.g. heel-strikes or toe-off. Blending occurs in the quaternion space, and alignment is achieved using a timewarping incremental scheme to enforce monotonicity. On top of this, they use scattered data interpolation to create the mapping from high-level parameters to blend weights. One of the control parameters they exploit is the rotation angle, which forces the character to follow a ground path;

- Kovar and Gleicher [KG03] introduce the concept of registration curves. Each curve is composed by: a time warp curve, establishing the correspondence between the frames of two motions; a coordinate alignment curve that builds on top of the time warp curve, and forces the translation and orientation of the matched frames to be the same; and, finally, a set of constraint matches that solve conflicts between the two sequences, e.g. foot placement. Linear blending is then used to achieve the final animation. This technique can also be used to control smaller transitions, in which case, the centre of transition and its length needs to be provided;

- Treuille et al. [TLP07] rely on a model to stitch motion capture clips, and a controller to choose the next clip. Animation sequences are achieved by overlapping the clips

and linearly blending them while respecting each clip's constraints. Additionally, the body is reoriented according to some of its joints. They then use reinforcement learning to create several policies, such as navigation or obstacle avoidance. Since these objectives have different properties in each state, they consider all properties when learning the control policies. By doing this, they can just change the objective and the animation seamlessly shifts to the new goal;

- Wang and Bodenheimer [WB08] use the cost function of [LCR+02] as a distance metric to determine the best blending positions. However, they learn the influence of the joints by analysing motion capture data, as opposed to a manual definition. They then propose two approaches for blending: one by testing multiple blend durations and choosing the one that minimises the geodesical distance between joints; the other following the assumption that joints' changes are naturally smooth. Hence, they analyse the difference between the start and end positions, choosing the blend length that allows joints' velocity changes to be more smooth;

- Huang and Kallman [HK10] approach motion blending purely as an optimisation of the weights that allows spatial constraints to be minimised. Optimisation occurs first by selecting the motions that best satisfy the constraints, following an initial guess of the weights obtained using RBF interpolation. The blending weights are then minimised according to the error functions associated to the constraints, such as end effectors positions;

- Mukai [Muk11] first reduces the dimension of the data, and then converts it into a cycle in the latent space, by close-curved fitting. A motion ring is created by combining temporally aligned motion cycles. However, with many cycles, repetitions are bound to happen. So they first eliminate redundant motions and then trim down the cycles by keeping only the most dissimilar motions. Synthesis occurs by traversing the motion ring, with the blend weights determined by contact spheres. The weights of these spheres are obtained using sampling over the blending space where foot striking occurs. When the character is moving, the weight is retrieved from the contact sphere which collides with the ground. If multiple spheres are detected, the one that best matches the user input is selected;

- Oshita [Osh14] presents a technique that allows the user to just specify the sequence of motions, with their optimal blend range and weights automatically found. Different ways to blend are introduced, such as motion-to-motion or motion-to-pose. The one used is chosen based on constraints like foot contact and velocity, or a manually specified core-motion constraint. The blending type is finally applied independently

for the upper and lower part of the body, thus allowing different blending ranges and weights respectively;

It is interesting to see that issues like alignment are common on all previous methods. This is not a surprise, as it is extremely difficult to capture different sequences exactly in the same conditions, and that would simply allow concatenation directly into each other. Issues like this provide further insights on the kind of challenges that need be addressed when approaching automatic generation of facial animation. **Parametric motion graphs** rely on more manual effort than motion graphs when creating nodes, or groups of nodes, that encode similar motions. This is also known as the process of defining the parametrised motion space. While this can lead to a simpler control, it is important to note that creating a transition between parametrised spaces is even harder than with normal motion graphs. In the latter, the poses are well defined à priori, which is not the case with parametric spaces that can generate an almost infinite number of poses. This makes preparing for all the transitions à priori infeasible.

- Gleicher et al. [GSKJ03] introduced a structure where each node has a group of frames, or match sets, common to multiple samples. Edges contain different clips of motion, with the nodes effectively serving as transition points between the edges. The graph is created one node at a time, with the user specifying either the groups of frames or by choosing a frame. In the last option, the frames' group is automatically extracted by analysing the DB with the distance metric of [KGP02]. Transitions are found using displacement maps and by forcing C^1 continuity for match sets, whose poses have been aligned around a representative frame;
- Shin and Oh [SO06] introduce a structure called fat graphs, where the nodes contain only a frame and the edges one or more motions. The nodes are automatically identified and proposed to the user. This process is based on the frames common to the highest number of sequences, and follows, again, the distance metric of [KGP02]. When multiple sequences between the same nodes occur, these are temporally aligned and, if similar enough, a parametrically controlled fat edge is created. Since it is impossible to control all existing parametrisations, they assume the motion is controlled by specifying certain joint positions (e.g. the foot is specified to generate kicking). The user selects the key joint for each fat edge, and the parameter space is created using the approach of [KG04]. On top of this, they implement a method to translate input from 2D control axis, e.g. in a joystick, to this joint parameterisation;
- Similarly, Heck and Gleicher [HG07] use the work of [KG04] to provide the content for the main nodes of a parametric motion graph. The author chooses which

nodes should connect to each other, and the edges are established using the metric of [KGP02]. The actual transition is, however, established only between frames in the end of the starting node and the frames in the beginning of the end node. To solve the issue of blending two parametric motions, they sample the parameter space of each node and calculate the cost of each transition. If below a certain threshold, they will add it to the edge;

- Beaudoin et al. [BCvdPP08] automatically create a structured graph by identifying motifs in the captured data. They first apply PCA to the sequences to reduce the number of features, and then they use k-means to cluster all the frames. Each cluster is associated to a letter, so a sample can be represented as a sequence of letters. After removing sequential repetitions of letters, they find patterns, i.e. the motifs. They additionally create an adjacency matrix between clusters' hyperspheres, which allows small differences in the strings while still placing them in the same motif. A graph is created with motifs, as the nodes, and with the edges established whenever different motifs share some poses;

- Casas et al. [CTGH12] manually create a graph structure from several sequences of consistent meshes obtained using 4D motion capture. Parametric control over each node is achieved by creating an inverse mapping function with [KG04]. Transitions are computed in run-time based on the current and destination nodes, by minimizing a cost function that accounts for latency, shape and motion. Blending is achieved with a hybrid non-linear approach, where they precompute the differences between linear and non-linear mesh blending. In real-time, they first find the linear interpolation parameters and add the stored non-linear information;

D.3.3 Statistical-based approaches

Statistical methods learn the motion models from the variation of recorded motion data. These approaches lean heavily on machine learning concepts such as: PCA [EMMT04, MC12], Gaussian Processes [GMHP04, WFH08, UK12, LWH$^+$12], reinforcement learning [LWB$^+$10], Linear/nonlinear dynamic systems [LWS02, Bis05], and deep learning [THR11, FLFM15, HSK16]. Additionally, statistical-based methods can also be combined with graphs [LWS02, UK12]. These techniques fall in grey areas, hence we have loosely tried to identify the core of the method. If it involved graphs, then it would be placed in either of the previous graph related sections (Sec. D.3.2 or D.3.1), otherwise, it will be discriminated in the following:

- Li et al. [LWS02] introduce the concept of motion textures, a two-layer statistical model that, at the lowest level, has several linear dynamic systems (motion texton), and on top, has a distribution of textons and respective transition probabilities. While the first models the local dynamics, the second deals with the global combination of individual motions. Textons are optimised using maximum likelihood in an incremental approach, i.e. they try to fit consecutive frames into a texton and, when the error accumulates over a threshold, they create another texton. Motion synthesis occurs by specifying the start/end textons, and the path is calculated with Dijkstra, applied using weights obtained from the transition probabilities;

- Egges et al. [EMMT04] capture idle motions and manually segment them into their basic movements, e.g. "balance on the left foot". Afterwards, they convert the motion features into PCA space and create a graph. The probability of transitioning is obtained by comparing the number of basic movement transitions against the total number of transitions involving those basic movements. Synthesis occurs by randomly selecting the next motion, following the transition probabilities. Noise is added to each principal component, via Perlin noise [Per85];

- Despite Grochow et al. [GMHP04] not providing a full method capable motion synthesis, which requires a small number of joint positions every frame, it is still one of the most commonly referred techniques in this field. The authors obtain the probability distribution between all the possible poses, via a Scaled Gaussian Process Latent Variable Model (GPLVM) that converts the input features into a low dimensional representation (latent space). The samples allow learning the parameters that define this space via optimisation. Animation is generated by providing the position of a reduced number of joints through e.g. a trajectory curve, and the remaining joint configuration is automatically obtained;

- Bissacco [Bis05] approaches the problem using switching linear dynamic systems, which contain different linear systems for modelling separate parts of the motion. These are activated whenever a certain condition is met, i.e. switch. They first segment the input to detect the switch instants and then use the segments to learn the linear systems. Motion is synthesised by choosing an initial state, and then alternating between the linear systems. The number of frames generated on a specific system is found by sampling of the normal distribution, determined from the sequences' length;

