

Automated Processes and Intelligent Tools in CG Media Production

Marco Romeo

TESI DOCTORAL UPF / 2016

DIRECTOR DE LA TESI

Prof. Dr. Josep Blat Gimeno

DEPARTMENT OF INFORMATION AND COMMUNICATION
TECHNOLOGIES



A Lorena.

Acknowledgements

I would like to thank Josep Blat and Alun Evans for their help and guidance.

To Santi Fort and Marcelo Dematei who believed in me and expanded my understanding of media.

Many thanks to all the companies I worked with and the incredible teams I had the luck to join. Special mentions are Javier Agenjo, Joan Abadia, Helge Mathee, Vincenzo Leombruno, Lluís Casals, Bor Arroyo, Hannes Ricklefs, Jordi Cardus, Javier Mansilla and Jared Auty; thanks mates.

I would like to thank all my friends who have always been extremely supportive and helped me in the hard times when I was not able to join the party.

Many thanks to Lydia who helped me to manage all the bureaucracy, even after leaving Barcelona. A big chunk of this happened because of her.

Finally, I would like to thank Lorena and my families in Bari and Durango. You all understood how much help I needed to go through this and were with me all the time.

Abstract

Modern media production heavily relies on computer generated content, such as 3D animations and digital visual effects. Those complex assets populate videogames, films, television, mobile devices and the Internet, but their creation is still a complex task requiring a lot of human intervention, and with non standard complex workflows, which are prone to errors, and demand more production effort.

The thesis contributes in two main areas of computer generated content: the oldest research is geared towards the automation of rigging, of emotional facial expression and emotional movements; and towards semi-automated generation of clips; in the most recent research, based on extensive work with the industry, the thesis defines a generic computer generated media production workflow, including production processes and a sample pipeline, and starting from this workflow, approaches to automate the most critical steps are shown and discussed with both scientific and industry eyes. In summary, this thesis contributes with a series of algorithms and tools that are “aware” of the pipeline, and assist the user in the production process, thus: *intelligent tools*.

Resum

La producció moderna dels mitjans descansa de forma important sobre continguts generats per ordinador, tals com animacions 3D, i efectes visuals digitals. Aquests actius complexos poblen videojocs, pel·lícules, televisió, dispositius mòbils i internet, però crear-los encara és una feina complexa, que requereix molta intervenció humana, té un flux de treball complex i no estàndard, procliu als errors, i demana un esforç important.

Aquesta tesi contribueix en dues àrees de contingut generat per ordinador: la investigació inicial es dirigeix a l'automatització del "rigging", de l'expressió facial emocional i de moviments expressius, i de la generació semiautomàtica de "clips"; a la investigació més recent, basada en un treball extens amb la indústria, la tesi defineix un flux genèric de producció de mitjans generats per ordinador, inclosos processos de producció i un model de "pipeline", i, a partir d'aquest flux, es mostren i es discuteixen unes aproximacions a l'automatització dels passos més crítics des d'una perspectiva tant industrial com acadèmica. En resum, la tesi contribueix amb un conjunt d'algorismes i eines "familiaritzats" amb el "pipeline", que ajuden l'usuari en el procés de producció, és a dir, *eines intel·ligents*.

Resumen

La producción moderna de los medios descansa de forma importante en contenidos generados por ordenador, tales como animaciones 3D, y efectos visuales digitales. Estos activos complejos pueblan los videojuegos, películas, televisión, dispositivos móviles e internet, pero crearlos es aún una faena compleja, que demanda mucha intervención humana, implica un flujo de trabajo complejo no estándar, proclive a errores, y exige un esfuerzo importante.

Esta tesis contribuye en dos áreas de contenido generado mediante ordenador: la investigación inicial se dirige a la automatización del “rigging”, de la expresión facial emocional i de movimientos expresivos, i de la generación semiautomática de “clips”; en la investigación más reciente, basada en un trabajo extenso en la industria, la tesis define un flujo genérico de producción de medios generados mediante ordenador, incluyendo procesos de producción y un modelo de “pipeline”, y, a partir de este flujo, se muestran y discuten unas aproximaciones a la automatización de los pasos más críticos desde una perspectiva tanto industrial como académica. En resumen, la tesis contribuye con un conjunto de algoritmos y herramientas “familiarizados” con el “pipeline”, que ayudan al usuario en el proceso de producción, es decir, unas herramientas inteligentes.

Contents

Abstract.....	vii
Resum.....	viii
Resumen.....	ix
Contents.....	x
List of figures	xv
List of tables	xxvi
1. INTRODUCTION	1
1.1. Production and asset management for efficient pipeline and workflows in CGI media: where is my data and where am I?.....	2
1.2. Full production tracking: intelligent pixels, rendering and management	4
1.3. Creative automation: make it move	6
1.4. Fully automated animation production: the quest for the magic button!	9
1.5. Summary	11
1.6. References.....	12
2. PRODUCTION AND ASSET MANAGEMENT FOR EFFICIENT PIPELINE AND WORKFLOWS IN CGI MEDIA.....	17
2.1. Existing techniques, technologies and tools	19
2.2. Abstract production workflow formulation	26
2.3. Detailed CG workflow description	27
a) The pre-production stage.....	29
b) The production stage	33

c)	The post-production stage	48
d)	Integration in a generic film production workflow	50
2.4.	Specialized production workflows.....	51
2.5.	Data flow strategies	53
2.6.	Task management	53
2.7.	Tying all together.....	55
2.8.	Conclusions.....	57
2.9.	References.....	58
3.	FULL PRODUCTION TRACKING: INTELLIGENT PIXELS, RENDERING AND MANAGEMENT	61
3.1.	Intelligent rendering of dailies: automation, layering and reuse of rendered assets.....	64
a)	Introduction	64
b)	Design overview.....	65
c)	Conclusions and future work	66
d)	References	67
3.2.	Full tracking dailies	68
a)	RVGun	68
b)	Evaluation.....	70
3.3.	The digital production pinboard	73
a)	Design.....	74
b)	Evaluation.....	77
3.4.	Render-embedded production tracking.....	78
a)	Method	79
3.5.	Conclusions.....	82
3.6.	References.....	83
4.	CREATIVE AUTOMATION: MAKE IT MOVE	85
4.1.	A reusable model for emotional biped walk-cycle animation with implicit retargeting.....	88

a)	Introduction	89
b)	Related work	91
c)	Overview of approach	92
d)	Conceptual model.....	93
e)	Implicit retargeting of data.....	97
f)	Discussion and future work	100
g)	References	101
4.2.	The MASKLE: automatic weighting for facial animation	102
a)	Introduction	103
b)	Related work	104
c)	Maskle overview	105
d)	Automatic fitting	107
e)	Weight transfer.....	111
f)	Evaluation	112
g)	Discussion and future work.....	114
h)	References	117
4.3.	A synthetic model for automatic and real-time facial expression animation	119
a)	Introduction and background	119
b)	Related work	122
c)	Contribution	125
d)	A synthetic model for expression representation	125
e)	Muscle groups of the model	130
f)	Interpolation formula	131
g)	Evaluation.....	132
h)	Conclusions and future work	137
i)	References.....	139
j)	An emotional interface for the synthetic model.....	141
k)	Evaluation of the interface	146

4.4.	Industrial evaluation of the Maskle (4.2) and the facial expression animation system (4.3).....	148
a)	Industrial evaluation of the Maskle.....	148
b)	Industrial evaluation of the facial expression animation system.....	150
4.5.	References.....	152
5.	FULLY AUTOMATED ANIMATION PRODUCTION: THE QUEST FOR THE MAGIC BUTTON!.....	155
5.1.	Assisted animated production creation and programme generation.....	159
a)	Introduction.....	159
b)	Related work.....	161
c)	Workflow overview.....	162
d)	The programme editor.....	167
e)	Output.....	172
f)	Use cases.....	173
g)	Conclusions and future work.....	178
h)	References.....	179
i)	Integrating emotional work.....	180
5.2.	Domain specific sign language animation for virtual characters.....	181
a)	Introduction.....	181
b)	Related work.....	183
c)	Methodology.....	184
d)	Borja: sports reporter.....	190
e)	Results and conclusions.....	195
f)	References.....	197
5.3.	Measuring the facial expressivity impact in virtual characters for sign language communication.....	198
a)	Introduction.....	199
b)	Facial expressions in question types.....	199

c)	Experiment	200
d)	Data processing and results	204
e)	Conclusions	205
f)	References	206
g)	Evaluation design for the international sign system virtual signer	206
5.4.	References	207
6.	CONCLUSIONS	209
	BIBLIOGRAPHY	217

List of figures

Fig. 2.1.1. A graph representing a possible VFX pipeline from the book Production Pipeline Fundamentals for Film and Games [Dunlop, 2014], pp. 28.....	20
Fig. 2.1.2. Shotgun enables users and production coordinators to review tasks by user.....	21
Fig. 2.1.3. In Shotgun the different stages of production are accessible through a excel-like data sheet. Each stage is represented by a column, while shots are described in rows. This information can be edited.....	22
Fig. 2.1.4. Tactic provides an advanced UI to build production workflows and pipelines as dependency graphs.....	23
Fig. 2.1.5. The relationship across the tables of the database can be specified through a dependency graph in Tactic.....	23
Fig. 2.1.6. For each shot displayed as a row, Tactic shows the different production stages in editable columns.....	24
Fig. 2.3.1. Diagram showing the overall processes involved in modern CGI media production. In both feature animation and VFX production, there are three main stages, pre-production, production and post-production.....	28
Fig. 2.3.2. The figure shows the sub-stages involved in the pre-production stage. Through Assets Definition all the required assets for the production are defined. Then, all of the following sub-stages produce data that is then used in the following production and post-production stages to guide the artist work.....	28

Fig. 2.3.3. Art Design for a character from the Noa&Max series. (a) orthographic view; (b) model sheet to describe facial poses; (c) model sheet for body poses; (d) a drawing showing the character in context.....	30
Fig. 2.3.4. Design for two elements from the Noa&Max series. (a) design of a backpack; (b) design of the elements of a sofa for an environment (sofa, pillows, blanket).....	31
Fig. 2.3.5. Color script for the film The Incredibles (Brad Bird, 2004). The complete film is depicted in different panels showing the main colour scheme for each sequence.....	31
Fig. 2.3.6. Storyboard for one shot of the short-film No Pets Allowed. The panel describes a change of the camera lens: a zoom-out (in red).....	32
Fig. 2.3.7. Diagram of the various processes involved in the production stage. Some processes are required (e.g. rigging) and others are optional (e.g. techanim).....	34
Fig. 2.3.8. The schematic representation of the data flow and logic involved in the Blocking Modeling process described in the text. At any iteration of the process, the result can be delivered as a new version; thus, the model is evaluated prior to its storage and registration in the database. Errors can be fixed or delivery can still be approved with issues.....	36
Fig. 2.3.9. The modeling refinement process starts from a blocking model and aims at adding more detail on the geometry and produce other necessary data as the UV maps used for texture mapping and the target geometry to be used for blend shapes and pose space deformation.....	37

Fig. 2.3.10. The layout rig is the first (and required) controllable version of a geometry.....	39
Fig. 2.3.11. The simple rigging process aims at creating a skeletal structure to deform the geometry. The skeletal deformation is controlled by setting skin weights on the skinning deformer and by driving the deform skeleton through a control skeleton that implements features such as stretch and squash, bending and twisting. Other deformation is also obtained by controlling the target models from the refinement modeling process through a controlling system that can be driven by controllers or by configurations of the deformation skeleton (pose space deformation).....	40
Fig. 2.3.12. The advanced rig is obtained by adding advanced simulation solutions over the simple rig. Simulation can aim at producing realistic behaviours such as volume preservation, wrinkles and muscle dynamics..	41
Fig. 2.3.13. In layout the rigged assets are placed in the three-dimensional scene to match the indications from the script and the director. The camera is also set in the layout process; position, orientation, focal length, aspect ratio, focus distance and shutter angle are key factors that determine the shot and its render..	42
Fig. 2.3.14. In the blocking of animation, the transformations from layout are merged with more detailed keyframes. By merging layers of animation from different processes it is possible to make the workflow non-destructive.....	43
Fig. 2.3.15. The animation refinement modifies the blocking animation by changing the set of keyframes (keyframes can be deleted, added or changed in value and time) and creating animation curves that define the interpolation between two keyframes.....	44

Fig. 2.3.16. Postproduction is made of three consecutive processes, where the second one (relight) is optional.....	49
Fig. 2.3.17. The film production workflow incorporates further aspects that are not directly related to CG media. These (e.g. live action shooting) are the processes that constitute the Film Production stage, while the production and post-production stages related to CG seen in the two previous sections are merged to form the Film Post-Production stage.....	51
Fig. 2.4.1. The workflow seen in section 2.3 can be split into two graphs, depending on the desired final outcome. The <i>Asset creation</i> graph (top) follows the processes required to build an asset to be used in a shot. The <i>Shot creation</i> graph (down) describes the interaction of processes to complete an animated shot.....	51
Fig. 2.4.2. As two graphs can run concurrently in different departments, it is important to remark that they can plug into each other to enable artists to work at different levels of the production, avoiding delays caused by waiting for pipeline completion. In the figure, dashed lines represent interactions of processes across two workflows.....	52
Fig. 2.7.1. Graph showing the integration and interaction between workflows, pipelines, DAM and DPM systems.....	56
Fig. 3.1.1. In this graph, two animated characters (bob and jim) need to be previewed with a certain camera. Version 6 of bob is new and will be rendered, while version 5 of jim was already rendered with the desired camera and will be reused. The render of bob is assetized for further use.....	65

Fig.3.2.1. A screenshot of the RVGun interface (left upper corner). The user is presented with a list of possible choices that are suited to his/her tasks in the production. The available options are appropriate to the user task thanks to the production management system, helping the user reduce time and possible mistakes.....	69
Fig. 3.2.2. The quicktime video is created with a slate that summarizes the main information about the daily.....	70
Fig. 3.2.3. After all the fields are filled and the user submits the work, the asset management system registers the entry. This is now accessible from the production management system (in this case Shotgun).....	70
Fig. 3.3.1. With the Digital Pinboard each panel in a sequence can describe multiple properties. One key feature is the ability to specify the 3D assets that have to appear in the shot and define their typology: character (CHR), prop (PRP) and set (SET).....	75
Fig. 3.3.2. A complete sequence can be described as a set of panels. Each shot is described by one or multiple panels that contain all the relevant information for the pipeline and the artists. This information includes description of the shot, camera description, the list of assets, sounds, actions and dialogues. The panel also shows the status and the feedback for the shot	76
Fig. 3.3.3. The active (left) area of the pinboard is the one that contains the active assets. The other region (lighter gray) can be used to experiment with data, waiting for it to be enabled.....	77

Fig. 3.4.1. A schematic view representing the flow of data to embed tracking and reviewing assets in rendered pixels. The tracking data (assets' identification code) is managed from and into the database thanks to the DAMS and DPMS, then it is fed to the three-dimensional and bi-dimensional authoring softwares so that it can be embedded in the rendering process. This embedded data can be accessed through a tool designed to review frames and query the database to gather the information related to the tracking data of selected pixels. Finally, the information can be reviewed and modified so that the database gets updated with the required changes.....	78
Fig. 3.4.2. A custom tool can be used to define the assets in a shot and the required tasks for its completion. This information is stored in the DAMS and DPMS to be used later in the pipeline.....	80
Fig. 3.4.3. A shot animated in Autodesk Maya. The contents were loaded through a pipeline that generated a pass per each asset, encoding the asset's DAMS code.....	80
Fig. 3.4.4. This figure shows a rendered frame from the shot, open in my prototype review tool. When clicking on a pixel, the tool extracts the code of the asset depicted in the selected pixel and retrieves all the relative production data from the DAMS and DPMS.....	81
Fig. 4.1.1. Diagrammatical overview of the system architecture.....	92
Fig. 4.1.2. The four poses of our model. The circle represents the <i>apparent centre of gravity</i> – the area of the body that is leading the character while walking. The associated curve shows how the spine is curved in each pose.....	94

Fig. 4.1.3. Still images of real motion capture data, grouped by emotion. Of the two shaded characters in each cell of the table, the one on the right is representing data from the female actress while the character on the left is the male actor. In the upper left corner of each cell is the equivalent pose from the conceptual model (see Figure 4.1.2).....	96
Fig. 4.1.4. Screenshots of the results of the plugin developed for Autodesk 3D Studio Max. (a), (b) and (b) show a standard walk animation applied to characters of different dimensions. (d), (e) and (f) show the same characters with the ‘Sad’ emotion blended in. In (c) and (f), the characters hip joint has been scaled artificially to three times the width, yet the retargeting still works perfectly.....	99
Fig. 4.2.1. View of the Maskle system post-initialisation step.....	108
Fig. 4.2.2. Final position of the Maskle for weight transfer.....	110
Fig. 4.2.3. Sample weight profiles for one bone of a manually-weighted face (blue line) and automatically weighted face (red dashed line). Figure (a) shows the results for the Maskle, Figure (b) for the envelope algorithm. The envelope algorithm shows a much greater level of noise and erroneous calculations.....	116
Fig. 4.2.4. Visual differences between Maskle-weighted and envelope-weighted faces, comparing a simple expression created by moving the bones of the rig. (a) shows the unanimated face; (b) shows the expression applied to the Maskle-weighted face; (c) shows the same expression applied to the envelope-weighted face.....	116