- Wang et al. [WFH08] approach synthesis as a problem of modelling non-linear time series via Gaussian Process Dynamic Models (GPDMs). This model is comprised of a mapping between the skeleton to a latent space, and by a dynamical model

in this space, which deals with the temporal dependence between the poses. They experiment several techniques to create this mapping ranging from MAP to manually tuning some of the estimated parameters. Synthesis uses the distribution present in the current position/pose in the latent space, by mean prediction. It is important to note that each movement has its own time series, obtained from similar segments;

- Lee et al. [LWB+10] introduce the concept of motion fields, where original poses form initial space states that contain the joint/velocity information. These states form the basis for a continuous space, where a pose is created by combining information from its k-nearest neighbour states. Each state has set of actions, e.g. move in a certain direction, learned with reinforcement learning. This allows the next pose to be created using a convex combination of its k-nearest neighbours;
- Taylor et al. [THR11] used different types of conditional Restricted Boltzmann Machine (RBM) to model the motion as time-series. In its basic form, this approach relies in two layers of nodes, one hidden and one real-valued visible layer. Generation is done online, with the past few steps used to determine the biases of the visible and hidden units, and combined with alternating Gibbs sampling;
- Min and Chai [MC12] present another extension on the concept of motion graphs by implementing a generative statistical model in each node, referred to as morphable motion primitive. Each node models the style variations of a base motion, and the global dynamics are defined as the edges. Data is segmented according to the contact points, with similar segments assigned to each motion primitive. The model is created using functional PCA, and the prior distribution function is extracted following a Gaussian Mixture Model. The goal of this distribution is to force synthesis to be close to the captured data. Synthesis builds on top of a MAP framework, where the user provides a command, and the nodes that best satisfy it are chosen. They additionally embed semantic information in both, the original data and motion primitives, which allows fine control over the animation and can be used as a driving command;
- Levine et al. [LWH+12] propose an approach based on GPLVMs. Similar to previous approaches, it converts motion parameters into a smaller space. This method accepts the motion samples with the respective controller parameters, e.g. direction, and the reward function, e.g. moving in the desired direction. The GPLVM is enhanced with a connectivity prior to enforce a dense connectivity within the latent space. Non-parametric approximate dynamic programming is used to precompute the policies that drive synthesis;
- Ukita and Kanade [UK12] combine GPDM with motion graphs for human tracking. Samples can easily be represented in the latent space, however, the more complex

issue is how to handle the transitions. They present a topologically-constrained model that arranges the latent space so the transitions between the poses occur easily. This is created by comparing the pairs of latent pose values and using this info to reconstruct the GPDM so their values closer. Sample poses around the extracted end points are interpolated in the observation space to create intermediate poses in the latent space. A motion graph is constructed in the latent space, with the path that connects two values being the node, and the latent variable that connects two paths being the edge. Finally, Dijkstra is applied between all pairs of original nodes/edges. The ones not included in any of the paths are discarded. They have applied this model to motion tracking, however, this could also be used in motion synthesis;

- Fragkiadaki et al. [FLFM15] propose a model that, at its core, has a recurrent neural network and incorporates non-linear encoder and decoder networks before and after respectively, also referred as ERD (Encoder-Recurrent-Decoder). The encoder transforms the data to a representation where learning is easier, while the decoder does the opposite task of going back to the original data. Both are fully connected networks. The recurrent layer learns the dynamics, which are represented on its weights. ERDs automatically, and simultaneously, learn both the representation and dynamics. Prediction occurs based on the current pose. Only the first pose needs to be provided, with the remaining ones generated by the model;
- Holden et al. [HSK16] combine several neural networks to model and synthesize the motion. They first train a convolutional autoencoder that models the motion data, and does not require motion segmentation or alignment. On top of this, they train a deep forward neural network, responsible for converting the high-level parameters into the low-level parameters of motion contained in the hidden units of the encoder. To solve ambiguities of the parameterisations, they train an additional network that extracts the foot contacts on the ground. They also allow changes in animation style by controlling the values of key hidden units;

Interestingly, some statistical-based approaches [FLFM15, HSK16] automatically remove the effects of alignment when creating the generalised model. As such, moving away from the need to explicitly solve this issue.

D.3.4 Comparing example-based approaches

The current chapter presents by no means a thorough look at the state of the art in the example-based body synthesis, but it allows seeing some trends and common steps required

to automate body animation. Most recent results already show very good quality so, arguably, quality is not the best differentiating feature. Additionally, quality is, per se, hard to evaluate. Instead, we opt to empirically compare the amount of *input* against *control* of the animations and *ease of use*. Parametric approaches appear to be the most *intuitive*, as they require either choosing the blended/retrieved animations or specifying the type of motion included in the nodes of a graph. This means the author implicitly knows how the final animation will look like. Additionally, parametric motion graphs rely on a structure that can be used to provide visual aids to the type of generated animation. Non-parametric motion graphs also benefit from this aspect, but, as the graph is unstructured, it is more difficult to identify the relevant node/edges. Motion graphs and statistical-based approaches usually require lower-level parameters, such as end effectors position, and need a specific reward/optimisation function, which makes them less intuitive to control. Statistical-based approaches also tend to work in a black box format, being very hard to interpret their internal representation. Regarding *input*, the tables are turned. Parametric approaches require more input in both the configuration and animation stages. Motion graphs and statistical-based approaches appear to be almost equivalent, with the latter, arguably, having the edge since it can completely remove the aligning process. Finally, regarding *control*, motion graphs provide the worst control overall. Zhao and Safonova [ZS09] focused their efforts on improving the connectivity, which effectively reduces the motion transitioning lag. However, there is the issue of the motion graph's underlying representation being discrete, which, given that motion is continuous, leads the model having to fill in the missing gaps. Some parametric approaches already enter the continuous domain, but they are still limited to the direct combination of samples. Statistical methods can generalise the motion, so they have the highest capacity to learn the transitions, thus providing the best control.

References

[3D17] Artec 3D. Artec 3d scanners. http://www.artec3d.com, 2017. Accessed: 05/04/2017.

[ACR09] Zara Ambadar, Jeffrey. Cohn, and LawrenceIan Reed. All smiles are not created equal: Morphology and timing of smiles perceived as amused, polite, and embarrassed/nervous. *Journal of Nonverbal Behavior*, 33(1):17–34, 2009.

[AD07] Ali Arya and Steve DiPaola. Multispace behavioral model for face-based affective social agents. *J. Image Video Process.*, 2007(1):4–4, January 2007.

[AF02] Okan Arikan and D. A. Forsyth. Interactive motion generation from examples. *ACM Trans. Graph.*, 21(3):483–490, July 2002.

[AFO03] Okan Arikan, David A. Forsyth, and James F. O'Brien. Motion synthesis from annotations. In *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, pages 402–408, New York, NY, USA, 2003. ACM.

[AFO05] Okan Arikan, David A. Forsyth, and James F. O'Brien. Pushing people around. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pages 59–66, New York, NY, USA, 2005. ACM.

[AIM17] Artificial intelligence markup language - aiml. http://www.alicebot.org/, 2017. Accessed: 25/1/2017.

[And12] Christina Anderson. *Idiosyncratic Facial Movement in Face Perception and Recognition*. PhD thesis, Universität Würzburg, Fakultät für Humanwissenschaften, 2012.

[And15] Carolyn J. Anderson. Multivariate normal distribution. Lectures - University of Illinois, 2015. Last acessed on: 21–09–15.

[APM07] Ali Arya, Avi Parush, and Alicia McMullan. Perceptually valid facial expression blending using expression units. In *ACM SIGGRAPH 2007 Posters*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[Arg73] Michael Argyle. *Social interaction*, volume 103. Transaction Publishers, 1973.

[Ari12] Dynamic animation and control environment. `http://www.arishapiro.com/dance/`, 2012. Accessed: 12/8/2016.

[ARL+09] O. Alexander, M. Rogers, W. Lambeth, M. Chiang, and P. Debevec. Creating a photoreal digital actor: The digital emily project. In *CVMP - Conference for Visual Media Production*, pages 176–187, Nov 2009.

[ARV07] B. Amberg, S. Romdhani, and T. Vetter. Optimal step nonrigid icp algorithms for surface registration. In *In proceedings of the conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, Minneapolis, MN, USA, June 2007. IEEE Computer Society Press.

[ATL12] Ken Anjyo, Hideki Todo, and J.P. Lewis. A practical approach to direct manipulation blendshapes. *Journal of Graphics Tools*, 16(3):160–176, 2012.

[AZ10] Olusola O. Aina and Jian Jun Zhang. Automatic muscle generation for physically-based facial animation. In *ACM SIGGRAPH 2010 Posters*, SIGGRAPH '10, pages 105:1–105:1, New York, NY, USA, 2010. ACM.

[BBA+07] Bernd Bickel, Mario Botsch, Roland Angst, Wojciech Matusik, Miguel Otaduy, Hanspeter Pfister, and Markus Gross. Multi-scale capture of facial geometry and motion. *ACM Trans. Graph.*, 26(3), July 2007.

[BBBB+03] Enrique Bailly-Bailliére, Samy Bengio, Frédéric Bimbot, Miroslav Hamouz, Josef Kittler, Johnny Mariéthoz, Jiri Matas, Kieron Messer, Vlad Popovici, Fabienne Porée, Belen Ruiz, and Jean-Philippe Thiran. The banca database and evaluation protocol. In *Proceedings of the 4th International Conference on Audio- and Video-based Biometric Person Authentication*, AVBPA'03, pages 625–638, Berlin, Heidelberg, 2003. Springer-Verlag. `http://www.ee.surrey.ac.uk/CVSSP/banca/`.

[BCvdPP08] Philippe Beaudoin, Stelian Coros, Michiel van de Panne, and Pierre Poulin. 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, 2008. Eurographics Association.

[BE12] B. Basten and A. Egges. Motion transplantation techniques: A survey. *IEEE Computer Graphics and Applications*, 32(3):16–23, May 2012.

[Bea12] Andy Beane. *3D Animation Essentials*. John Wiley & Sons, 2012.

[BGY+13] Kiran S. Bhat, Rony Goldenthal, Yuting Ye, Ronald Mallet, and Michael Koperwas. High fidelity facial animation capture and retargeting with contours. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, pages 7–14, New York, NY, USA, 2013. ACM.

[BHB+11] Thabo Beeler, Fabian Hahn, Derek Bradley, Bernd Bickel, Paul Beardsley, Craig Gotsman, Robert W. Sumner, and Markus Gross. High-quality passive facial performance capture using anchor frames. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 75:1–75:10, New York, NY, USA, 2011. ACM.

[BHPN01] The Duy Bui, Dirk Heylen, Mannes Poel, and Anton Nijholt. Generation of facial expressions from emotion using a fuzzy rule based system. In *Proceedings of the 14th Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence*, AI '01, pages 83–94, London, UK, UK, 2001. Springer-Verlag.

[BHPS10] Derek Bradley, Wolfgang Heidrich, Tiberiu Popa, and Alla Sheffer. High resolution passive facial performance capture. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 41:1–41:10, New York, NY, USA, 2010. ACM.

[Bir70] Ray L. Birdwhistell. *Kinesics and context: Essays on body motion communication*. University of Philadelphia Press, Philadelphia, 1970.

[Bis05] A. Bissacco. Modeling and learning contact dynamics in human motion. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 421–428 vol. 1, June 2005.

[BLF+06] Marian Stewart Bartlett, Gwen Littlewort, Mark G Frank, Claudia Lainscsek, Ian R Fasel, and Javier R Movellan. Automatic recognition of facial actions in spontaneous expressions. *Journal of mul-*

timedia, 1(6):22–35, 2006. http://mplab.ucsd.edu/grants/project1/research/rufacs1-dataset.html.

[BMS12] Tanja Bänziger, Marcello Mortillaro, and Klaus R Scherer. Introducing the geneva multimodal expression corpus for experimental research on emotion perception. *Emotion*, 12(5):1161, 2012. http://www.affective-sciences.org/en/gemep/.

[BPL+03] George Borshukov, Dan Piponi, Oystein Larsen, J. P. Lewis, and Christina Tempelaar-Lietz. Universal capture: Image-based facial animation for "the matrix reloaded". In *ACM SIGGRAPH 2003 Sketches &Amp; Applications*, SIGGRAPH '03, pages 1–1, New York, NY, USA, 2003. ACM.

[BS10] Tanja Bänziger and Klaus R Scherer. Introducing the geneva multimodal emotion portrayal (gemep) corpus. *Blueprint for affective computing: A sourcebook*, pages 271–294, 2010. http://www.affective-sciences.org/en/gemep/.

[BSG10] R. Bidarra, R. Schaap, and K. Goossens. Growing on the inside: Soulful characters for video games. In *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*, pages 337–344. IEEE Computer Society, 2010.

[BSK+07] Rob Bredow, David Schaub, Daniel Kramer, Matthew Hausman, Danny Dimian, and R. Stirling Duguid. Surf's up: The making of an animated documentary. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07, pages 1–123, New York, NY, USA, 2007. ACM.

[BW95] Armin Bruderlin and Lance Williams. 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, 1995. ACM.

[BWP13] Sofien Bouaziz, Yangang Wang, and Mark Pauly. Online modeling for realtime facial animation. *ACM Trans. Graph.*, 32(4):40:1–40:10, July 2013.

[CBE+15] Matthew Cong, Michael Bao, Jane L. E, Kiran S. Bhat, and Ronald Fedkiw. Fully automatic generation of anatomical face simulation models. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on*

Computer Animation, SCA '15, pages 175–183, New York, NY, USA, 2015. ACM.

[CBZB15] Chen Cao, Derek Bradley, Kun Zhou, and Thabo Beeler. Real-time high-fidelity facial performance capture. *ACM Trans. Graph.*, 34(4):46:1–46:9, July 2015.

[CCP02] Berardina De Carolis, Valeria Carofiglio, and Catherine Pelachaud. From discourse plans to believable behavior generation. *Proceedings of the 2nd International Conference on Natural Language Generation (INLG 2002)*, 2002.

[Cen11] Porto Interactive Center. Learning of facial emotions using serious games. http://www.portointeractivecenter.org/lifeisgame/, 2011. Accessed: 15/4/2017.

[Cha07] Alex Champandard. Behavior trees for next-gen game ai. In *Game Developers Conference - AI Summit*, 2007.

[CHP89] J. E. Chadwick, D. R. Haumann, and R. E. Parent. Layered construction for deformable animated characters. *SIGGRAPH Comput. Graph.*, 23(3):243–252, July 1989.

[CHZ14] Chen Cao, Qiming Hou, and Kun Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph.*, 33(4):43:1–43:10, July 2014.

[CK05] Byoungwon Choe and Hyeong-Seok Ko. Analysis and synthesis of facial expressions with hand-generated muscle actuation basis. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[CKH11] D. Cosker, E. Krumhuber, and A. Hilton. A facs valid 3d dynamic action unit database with applications to 3d dynamic morphable facial modeling. In *2011 International Conference on Computer Vision*, pages 2296–2303, Nov 2011. `http://www.cs.bath.ac.uk/~dpc/D3DFACS/`.

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2009.

[CPB+94] Justine Cassell, Catherine Pelachaud, Norman Badler, Mark Steedman, Brett Achorn, Tripp Becket, Brett Douville, Scott Prevost, and Matthew

Stone. 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. ACM, 1994.

[CPSMS72] Yaron Canetti, Mark Piretti, Raffaele Scaduto-Mendola, and Jason Schleifer. Character setup from rig mechanics to skin deformations: A practical approach. In *ACM SIGGRAPH 2002 Course Notes*, SIGGRAPH '02, pages 1–163, New York, NY, USA, 20072. ACM.

[CTGH12] Dan Casas, Margara Tejera, Jean-Yves Guillemaut, and Adrian Hilton. 4d parametric motion graphs for interactive animation. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '12, pages 103–110, New York, NY, USA, 2012. ACM.

[CVB01] Justine Cassell, Hannes Högni Vilhjálmsson, and Timothy Bickmore. 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, 2001. ACM.

[CWLZ13] Chen Cao, Yanlin Weng, Stephen Lin, and Kun Zhou. 3d shape regression for real-time facial animation. *ACM Trans. Graph.*, 32(4):41:1–41:10, July 2013.

[CXH03] Jin-xiang Chai, Jing Xiao, and Jessica Hodgins. Vision-based control of 3d facial animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 193–206, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[Dav11] Martin Davies. Concept mapping, mind mapping and argument mapping: what are the differences and do they matter? *Higher Education*, 62(3):279–301, 2011.

[DCFN06] Zhigang Deng, Pei-Ying Chiang, Pamela Fox, and Ulrich Neumann. Animating blendshape faces by cross-mapping motion capture data. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, I3D '06, pages 43–48, New York, NY, USA, 2006. ACM.

[DDLT02] Frédéric Devillers, StÃ©phane Donikian, Fabrice Lamarche, and Jean-Francois Taille. A programming environment for behavioural animation.

The Journal of Visualization and Computer Animation, 13(5):263–274, 2002.

[DeC00] Dimitris DeCarlo, Douglas nd Metaxas. Optical flow constraints on deformable models with applications to face tracking. *International Journal of Computer Vision*, 38(2):99–127, 2000.

[DGL09] Zhigang Deng, Qin Gu, and Qing Li. Perceptually consistent example-based human motion retrieval. In *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games*, I3D '09, pages 191–198, New York, NY, USA, 2009. ACM.