Fig. 4.2.5. Example of expressions successfully created using the Maskle on a cartoon character with pronounced features. (a) is the unanimated face, (b) and (c) are expressions created after the Maskle has weighted the face.....	117
Fig. 4.2.6. Examples of expressions created on 3D models animated using the Maskle (images © Merja Nieminen, Crucible Studio / University of Art and Design Helsinki 2008.....	117
Fig. 4.3.1. Sub-cutaneous muscles.....	122
Fig. 4.3.2. Superville's schemes.....	123
Fig. 4.3.3. Plasencia's schemes.....	124
Fig. 4.3.4. Our model's schemes.....	127
Fig. 4.3.5. The points of action and movements proposed by our model.....	131
Fig. 4.3.6. Whissell's activation(vertical axis) - evaluation(horizontal) space.....	134
Fig. 4.3.7. Table of viewer's test results.....	135
Fig. 4.3.8. Viewer's test results dispersion graph.....	135
Fig. 4.3.9. Per-expression results obtained with the viewer test evaluation. The bars in the chart represent ratios of matching emotional expressions recognized. Red colour indicates recognition rate for exact expression, while yellow represents the ratio of recognition as the closest emotion in the whissell wheel.....	136
Fig. 4.3.10. Our defined wheel for Activation (vertical axis) and Evaluation (horizontal).....	142
Fig. 4.3.11. Screenshot of interface as a Maya plug-in....	145
Fig. 4.3.12. Screenshot of a standalone interface controlling a character's face in real-time.....	145
Fig. 4.3.13. Chart showing overview of evaluation results.....	147
Fig. 5.1.1. Programme Editor and real-time preview window.....	160
Fig. 5.1.2. 3D Studio Max direct export to the framework.....	166
Fig. 5.1.3. Basic programme structure.....	169
Fig. 5.1.4. Camera and audio components added to programme description.....	169
Fig. 5.1.5. Addition of character "Bruce" to the scene.....	171

Fig. 5.1.6. Tracks with two characters and several components added to the scene.....	172
Fig. 5.1.7. Sam the virtual weatherman.....	174
Fig. 5.1.8. Sam explaining today's weather.....	175
Fig. 5.1.9. Virtual stock market presenter.....	175
Fig. 5.1.10 Manolo the virtual bar-tender.....	176
Fig. 5.1.11. Programme Generator Server. On the left the user can specify some characteristics, including text to utter, on the right the resulting video is displayed.....	177
Fig. 5.1.12. Screenshots of the emotional interface (section 4.3.10) working in the Programme Editor. The emotional interface is interpolating the shapes described in the synthetic model for facial expression (section 4.3). (a) shows disgust (approximately activation -0.9, evaluation 0.1). (b) shows happiness (approximately activation 0.6, evaluation 0.3).....	180
Fig. 5.2.1. The animation system receives an XML file that contains the description of the clips that has to be concatenated in order to build the complete animation. Depending on whether it's a pre-made phrase, a number or a word, the system retrieves and merges the appropriate animated clips to build the signed contents.....	186
Fig. 5.2.2. Screenshot of a signing expert (prelingually deaf) with virtual character overlay.....	187
Fig. 5.2.3. The flow of information through the different processes that create an animation clip from input.....	189
Fig. 5.2.4. Figure shows how blending can be made between two clips. Left: interpolating from last frame of Clip A to first frame of Clip B; or Right: cross fading from Clip A to Clip B..	190

Fig. 5.2.5. Art design for Borja. Design includes different clothing for different weather conditions. The design shows a sporty style for the character to make it closer to the sailing world, bigger eyes to empathize with audience and big hands to make signs more readable through streaming videos.....	192
Fig. 5.2.6. Scheme of the news generation system.....	193
Fig. 5.2.7. Webpage of the Spanish National Radio showing Borja's videos for the Barcelona World Race.....	194
Fig. 5.2.8. Screenshot of Borja in action.....	195
Fig. 5.2.9. Screenshot of RTVE ranking page showing Borja popularity.....	195
Fig. 5.2.10. Chart showing the 15.8% greater in popularity of Borja's videos over all other Barcelona World Race multimedia. Also shown in the Standard Deviation bar.....	196
Fig. 5.3.1. Two stills of a real person signing two types of questions, "yes/no" (left) and "wh-?" (right).....	200
Fig. 5.3.2. Snapshot of the web interface (based on Flash). The buttons are placed below. "Abierta" and "Si/No", translated to English, mean "Wh-?" and "Yes/No".....	202
Fig. 5.3.3. Two stills of a RS signing "what" in LSC, one with coherent facial expression (left) and another with incoherent facial expression (right).....	203
Fig. 5.3.4. A real signer and an animated character signing.....	203
Fig. 5.3.5. Time ratio averages and standard error for correct distinction of questions in coherent and incoherent cases for RS(a) and VS(b). Distinction correctness of questions in coherent and incoherent cases for RS(c) and VS(d).....	205

Fig. 5.3.6. Time ratio averages' comparison between RS with coherent facial expression, RS with incoherent facial expression, VS with coherent facial expression and VS with incoherent facial expression..... 205

List of tables

Table 3.2.1. Timing details for the operations required to register a daily on a task with Shotgun. The times varied a lot across the users but it is interesting to notice that they spent similar time (104 seconds in average) in “others” actions which were not required.....	71
Table 3.2.2. Mistakes (denoted by the symbol ✕) occurred when using Shotgun to register daily. Tags were almost omitted and the file location is wrong in 2 cases out of five. Having a wrong file location is a big issue in production as supervisors would review the wrong daily, could generate confusion and will finally require to repeat the process.....	71
Table 3.2.3. In RVGun the times spent in the various processes are reduced. Most of the time is spent in generating and uploading the video and the thumbnail, which in this system, is not performed by the user. The time taken to write the tags is very low as the system already provides a set of default tags for the selected task.....	72
Table 3.2.4. All processes produced the expected results (✓) with RVGun.....	72
Table 3.3.1. Table of relevant properties for different types of assets.....	75
Table 4.1.1. The four poses of the model, and examples of associated emotions.....	94
Table 4.2.1. The mean differences between manually weighted faces and automatically weighted faces. A sample of each model is shown in Figures 4.2.4, 4.2.5 and 4.2.6.....	114
Table 4.3.1. The poses proposed, modelled on two different character’s faces.....	128
Table 4.3.2. Some proposed facial expressions.....	129

Table 4.3.3. Set of experimental interpolation values between facial poses.....	130
Table 4.3.4. Comparison of our synthetic model with the animated 3D scans of Wallraven <i>et al.</i> , measured in percentage of viewer recognition. * indicates a mean value of corresponding emotions from our model. The table shows that the expressions obtained with our synthetic model are as believable as the ones obtained with a 3D scan of a real human’s face and expression..	137
Table 4.3.5. Extreme poses of the face, and the Activation-Evaluation values assigned to them. See Table 4.3.4 for interpolation values.....	143
Table 4.3.6. Interpolation values for the poses described in table 4.3.5.....	144
Table 4.4.1. Evaluation results of the use of the MASKLE in a professional production context (extracted from SALERO deliverable <i>D9.6.3: Third Phase Experimental Productions and Evaluation Report</i> , p. 35).....	149
Table 4.4.2. Feedback received from artists using the Synthetic model for facial expression to create the installation Alan01 (Extracted from SALERO deliverable <i>D9.6.3: Third Phase Experimental Productions and Evaluation Report</i> , p. 56).....	151
Table 5.3.1. List of questions used in the experiment (translated to English from Catalan). The questions are divided in two groups, depending on the type of question.....	201

1. INTRODUCTION

This thesis is focused on CG (Computer Graphics) media. More specifically, it presents a series of contributions mainly intended to improve the management and automation of some key processes.

Chapter 2 presents a (novel) formalization of workflows and pipelines in CG media. Chapter 3 presents some improvements in workflows, which are based on the concepts presented in the previous chapter. Chapter 4 presents some solutions to aid and automate the animation and rigging of virtual characters. Chapter 5 focuses on automating the complete production process of short animation TV and web programmes by making use of pre-made clips and templates.

The research contained in chapters 4 and 5 is older than that of the earlier chapters. It was mainly framed in the context of European projects in collaboration with industry. The research presented in chapters 2 and 3 is more recent and draws on my experience in different media companies (the most recent one being the Moving Picture Company, MPC) and productions.

This separation between recent industrial and previous academic research is a direct result of a shift in the understanding of which production aspects are crucial for automation. When the work in the thesis began, there was indication that the most important aspect was to automate some facets of creative tools. The goal was to reduce the time invested in non creative (thus automatable) processes and enable new formats to be rapidly delivered on new media supports, such as the web and mobile devices. The automation of complex processes, as the ones described in chapter 5, unfortunately comes at the expense of creative flexibility, thus the work in chapter 4 was aiming at providing ways to reintroduce this creative flexibility, mainly focusing on virtual characters.

While working with companies and industrial research partners to integrate automated solutions, it became evident that any automated or intelligent process requires an infrastructural foundation that enables the proper storage and management of production data (e.g., human resources, assets and tasks). This way, the focus of interest turned into pipelines, workflows, digital asset management (DAM)

and digital production management (DPM), as these were identified as being at the core of the required foundation. Studying this issue as described in chapter 2, proved that the more advanced the infrastructure is, the harder it becomes to access and manage the data. Even though it also revealed the opportunity for further advances. In fact, in chapter 3 I describe how this management can be performed efficiently, while empowering the tools with all the benefits of a proper infrastructure.

As some of the contents have already been published or are written for submission, they are presented in a way that follows closely the original paper. Thus, related references are included as a subsection. All references for new contents are listed at the end of every chapter. At the end of the document, a list of the complete bibliography is provided.

This thesis is then structured to reproduce a bottom-up perspective on my journey and the order of chapters represents the order of execution that each issue and work should be addressed from an industrial perspective. Next, I outline the contributions in each chapter, and discuss the relevance of each of them in more detail.

1.1. Production and asset management for efficient pipeline and workflows in CGI media: where is my data and where am I?

In chapter 2 I introduce key concepts of the workflows and pipelines for digital media, more especially focused on CG media. The chapter makes a more formal presentation of the concepts than those existing so far in the scarce literature available on the topic. The presentation is geared towards enhancing the efficiency and robustness of the pipeline, and the concepts are the basis for the pipeline innovations I present in chapter 3.

In recent years, international media companies have been all looking at workflows and pipelines as one of their biggest challenges for the future, these words being used without a very precise meaning or conceptualization. That workflows and pipelines are key challenges is currently confirmed by a lot of activities that

demonstrate the increasing interest by the academia and the industry to advance in solving the issues under those concepts. Recently, books appeared which were written by recognised members of the best animation, game and visual effects companies in the media industry [Polson, 2014][Dunlop, 2014]. The academic interest has also grown, with journals now focusing on such type of issues like the Journal of Digital Asset Management [Journal of Digital Media Management, 2015], with special panels as *Global VFX Pipelines* [SIGGRAPH, 2015][Chung, 2011] hosted by SIGGRAPH and technical talks in conferences like SIGGRAPH [Johnson et al., 2014][Meeres-Young, 2010]. On the other hand commercial solutions have proliferated, providing media companies with ways to define and manage their workflows and pipelines.

There is yet little formal work on how to design workflows and pipelines for CGI media production, as shown in the state of the art discussed in chapter 2, which discusses academic, commercial, and the internal media industry solutions, at least from the information available. Furthermore, there is little evidence of the benefits that would come from doing so. The prototypical solution I introduce in chapter 2 defines the concepts supporting workflows and pipelines. It provides an integrated approach that merges the concepts of workflow and pipeline to improve the management and traceability of production. This is not a standard approach in the industry or research works, discussed in the state of the art, and could be published in the future.

Besides being a novel approach, it has significant potential as demonstrated by several different innovative proposals to increase productivity and robustness across several production processes described and discussed in chapter 3. In this sense, chapter 3 demonstrates the first steps in validating the novel approach, and, at the same time it is one of the first evidences that formalizing workflows and pipelines as in our approach leads to benefits.

1.2. Full production tracking: intelligent pixels, rendering and management

On the basis of the approach of chapter 2, chapter 3 introduces and discusses 4 different innovative production systems that take advantage of those engineered pipelines, workflows, production and asset management systems. Because the data is properly registered and managed, it is possible to connect different systems, sharing such data or even build newer systems to show, edit and control the data.

The work described was carried out in different companies and was strictly related to the production needs of the moment. The evaluation of these works is thus directly related to their use in production, the improvement of the overall productivity and facilitating the workflow for production coordinators and artists.

The first contribution *improves the quality of the daily renders and reduces the time required to compute them*, was undertaken at the Moving Picture Company (MPC) and has been recently published [Romeo et al. 2015].

Romeo, M., Auty, J. & Fagnou, D. (2015). Intelligent rendering of dailies: automation, layering and reuse of rendered assets. *Proceedings of the 12th European Conference on Visual Media Production (CVMP '15)*. ACM, New York, article 15.

The system, called *Renderflow*, integrates the company Asset Management System and workflows into one centralized solution. This way, all assets from a shot can be rendered using state of the art renderers (e.g. Mental Ray or RenderMan) and, most importantly, the resulting renders can be assetized for later reuse, which is a key feature of the system. Renderflow uses a compositing technique that merges together different images by sorting their pixels by their distance (or depth) from the camera. This makes it possible to render each asset on a separate layer and compose it at a later stage. This way, each layer gets assetized and can be reused if no changes are made to the assets contained in the layer. If shots with many complex assets are considered (numbers can go up to 200 and above), the changes on one single asset will

result into a new high quality render being produced very rapidly on the basis of the proposed solution.

The second contribution *integrates the Digital Production Management System into a media player*. This way, artists can easily bind videos and images to their tasks or assigned assets, as navigating the DPM when assets and tasks are very large, as it is usually the case, becomes tedious for the artists. On the other hand, the number of tasks and assets also impacts on the amount of dailies (videos or image sequences) created, complicating the browsing and access of the data. The prototype introduced was designed at StudioNest in 2010 to overcome these problems. The system integrates Autodesk Shotgun [Shotgun, 2015], RV [RV, 2015] and the studio pipeline to give artists easy and personalized access to the relevant information and was named *RVGun*. The system attracted a very positive interest of both software companies (Shotgun and Tweak Software) which were recently bought by Autodesk. The validity of the tool is further supported by the *ReviewTool* developed by MPC which was published later [Fagnou et al. 2013], and which is based on the same ideas. This software uses the same approach and extends it further by also providing a pipeline-based editing User Interface (UI), which enables users to see shots in different contexts. Contexts can vary from simple editing cuts to the different stages of the different departments. The UI is a key aspect of the third contribution, discussed next.

The third contribution is a production tool, the *Production Pinboard*, that integrates the ideas of production management, pre-production and digital pinboards. Production management commonly happens through Excel-like tables, and it is hard to browse information rapidly without filtering the data. Unfortunately filtering the data reduces the context. Moreover, data sheets do not present information in a way that is visually meaningful. The Production Pinboard is a new way to access pre-production data. Through the software it is possible to define shots and assets, arrange them in meaningful ways and visually navigate and edit the data without losing context. The work was undertaken during 2014, at Bee and Birds animation studio in Barcelona. It was not extensively tested in production due to the closure of the studio but its development was informally evaluated and driven by the person who is now Layout lead at MPC.

The fourth contribution is *a method that enables Asset Management data to be embedded in rendered images*. Three-dimensional scenes are described by a live hierarchical structure (or scene graph) [SIGGRAPH2013 USD Presentation, 2013, Brutzman & Daly, 2007]; because of this, it is easy to add metadata to tag different parts of the scene in the Digital Asset Management System (DAMS). Unfortunately, images commonly have a flat structure that does not allow to hierarchically cluster pixels, making it impossible to properly add per-asset metadata. As a result, once a frame is rendered, it is impossible to retrieve the asset that was used to render a certain pixel and its data. I propose a method that encodes a link to the DAMS entry of each asset rendered in a pixel of an image. By generating an extra pass that writes the asset code ID in the image, it is then possible to query a pixel to get the ID and retrieve all the asset information from the DAMS and task information from the DPM. I developed this technique in 2014 and I am currently designing an evaluation strategy prior to its publication.

1.3. Creative automation: make it move

Chapter 4 deals with research contributions towards finding solutions to aid and automate the animation and rigging of virtual characters, one of the many processes within the context of CG production. I outline and discuss three contributions next.

The first contribution is related to *automated full body animation*, and focuses specifically on tweaking walk cycle animations. This research is oriented to find ways to automatically modify and apply walk-cycle animations to various bipedal characters of any proportion to represent their emotional state through motion. Our research contributes by creating a model for emotional character walking, and defining a series of proportional variables, whose values are based on motion capture, which can be changed to create different emotional walk cycles. As we avoid values in world coordinates and work solely with proportions, the system implicitly retargets the data to any biped body shape, regardless of the size and structure of the skeleton. The results were presented in

Articulated Motion and Deformable Objects (AMDO) 2010 [Romeo et al., 2010].

Romeo, M., Dematei, M., Bonequi, J., Evans, A. & Blat, J. (2010). A reusable model for emotional biped walk-cycle animation with implicit retargeting. *Proceedings of the 6th international conference on Articulated motion and deformable objects*. Springer-Verlang, Berlin, pp 270-279.

The previous approaches which are most similar to our research, in a sense, are Densley and Willis [Densley & Willis, 1997] and Meredith and Maddock [Meredith & Maddock, 2005]. Our approach combines their philosophies but underpins it with a more formal and focused theoretical background. We tackle the issue of walk-cycle animation inspiring us in traditional character animation techniques, applied within a modern computing and mathematical framework.