[DGLG11] A. Dhall, R. Goecke, S. Lucey, and T. Gedeon. Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 2106–2112, Nov 2011.

[DGLG12] A. Dhall, R. Goecke, S. Lucey, and T. Gedeon. Collecting large, richly annotated facial-expression databases from movies. *IEEE MultiMedia*, 19(3):34–41, July 2012.

[Dij59] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[DiP13] Steve DiPaola. *Face, Portrait, Mask: Using a Parameterised System to Explore Synthetic Face Space*, pages 213–227. Springer London, London, 2013.

[DN08] Zhigang Deng and Junyong Noh. Computer facial animation: A survey. In Zhigang Deng and Ulrich Neumann, editors, *Data-Driven 3D Facial Animation*, pages 1–28. Springer London, 2008.

[Dyn16] Dynamixyz grabber/analyzer/bridge. `http://www.dynamixyz.com/`, 2016. Accessed: 7/8/2016.

[EBDP96] Irfan Essa, Sumit Basu, Trevor Darrell, and Alex Pentland. Modeling, tracking and interactive animation of faces and heads using input from video. In *Proceedings of the Computer Animation*, CA '96, pages 68–, Washington, DC, USA, 1996. IEEE Computer Society.

[EF71] P Ekman and W Friesen. Constants across cultures in the face and emotion. *J. Personality and Social Psychology*, 17:124–129, 1971.

[EF78] Paul Ekman and Wallace V. Friesen. *The Facial Action Coding System: A technique for the measurement of facial movement*. Consulting Psychologists Press, Palo Alto, California, 1978.

[EFH02] Paul Ekman, Wallace V. Friesen, and Joseph C. Hager. *Facial Action Coding System - Investigator's Guide*. 2002.

[Ekm03] Paul Ekman. *Emotions revealed: recognizing faces and feelings to improve communication and emotional life*. Times Books, 2003.

[EKMT03] Arjan Egges, Sumedha Kshirsagar, and Nadia Magnenat-Thalmann. *A Model for Personality and Emotion Simulation*, pages 453–461. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[EMH14] Christopher Evans, Lars Martinsson, and Sascha Herfort. Building an empire: Asset production in ryse. In *ACM SIGGRAPH 2014 Courses*, SIGGRAPH '14, pages 18:1–18:49, New York, NY, USA, 2014. ACM.

[EMMT04] A. Egges, T. Molet, and N. Magnenat-Thalmann. Personalised real-time idle motion synthesis. In *Computer Graphics and Applications, 2004. PG 2004. Proceedings. 12th Pacific Conference on*, pages 121–130, Oct 2004.

[EMT05] A. Egges and N. Magnenat-Thalmann. Emotional communicative body animation for multiple characters. In *International Workshop on Crowd Simulation (V-Crowds)*, pages 31–40, November 2005.

[Epi16] Epic. Unreal engine 4. `https://www.unrealengine.com/` and `https://docs.unrealengine.com/latest/INT/Engine/Animation/PhysicallyDrivenAnimation/`, 2016. Accessed: 12/8/2016.

[Fac15] Faceware. Facial rigs faceware. `http://facewaretech.com/sdm_categories/rigs/`, 2015. Accessed: 15/4/2017.

[Fac16a] Faceshift studio. `http://faceshift.com/studio/2015.2/introduction.html`, 2016. Accessed: 7/8/2016.

[Fac16b] Faceware analyser and retargeter. `http://facewaretech.com/products/software/`, 2016. Accessed: 7/8/2016.

[FBM11] Adso Fernández-Baena and David Miralles. Progressive transitions using body part motion graphs. In *SIGGRAPH Asia 2011 Posters*, SA '11, pages 3:1–3:2, New York, NY, USA, 2011. ACM.

[FGG+99] M Frey, P Giovanoli, H Gerber, M Slameczka, and E StÃ¼ssi. Three-dimensional video analysis of facial movements: a new method to assess the quantity and quality of the smile. *Plastic And Reconstructive Surgery*, 104(7):2032 – 2039, 1999.

[FJA+14] Graham Fyffe, Andrew Jones, Oleg Alexander, Ryosuke Ichikari, and Paul Debevec. Driving high-resolution facial scans with video performance capture. *ACM Trans. Graph.*, 34(1):8:1–8:14, December 2014.

[FLFM15] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4346–4354, 2015.

[For03] Joe Fordham. Middle-earth strikes back. In *Cinefex*. 2003.

[FSOnO12] Tiago Fernandes, José Serra, Juan Órdo nez, and Verónica Orvalho. Mind maps as behavior controllers for virtual characters. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '12, pages 2291–2296, New York, NY, USA, 2012. ACM.

[Geb05] Patrick Gebhard. Alma: A layered model of affect. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '05, pages 29–36, New York, NY, USA, 2005. ACM.

[Gil16] Jonas Gillberg. Ai behavior editing and debugging in 'tom clancy's the division'. In *Game Developers Conference - AI Summit*, 2016.

[GKN08] M. Grimm, K. Kroschel, and S. Narayanan. The vera am mittag german audio-visual emotional speech database. In *2008 IEEE International Conference on Multimedia and Expo*, pages 865–868, June 2008.

[Gle08] Michael Lee Gleicher. Graph-based motion synthesis: An annotated bibliography. In *ACM SIGGRAPH 2008 Classes*, SIGGRAPH '08, pages 49:1–49:11, New York, NY, USA, 2008. ACM.

[GMHP04] Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popović. Style-based inverse kinematics. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 522–531, New York, NY, USA, 2004. ACM.

[Gol16] Golaem. `http://golaem.com/content/product/golaem-features#customAssets`, 2016. Accessed: 13/8/2016.

[GP06] H. Gunes and M. Piccardi. A bimodal face and body gesture database for automatic analysis of human nonverbal affective behavior. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 1, pages 1148–1153, 2006.

[GP12] T. Geijtenbeek and N. Pronost. Interactive character animation using simulated physics: A state-of-the-art review. *Computer Graphics Forum*, 31(8):2492–2515, 2012.

[GRVT+06] A. GarcÃ?a-Rojas, F. Vexo, D. Thalmann, A. Raouzaiou, K. Karpouzis, S. Kollias, L. Moccozet, and N. Magnenat-Thalmann. Emotional face expression profiles supported by virtual human ontology. *Computer Animation and Virtual Worlds*, 17(3-4):259–269, 2006.

[GS10] Marco Gillies and Bernhard Spanlang. Comparing and evaluating real time character engines for virtual environments. *Presence: Teleoper. Virtual Environ.*, 19(2):95–117, April 2010.

[GSBS01] Maia Garau, Mel Slater, Simon Bee, and Martina Angela Sasse. The impact of eye gaze on communication using humanoid avatars. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '01, pages 309–316, New York, NY, USA, 2001. ACM.

[GSC+15] Shihui Guo, Richard Southern, Jian Chang, David Greer, and Jian Jun Zhang. Adaptive motion synthesis for virtual characters: A survey. *Vis. Comput.*, 31(5):497–512, May 2015.

[GSKJ03] Michael Gleicher, Hyun Joon Shin, Lucas Kovar, and Andrew Jepsen. Snap-together motion: Assembling run-time animations. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics*, I3D '03, pages 181–188, New York, NY, USA, 2003. ACM.

[GSV+03] Maia Garau, Mel Slater, Vinoba Vinayagamoorthy, Andrea Brogni, Anthony Steed, and M. Angela Sasse. The impact of avatar realism and eye gaze control on perceived quality of communication in a shared immersive virtual environment. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 529–536, New York, NY, USA, 2003. ACM.

[GVWT13] Pablo Garrido, Levi Valgaert, Chenglei Wu, and Christian Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph.*, 32(6):158:1–158:10, November 2013.

[Hal06] Judith A. Hall. *Women's and Men's Nonverbal Communication: Similarities, Differences, Stereotypes, and Origins*, pages 201–219. SAGE Publications, Inc., 2006.

[Hav06] Parag Havaldar. Sony pictures imageworks. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[Hav16] Havok. Havok physics. `http://www.havok.com/physics/`, 2016. Accessed: 12/8/2016.

[HCTW11] Haoda Huang, Jinxiang Chai, Xin Tong, and Hsiang-Tao Wu. Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition. *ACM Trans. Graph.*, 30(4):74:1–74:10, July 2011.

[HG07] Rachel Heck and Michael Gleicher. Parametric motion graphs. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, I3D '07, pages 129–136, New York, NY, USA, 2007. ACM.

[HK10] Yazhou Huang and Marcelo Kallmann. Motion parameterization with inverse blending. In *Proceedings of the Third International Conference on Motion in Games*, MIG'10, pages 242–253, Berlin, Heidelberg, 2010. Springer-Verlag.

[HKG06] Rachel Heck, Lucas Kovar, and Michael Gleicher. Splicing upper-body actions with locomotion. *Computer Graphics Forum*, 25(3):459–466, 2006.

[HMYL15] P. L. Hsieh, C. Ma, J. Yu, and H. Li. Unconstrained realtime facial performance capture. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1675–1683, June 2015.

[HNMP10] Sylwia Hyniewska, Radosław Niewiadomski, Maurizio Mancini, and Catherine Pelachaud. Expression of affects in embodied conversational agents. *A blueprint for an affective computing: A Sourcebook and Manual*, pages 213–221, 2010.

[HOP+05] David Hanson, Andrew Olney, Steve Prilliman, Eric Mathews, Marge Zielke, Derek Hammons, Raul Fernandez, and Harry Stephanou. Upending the uncanny valley. In *Proceedings of the 20th National Conference on*

Artificial Intelligence - Volume 4, AAAI'05, pages 1728–1729. AAAI Press, 2005.

[HOT06] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, July 2006.

[HR10] Gregor Hofer and Korin Richmond. Comparison of HMM and TMDN methods for lip synchronisation. In *Proc. Interspeech*, pages 454–457, Makuhari, Japan, September 2010.

[HSK16] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4):138:1–138:11, July 2016.

[HTCH15] Peng Huang, Margara Tejera, John Collomosse, and Adrian Hilton. Hybrid skeletal-surface motion graphs for character animation from 4d performance capture. *ACM Trans. Graph.*, 34(2):17:1–17:14, March 2015.

[Inv] Singular Inversions. Facegen. `http://facegen.com/`.

[JKK+11] Yvonne Jung, Arjan Kuijper, Michael Kipp, Jan Miksatko, Jonathan Gratch, and Daniel Thalmann. Believable virtual characters in human-computer dialogs. In *in Proc. of the 32nd Annual Conference of the European Association for Computer Graphics - Eurographics*, 2011.

[JW12] Qiang Ji and Yue Wu. Nvie - feature points. `http://nvie.ustc.edu.cn/Our%20Work.html`, 2012. Accessed: 15/4/2017.

[KB84] Doris H. U. Kochanek and Richard H. Bartels. Interpolating splines with local tension, continuity, and bias control. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 33–41, New York, NY, USA, 1984. ACM.

[KCBW12] Kathrin Kaulard, Douglas W. Cunningham, Heinrich H. Bulthoff, and Christian Wallraven. The mpi facial expression database - a validated database of emotional and conversational facial expressions. *PLOS ONE*, 7:1–18, 03 2012. `https://www.b-tu.de/en/graphic-systems/databases/the-large-mpi-facial-expression-database`.

[KCT+13] Mubbasir Kapadia, I-kao Chiang, Tiju Thomas, Norman I. Badler, and Joseph T. Kider, Jr. Efficient motion retrieval in large motion databases. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '13, pages 19–28, New York, NY, USA, 2013. ACM.

[Ker09] Isaac V. Kerlow. *The Art of 3D Computer Animation and Effects*. John Wiley & Sons, 2009.

[KG03] Lucas Kovar and Michael Gleicher. Flexible automatic motion blending with registration curves. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 214–224, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[KG04] Lucas Kovar and Michael Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Trans. Graph.*, 23(3):559–568, August 2004.

[KGP02] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 473–482, New York, NY, USA, 2002. ACM.

[Kin09] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[Kit] Midori Kitagawa. Golaem. `http://www.utdallas.edu/atec/midori/Handouts/procedural_animation.htm`. Accessed: 13/8/2016.

[KMMT01] S. Kshirsagar, T. Molet, and N. Magnenat-Thalmann. Principal components of expressive speech animation. In *Proceedings. Computer Graphics International 2001*, pages 38–44, 2001.

[KMT02] Sumedha Kshirsagar and Nadia Magnenat-Thalmann. A multilayer personality model. In *Proceedings of the 2Nd International Symposium on Smart Graphics*, SMARTGRAPH '02, pages 107–115, New York, NY, USA, 2002. ACM.

[KMTT92] Prem Kalra, Angelo Mangili, Nadia Magnenat Thalmann, and Daniel Thalmann. Simulation of facial muscle actions based on rational free form deformations. *Computer Graphics Forum*, 11(3):59–69, 1992.

[KTC00] Takeo Kanade, Yingli Tian, and Jeffrey F. Cohn. Comprehensive database for facial expression analysis. In *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000*, FG '00, pages 46–, Washington, DC, USA, 2000. IEEE Computer Society. `http://www.consortium.ri.cmu.edu/ckagree/`.

[LA10] J. P. Lewis and Ken Anjyo. Direct manipulation blendshapes. *IEEE Comput. Graph. Appl.*, 30(4):42–50, July 2010.

[LAGP09] Hao Li, Bart Adams, Leonidas J. Guibas, and Mark Pauly. Robust single-view geometry and motion reconstruction. *ACM Trans. Graph.*, 28(5):175:1–175:10, December 2009.

[Lak06] Jessica L. Lakin. *Automatic Cognitive Processes and Nonverbal Communication*, pages 59–79. SAGE Publications, Inc., 2006.

[LAR+14] J. P. Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Fred Pighin, and Zhigang Deng. Practice and theory of blendshape facial models. In *in Proc. of the 35th Annual Conference of the European Association for Computer Graphics - Eurographics*, pages 199–218. ACM, 2014.

[Las87] John Lasseter. Principles of traditional animation applied to 3D computer animation. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques - SIGGRAPH '87*, volume 21, pages 35–44, New York, New York, USA, 1987. ACM Press.

[LCF00] J. P. Lewis, Matt Cordner, and Nickson Fong. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 165–172, New York, NY, USA, 2000.

[LCK+10] P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *In proceedings of Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 94–101, San

Francisco, CA, USA, 2010. IEEE Computer Society Press. `http://www.pitt.edu/~emotion/ck-spread.htm`.

[LCP+11] P. Lucey, J. F. Cohn, K. M. Prkachin, P. E. Solomon, and I. Matthews. Painful data: The unbc-mcmaster shoulder pain expression archive database. In *Face and Gesture 2011*, pages 57–64, March 2011.

[LCR+02] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 491–500, New York, NY, USA, 2002. ACM.

[LD08] Q. Li and Z. Deng. Orthogonal-blendshape-based editing system for facial motion capture data. *IEEE Computer Graphics and Applications*, 28(6):76–82, Nov 2008.

[Leo10] David Leopold. *Dynamic Facial Signaling: A Dialog between Brains*. MIT Press, Cambridge, MA, USA, 2010.

[LL04] Jehee Lee and Kang Hoon Lee. Precomputing avatar behavior from human motion data. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '04, pages 79–87, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[LL13] Sukwon Lee and Sung-Hee Lee. An efficient motion graph searching algorithm for augmented reality characters. In *Proceedings of the First International Conference on Distributed, Ambient, and Pervasive Interactions - Volume 8028*, pages 449–458, New York, NY, USA, 2013. Springer-Verlag New York, Inc.

[LM08] Brent J. Lance and Stacy C. Marsella. A model of gaze for the purpose of emotional expression in virtual embodied agents. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '08, pages 199–206, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

[LTO+15] Hao Li, Laura Trutoiu, Kyle Olszewski, Lingyu Wei, Tristan Trutna, Pei-Lun Hsieh, Aaron Nicholls, and Chongyang Ma. Facial performance sensing head-mounted display. *ACM Trans. Graph.*, 34(4):47:1–47:9, July 2015.

[LU71] Rudolf Laban and L Ullmann. *The mastery of movement*. Plays Inc, 1971.

[LWB+10] Yongjoon Lee, Kevin Wampler, Gilbert Bernstein, Jovan Popović, and Zoran Popović. Motion fields for interactive character locomotion. *ACM Trans. Graph.*, 29(6):138:1–138:8, December 2010.

[LWH+12] Sergey Levine, Jack M. Wang, Alexis Haraux, Zoran Popović, and Vladlen Koltun. Continuous character control with low-dimensional embeddings. *ACM Trans. Graph.*, 31(4):28:1–28:10, July 2012.

[LWHW12] Christian Lang, Sven Wachsmuth, Marc Hanheide, and Heiko Wersing. Facial communicative signals - valence recognition in task-oriented human-robot interaction. *I. J. Social Robotics*, 4(3):249–262, 2012.

[LWS02] Yan Li, Tianshu Wang, and Heung-Yeung Shum. Motion texture: A two-level statistical model for character motion synthesis. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 465–472, New York, NY, USA, 2002. ACM.

[LXC+15] Yilong Liu, Feng Xu, Jinxiang Chai, Xin Tong, Lijuan Wang, and Qiang Huo. Video-audio driven real-time facial animation. *ACM Trans. Graph.*, 34(6):182:1–182:10, October 2015.

[LYCS11] Jia Liu, Mingyu You, Chun Chen, and Mingli Song. Real-time speech-driven animation of expressive talking faces. *International Journal of General Systems*, 40(4):439–455, 2011.