The second contribution, the *Maskle*, is an automated facial rigging system whose main purpose is to provide a simple but functional deformation system for the face, making possible to reduce the time spent by artists for the rigging task and ensure that the facial rig is coherent across characters, enabling reuse and automation. The work, published in the *International Conference on Computer Graphics Theory and Applications (GRAPP 2009)* [Evans, 2009], is based on the encoding of deformation parameters (eg. linear skinning weights) on a low resolution face template for future reuse.

Evans, A., Romeo, M., Dematei, M. & Blat, J. (2009). The Maskle: Automatic Weighting for Facial Animation. *International Conference on Computer Graphics Theory and Applications (GRAPP)*.

In fact, thanks to dedicated tools and a fitting algorithm, the template is easy to setup onto diverse faces and the deformation parameters can then be automatically transferred to make them animatable. This idea was gestated as a solution to enable future automation of characters' faces; having a standardized facial rig, makes it possible to define techniques and algorithms for its animation and reuse. The work was evaluated in three experimental

productions within the SALERO European project, showing productivity benefits [Third Phase Experimental Productions and Evaluation Report SALERO 2009], up to 80%, and no tangible difference in qualitative comparison. The system proved well suited to provide a working facial rig for humanoid faces and it was the only available tool of this kind at that moment. Commercial products do not incorporate yet any similar tool/technique and the skinning algorithms have unfortunately not improved either, making the Maskle a still relevant contribution today.

The third contribution is a *Synthetic Model for Facial Expression*. At the time, a lot of the productions of the industrial partners in the research projects undertaken lacked facial expressivity; the main motivations were:

- Characters are difficult to setup to express emotions through the face;
- Animation of the expressivity of the face is hard to automate;
- The complexity of facial rigs to provide characters with expressivity limits the frame-rate of realtime applications and requires higher processing times for off-line productions.

Our approach is based on a simple set of five facial shapes, plus the articulated deformation produced by the jaw bone, which we show that can be combined together to represent a wide range of emotions. These combinations are arranged into a Whissell wheel [Whissell, 1989][Plutchnick & Kellerman, 2015] (alternatively known as bidimensional valence-arousal emotional model) to easily set and interpolate different emotional facial expressions. This solved the barriers for facial emotional animation indicated above.

The quality of the resulting facial expressions was evaluated through an online survey, which showed that most of the obtained emotions were easy to understand across three different countries. The work was also used and evaluated in an experimental production [Tuomola et al., 2009, Alan01, 2010], receiving a mixed response from the artists working on it, finding it useful and easy to

use, but not flexible enough at times. On the other hand, 96% of the spectators visiting the installation who took part in a survey, rated the quality of facial animation as excellent, enlarging the positive results of the online survey mentioned above.

The work was not published at the time, but the new trends in real-time online 3D graphics [Evans et al., 2014] and the renewed interest in modeling emotional expressivity for virtual characters [Hyde et al., 2014][Faita et al., 2015] may set the ground for a publication.

1.4. Fully automated animation production: the quest for the magic button!

Chapter 5 focuses on automating the complete production process of short animation TV and web programs by making use of pre-made clips and templates. This approach was used to generate web content, multimedia messages, and short TV programmes for a period of more than five years.

The automation can be extended from the creation of the contents, to the creation of the rendered result. In the chapter I present two research works that have successfully been used to automatically produce media contents for the web, for multimedia messages and television.

The first contribution is *Automated template-based production*. Automating the complete production process had been seldom addressed in CG production, although more work exists on video editing, as discussed in the state of the art analyzed in chapter 5, the key factors to reduce the production time are the simplification of the three-dimensional assets and animation creation, plus the acceleration of the rendering and editing processes. The solution proposed addresses these issues through a system inspired by videogame technology, i.e. creating real-time 3D graphics based on GPU use. The system can blend animation clips together, apply appropriate transformations on 3D geometries, shade and render them in real-time. The system can also apply audio on rendered videos, which can be processed to automatically synchronize the

facial animation and the audio. These features were all integrated within the GTI (Grup de Tecnologies Interactives, Universitat Pompeu Fabra) framework, a collection of Open Source libraries for different aspects of real-time interactive 3D graphics, which could be scripted. Another aspect to be taken into account is how the animations and other contents have to be selected and combined to provide a narrative context to the produced media. In fact, a lot of time can be spent in defining the narrative of a video and deciding which assets better communicate the intended message. In this case we provided a template-based system that enabled production companies to prepare templates that can be fed with live data to properly choose the contents which are suitable, to generate and display the 3D animations. These templates could be prototyped through an interactive system branded *Programme Editor* that enables non-technical users to access all the features of the GTI framework. The system can preview videos of the contents selected by the user, set the variables that drive the logic behind their automated selection and finally render a video or store a template. The system uses PVML, our own variant of SMIL. This work was published in *Advances in Computer Entertainment (ACE)*, 2009 [Abadia et al., 2009] and is the result of a few years of development, where I contributed in different ways by driving the changes to its overall design and interface to accommodate real production requirements, iteratively testing the system, and championing its adoption in different projects.

Abadia, J., Evans, A., Ganzales, E., Gonzales, S., Soto, D., Fort, S., Romeo, M. & Blat, J. (2009). Assisted Animated Production Creation and Programme Generation. *ACM Advances in Computer Entertainment Technology*. ACM, New York, pp. 207-214.

The second contribution is *Automatic generation of Sign Language*, a specific area which has attracted a lot of research, that of Sign Language (such as the different European projects ViSiCAST, eSIGN, CogViSys, Dicta-Sign). Our research mainly focuses on the automatic generation of sign language videos with virtual characters. As a system, its approach is similar to that of the Programme Editor, using templates. The domain is specific as the system generates signed sport news, based on heterogeneous raw

data collected from multiple sources. This system was created to provide accessible news to the deaf audience of the Barcelona World Race sailing regatta around the world and the resulting videos were video streamed in the Spanish National Radio (RNE) website. There are different levels of evaluation of its results. Two of them are relevant for the system evaluation:

- The system was successful in producing automatically animation clips, being able to run publicly during the event;
- The textual transcript of the automatically generated news was checked by RNE journalists who did not suggest any changes, proving the soundness of the news generator. More importantly, in terms of sign language understandability, and complex animation;
- There was an indirect proof that the sign language was understandable in terms of the popularity of the videos in the web site.

The research was presented in *GRAPP 2014* [Romeo et al, 2014].

Romeo, M., Evans, A., Pacheco, D. & Blat, J. (2014). Specific Sign Language Animation for Virtual Characters. *GRAPP 2014 - Proceedings of the 9th International Conference on Computer Graphics Theory and Applications*. SciTePress, pp. 487-494.

We also propose a more direct, detailed evaluation of the Sign Language which our systems were able to generate, specifically in terms of the understandability related to the facial expressivity, and discuss the limitations and further work.

1.5. Summary

In summary, this thesis presents several contributions to the automation of CG media generation, some of them in specific production aspects, and some in the improvement of workflows and pipelines. The evaluation is based on a mix of lab-based, and

production based strategies, while some of the evaluation in deeper and broader aspects is still ongoing.

1.6. References

3DS Max, computer software. (2015). Available from: <<http://www.autodesk.com/products/3ds-max>>. [13 December 2015].

Abadia, J., Evans, A., Gonzales, E., Gonzales, S., Soto, D., Fort, S., Romeo, M., Blat, J.: Assisted animated production creation and programme generation, *Proceedings of the International Conference on Advances in Computer Entertainment Technology*, 207-214, ACM, 2009.

Alan01. (2010). Available from: <<http://mlab.taik.fi/alanonline/#>>. [13 December 2015].

Brutzman, D. & Daly L. (2007). X3D: Extensible 3D Graphics for Web Authors, Morgan Kaufmann Publishers Inc. San Francisco.

Chung, H. J. (2011). 'Global Visual Effects Pipelines: An Interview with Hannes Ricklefs', *Media Fields Journal*, no. 10.

Densley, D. & Willis, P. (1997). Emotional Posturing: A Method Towards Achieving Emotional Figure Animation. *IEEE Computer Animation 1997*. IEEE Computer Society, Washington DC, pp 8-14.

Dunlop, R. (2014). *Production Pipeline Fundamentals for Film and Games*, Focal Press, New York.

Evans, A., Romeo, M., Dematei, M. & Blat, J. (2009). The Maskle: Automatic Weighting for Facial Animation. *International Conference on Computer Graphics Theory and Applications (GRAPP)*.

Evans, A., Romeo, M., Bahrehmand, A., Agenjo, J. & Blat, J. (2014). 3D graphics on the web: A survey. *Computers and Graphics*. vol. 41, no. 0, pp. 43-61.

Fagnou, D., Cameron C., & Valdez, A. (2013). ReviewTool: a database-driven visual effects editing application. *ACM SIGGRAPH 2013 Talks*, ACM, New York, article 28.

Faita, C., Vanni, F., Lorenzini, C., Carrozzino, M., Tanca, C. & Bergamasco, M. (2015). Perception of Basic Emotions from Facial Expressions of Dynamic Virtual Avatars, *Lecture Notes in Computer Science*. Springer International Publishing, Switzerland, vol. 9254, pp. 409-419.

Hyde, J., Kiesler, S., Hodgins, J. K. & Carter E. J. (2014). Conversing with Children: Cartoon and Video People Elicit Similar Conversational Behaviors. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, pp. 1787-1786.

Johnson, C., Tobiska, J., Tomlinson, J., Van de Bosh, N. & Whaley, W. (2014). A framework for global visual effects production pipelines. *ACM SIGGRAPH 2014 Talks*, ACM, New York, article 57.

Journal of Digital Media Management. (2015). Available from: <<http://www.henrystewartpublications.com/jdmm>>. [13, December, 2015].

Kennaway, J. R., Glauert, J. R. W., Zwitterlood, I. (2007). Providing signed content on the Internet by synthesized animation. *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 14, no. 3, Art. 15, pp 1-29.

Meeres-Young, G., Ricklefs, H. & Tovell, R. (2010). Managing thousands of assets for the prince of persia city of Alamut. *Proceedings of ACM SIGGRAPH 2010 talks*.

Meredith. M. & Maddock, S. (2005). Adapting Motion Capture Data Using Weighted Real-Time Inverse Kinematics. *ACM Computer in Entertainment*, vol. 3, no. 1 pp. 5-5.

Plutchnik R. & Kellerman H. (2015), The Measurement of Emotions, *Emotion: Theory, Research and Experience*, vol. 4, pp. 113-129.

Polson B. (2014). CG pipeline design patterns. *Proceedings of the Fourth Symposium on Digital Production*, pp. 29-29.

Romeo, M., Dematei, M., Bonequi, J., Evans, A. & Blat, J. (2010). A reusable model for emotional biped walk-cycle animation with implicit retargeting. *Proceedings of the 6th international conference on Articulated motion and deformable objects*. Springer-Verlang, Berlin, pp 270-279.

Romeo, M., Evans, A., Pacheco, D. & Blat, J. (2014): Specific Sign Language Animation for Virtual Characters, *GRAPP 2014 - Proceedings of the 9th International Conference on Computer Graphics Theory and Applications*, 487-494, SciTePress.

Romeo, M., Auty, J. & Fagnou, D. (2015). Intelligent rendering of dailies: automation, layering and reuse of rendered assets. *Proceedings of the 12th European Conference on Visual Media Production (CVMP '15)*. ACM, New York, article 15

RV, computer software. (2015). Available from: <<http://www.tweaksoftware.com/products/rv>>. [13 December 2015].

Shotgun, computer software. (2015). Available from: <<http://www.autodesk.com/products/shotgun>>. [13 December 2015].

SIGGRAPH2013 USD Presentation. (2013). Available from: <http://graphics.pixar.com/usd/s2013_presentation.html>. [13 December 2015].

SIGGRAPH 2015, Global VFX Pipelines. (2015). Available from: <<http://s2015.siggraph.org/attendees/birds-feather/sessions/global-vfx-pipelines>>. [13, December, 2015].

Salero: Semantic Audiovisual Entertainment Reusable Objects. (2006). Available from: <<http://www.salero.eu>>. [13 December 2015].

Third Phase Experimental Productions and Evaluation Report
SALERO. (2009). Available from:
</http://salero.eu/en/resources/deliverables.html>. [13 December
2015].

Tuomola, M. L., Korpilahti, T., Pesonen, J., Singh, A., Villa, R.,
Punitha, P., Feng, Y. & Jose, J. M. (2009). Concept, content and the
convict. *Proceedings of the 17th ACM international conference on
Multimedia*, ACM, New York, pp. 1063-1072.

Whissell C. M. (1989). *The Dictionary of Affect in Language*.

2. PRODUCTION AND ASSET MANAGEMENT FOR EFFICIENT PIPELINE AND WORKFLOWS IN CGI MEDIA

In this chapter I introduce key concepts of the workflows and pipelines for digital media, more especially focused on CG (Computer Graphics) media. This chapter makes a more formal presentation of the concepts than those existing so far in the scarce literature, and is geared towards enhancing the efficiency and robustness of the pipeline. The concepts are needed for the pipeline innovations I present in the following chapter.

The complexity of modern media production has grown a lot in many industries, including film, commercial, television and game companies. This complexity directly translates into large numbers of assets to be created and managed through all the production processes.

The creation of assets requires several production stages to happen in different departments and by many collaborating individuals (artists and technicians). At each stage, a varying number of versions may be produced, until one version is considered the suitable candidate. The creation of an asset is thus made of different processes; these may be sequential (one process may be used as the starting point for the next stage) or parallel (two processes are used together to create certain aspects of the asset). Furthermore, the creation of an asset may require the creation of other assets, and consequently assets may be made of other assets (e.g., the characters in a shot or the textures on a character).

The processes have to be designed and combined to efficiently separate tasks across departments and enable them to work in parallel on different aspects of the production. This requires a production workflow to be engineered in conjunction with all the production stakeholders. The production processes can generate a large amount of data that represent the various versions of the desired result and the finalized result. How to properly manage, bundle and make this data accessible is still an open question and there still is a struggle to engineer an efficient pipeline.

During my research I had the opportunity to work and discuss with a few international companies and found that the productivity issues they were facing were mostly referred to a lack of workflow or pipelines. That workflows and pipelines are key challenges is now confirmed by a lot of activities that demonstrate the increasing interest by the academia and the industry to advance in solving these problems. Recently, books appeared which were written by recognized members of the best animation, game and visual effects (VFX) companies in the media industry [Polson, 2014] [Dunlop, 2014]. The academic interest has also grown, with journals now focusing on issues like Digital Asset Management [Journal of Digital Media Management, 2015], special panels as *Global VFX Pipelines* [SIGGRAPH, 2015] [Chung, 2010] hosted by SIGGRAPH and technical talks in conferences like SIGGRAPH [Johnson et al. 2014] [Meeres-Young 2010]. Finally there is a proliferation of commercial solutions to provide companies with ways to define and manage their workflows and pipelines.

While the conversation is currently focusing in understanding their importance, there is yet little formal work on how to design workflows and pipelines for CGI media production. Moreover, the relation between workflow, pipeline, DAM and DPM systems is even less clear in the industry. Because of this, various situations may be currently found in the industry. The vast majority of production companies does not adopt any of these technologies, relying on naming conventions and intricate folder structures to try to keep data organized and searchable. Other companies have some kind of workflow or pipeline, but they still heavily rely on manual management of files and folders. Fewer have already integrated a DPM system to help artists and managers to track production tasks. Finally, there is an extremely limited number of companies that are also using a DAM system but still have not developed any integration across these four technologies. This chapter proposes a formalization and a subsequent integration strategy. Besides supporting the research contributions presented in chapter 3, the content of chapter 2 has been written with the aim of being a valuable source of information for companies willing to move toward a more robust and productive production environment.

In this chapter I describe a prototypical solution that defines the concepts supporting workflows and pipelines. The resulting idea is

one of an integrated approach that merges the concepts of workflow and pipeline to improve the management and traceability of production. This is not a standard approach in the industry or research works and has significant potential as illustrated by several different novel proposals described and discussed in the following chapter. In fact, in this chapter I set the foundation to chapter 3 where I will present a series of tools and systems that aim to increase productivity and robustness across several production processes. The content of this chapter allows to envision chapters 4 and 5 into the right production perspective as well.

In the following section, the related work from different sources is presented. Then I introduce different concepts related to workflows and define the pipeline for a subset of it. I finally introduce more advanced concepts to help understanding task management, data flow and interaction across pipelines, workflow, DAM and DPM systems.

I am aiming at submitting the contents of this chapter for publication in the Journal of Digital Asset Management [Journal of Digital Media Management, 2015], which has already published related work from industry leaders.

2.1. Existing techniques, technologies and tools

Production and asset management, as well as media pipelines and workflows are of big interest in the industry and are recently receiving growing interest in the literature, with publications in books and conferences. In this section I present solutions and studies, including a review of existing tools and interviews to relevant members of production companies.

In the recently published book *Production Pipeline Fundamentals for Film and Games* [Dunlop, 2014] several experts from the videogame and VFX industry share their knowledge and lessons learnt about media pipelines. The book (page 25) defines the *Pipeline* as “the glue that holds together the work of each artist involved in a production” and, while the *workflow* concept is not clearly defined, it is mentioned across the book to highlight the

importance of functionally tying it to the pipeline. It also provides some examples of pipelines for games and VFX (figure 2.1.1), which show the highly complex interaction between different aspects of production.

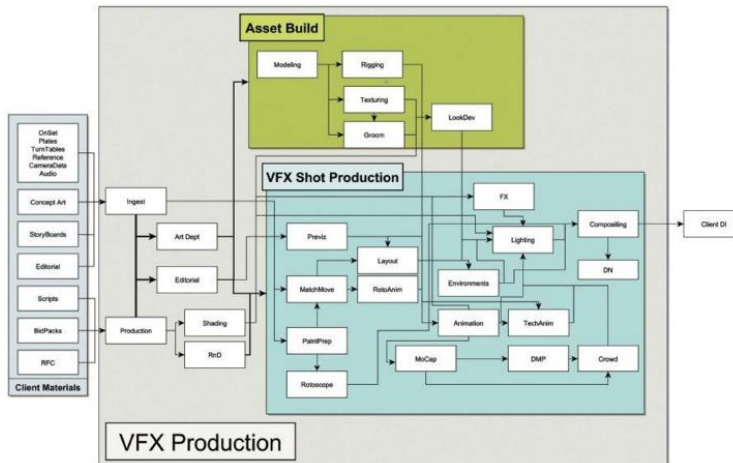


Figure 2.1.1: A graph representing a possible VFX pipeline from the book Production Pipeline Fundamentals for Film and Games [Dunlop, 2014], pp. 28.

In an interview [Chung, 2011], Hannes Ricklefs, currently Head of Software at the Moving Picture Company, commented about his “Global Visual Effects Pipeline” panel at Siggraph: “Asset management, automation and workflows are some of the last remaining challenges within the VFX industry”. While the benefits of automating the pipeline through DAM and DPM systems have not been measured and no methodologies still exist for this purpose, some authors have described possible scenarios. Dittmer [Dittmer, 2012] describes how a DAM can be used to automate the workflow and use the status of job tasks to drive the visibility and accessibility of information by users. This is very important as a media production may suffer from a pollution of assets, files, folders and tasks and the user may find it difficult to navigate if no intelligent filtering is performed to let him/her see what only matters, in his/her own production context.

Thanks to the acknowledgement of this need, in the last decade software companies have been starting to develop tools designed to provide artists and media production companies an intuitive approach to the management of production and assets. The first

successful incarnation of this idea was Alienbrain [Alienbrain, 2015], which at the beginning of the 2000 provided a versioning system for assets and a visual interface for artists to use. However, Alienbrain is not popular in the industry nowadays, thus I will briefly describe the two most relevant products today: Shotgun and Tactic.

Shotgun. Shotgun [Shotgun, 2015] is probably the most famous product of its kind. It mostly focuses on production tracking, building its strength on the creation and managing of production tasks. Assets are managed as results of the work performed on the tasks and can be versioned while progressing through completion.

The Shotgun system registers every interaction on a log. The log, more than being a simple record of the transactions, can also be used to trigger specific automated processes. It is in fact possible to track all the activity by every user on every task (figure 2.1.2) and, depending on this, perform predefined actions. For example, one simple automation could be the dispatch of an email to the supervisors when a new version is ready for review, but more complex behaviours can be performed. In fact, thanks to its Python API, it is possible to execute other tools (e.g., generating a geometric cache of an animated character whenever an animation is approved).

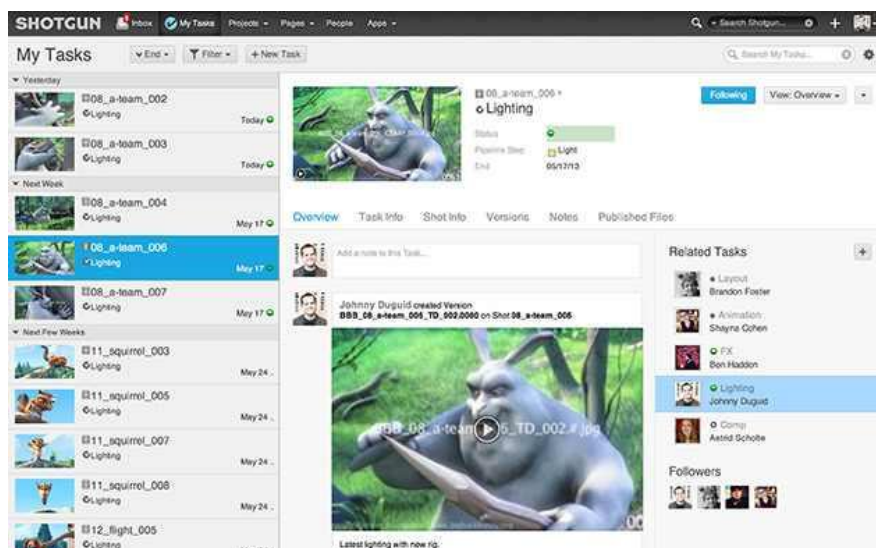


Figure 2.1.2: Shotgun enables users and production coordinators to review tasks by user.

The system runs in a browser and enables users to collaborate by attaching written comments to asset versions. Furthermore, with the software having been recently bought by Autodesk, it is currently provided with a flexible set of tools to interact with other Autodesk software and with the other recent acquisition RV [RV, 2015]. Thanks to the integration with RV, it is possible to perform reviews on still images and sequences, creating annotations and even syncing the review session across different users around the world. These acquisition moves demonstrate as well how much production tracking is a hot topic for the industry.

Shotgun runs on a PostgreSQL database [PostgreSQL, 2015], which can be managed through a graphical User Interface UI (figure 2.1.3) or through a textual terminal.

The screenshot shows the 'Shots' interface in Shotgun. It features a table with columns for Shot Code, Status, Thumbnail, and several production stages: LAY (Layout), ANM (Animation), FX (Effects), LGT (Lighting), and CMP (Compositing). Each stage has a 'Status' column. The rows represent individual shots, such as '08_a-team_001' through '08_a-team_004'. The interface includes a toolbar with options like 'Sort', 'Group', 'Fields', '+ Shot', 'More', and 'Pipeline'.

Shot Code	Status	Thumbnail	LAY Status	ANM Status	FX Status	LGT Status	CMP Status	Sequence
08_a-team (20)								
08_a-team_001	●		●	●			●	08_a-team ●
08_a-team_002			●	●	●	●	●	08_a-team ●
08_a-team_003			●	●	●	●	●	08_a-team ●
08_a-team_004			●	●	●	●	●	08_a-team ●

Figure 2.1.3: In Shotgun the different stages of production are accessible through a excel-like data sheet. Each stage is represented by a column, while shots are described in rows. This information can be edited.

Tactic. Southpaw Technology offers an OpenSource solution that focuses its attention on the workflow. In Tactic [Noteboom, 2013] it is possible to explicitly define all the production steps and easily build workflows through a node-based dependency graph. Multiple workflows can be defined for a single pipeline, constituting the set of tasks to be completed to deliver the product (figure 2.1.4).

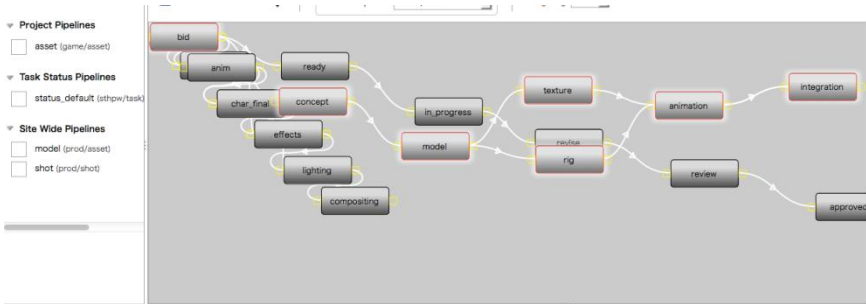


Figure 2.1.4: Tactic provides an advanced UI to build production workflows and pipelines as dependency graphs.

Each production stage may have one or more associated asset types and each asset can be defined by a set of tags and a mixture of file types as single file, file sequence and file sets, to cover all the production needs. The relationship among assets is also defined through a node system that describes how different assets get bundled together and the dependencies across them (figure 2.1.5).

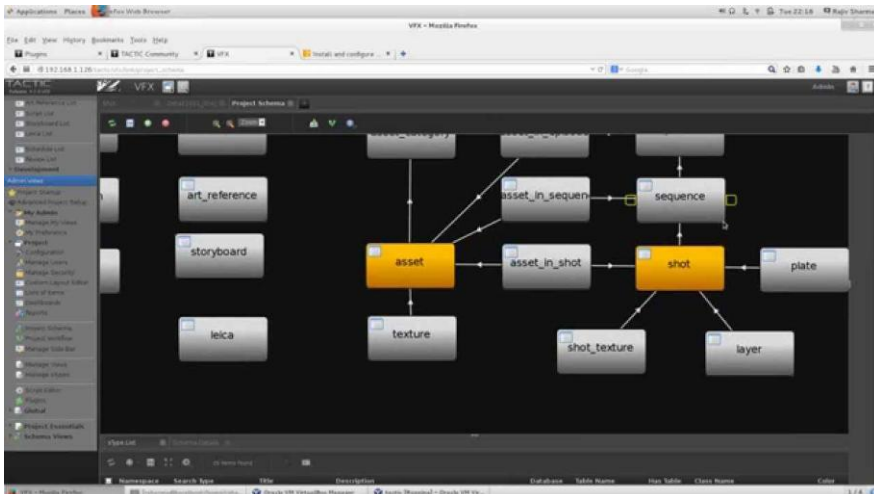


Figure 2.1.5: The relationship across the tables of the database can be specified through a dependency graph in Tactic.

Tactic provides an event triggering system that can be used to define automatic processes to be run when selected events occur. In this way it is possible to automatically run post-processes over

delivered assets, to generate new assets or to alter the status of production tasks.

Bundled with a simple interface (figure 2.1.6), it gives easy access to all its functionalities and enables expert users to implement custom widgets to create new views on the data (internally managed in a PostgreSQL database).

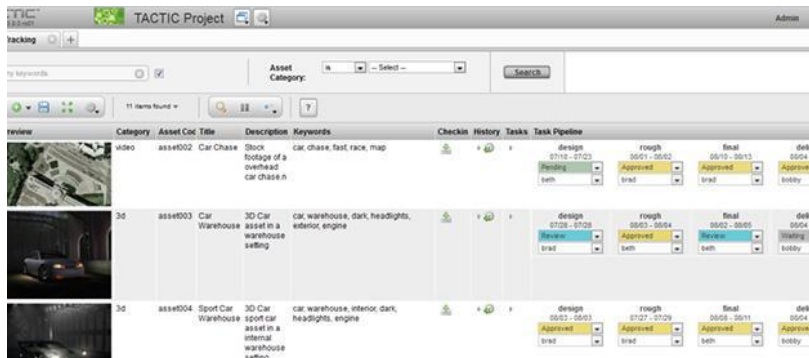


Figure 2.1.6: For each shot displayed as a row, Tactic shows the different production stages in editable columns.

In the industry, these commercial solutions have been adopted by some studios at different levels, but very few information is shared about how the integration is performed in the pipeline. This is also true when considering in-house developments as *Pronto* [Ricklefs, 2013], the Moving Picture Company DPM system.

To better understand the benefits of Digital Asset Management (DAM) in the media industry, it is interesting to quote what Avid’s white paper on “Understanding the Return on Investment of Media Asset Management Systems” [Avid, 2013] mentioned. In this paper, a series of clear benefits are identified:

- Improved utilization of production infrastructure;
- Collaboration across departments;
- Dramatically improved location-independent workflows;
- Process visibility and optimization;
- Automation of manual processes;
- Consistent metadata capture and protection through the media lifecycle;
- Improved discovery and utilization of archived assets;

- Production and multi-platform distribution combined into a single, file-based workflow;
- Less production cost and time to create higher quality content;
- Streamlined work order management and resource scheduling;
- Reduced distribution time and improved quality of the end product;
- Creation of an agile infrastructure that supports changing needs;
- Expanded supply chain options;
- Lower aggregate storage cost through tiered storage management.

All of them fit the needs of the ever evolving media industry, with lots of data being generated and stored every day, many artists and technicians working on and sharing them. Furthermore, considering the delocalization of production, due to tax benefits, the third benefit “Dramatically improved location-independent workflows” is crucial to the sustainability of the industry. The document also provides some projections of performance gains in different scenarios. For example, the time to publish video news on the web is expected to be reduced by 90% when using a workflow that implements a DAM system.

Sony Pictures Entertainment sees the importance of Digital Asset Management also in the afterlife of a production or an asset itself. This fits in the most recent trends of transmedia production, as the same asset should be ideally used for many purposes. In fact they recognize three dimensions required to address metadata [Yasur, 2013]:

- *Immersion*: to enable customers and business units to interact with assets at a granular level, meaning frames, clips and even versions of the asset;
- *Expansion*: extend metadata to related material that was instrumental to the main production (e.g., trailers and behind the scenes) to enable re-use for future products;

- *Ubiquity*: to account for the effects of globalization it is necessary to reduce and optimize the creation of metadata from different locations and sources, but also to improve how information is provided in different locations.

The work presented in the following sections aims at providing the core knowledge about CG media production in terms of pipeline and workflow and describes a strategy to tie all the concepts together toward an automated pipeline.

2.2. Abstract production workflow formulation

Production workflows can be defined by using hierarchies of nested structures of connected elements. The levels in the hierarchy are not homogeneous as the types of processes, connections and data differ. Each level describes the production process at different granularity, thus each level of the workflow describes a workflow in itself, with its contexts and rules. Each level can implement the different element types.

Data: is the digital information produced through both manual and automated processes.

Asset: is the database entry description of the data that unambiguously defines how sets of data collaborate to conform a specific creative resource. Assets can be used to define audio sources, bi-dimensional images (as textures or rendered frames), three-dimensional geometries, animations, etc.

Check-in / check-out: data has to be checked in and out, to respectively store or load it. Through check-out the database gets queried and the relevant data is retrieved from the storage. On check-in, output files get saved on the storage and metadata is associated in the database. Depending on the adopted data flow strategy (see section 2.5), this process has to guarantee that metadata is properly propagated.

Plug: processes receive input data and produce output. These are managed through plugs.

Operator: a specialized, automated or manual, computational element that operates on data (e.g. modelling the geometry of a character).

Evaluation: each sequence of processes and operators should produce a result that obeys to certain rules. In the evaluation these rules are checked against the data. If the evaluation is not passed, data does not get checked-in and communication is performed to inform about the issues.

Trigger: defines the behaviour of the workflow whenever new data is checked-in. Triggers can fire various automated systems as geometries caching, communication or production management.

Process: a set of data, operators, checks and evaluations that serve a common goal (e.g., modeling or animation).

Flow: defines how two sub-stages are connected. Two connected stages share their data, depending on the properties of the specified flow.

Stage: a set of processes that represents a specific production domain. Different stages require different artistic and technical skill-sets. The data flows across stages in serial manner.

2.3. Detailed CG workflow description

In this section I define a formal media production workflow, aiming at consistency and flexibility. This formalisation is the result of a series of interviews with department supervisors from different production companies, and my own experience in companies. The goal of such interviews was to understand the internal functioning of each department, their workflow and control procedures. This process also helped in defining the relationship and constraints within each department and among them, being then formalized in data checks and triggers. The evaluations at delivery will also be briefly described; on delivery, the content is evaluated against a set

of rules for the asset and, on failure, the user is asked to decide whether to proceed with delivery or address the issues.

The first categorization needed for our workflow formulation is determined by the three main stages of the computer generated (CG) media production processes: pre-production, production and post-production (Figure 2.3.1). This differs from the overall media production, where other components may be considered as the shooting of live action footage, causing a shift in processes (as seen in section 2.3.4).

In pre-production (section 2.3.1) all the narrative is defined and the first set of requirements is made. Decisions about the assets that need to be created are taken at this stage and extra contents to ease the work in the next stages are created: storyboards, animatics and pre-visualisations. This is the stage where most of the initial information about the production has to be introduced in the production management system.

In the production stage (section 2.3.2), all the required assets get created and combined together to animate all the shots, lit and render them. In the production stage I will schematically describe some of the processes to show the complex data flow and logic that lays behind production processes.

The resulting data from the production stage, gets into post-production (section 2.3.3) where final post-processing is applied to the renders. The post processing can involve the compositing of different layers to create the final look for the shot or the modification of the lighting or color information.

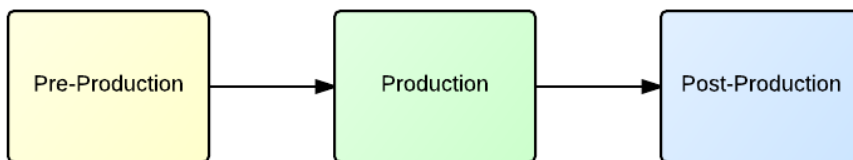


Figure 2.3.1: Diagram showing the overall processes involved in modern CGI media production. In both feature animation and VFX production, there are three main stages, pre-production, production and post-production.

It is important to note that the most recent trends in the industry

lead to blurring the separation between the production and post-production stages. However, for the sake of formality we require for this description, we will still consider them as two separate stages, while mentioning possible integration in our further analysis.

a) The pre-production stage

The pre-production stage is a key element in our workflow formalisation. At this stage, the narrative is studied and turned into data of different kinds. The key aspect is that here all the details to bring the script to life are turned into images and will then drive the artistic creation in the production stage. Most importantly, in order to allow metadata propagation, all the assets required for the production have to be defined in the *Assets Definition* sub-stage.