[LYYB13] Hao Li, Jihun Yu, Yuting Ye, and Chris Bregler. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph.*, 32(4):42:1–42:10, July 2013.

[Mac14] Simon Mack. Parametric blending & fbik: Latest developments in character animation with morpheme. Computer Entertainment Developers Conferece - CEDEC, 2014. `https://youtu.be/DihDTOWf_Zg` Accessed: 11/8/2016.

[MAO+11] José Carlos Miranda, Xenxo Alvarez, João Orvalho, Diego Gutierrez, A. Augusto Sousa, and Verónica Orvalho. Sketch express: Facial expressions made easy. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*, SBIM '11, pages 87–94, New York, NY, USA, 2011. ACM.

[Mas16] Massive prime. `http://www.massivesoftware.com/features.html`, 2016. Accessed: 13/8/2016.

[Mat90] David Matsumoto. Cultural similarities and differences in display rules. *Motivation and Emotion*, 14(3):195–214, 1990.

[Mat06] David Matsumoto. *Culture and Nonverbal Behavior*, pages 219–237. SAGE Publications, Inc., 2006.

[MBC01] Mark Mizuguchi, John Buchanan, and Tom Calvert. Data driven motion transitions for interactive games. In *Eurographics 2001 - Short Presentations*. Eurographics Association, 2001.

[MBXL08] Xia Mao, Haiyan Bao, Yuli Xue, and Zheng Li. *Layered fuzzy facial expression generation: social, emotional and physiological*. INTECH Open Access Publisher, 2008.

[MC12] Jianyuan Min and Jinxiang Chai. Motion graphs++: A compact generative model for semantic motion analysis and synthesis. *ACM Trans. Graph.*, 31(6):153:1–153:12, November 2012.

[Meh97] Albert Mehrabian. Analysis of affiliation-related traits in terms of the pad temperament model. *The Journal of Psychology*, 131(1):101–117, 1997.

[Meh07] Albert Mehrabian. *Nonverbal Communication*. Aldine Transaction, 2007.

[MH11] David Matsumoto and Hyi Sung Hwang. Evidence for training the ability to read microexpressions of emotion. *Motivation and Emotion*, 35(2):181–191, 2011.

[MHM10] Tim K. Marks, John R. Hershey, and Javier R. Movellan. Tracking motion, deformation, and texture using conditionally gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(2):348–363, 2010. `http://mplab.ucsd.edu/wordpress/?page_id=1207`.

[MJ92] Robert R. McCrae and Oliver P. John. An introduction to the five-factor model and its applications. *Journal of Personality*, 60(2):175–215, 1992.

[MM76] H McGurk and J MacDonald. Hearing lips and seeing voices. *Nature*, 264(5588):746–748, 1976.

[MMB+13] S. M. Mavadati, M. H. Mahoor, K. Bartlett, P. Trinh, and J. F. Cohn. Disfa: A spontaneous facial action intensity database. *IEEE Transactions on Affective Computing*, 4(2):151–160, April 2013. `http://www.engr.du.edu/mmahoor/DISFA.htm`.

[MMKJ99] K. Messer, J. Matas, J. Kittler, and K. Jonsson. Xm2vtsdb: The extended m2vts database. In *In Second International Conference on Audio and Video-based Biometric Person Authentication*, pages 72–77, 1999. `http://www.ee.surrey.ac.uk/CVSSP/xm2vtsdb/`.

[Mor70] M. Mori. The uncanny valley. In *Energy 7*, pages 33–35, 1970.

[Mot16a] Natural Motion. Euphoria. `http://www.naturalmotion.com/middleware/euphoria`, 2016. Accessed: 12/8/2016.

[Mot16b] Natural Motion. Morpheme. `http://www.naturalmotion.com/middleware/morpheme`, 2016. Accessed: 12/8/2016.

[Mov14] Mova contour. `http://www.mova.com/` & `http://www.fordhamiplj.org/2016/03/08/the-curious-case-of-mova-technology//`, 2014. Accessed: 9/2014 & 7/8/2016.

[MR76] Albert Mehrabian and JA Russell. The three dimensions of emotional reaction. *Psychology Today*, 10(3):57–61, 1976.

[MR06] Meinard Müller and Tido Röder. Motion templates for automatic classification and retrieval of motion capture data. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, pages 137–146, Aire-la-Ville, Switzerland, Switzerland, 2006.

[Muk11] Tomohiko Mukai. Motion rings for interactive gait synthesis. In *Symposium on Interactive 3D Graphics and Games*, I3D '11, pages 125–132, New York, NY, USA, 2011. ACM.

[MVCP10] G. McKeown, M. F. Valstar, R. Cowie, and M. Pantic. The semaine corpus of emotionally coloured character interactions. In *2010 IEEE International Conference on Multimedia and Expo*, pages 1079–1084, July 2010.

[MZ07] Mikiko Matsunaga and Victor B Zordan. A dynamics-based comparison metric for motion graphs. In *Computer Graphics International (CGI)*, volume 2007, 2007.

[NBPZ05] Elena Not, Koray Balci, Fabio Pianesi, and Massimo Zancanaro. Synthetic characters as multichannel interfaces. In *Proceedings of the 7th International Conference on Multimodal Interfaces*, ICMI '05, pages 200–207, New York, NY, USA, 2005. ACM.

[NCLC10] William W. L. Ng, Clifford S. T. Choy, Daniel P. K. Lun, and Lap-Pui Chau. Synchronized partial-body motion graphs. In *ACM SIGGRAPH ASIA 2010 Sketches*, SA '10, pages 28:1–28:2, New York, NY, USA, 2010. ACM.

[NHP09] Radosław Niewiadomski, Sylwia Hyniewska, and Catherine Pelachaud. *Modeling Emotional Expressions as Sequences of Behaviors*, pages 316–322. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[NN01] Jun-yong Noh and Ulrich Neumann. Expression cloning. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 277–288, New York, NY, USA, 2001. ACM.

[Nol06] Patricia Noller. *Nonverbal Communication in Close Relationships*, pages 403–421. SAGE Publications, Inc., 2006.

[OBP+12] Veronica Orvalho, Pedro Bastos, Frederic Parke, Bruno Oliveira, and Xenxo Alvarez. A facial rigging survey. In *in Proc. of the 33rd Annual Conference of the European Association for Computer Graphics - Eurographics*, pages 10–32. ACM, 2012.

[OCC88] Andrew Ortony, Gerald L Clore, and Allan Collins. *The Cognitive Structure of Emotions*, volume 18. Cambridge University Press, 1988.

[OHS+05] Alice J. O'Toole, Joshua Harms, Sarah L. Snow, Dawn R. Hurst, Matthew R. Pappas, Janet H. Ayyad, and Herve Abdi. A video database of moving faces and people. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(5):812–816, May 2005. `http://www.utdallas.edu/bbs/facelab/database/`.

[ONP14] Magalie Ochs, Radoslaw Niewiadomski, and Catherine Pelachaud. *Facial Expressions of Emotions for Virtual Characters*. Oxford University Press, 2014.

[ONPS05] Magalie Ochs, Radosław Niewiadomski, Catherine Pelachaud, and David Sadek. *Intelligent Expressions of Emotions*, pages 707–714. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[Opt16] Optitrack expression. `http://optitrack.com/products/expression/`, 2016. Accessed: 7/8/2016.

[Orf96] Sophocles Orfanidis. *Introduction to Signal Processing*. Englewood Cliffs, NJ, Prentice Hall, 1996.

[Osh14] Masaki Oshita. Interactive motion synthesis with optimal blending. *Comput. Animat. Virtual Worlds*, 25(3-4):313–321, May 2014.

[Osi10] Jason Osipa. *Stop Staring: Facial Modeling and Animation Done Right*. John Wiley & Sons, 2010.

[OZS08] VerÃ^{3}nica Costa Orvalho, Ernesto Zacur, and Antonio Susin. Transferring the rig and animations from a character to different face models. *Computer Graphics Forum*, 27(8):1997–2012, 2008.

[Par72] Frederick I. Parke. Computer generated animation of faces. In *Proceedings of the ACM annual conference on - ACM'72*, page 451, New York, New York, USA, August 1972. ACM Press.

[Par74] Frederic Ira Parke. *A Parametric Model for Human Faces.* PhD thesis, The University of Utah, 1974.

[Par82] F. I. Parke. Parameterized models for facial animation. *IEEE Computer Graphics and Applications*, 2(9):61–68, Nov 1982.

[Par02] Aldo Paradiso. An algebra for combining mpeg-4 compliant facial animations. In *Proc. Int. Workshop Lifelike Animated Agents: Tools, Affective Functions, and Applications*, 2002.

[Par07] Rick Parent. *Computer Animation: Algorithms and Techniques*. Morgan Kaufmann, 2nd edition, 2007.