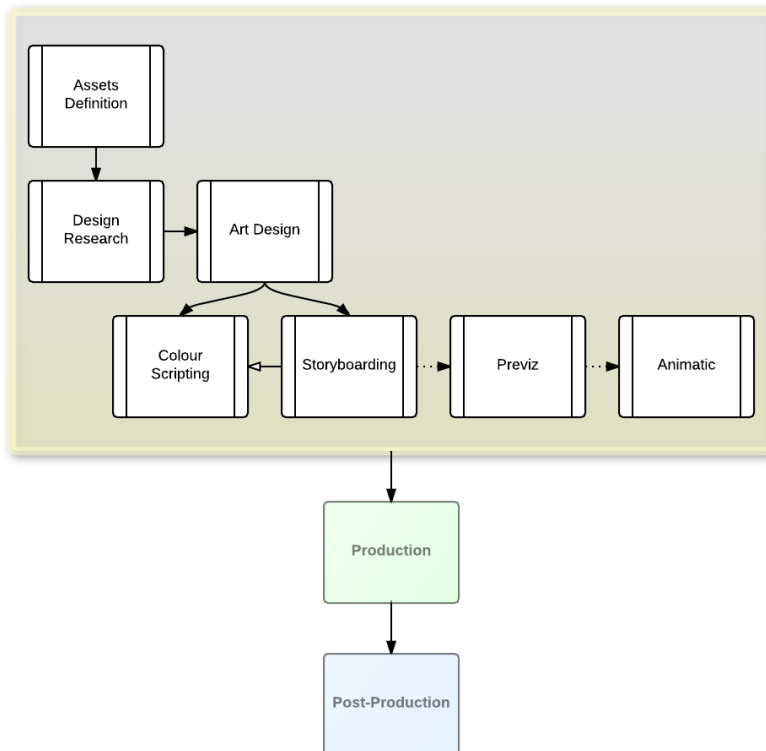


Figure 2.3.2: The figure shows the sub-stages involved in the pre-production stage. Through Assets Definition all the required assets for the production are

defined. Then, all of the following sub-stages produce data that is then used in the following production and post-production stages to guide the artist work.

Asset Definition. From the script all the required assets are identified. The assets can be shots, three-dimensional geometries, textures, environments. Starting from the definitions, all the production tasks aim to bring the assets to completion. The data to define an asset directly relates to the type (e.g. scene, shot, character, environment, etc.) of the asset.

Design Research. Each asset has to be designed in order to be then properly created for the screen. This sub-stage aims at collecting references and study design possibilities for each asset.

Art Design. Using the resulting material from the Design Research sub-stage, key aspects of an asset are graphically defined. The art design includes technical information to properly model the geometry of the asset, commonly delivered as orthographic views and drawings of the relevant details. Furthermore, other aspects are covered to set a guide for later artists to use. For example, designers define a set of basic shapes and poses for characters' body and face, to drive the work of modellers, riggers and animators as described in section 2.3.2. Finally, color and textures are defined in the Art Design. In figure 2.3.3 and figure 2.3.4 I show the design work made for the TV series pilot Noa&Max [Noa&Max, 2013].

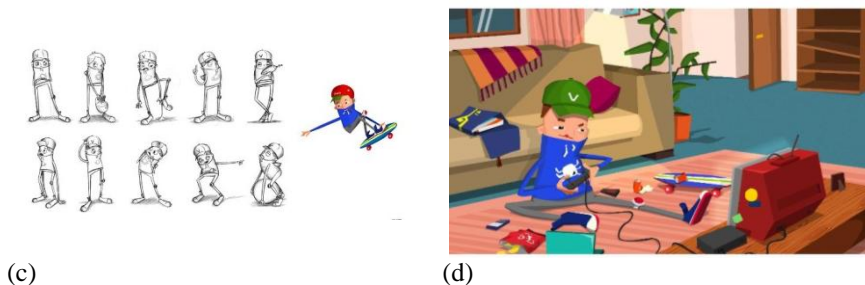
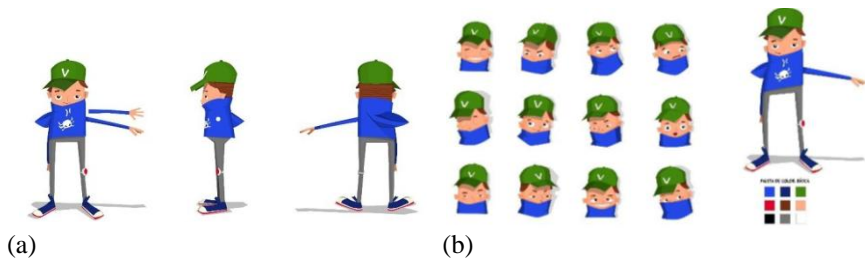
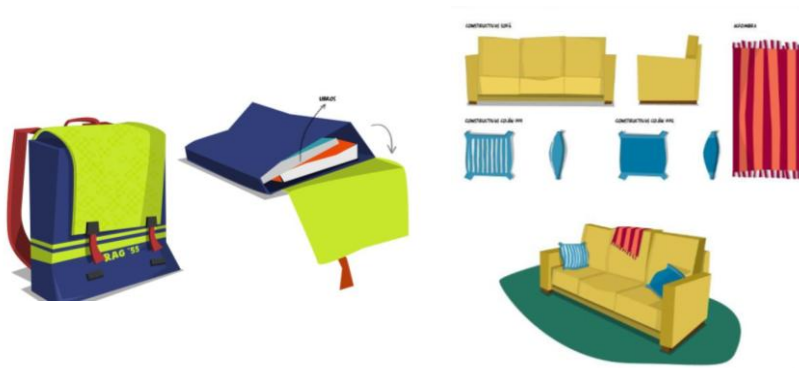


Figure 2.3.3: Art Design for a character from the Noa&Max series. (a) orthographic view; (b) model sheet to describe facial poses; (c) model sheet for body poses; (d) a drawing showing the character in context.



(a) (b)
Figure 2.3.4: Design for two elements from the Noa&Max series. (a) design of a backpack; (b) design of the elements of a sofa for an environment (sofa, pillows, blanket).

Colour Scripting. For each shot or sequence, it is important to sketch the lighting and mood for future reference. This stage has a big relation with the narrative of the final product and it consists of a series of coloured drawings that approximate the desired final perceived look.

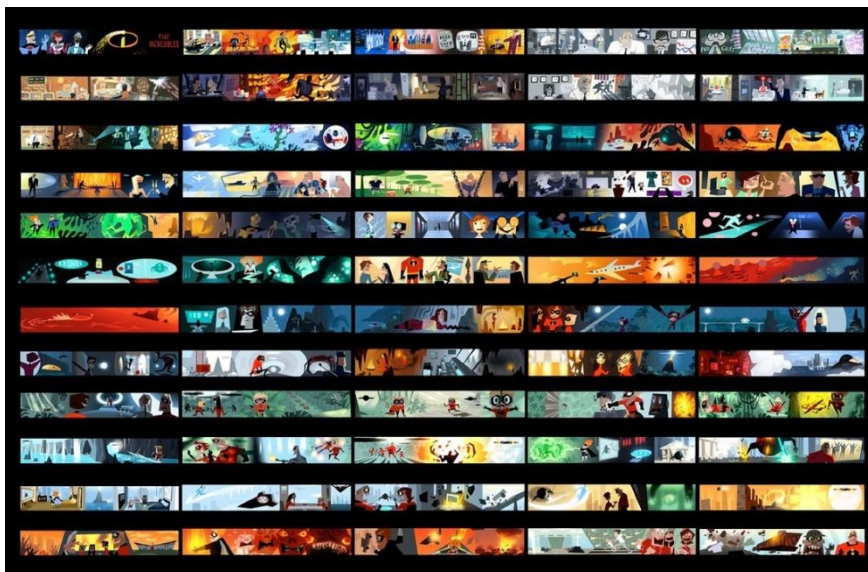


Figure 2.3.5: Color script for the film *The Incredibles* (Brad Bird, 2004). The complete film is depicted in different panels showing the main colour scheme for each sequence.

Storyboarding. A storyboard is a collection of drawn panels that describe each shot related to a given script. The drawing approximates the camera angle desired by the director and also describes the configuration of the space and characters within the shot. The storyboard can describe a single shot by means of multiple panels if the action is too complex. In the storyboard, the camera movement is also described. Figure 2.3.6 shows a storyboard panel from a shot for *No Pets Allowed*, an animated short-film produced at Sixbirds Production and Studio Nest while I was part of the pipeline development team.

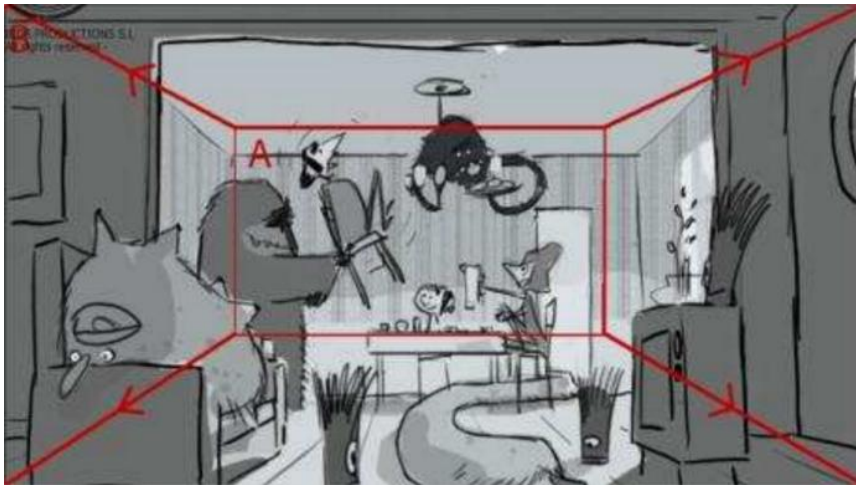


Figure 2.3.6: Storyboard for one shot of the short-film *No Pets Allowed*. The panel describes a change of the camera lens: a zoom-out (in red).

Animatic. Once the storyboard is available, it is possible to edit in time the different panels to have a first rough version of the film (or other media format) to review. The animatic can help in determining whether the timing between shots is right and the overall rhythm. Furthermore an animatic with audio can be used to review dialogues duration. Finally, it gives the opportunity to review whether the continuity across shots is preserved.

Previz. The previz is an advanced type of animatic that is obtained by animating draft versions of the characters within a rough representation of the environments. Voices are normally made-up and the cameras match the desired angle. The result is a very unpolished version of the film and all the contents are disposable but it is a powerful tool for directors to preview their ideas. Animatics are very important for feature animation and VFX but they can also be used to prepare the shooting of live action footage.

b) The production stage

The production stage is a complex set of processes that cover the life of assets from their creation through modeling, until their rendering. Figure 2.3.7 shows the processes involved in the production stage. It starts with the modeling of each asset, which are then rigged to be posed in the virtual set and textured. The textures are used to define the shading of the asset's surface or basic properties of its groom (which can also be rigged to provide artistic control to animators). Each rigged asset is placed in the scene by layout, where also cameras are defined. If necessary, an asset can be animated and, in some cases, several assets can form a simulated crowd. In technical animation (or techAnim), fixes or dynamics (eg. cloth simulation) required to improve the shot are put in place. In some cases effects are added to increase the detail of scenes with simulations of fluids, smoke or rigid and soft body dynamics, just to name some. Finally, the shot with all the assets is lit and rendered.

In this section, special attention will be put on the Modeling, Rigging, Layout and Animation processes. For these, I present schematics that define the data flow within the process. This is meant as an example of the need to engineer processes in a formal and unequivocal manner. The complexity of processes, data and their flow is such that without a clear definition, it is likely that the production workflow and pipeline are undermined, requiring constant changes and tweaks. It is important to consider that changes in the assets and processes produce data inconsistencies, making older data produced with a different workflow and pipeline not reusable as not interpretable anymore.

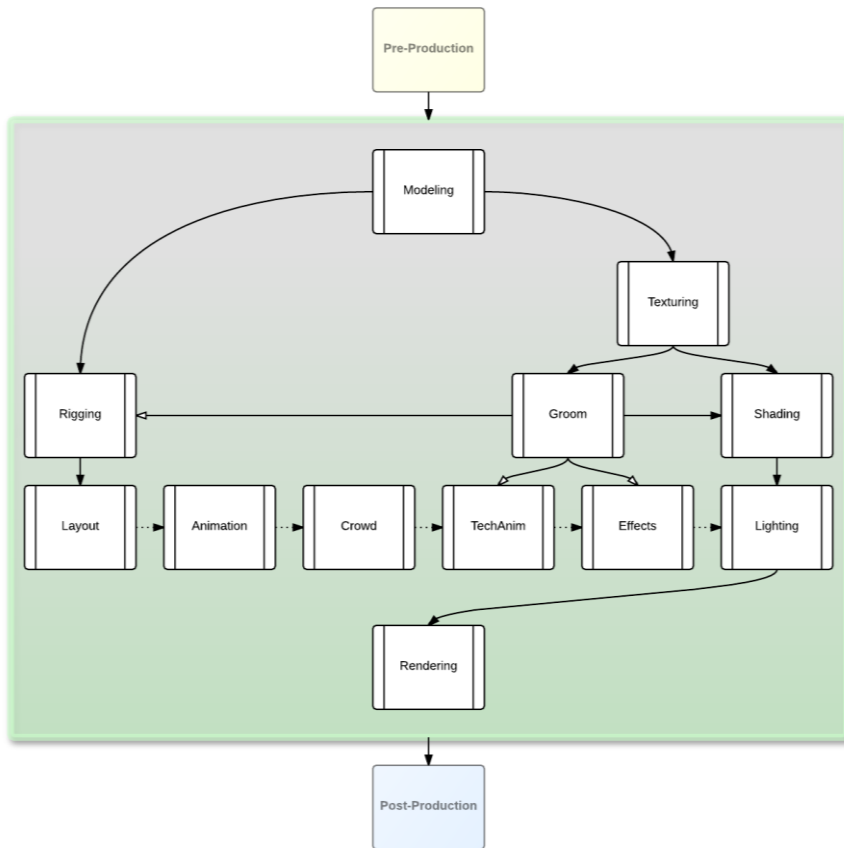


Figure 2.3.7: Diagram of the various processes involved in the production stage. Some processes are required (e.g. rigging) and others are optional (e.g. techanim).

Modeling. In the modeling process, two consecutive processes can be identified: blocking and refinement. The geometry produced in the blocking process is used to start the refinement. As a quick summary, in the blocking, the rough proportions, volume and edge flow of the asset is determined, while in refinement all the details are modelled and the asset reaches its final shape. In both cases, the delivered result can be used as a template later in production.

In the blocking, the overall shape and volume of the model is defined. The process (figure 2.3.8) requires design information to get started; design research for references and the art design to guide the work. The modeling process can also start from previous

geometry as templates. With all the required data being retrieved and loaded in the scene, the modeler can start modeling and delivering the resulting geometry for versioning at any moment. On delivery, the model mesh is evaluated for issues as faces that share all edges (lamina faces) or inconsistent normals.

The refinement process (figure 2.3.9) starts from the latest approved version of the asset's blocking model to add details, create texture mapping coordinates (UVmap) and target shapes to be used in conjunction with a blend shape or pose space deformation system [Lewis et al., 2000]. Thus, the refinement process can be delivered as three separate assets: base model, UV map and target model. Once the first version is delivered, next versions will start by collecting and merging these three pieces of data. As for the blocking, the data to be delivered is evaluated for each case.

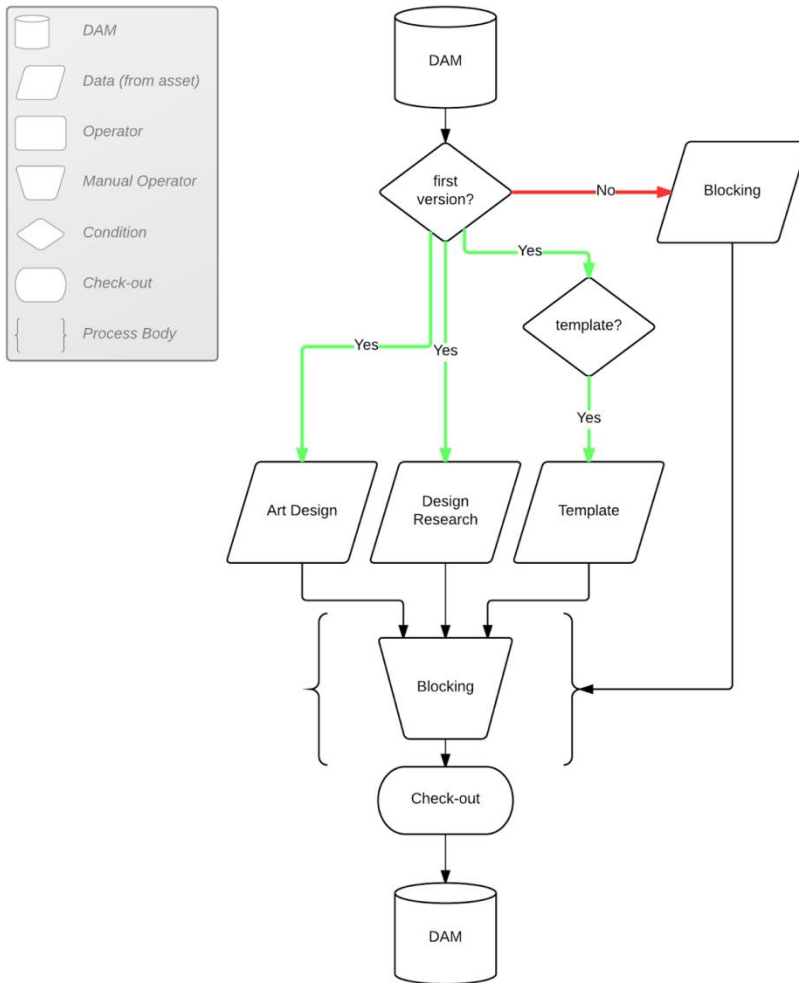


Figure 2.3.8: The schematic representation of the data flow and logic involved in the Blocking Modeling process described in the text. At any iteration of the process, the result can be delivered as a new version; thus, the model is evaluated prior to its storage and registration in the database. Errors can be fixed or delivery can still be approved with issues.