[PB81] Stephen M. Platt and Norman I. Badler. Animating facial expressions. *SIGGRAPH Comput. Graph.*, 15(3):245–252, August 1981.

[PB03] Catherine Pelachaud and Massimo Bilvi. *Computational Model of Believable Conversational Agents*, pages 300–317. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[PBS91] Catherine Pelachaud, Norman I. Badler, and Mark Steedman. *Linguistic Issues in Facial Animation*, pages 15–30. Springer Japan, Tokyo, 1991.

[Pel09] Catherine Pelachaud. Modelling multimodal expression of emotion in a virtual agent. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 364(1535):3539–3548, 2009.

[Pep] Can drake save next-gen?: Interview to josh scherr and ricky cambier.

[Per85] Ken Perlin. An image synthesizer. *SIGGRAPH Comput. Graph.*, 19(3):287–296, July 1985.

[Per97] Ken Perlin. Layered compositing of facial expression. In *ACM SIGGRAPH 97 Visual Proceedings: The Art and Interdisciplinary Programs of SIGGRAPH '97*, SIGGRAPH '97, pages 226–227, New York, NY, USA, 1997. ACM.

[PF03] Igor S. Pandzic and Robert Forchheimer, editors. *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. John Wiley & Sons, Inc., New York, NY, USA, 2003.

[PG96] Ken Perlin and Athomas Goldberg. Improv: A system for scripting interactive actors in virtual worlds. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 205–216, New York, NY, USA, 1996. ACM.

[PP10] T. Pejsa and I.S. Pandzic. State of the Art in Example-Based Motion Synthesis for Virtual Characters in Interactive Applications. *Computer Graphics Forum*, 2010.

[PSS99] Frédéric Pighin, Rick Szeliski, and David H. Salesin. Resynthesizing facial animation through 3d model-based tracking. In *Seventh International Conference on Computer Vision (ICCV'99)*, pages 143–150, Kerkyra, Greece, September 1999. IEEE Computer Society.

[PSS02] Sang Il Park, Hyun Joon Shin, and Sung Yong Shin. On-line locomotion generation based on motion blending. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, pages 105–111, New York, NY, USA, 2002. ACM.

[PVRM05] M. Pantic, M. F. Valstar, R. Rademaker, and L. Maat. Web-based database for facial expression analysis. In *Proceedings of IEEE Int'l Conf. Multimedia and Expo (ICME'05)*, pages 317–321, Amsterdam, The Netherlands, July 2005.

[PW08] Frederic I. Parke and Keith Waters. *Computer Facial Animation*. A K Peters, 2008.

[QCM10] Rossana B. Queiroz, Marcelo Cohen, and Soraia R. Musse. An extensible framework for interactive facial animation with facial expressions, lip synchronization and eye behavior. *Comput. Entertain.*, 7(4):58:1–58:20, January 2010.

[RAB+14] Kerstin Ruhland, Sean Andrist, Jeremy B. Badler, Christopher E. Peters, Norman I. Badler, Michael Gleicher, Bilge Mutlu, and Rachel McDonnell. Look me in the eyes: A survey of eye and gaze animation for virtual agents and artificial systems. In Sylvain Lefebvre and Michela Spagnuolo, editors, *Eurographics (State of the Art Reports)*, pages 69–91. Eurographics Association, 2014.

[Ras16] Jakob Rasmussen. Are behavior trees a thing of the past? `http://www.gamasutra.com/blogs/JakobRasmussen/20160427/271188/Are_Behavior_Trees_a_Thing_of_the_Past.php`, April 2016. Accessed: 15/4/2017.

[RHKK11] Taehyun Rhee, Youngkyoo Hwang, James Dokyoon Kim, and Changyeong Kim. Real-time facial animation from live video tracking. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, pages 215–224, New York, NY, USA, 2011. ACM.

[RNTH03] ZsÃ3fia Ruttkay, Han Noot, and Paul Ten Hagen. Emotion disc and emotion squares: Tools to explore the facial expression space. *Computer Graphics Forum*, 22(1):49–53, 2003.

[ROL90] William T. Reeves, Eben F. Ostby, and Samuel J. Leffler. The menv modelling and animation environment. *The Journal of Visualization and Computer Animation*, 1(1):33–40, 1990.

[RP07] Paul S. A. Reitsma and Nancy S. Pollard. Evaluating motion graphs for character animation. *ACM Trans. Graph.*, 26(4), October 2007.

[RRÉM+06] Sylvain Roy, Cynthia Roy, Catherine Éthier-Majcher, Isabelle Fortin, Pascal Belin, and Frédéric Gosselin. Stoic: A database of dynamic and static faces expressing highly recognizable emotions. Technical report, Université de Montréal, Département de Psychologie, 2006. Accessed: 15/4/2017.

[RRG95] Anand S. Rao, Anand S Rao, and Michael P Georgeff. BDI Agents: From Theory to Practice. *In Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95*, pages 312 – 319, 1995.

[RSF06] James M. Tyler Robert S. Feldman. *Factoring in Age: Nonverbal Communication Across the Life Span*, pages 181–201. SAGE Publications, Inc., 2006.

[RSSL13] F. Ringeval, A. Sonderegger, J. Sauer, and D. Lalanne. Introducing the recola multimodal corpus of remote collaborative and affective interactions. In *2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, pages 1–8, April 2013.

[RTR+11] Etienne B. Roesch, Lucas Tamarit, Lionel Reveret, Didier Grandjean, David Sander, and Klaus R. Scherer. Facsgen: A tool to synthesize emotional facial expressions through systematic manipulation of facial action units. *Journal of Nonverbal Behavior*, 35(1):1–16, 2011.

[Rus91] J A Russell. Culture and the categorization of emotions. *Psychological Bulletin*, 110(3):426–450, 1991.

[RZS10] Cheng Ren, Liming Zhao, and Alla Safonova. Human motion synthesis with optimization-based graphs. *Computer Graphics Forum*, 29(2):545–554, 2010.

[SACR06] Karen L Schmidt, Zara Ambadar, Jeffrey F Cohn, and L Ian Reed. Movement differences between deliberate and spontaneous facial expressions: Zygomaticus major action in smiling. *Journal of Nonverbal Behavior*, 30(1):37–52, 2006.

[SAHM08] Joshua M Susskind, Adam K Anderson, Geoffrey E Hinton, and Javier R Movellan. *Generating facial expressions with deep belief nets*. INTECH Open Access Publisher, 2008.

[SBP97] M. D. Sadek, P. Bretier, and F. Panaget. Artimis: Natural dialogue meets rational agency. In *Proceedings of the Fifteenth International Joint Conference on Artifical Intelligence - Volume 2*, IJCAI'97, pages 1030–1035, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[SBR+14] Mark Sagar, David Bullivant, Paul Robertson, Oleg Efimov, Khurram Jawed, Ratheesh Kalarot, and Tim Wu. A neurobehavioural framework for

autonomous animation of virtual human faces. In *SIGGRAPH Asia 2014 Autonomous Virtual Humans and Social Robot for Telepresence*, SA '14, pages 2:1–2:10. ACM, 2014.

[Sid] SideFX. Houdini 15.5. `https://www.sidefx.com/tutorials/masterclass-crowds-in-houdini-155/`.

[Slo15] Robin James Stuart Sloan. *Virtual Character Design for Games and Interactive Media*. A K Peters/CRC Press, 2015.

[SLS+12] Yeongho Seol, J.P. Lewis, Jaewoo Seo, Byungkuk Choi, Ken Anjyo, and Junyong Noh. Spacetime expression cloning for blendshapes. *ACM Trans. Graph.*, 31(2):14:1–14:12, April 2012.

[SMMH12] I. Sneddon, M. McRorie, G. McKeown, and J. Hanratty. The belfast induced natural emotion database. *IEEE Transactions on Affective Computing*, 3(1):32–41, Jan 2012.

[SNF05] Eftychios Sifakis, Igor Neverov, and Ronald Fedkiw. Automatic determination of facial muscle activations from sparse motion capture marker data. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 417–425, New York, NY, USA, 2005. ACM.

[SO06] Hyun Joon Shin and Hyun Seok Oh. 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, 2006. Eurographics Association.

[SOC16] José Serra, Verónica Orvalho, and Darren Cosker. Behavioural facial animation using motion graphs and mind maps. In *Proceedings of the 9th International Conference on Motion in Games*, MIG '16, pages 161–166, New York, NY, USA, 2016. ACM.

[SP04] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, 23(3):399–405, August 2004.

[SSB09] Nicolas Stoiber, Renaud Seguier, and Gaspard Breton. Automatic design of a control interface for a synthetic face. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pages 207–216, New York, NY, USA, 2009. ACM.

[SSRMF06] Eftychios Sifakis, Andrew Selle, Avram Robinson-Mosher, and Ronald Fedkiw. Simulating speech with a physics-based facial muscle model. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, pages 261–270, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[SWTC14] Fuhao Shi, Hsiang-Tao Wu, Xin Tong, and Jinxiang Chai. Automatic acquisition of high-fidelity facial performances using monocular videos. *ACM Trans. Graph.*, 33(6):222:1–222:13, November 2014.