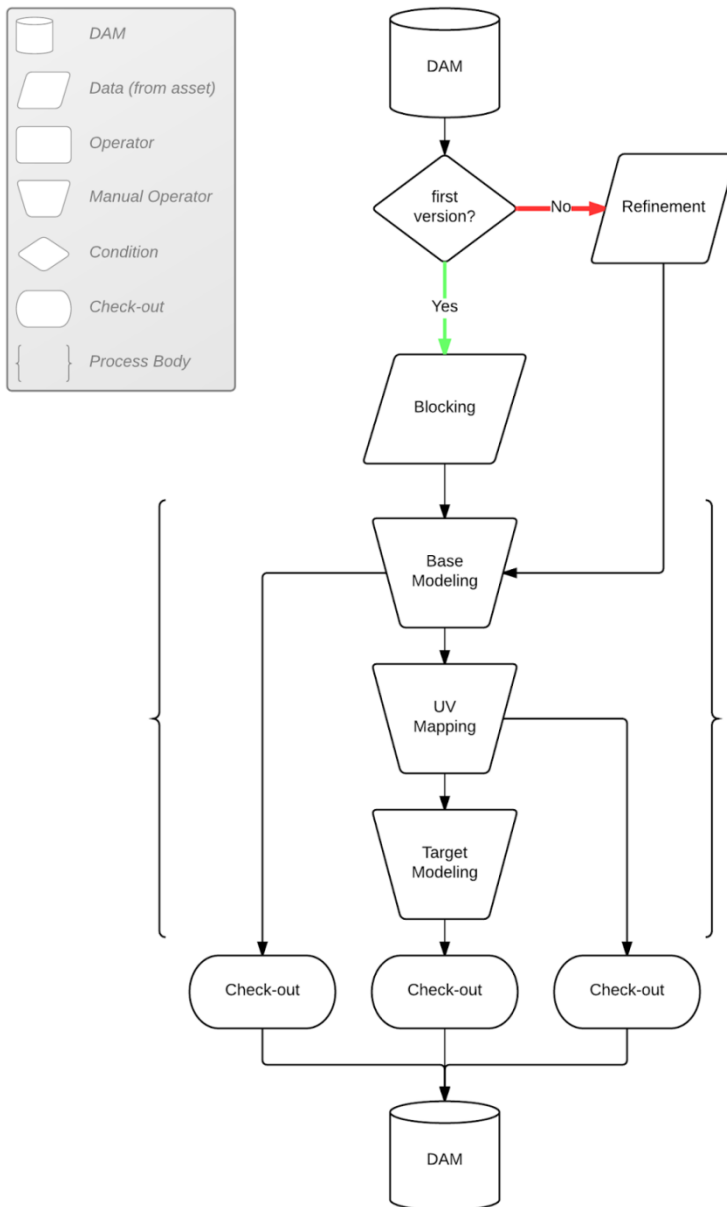


Figure 2.3.9: The modeling refinement process starts from a blocking model and aims at adding more detail on the geometry and produce other necessary data as the UV maps used for texture mapping and the target geometry to be used for blend shapes and pose space deformation.

Rigging. The goal of the rigging process is to provide a geometry with a control system that enables artists to easily access the required functionalities of the digital object. For example, in the case of characters or vehicles this control system contains the set of animation controls, plus other attributes to control practical aspects as the visibility. There are three different levels of complexity for the rig: layout, simple and advanced. The layout rig is solely for placement and very basic parametrization or animation of the object in the virtual set; the simple rig is intended for rapid interaction with all the deformation abilities of the object; the advanced rig implements enhanced deformation and simulation techniques that cannot be computed rapidly and require an offline solution. The three types of rig add complexity using the previous one as a starting point so that any setting used on a layout or simple rig will still be properly evaluated on an advanced rig. It is important to mention that, while the layout rig is required, the other two are not, as static environment geometries do not require any rig more complex than the layout one. Even though, as they build hierarchically, it is important to have a simple rig if the advanced one is necessary.

The layout rig (figure 2.3.10) is intended to be used by the layout department to populate the virtual set and prepare a first staging of characters in the scene. Furthermore, layout artists use rigged cameras to precisely match the camera angle, lens and movement desired by the director for each shot. To facilitate the interoperability across departments and avoid delays in production, the layout rig can be started using the approved blocking model, waiting for the refined model to be completed. A first version of the layout rig can also be obtained automatically, as a trigger from modeling, by simply binding the asset's geometry to a single joint and controller, which is sufficient for simple placement of objects in the 3D space. On top of this first version, it is possible to build a more complex skeleton or deformation system for simple animations. The evaluation check for the delivery of the layout rig reviews that all geometries are controlled by at least one deformer.

The simple rig iterates over the layout rig. In this part of the process (figure 2.3.11), the rigger adds further complexity to the layout rig by creating a complex deformation system that can be animated interactively. In the case of characters the simple rig consists of a

hierarchical skeleton structure that can be controlled by a set of intuitive controllers. The skeleton is commonly bound to the geometry using a linear or double quaternion skin deformer. The geometry can also be deformed by other performant deformers (e.g. blend shape). In a simple rig, the skeletal structure can also be built to perform forward and inverse kinematic, with twisting, bending and stretching joint chains.

Finally, as shown in figure 2.3.12, simulation can be added to the simple rig and obtain an advanced rig. An advanced rig should be used to add realistic effects on a deforming object through simulation. Muscle dynamics, skin wrinkles or soft bodies are achieved through complex simulation solvers that can hardly be computed at interactive rates. Hence, the need have these features in a separate rig that will be computed off-line.

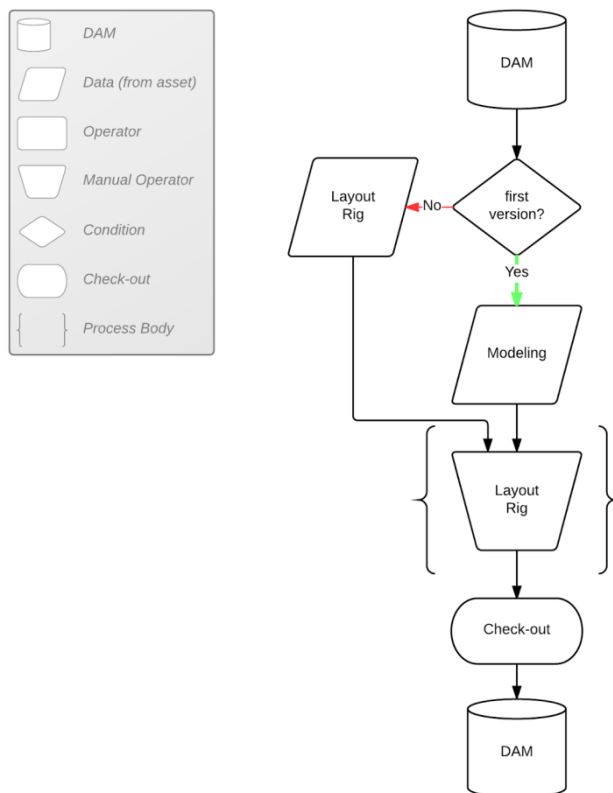


Figure 2.3.10: The layout rig is the first (and required) controllable version of a geometry.

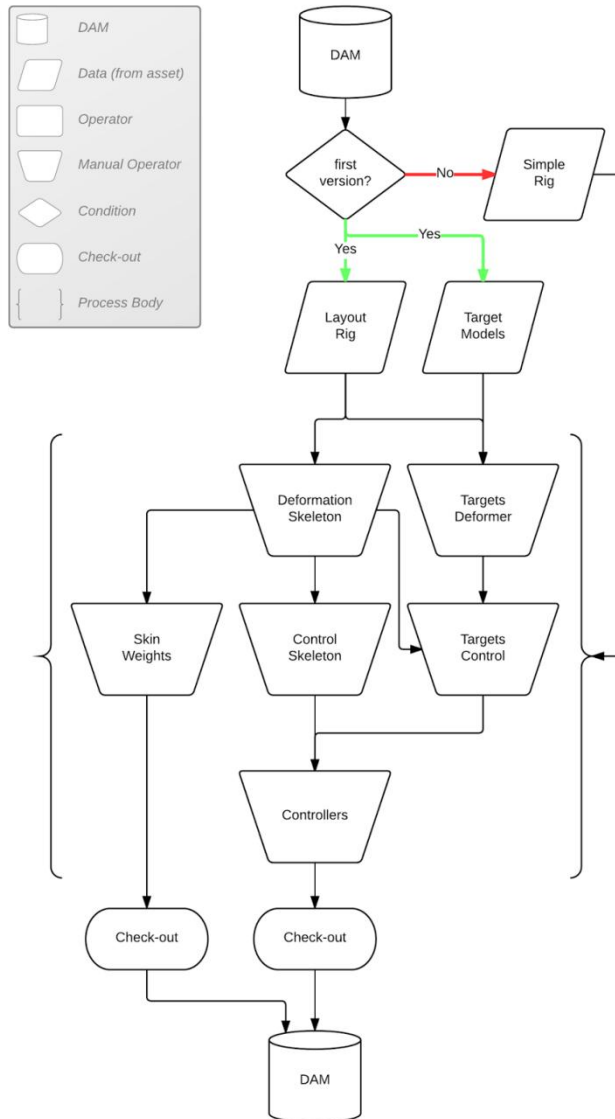


Figure 2.3.11: The simple rigging process aims at creating a skeletal structure to deform the geometry. The skeletal deformation is controlled by setting skin weights on the skinning deformer and by driving the deform skeleton through a control skeleton that implements features such as stretch and squash, bending and twisting. Other deformation is also obtained by controlling the target models from the refinement modeling process through a controlling system that can be driven by controllers or by configurations of the deformation skeleton (pose space deformation).

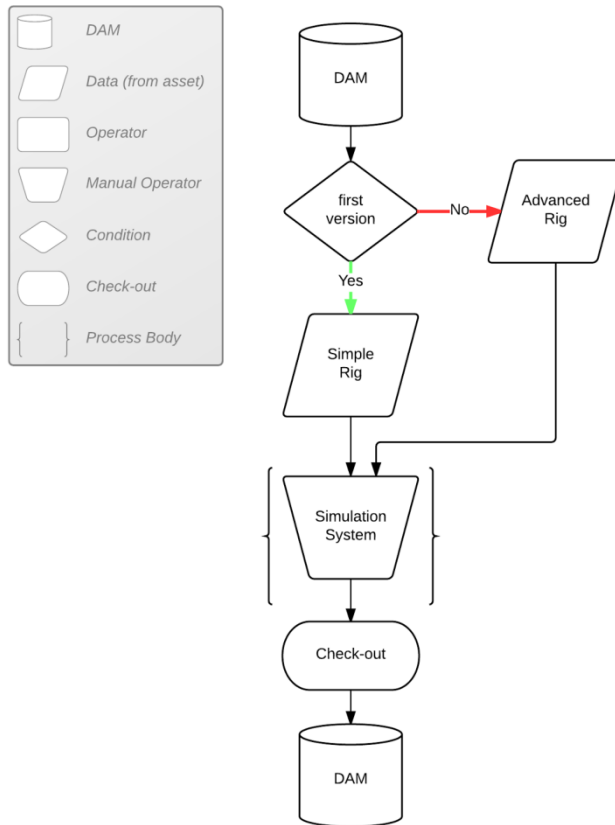


Figure 2.3.12: The advanced rig is obtained by adding advanced simulation solutions over the simple rig. Simulation can aim at producing realistic behaviours such as volume preservation, wrinkles and muscle dynamics.

Layout. In the layout process all available assets are loaded into a scene and placed at the right position, trying to match the information from the storyboard. An asset needs to have at least an approved layout rig version in order to be considered for layout. The layout can start even if not all the assets are available and, as more assets get done, they have to be automatically added to the layout scene.

Another key aspect of layout is camera placement and setup. The camera has to be properly placed and oriented, while all the lens (e.g. focal length, focus distance) and film (e.g. offset, aspect ratio) properties are also set to match the desired shot.

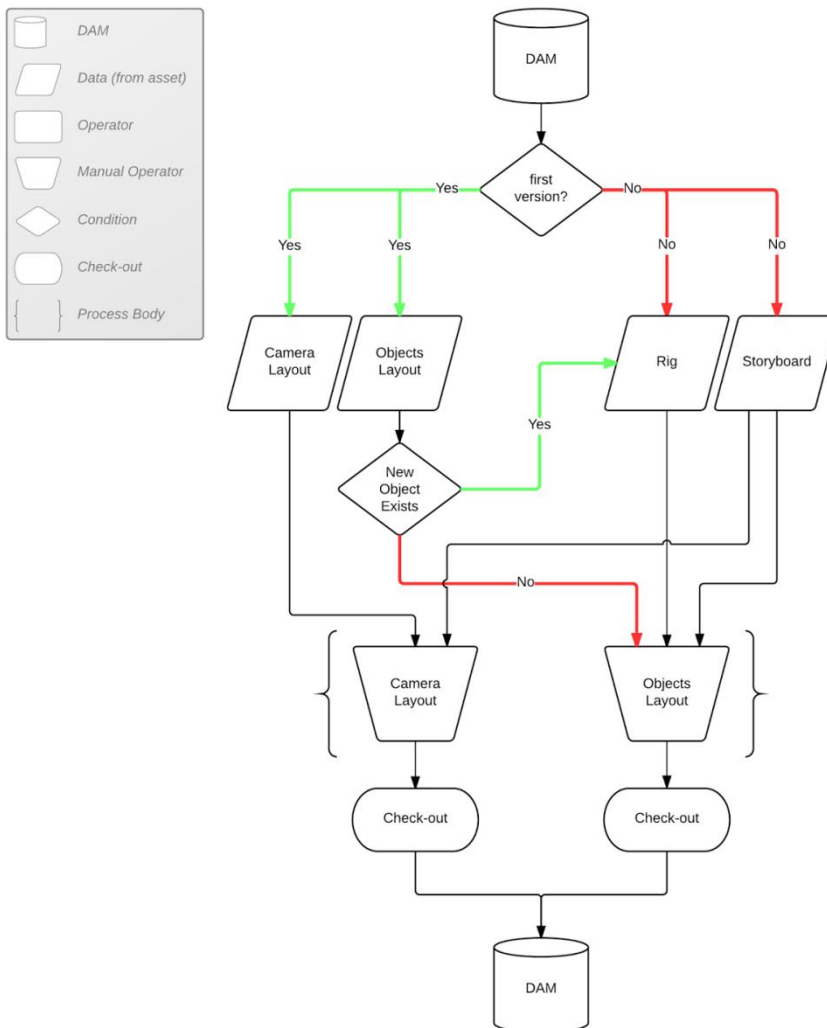


Figure 2.3.13: In layout the rigged assets are placed in the three-dimensional scene to match the indications from the script and the director. The camera is also set in the layout process; position, orientation, focal length, aspect ratio, focus distance and shutter angle are key factors that determine the shot and its render.

Finally, a first simplistic version of the animation can also be created in the layout process to roughly describe the movement of assets through the set in time. The animation is also commonly defined for cameras.

Animation. In the animation process, the assets placed in the layout get properly animated to represent all the actions required by the script. Similarly to what was described for modeling, as animation is a time-expensive process, it is important to split it into two sub-tasks: blocking and refinement. In this way it is possible to review the work at different levels of complexity and avoid destructive changes.

In the blocking (figure 2.3.14) the main poses of the animation are defined, without caring too much about the interpolation between them. This is something that is looked after in the refinement (figure 2.3.15), as more poses are added and the interpolation curves between keyframes are shaped to convey the right feeling of speed, rhythm, weight and drama.

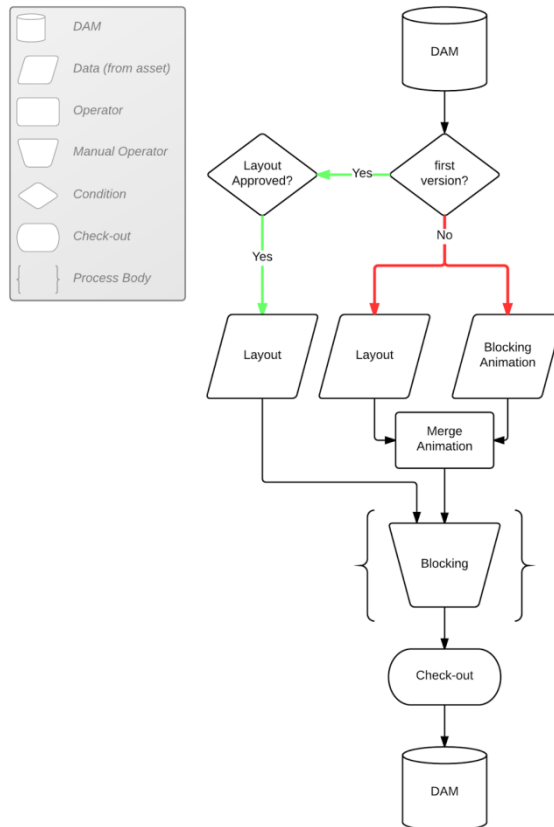


Figure 2.3.14: In the blocking of animation, the transformations from layout are merged with more detailed keyframes. By merging layers of animation from different processes it is possible to make the workflow non-destructive.

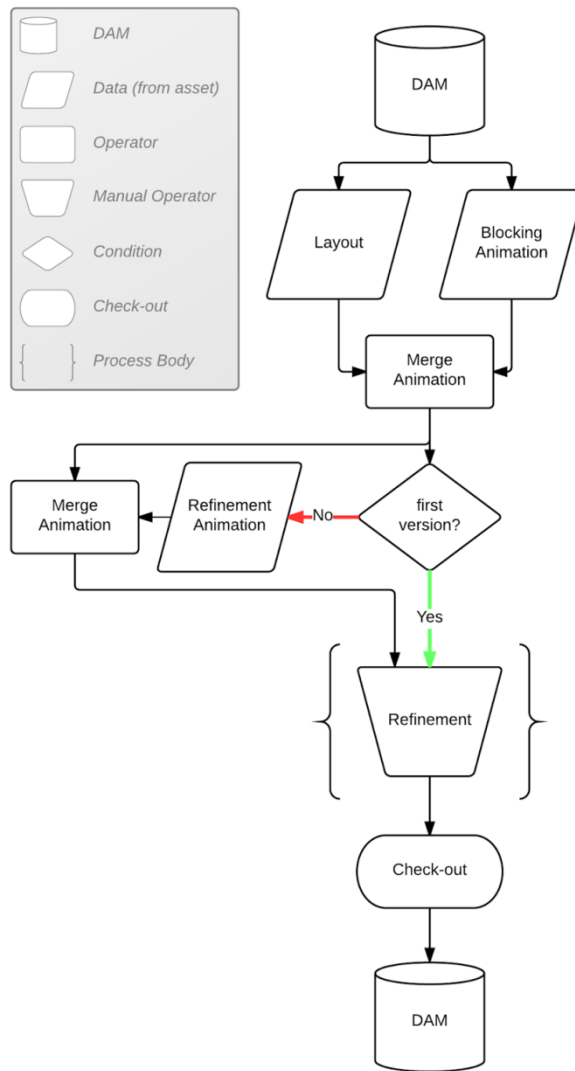


Figure 2.3.15: The animation refinement modifies the blocking animation by changing the set of keyframes (keyframes can be deleted, added or changed in value and time) and creating animation curves that define the interpolation between two keyframes.

When delivering the animation, a trigger could automate the generation of a geometry cache: a set of files containing the position of each deformed point through each frame of the animation. This cache is crucial for the later effects, techanim, lighting and rendering processes.

Shading. In the shading process artists define the material properties that determine how lights interact with mesh surfaces. Shaders are typically dependent on the engine used to render the frames. Depending on the complexity of the production and the size of the production company, preset shaders could be used from a library, or new custom shaders could be written by shader writers.

Recently, Sony Pictures Imageworks has released the Open Shading Library (OSL) [Open Shading Language, 2015], their shading writing language for Arnold Renderer [Arnold, 2015], with the goal to define an open standard shared across the industry. OSL has not gained ground in the industry yet, but this might change, as it already happened with OpenEXR, Alembic and OpenColorIO.

The shading process is strictly related to texturing, as each property of a shader that can affect differently on different parts of the surface can be driven by a painted texture. When procedural textures are required, their definition could be realized within the same shading process.

When shaders are defined, they are assigned to geometry components to define how different parts of geometries react to light. The resulting data is stored so that shaders and how they are assigned to geometries are treated separately as different assets.

As each shot might require a slightly different look, shading can be overridden in lighting to accommodate any creative need.

Lighting. The lighting process focuses on creating the lighting setup for shots and their contained geometries. Lights can be of different types (point, spot, area, etc.) and produce various kinds of shadows (e.g. depth map or raytrace). Commonly, lights are defined by their position and orientation, but when using photorealistic rendering engines, the set of parameters and algorithms can vary a

lot across different systems and no effort for formal standardization exists yet. These settings also need to be stored in the lighting setup.

Furthermore, in lighting, artists define how the image has to be separated in layers and passes. Layers define how different assets are separated in different files. For example, the background with some distant environment elements and the foreground with the acting characters could be rendered in two separate layers that are then composed together in the post-production stage. Passes are similar to layers but, rather than separating the image by assets, they provide further information about each pixel. A typical example of passes are the light components (diffuse, specular, reflections, etc.), surface properties (normals) or spatial information (e.g. camera depth or world position).

As mentioned previously, some shots might require the shading to be slightly tweaked to meet the creative requirements. Thus, the lighting process can act upon the shaders and deliver a shading override that applies only on the required shot.

The delivery consists of the set of lights, each with its settings (including shadows) and the shot rendering parameters.

Grooming. In groom, anything from short fur to long hairs or even grass is created by artists on geometries. The process operates geometry caches spawning follicles on top of the surface.

As for shading, texturing plays a crucial role for grooming as textures are used to determine all the parameters to comb and control the curves that describe the groom.

Crowd. Crowds are a specific extension of the animation and motion editing processes. Multiple characters (agents) are animated by applying a set of rules that trigger specific behaviours and related animations (clips). These animations come in the form of skeleton caches.

The crowd process is heavily based on the idea of instancing geometries and producing skeleton caches of the complete agents animation as output.

TechAnim. Delivered animations trigger a geometry caching process on delivery. When the caching happens on the advanced rig, it might modify the surface and volumes of the asset. In the TechAnim process, these caches are reviewed and any undesired effect is fixed by modifying the animation or the parameters in the advanced rig.

Furthermore, in TechAnim, any animation that happens through simulation (e.g. cloth simulation), rather than hand-crafted animation through controllers, is set up and performed to produce new sets of geometry caches.

Effects. In the effects process, dynamics are added to the shot in the form of particles (sparks, fluids, smoke, etc.) or meshes (iso-surfaces, soft bodies, rigid bodies, etc.).

When effects are applied on existing assets, the geometry caches of the specific assets are retrieved and used in the process.

The effects are delivered in the form of cached particles or cached geometries, depending on the typology of effects. The delivery is also accompanied by a file proprietary to the used Digital Content Creation package (DCC).

Texturing. In order to define various properties on three-dimensional surfaces for shading and groom, it is necessary to generate a texture map that describes these properties by means of pixels. The process of creating such maps is called texturing.

For the texturing to happen, it is necessary to have a way to represent the three-dimensional geometry as a bi-dimensional paintable canvas. This is commonly provided by UV maps but it can also be obtained using PTex [Burley & Lacewel, 2015], a technique which does not require manual unwrapping of UV maps and allows for different resolutions to be adopted on different parts of the mesh, depending on the requirements for the texture. Unfortunately, as PTex textures can be created and manipulated exclusively through specific software, UV maps are still the most used technique to drive textures.

The delivery of textures does not relate the texture to any other element but in the shading setup or groom setup, the binding between properties and textures is explicitly defined.

Rendering. The rendering process elaborates over the lighting process and is automated through a Rendering Management Software (RMS). The geometry and particle caches, the shading data, the light setup and the render settings (also set in lighting) are retrieved and used to render the frames for the desired shot.

c) The post-production stage

The post-production stage aims at bringing together all the images, layers and passes generated in the production stage to create the final frame. At this stage the artists can make all sort of adjustments, modifying the illumination of objects by changing how their surfaces are lit and altering the perceived look of their materials by modifying the merging of the different passes. The stage begins with the precomp process which is automated and provides artists with a good enough starting point.

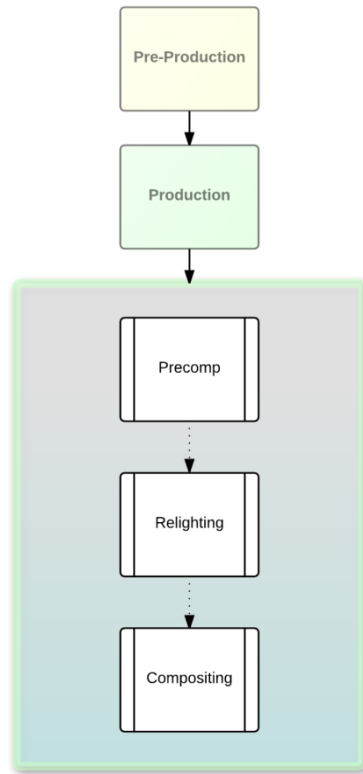


Figure 2.3.16: Postproduction is made of three consecutive processes, where the second one (relight) is optional.

Precomp. While the different layers and passes go through the rendering process, it is necessary to review the coherency of these layers and passes merged together as a single image. The precomp process produces a rough version of the finalized frames by using a set of standard rules for merging the passes and by compositing the different layers based on their distance from the camera (depth comp).

A simple (common) rule for merging passes is derived from using the following formula:

$$(d_f + s_f + rf_f + rr_f + i_f + sc_f) * o_f - sw_f \quad (1)$$

where d is the diffuse pass, s is the specular pass, rf is the reflection, rr is the refraction, i is the indirect contribution (e.g. final gather or

global illumination), sc is the sub-surface scatter contribution, o is the ambient occlusion and sw is the shadow. The subscript f represents the frame number from the rendered sequence.

This first merge can be done automatically as passes and layers are rendered, but artists can elaborate over this automated version to add quality and changes as the feedback is provided.

Relighting. After the precomp is ready and reviewed, directors and supervisors might request changes to the lighting of the three-dimensional objects rendered in the comped frames. As the render produces information about three-dimensional position of each pixel (world position pass), the normal at each pixel (normals pass) and their distance from the camera (depth pass), it is possible to create lights in the comp environment to modify their look.

Compositing. The draft compositing of the render layers from the precomp process, with the tweaks produced in the relighting process are now refined to obtain the final compositing. In this process, artists also ensure that any live-action footage and plate is seamlessly integrated with the Computer Generated contents.

d) Integration in a generic film production workflow

In the previous sections I have presented processes and stages strictly related to Computer Generated media, but it is possible to see how this has a strict link to other processes related to the creation of real-life assets. For example, in the VFX industry, a crucial part of the finalized product consists of live action footage (filming of actors or environments). Because of this, if we do not focus on CG media, it is necessary to rethink the graph seen in figure 2.3.1 by merging the production (section 2.3.3) into the post-production (section 2.3.3) and define a new production stage that is more related to the filming process (figure 2.3.17).

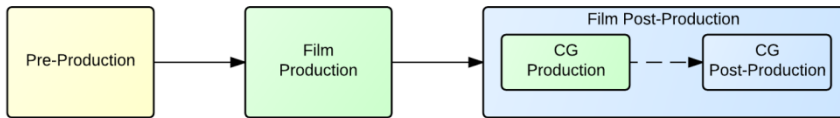


Figure 2.3.17: The film production workflow incorporates further aspects that are not directly related to CG media. These (e.g. live action shooting) are the processes that constitute the Film Production stage, while the production and post-production stages related to CG seen in the two previous sections are merged to form the Film Post-Production stage.

2.4. Specialized production workflows

In production it is necessary to separate the phases for the creation of different contents at different levels of granularity. Each level encompasses certain sub-stages of the workflow and the outcome is required by further levels.

In CG media production there are two main pipeline levels that can be identified as the *asset* and the *shot* (figure 2.4.1). In the asset pipeline each character, prop and environment gets created. In the shot pipeline all the previously created assets are used to set-up the scene and create the animation to be rendered.

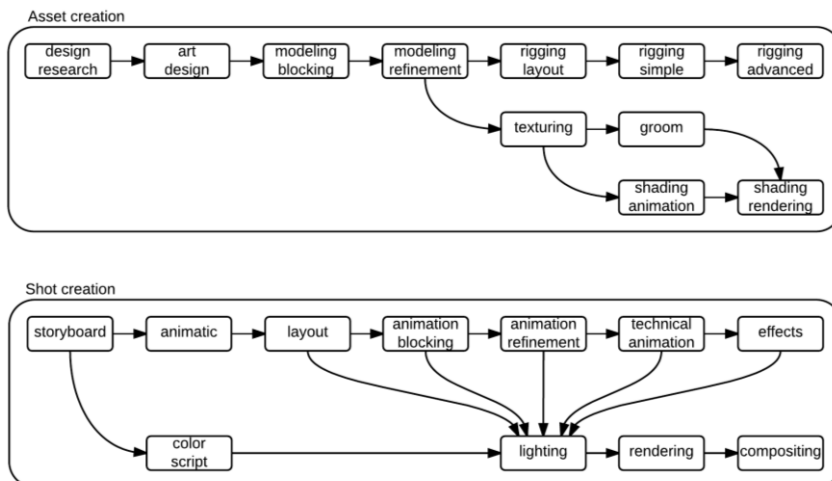


Figure 2.4.1: The workflow seen in section 2.3 can be split into two graphs, depending on the desired final outcome. The *Asset creation* graph (top) follows the processes required to build an asset to be used in a shot. The *Shot creation* graph (down) describes the interaction of processes to complete an animated shot.

The asset pipeline sub-stages are *modeling* (blocking and refinement), *shading* (animation and rendering), *rigging* (layout, animation and advanced) and *grooming*.

The shot pipeline sub-stages are *layout*, *animation* (blocking and refinement), *crowd*, *techanim*, *effects* and *lighting*.

Across the two pipelines there are bridges that enable the two to run in parallel. In fact, when the layout rig is ready for an asset, this can already be placed in the scene through the layout process in the shot pipeline. Figure 2.4.2 shows how the two pipelines work together. When the animation rig is finally ready, this should enable animators to move forward in the pipeline.

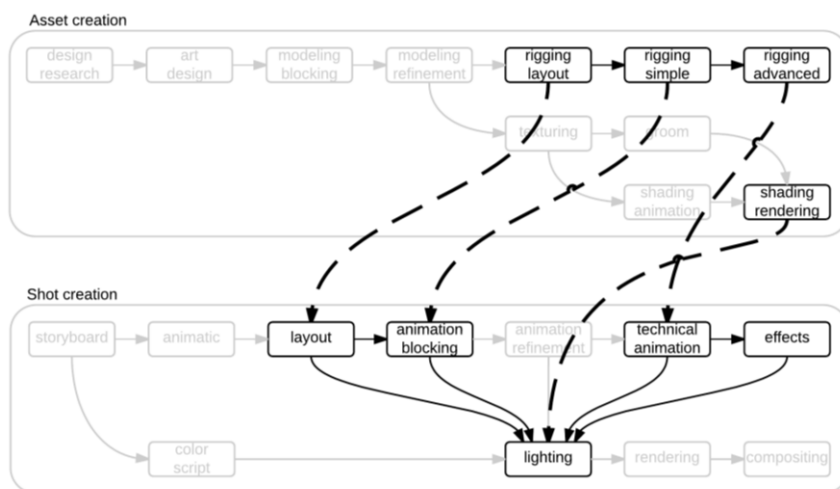


Figure 2.4.2: As two graphs can run concurrently in different departments, it is important to remark that they can plug into each other to enable artists to work at different levels of the production, avoiding delays caused by waiting for pipeline completion. In the figure, dashed lines represent interactions of processes across two workflows.

2.5. Data flow strategies

Data has to flow through the pipeline to feed the connected processes. To manage this flow, it is important to flag data to ensure that only the right asset versions propagate to further stages. Thus, it is necessary to define two flags for asset versions:

- *Latest*: it denotes the latest delivered version of an asset;
- *Approved*: it is assigned to any version that was reviewed and approved to be moved along the pipeline.

For example, whenever a new version of a model is approved, this has to flow towards the rigging process so that the new geometry is taken into account by the assigned rigger. Thus, defining how and when data has to flow across processes is a critical issue for production as it directly relates to the availability of assets to artists. This becomes evident when considering also automatic processes triggered on delivery; if a new animation version is approved, it will not be available for the effects process until its geometry cache is not generated.

There are two possible strategies for data flow: *push* and *pull*. If we consider two processes *A* and *B* with the asset *x* being delivered by *A* and used by *B* and asset *y* being delivered by *B*, we can describe the strategies as follows:

Push. When a new delivered version of *x* in *A* is approved, *B* is automatically updated and a new version of *y* is delivered.

Pull. When *B* gets updated, the system looks for any newer approved version of *x* and, if found, it is retrieved and a new version of *y* is delivered.

2.6. Task management

Each process in a workflow requires to be assigned to an artist and be tracked until its completion. This happens by the creation and

management of tasks. Tasks are typically managed through a DPM system and enable producers, managers and coordinators to define all the steps that will bring a production to delivery under the time and human resources constraints. Sometimes, it is even necessary to hand the process over to a different artist, which also needs to be tracked properly.

As tasks are related to production processes to happen on assets, the DPM and DAM systems have to be bound together and enable the creation of tasks only on registered assets entries. In this sense, the tasks should mirror the sequence of processes required by the production workflow for the asset to be completed.

A crucial aspect of task management is the control of the task status. Tasks can go through a series of status during the production, depending on the progress. The minimum set of status is:

- *Not Assigned*: a task is being created but no user is assigned to it;
- *Waiting*: a task that is assigned but is not planned yet to be started;
- *Pending*: a task that is scheduled to be started but has not yet commenced;
- *In Progress*: the work on the task is being carried on;
- *For Review*: a version of the asset produced through the task is ready to be reviewed;
- *Completed*: the task has been completed (and the resulting assets approved);
- *On Hold*: a task that needs to be paused for reasons such as waiting for other assets or changes in production priorities;
- *Cancelled*: a task that is not required anymore.

Depending on the processes adopted in a company the above list of status may be extended, but this is the minimum set that I identified as crucial in my work in production.