[TBS+98] J. Tromp, A. Bullock, A. Steed, A. Sadagic, M. Slater, and E. Frecon. Small group behaviour experiments in the coven project. *IEEE Computer Graphics and Applications*, 18(6):53–63, Nov 1998.

[THR11] Graham W. Taylor, Geoffrey E. Hinton, and Sam T. Roweis. Two distributed-state models for generating high-dimensional time series. *J. Mach. Learn. Res.*, 12:1025–1068, July 2011.

[TLJ07] Y. Tong, W. Liao, and Q. Ji. Rpi - isl facial expression databases. `https://www.ecse.rpi.edu/homepages/cvrl/database/ISL_Facial_Expression.htm`, 2007. Accessed: 15/4/2017.

[TLP07] Adrien Treuille, Yongjoon Lee, and Zoran Popović. Near-optimal character animation with continuous control. *ACM Trans. Graph.*, 26(3), July 2007.

[TMTM12] Sarah L. Taylor, Moshe Mahler, Barry-John Theobald, and Iain Matthews. Dynamic units of visual speech. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 275–284, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association.

[TW90] Demetri Terzopoulos and Keith Waters. Physically-based facial modelling, analysis, and animation. *The Journal of Visualization and Computer Animation*, 1(2):73–80, 1990.

[UK12] Norimichi Ukita and Takeo Kanade. Gaussian process motion graph models for smooth transitions among multiple actions. *Comput. Vis. Image Underst.*, 116(4):500–509, April 2012.

[VDSHFD11] Job Van Der Schalk, Skyler T Hawk, Agneta H Fischer, and Bertjan Doosje. Moving faces, looking places: validation of the amsterdam dynamic facial expression set (adfes). *Emotion*, 11(4):907, 2011. `http://aice.uva.nl/research-tools/research-tools.html`.

[VGS+06] V. Vinayagamoorthy, M. Gillies, A. Steed, E. Tanguy, X. Pan, C. Loscos, and M. Slater. Building Expression into Virtual Characters. In *in Proc. of the 27th Annual Conference of the European Association for Computer Graphics - Eurographics*, pages 21–61. ACM, 2006.

[Vic16] Vicon cara. `https://www.vicon.com/products/camera-systems/cara`, 2016. Accessed: 7/8/2016.

[VM06] Adam Jaworski Valerie Manusov. *Casting Nonverbal Behavior in the Media: Representations and Responses*, pages 237–257. SAGE Publications, Inc., 2006.

[VP10] M. F. Valstar and M. Pantic. Induced disgust, happiness and surprise: an addition to the mmi facial expression database. In *Proceedings of Int'l Conf. Language Resources and Evaluation, Workshop on EMOTION*, pages 65–70, Malta, May 2010.

[VWB+12] Levi Valgaerts, Chenglei Wu, Andrés Bruhn, Hans-Peter Seidel, and Christian Theobalt. Lightweight binocular facial performance capture under uncontrolled lighting. *ACM Trans. Graph.*, 31(6):187:1–187:11, November 2012.

[VWVBE+10] H. Van Welbergen, B. J. H. Van Basten, A. Egges, Zs. M. Ruttkay, and M. H. Overmars. Real time animation of virtual humans: A trade-off between naturalness and control. *Computer Graphics Forum*, 29(8):2530–2554, 2010.

[vWZR09] Herwin van Welbergen, Job Zwiers, and ZsÃ3fia M. Ruttkay. Real-time animation using a mix of physical simulation and kinematics. *Journal of Graphics, GPU, and Game Tools*, 14(4):1–21, 2009.

[W3Ca] W3C. Ontology web language - owl. `https://www.w3.org/TR/owl-features/`.

[W3Cb] W3C. Synchronized multimedia integration language - smil. `https://www.w3.org/AudioVideo/`.

[Wat87] Keith Waters. A muscle model for animation three-dimensional facial expression. *SIGGRAPH Comput. Graph.*, 21(4):17–24, August 1987.

[WB08] Jing Wang and Bobby Bodenheimer. Synthesis and evaluation of linear motion transitions. *ACM Trans. Graph.*, 27(1):1:1–1:15, March 2008.

[WBLP11] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. Realtime performance-based facial animation. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 77:1–77:10, New York, NY, USA, 2011. ACM.

[WFH08] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, Feb 2008.

[Whi] Cynthia M Whissell. A dictionary of affect in language. *Emotion: Theory, Research and Experience Vol 4. The Measurement of Emotions*.

[WLL+10] S. Wang, Z. Liu, S. Lv, Y. Lv, G. Wu, P. Peng, F. Chen, and X. Wang. A natural visible and infrared facial expression database for expression recognition and emotion inference. *IEEE Transactions on Multimedia*, 12(7):682–691, Nov 2010.

[WLP+14] Pengjie Wang, Rynson W.H. Lau, Zhigeng Pan, Jiang Wang, and Haiyu Song. An eigen-based motion retrieval method for real-time animation. *Computers & Graphics*, 38:255 – 267, 2014.

[WLVGP09] Thibaut Weise, Hao Li, Luc Van Gool, and Mark Pauly. Face/off: Live facial puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 7–16, New York, NY, USA, 2009. ACM.

[WLW+13] S. Wang, Z. Liu, Z. Wang, G. Wu, P. Shen, S. He, and X. Wang. Analyses of a multimodal spontaneous facial expression database. *IEEE Transactions on Affective Computing*, 4(1):34–46, Jan 2013.

[WN15] Yingying Wang and Michael Neff. Deep signatures for indexing and retrieval in large motion databases. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, MIG '15, pages 37–45, New York, NY, USA, 2015. ACM.

[WTP88] Jerry S Wiggins, Paul Trapnell, and Norman Phillips. Psychometric and geometric characteristics of the revised interpersonal adjective scales (ias-r). *Multivariate Behavioral Research*, 23(4):517–530, 1988.

[XCLT14] Feng Xu, Jinxiang Chai, Yilong Liu, and Xin Tong. Controllable high-fidelity facial performance transfer. *ACM Trans. Graph.*, 33(4):42:1–42:11, July 2014.

[XMLD07] Yu-Li Xue, Xia Mao, Zheng Li, and Wei-He Diao. *Modeling of Layered Fuzzy Facial Expression Generation*, pages 243–252. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[YCS+08] L. Yin, X. Chen, Y. Sun, T. Worm, and M. Reale. A high-resolution 3d dynamic facial expression database. In *2008 8th IEEE International Conference on Automatic Face Gesture Recognition*, pages 1–6, Sept 2008. `http://www.cs.binghamton.edu/~lijun/Research/3DFE/3DFE_Analysis.html`.

[YWL+13] Wen-Jing Yan, Qi Wu, Jing Liang, Yu-Hsin Chen, and Xiaolan Fu. How fast are the leaked facial expressions: The duration of micro-expressions. *Journal of Nonverbal Behavior*, 37(4):217 – 230, 2013.

[YZW+13] Mao Ye, Qing Zhang, Liang Wang, Jiejie Zhu, Ruigang Yang, and Juergen Gall. *A Survey on Human Motion Analysis from Depth Data*, pages 149–187. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[ZDA08] Veronica Zammitto, Steve DiPaola, and Ali Arya. A methodology for incorporating personality modeling in believable game characters. In *Proceedings of the 4th International Conference on Games Research and Development - CyberGames*, pages 24–31. China: Tsingua University Press, 2008.

[ZH99] Victor B. Zordan and Jessica K. Hodgins. *Tracking and Modifying Upper-body Human Motion Data with Dynamic Simulation*, pages 13–22. Springer Vienna, Vienna, 1999.

[Zha14] Bp4d-spontaneous: a high-resolution spontaneous 3d dynamic facial expression database. *Image and Vision Computing*, 32(10):692–706, 2014. `http://www.cs.binghamton.edu/~lijun/Research/3DFE/3DFE_Analysis.html`.

[ZMCF05] Victor Brian Zordan, Anna Majkowska, Bill Chiu, and Matthew Fast. Dynamic response for motion capture animation. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 697–701, New York, NY, USA, 2005. ACM.

[ZNKS09] Liming Zhao, Aline Normoyle, Sanjeev Khanna, and Alla Safonova. Automatic construction of a minimum size motion graph. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 27–35, New York, NY, USA, 2009. ACM.

[ZS09] Liming Zhao and Alla Safonova. Achieving good connectivity in motion graphs. *Graph. Models*, 71(4):139–152, July 2009.

[ZSCS04] Li Zhang, Noah Snavely, Brian Curless, and Steven M. Seitz. Spacetime faces: High resolution capture for modeling and animation. *ACM Trans. Graph.*, 23(3):548–558, August 2004.