2.7. Tying all together

To implement an efficient CG media production pipeline, it is important to have a comprehensive pipeline, integrated within a flexible workflow. The flexibility of the workflow is provided by the granularity of its processes and the specialization of clusters of processes to support a defined scope. If this happens, it is possible to redefine the workflow on a per show or per asset basis, adapting as much as possible to any production need. The pipeline has to deal with the data properly within the workflow, ensuring that all the nuts and bolts of each process are treated as assets. The outcome of a process can be made of multiple assets and these assets can be all or partially used by another process, also in conjunction with assets from a third process. Thus, we define the processes as the workflow's working units where existing assets are used as input and new ones are made for later use. A comprehensive pipeline is what manages this extra layer of granularity defined by the assets of all processes.

The downside of the assets granularity is the amount of data that is produced, as it becomes hard to retrieve and deliver it properly. This is a problem that can be solved using DAM systems that embed metadata into assets, making them easily retrievable and properly versioned. Furthermore, the use of a DPM can make visible only the data that matters to users. This way, an artist should see and access only the assets that are related to tasks assigned to him/her. Of course, every artist has the possibility to access any asset if necessary.

We can say that once the workflow is defined, the pipeline to manage the data is defined. The production elements (scenes, shots, characters, textures, etc.) are defined and their entities registered in the DAM system. For each element, a workflow is defined (this can be done using pre-made workflow templates), which then populates the tasks in the DPM system. The ability to define ad-hoc workflows depending on the element is key to a flexible overall production workflow, as it enables to define a detailed set of processes but selectively choose the ones involved in each case. Figure 2.7.1 summarizes this idea in a graph.

With the DPM and DAM working on the same dataset, it is then possible to drive the pipeline, by generating new assets through triggering automated processes depending on changes tracked by the DPM. For example, whenever a newer asset version is approved, this could automatically trigger the creation of newer versions of assets (even in different processes) that depend on it. This way, it is possible to implement an intelligent push strategy, where updates happen only on approved asset versions.

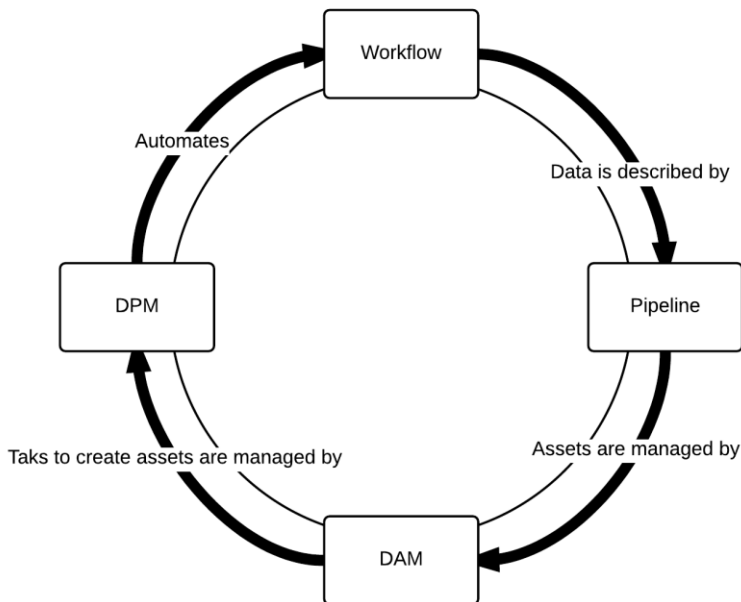


Figure 2.7.1: Graph showing the integration and interaction between workflows, pipelines, DAM and DPM systems.

Finally, it is important to consider always a certain level of flexibility and enable the pipeline to be broken or hacked. Media production is a challenging and ever evolving domain, where new requirements are defined every day. Thus, it is important to give developers the ability to force the workflow in directions that were not intended originally. It is crucial to design the pipeline which can take into account rapid patches and consider these changes as a prototype of some proper pipeline changes to come next.

2.8. Conclusions

In this chapter I presented some state-of-the-art as a result of collecting all the available information from existing software, books, conferences and websites. I have also defined a set of essential elements for a pipeline and showed how to use them to define a complete production pipeline. Furthermore, I discussed how data can flow through a pipeline and described how pipeline, workflows, DAM and DPM systems can be integrated to provide a reliable and flexible production environment.

The contents of this chapter are a direct result of several years of experience in the industry. The described pipeline and workflow is the outcome of various iterations to improve pipelines to produce contents varying from short films, videos for public events, TV programmes, internet videos and films. While the work focuses on the CG aspect of media production, it can be extended to incorporate other types of media.

This work does not cover the complexity related to transferring data across different locations and maintaining the versions of assets consistent. This is now a big challenge for large media companies and should be investigated in the future.

To conclude, it is important to say that it is practically impossible to design the perfect pipeline or workflow, but then it needs to be thought in a way that it makes previous decisions understandable and new changes easy to integrate while the production is running: “So a visual effects pipeline is not exactly a production line. It is more like the world’s longest Scalextric track. You race along it, switching lanes and doing loop-the-loops. Every so often you realize that you need to change the layout of the tracks, but you are rarely allowed to stop the cars. You can imagine Tom Cruise in one of those cars.” [Dunlop, 2014, p. 36]. I believe that the contents of this chapter provide a sufficient understanding of the important elements of a pipeline and how they relate to a production workflow, DAM and DPM systems.

2.9. References

- Arnold. (2015). Available from: <<https://www.solidangle.com/arnold/>>. [18 December 2015].
- Avid. (2013). Available from: <<http://connect.avid.com/MAM-ROI-Whitepaper.html>>. [23 December 2015].
- Burley, B. & Lacewell, D. (2008). Ptex: Per-Face Texture Mapping for Production Rendering. *Proceedings of the Nineteenth Eurographics conference on Rendering*. Eurographics Association, Aire-la-Ville, Switzerland, pp. 1155-1164.
- Chung, H. J. (2011). 'Global Visual Effects Pipelines: An Interview with Hannes Ricklefs', *Media Fields Journal*, no. 10.
- Dittmer, J. (2012). Bringing order to the creative department using metadata-driven workflows. *Journal of Digital Media Management* vol. 1, no. 2, pp. 176-182. Henry Stewart Publications LLP, London.
- Dunlop, R. (2014). *Production Pipeline Fundamentals for Film and Games*, Focal Press, New York.
- Johnson, C., Tobiska, J., Tomlinson, J., Van de Bosh, N. & Whaley, W. (2014). A framework for global visual effects production pipelines. *ACM SIGGRAPH 2014 Talks*, ACM, New York, article 57.
- Journal of Digital Media Management. (2015). Available from: <<http://www.henrystewartpublications.com/jdmm>>. [13, December, 2015].
- Lewis, J.P., Corder, M., & Fong, N. (2000). Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. *Proceedings of 27th annual conference on Computer graphics and interactive techniques*. ACM, New York, pp. 165-172.

Meeres-Young, G., Ricklefs, H. & Tovell, R. (2010). Managing thousands of assets for the prince of persia city of Alamut. *Proceedings of ACM SIGGRAPH 2010 talks*.

Noa&Max. (2013). Available from: <<http://www.noamax.tv/>>. [27 December 2015].

Noteboom, R. (2013). Configurable-WorkflowManagement. Available from: <<http://www.southpawtech.com/wp-content/uploads/2013/05/configurable-workflow-management.pdf>>. [27 December 2015].

Open Shading Language. (2015). Available from: <<http://opensource.imageworks.com/?p=osl>>. [18 December 2015].

Polson B. (2014). CG pipeline design patterns. *Proceedings of the Fourth Symposium on Digital Production*, pp. 29-29.

PostgreSQL. (2015). Available from: <<http://www.postgresql.org/>>. [27 December 2015].

Ricklefs, H. (2013). Pronto: scheduling the un-schedulable. *ACM SIGGRAPH 2013 Talks*, ACM, New York, article 29.

RV, computer software. (2015). Available from: <<http://www.tweaksoftware.com/products/rv>>. [13 December 2015].

Shotgun, computer software. (2015). Available from: <<http://www.autodesk.com/products/shotgun>>. [13 December 2015].

SIGGRAPH 2015, Global VFX Pipelines. (2015). Available from: <<http://s2015.siggraph.org/attendees/birds-feather/sessions/global-vfx-pipelines>>. [13, December, 2015].

Yasur, J.R. (2013). Metadata management in the entertainment industry: A case study from the studio perspective. *Journal of Digital Media Management*, vol. 2, no. 2, pp. 111-119. Henry Stewart Publications LLP, London.

3. FULL PRODUCTION TRACKING: INTELLIGENT PIXELS, RENDERING AND MANAGEMENT

This chapter describes innovative production systems that take advantage of engineered pipelines, workflows, production and asset management systems. Because the data is properly registered and managed, it is possible to connect different systems, sharing such data or even build newer systems to show, edit and control the data.

I started this work in 2010 as an extension to the work carried out to formalize the production pipeline and workflow described in the previous chapter. By improving the way data is managed in production, the goal is to improve several aspects:

- provide a global overview of the production
- increase the traceability of assets across their life
- re-use assets
- make assets more accessible to artists

The work described in the following 4 sections was carried out in different companies and was strictly related to the production needs of the moment. The evaluation of these works is thus directly related to their use in production, the improvement of the overall productivity and the easing of the workflow for production coordinators and artists.

In 3.1 I present recently published work [Romeo et al., 2015] done at the Moving Picture Company (MPC) to improve the quality of the daily renders and reduce the time required to compute them. The system, called *Renderflow*, reaches these goals by integrating the company Asset Management System and workflows into one centralized solution. This way, all assets from a shot can be rendered using state of the art renderers (e.g. Mental Ray or RenderMan) and, most importantly, the resulting renders can be assetized for later reuse. The reuse is a key feature of the system. We use a compositing technique that merges together different images by sorting their pixels by their distance (or depth) from the camera. This makes it possible to render each asset on a separate layer and compose it on a later stage. This way, each layer gets

assetized and can be reused if no changes are made to the assets contained in the layer.

When considering shots with many complex assets (numbers can go up to 200 and above), the changes on one single assets will result into a new high quality render very rapidly with the proposed solution.

Section 3.2 describes *a system that integrates the Production Management System into a media player*. This way, artists can easily bind videos and images to their tasks or assigned assets. In extensive media productions, the amount of tasks and assets can become very large, meaning that navigating the Digital Production Management System (DPMS) can become tedious for the artists. At the same time, the number of tasks and assets also affects the amount of dailies (videos or image sequences) created, complicating the browsing and access of the data.

In this section I show a prototype that was designed at StudioNest in 2010 to overcome this problem. The system integrates Autodesk Shotgun [Shotgun, 2015], RV [RV, 2015] and the studio pipeline to give artists easy and personalized access to the relevant information when delivering work. The system was named RVGun and during an evaluation with five industry professionals it proved to reduce the time taken to perform the related tasks by 61%. It also aroused the interest of the software companies (Shotgun and Tweak Software) developing the integrated products. These companies have now been bought by Autodesk. Some comments after I demoed the system in 2010:

“This is going to be extremely valuable to our clients, so I'd like to talk about how we can make a big release out of it.”
(Don Parker, Senior Director, Shotgun at Autodesk)

“This is a really great addition that is filling a missing piece in the whole workflow.” (Kevin Porterfield, Senior Software Engineer, Shotgun at Autodesk)

“...really looks like a large chunk of what's been missing for people who really want the 'tracking-aware playback'”

dream.” (Alan Trombla, Senior Software Engineering Manager)

MPC published later the ReviewTool they had developed [Fagnou et al., 2013], which is based on the same ideas. This software uses the same approach and extends it further by also providing a pipeline-based editing User Interface (UI), which enables users to see shots in different contexts. Contexts can vary from simple editing cuts to the different stages of the different departments.

In Section 3.3 I describe a production tool that integrates the ideas of production management, pre-production and digital pinboards. As production management commonly happens through Excel-like tables, it is hard to browse information rapidly without filtering the data. Unfortunately filtering the data reduces the context. Moreover, data sheets do not present information in a way that is visually meaningful.

The *Production Pinboard* as a new way to access pre-production data. Through the software it is possible to define shots and assets, arrange them in meaningful ways and visually navigate and edit the data without losing context.

The work was done during 2014, during my stay at Bee and Birds animation studio in Barcelona. It was not extensively tested in production due to the closure of the studio but its development was informally evaluated and driven by the person who is now Layout lead at MPC.

Finally in Section 3.4 I describe a method that enables Asset Management data to be embedded in rendered images. Three-dimensional scenes are described by a live hierarchical structure (or scene graph) [SIGGRAPH2013 USD Presentation, 2013] [Brutzman & Daly, 2007]; because of this, it is easy to tag different parts of the scene in the Digital Asset Management System (DAMS). Unfortunately, images commonly have a flat structure that does not allow to hierarchically cluster pixels, making it impossible to properly add per-asset metadata. As a result, once a frame is rendered, it is impossible to retrieve the asset that was used to render a certain pixel and its data.

I propose a method that encodes a link to the DAMS entry of each asset rendered in a pixel of an image. By generating an extra pass that writes the asset code ID in the image, it is then possible to query a pixel to get the ID and retrieve all the asset information from the DAMS and task information from DPMS.

I developed this technique in 2014, while working at *Bee and Bird Animation Studio*, but the sudden closure of the studio made it impossible to carry out its proper evaluation, which would require extensive use in production and professional users enquiries. With renewed opportunities at MPC I am now designing an evaluation strategy prior to its publication.

3.1. Intelligent rendering of dailies: automation, layering and reuse of rendered assets

a) Introduction

In visual effects production the amount of digital assets to be created is constantly increasing. These assets can be two dimensional images, three dimensional geometries or animated scenes with multiple characters, large scenarios and complex shading. In order to properly review the creation process for all these contents, it is necessary to rapidly generate accurate previews that show the progress and enable supervisors to provide valuable feedback. Unfortunately, given the complex nature of these assets, it is difficult to manually generate such previews, with the quality required to provide meaningful feedback in time for daily reviews.

From a technical standpoint, it is important to consider that MPC's pipeline is designed to work as a multi-layered reference tree of several assets of different types. Then, one asset is defined as a hierarchical structure of interconnected and interdependent assets [1, 3]. Because of this, it is important to ensure that, whatever version of an asset is rendered, all of its dependent assets (characters, animations, cameras, etc.) are properly sourced.

In this paper we describe an automated system called "RenderFlow" that takes advantage of our digital asset management system to

rapidly generate high quality renders of complex scenes by performing an automated gathering of required assets and reusing existing renders to save processing time. The system is integrated within our visual effects production pipeline and takes advantage of several commercially available products (e.g. Autodesk Maya and Pixar’s RenderMan).

b) Design overview

Our system is designed to build a dependency graph of nodes defining how to gather and render all the desired assets (figure 3.1.1). Once the main asset to be rendered is defined (e.g. a shot), the system queries our digital asset management system to retrieve all the dependent assets which are then configured in clusters (defined by the user). For each cluster the system builds one gathering node per asset and one node to render the cluster itself. It also creates one node to compose the output renders together and finally convert the comped render into an assetized daily.

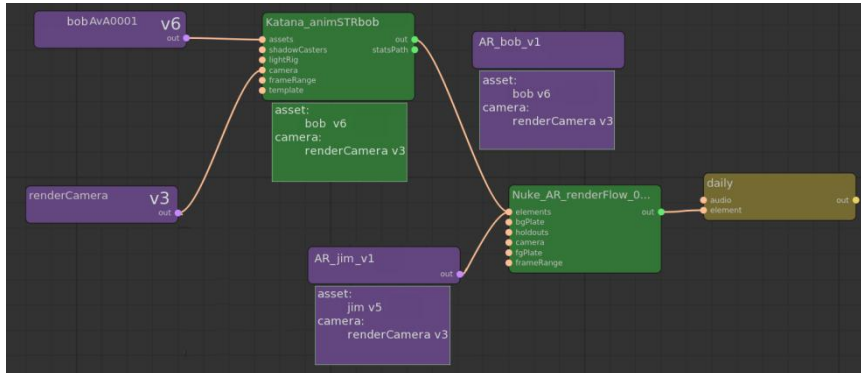


Figure 3.1.1: In this graph, two animated characters (bob and jim) need to be previewed with a certain camera. Version 6 of bob is new and will be rendered, while version 5 of jim was already rendered with the desired camera and will be reused. The render of bob is assetized for further use.

Each node triggered by the system runs using a different external process that can make use of in-house development, third-party software or both. These processes run over our studio render farm and can be parallelized, as different asset clusters can be rendered on different boxes at the same time.

One key feature of the system is the reuse of rendered asset clusters. Most of the time, scenes do not change completely; only small parts (and their dependent assets) may change through the day. In this case, everything that has not changed since the last render will be reused and comped, while only the recently modified parts of the scene will be gathered and rendered. Other approaches are also adopted to ensure that each asset is rendered only when it is visible through the camera by evaluating the assetized meshes and geometry caches against the camera frustum.

When required, rendering of dependent assets can be performed through different engines. We currently use both RenderMan and Mental Ray, but our system is easily expandable to use any other render engine capable of producing a depth pass.

The different layers are comped using their z-depth information. This ensures a fast and reliable process and enables the implementation of other effects, such as depth of field, screen space ambient occlusion [4] and fog. Deep compositing could also be performed to ensure proper compositing when dealing with transparencies and volumetrics, but its use would be limited, due to the large storage resources it requires.

Designed to be a core API to be used by developers and TDs to build their own specific tools, the system allows scripting of custom processes. By injecting python code into the system's nodes, it is possible to force the processes to perform actions that are specific to the needs of a certain department and are not meant to be general. One example is the rigging department, where camera and characters have to be transformed so that characters are always visible in the render.

c) Conclusions and future work

Our system enables us to rapidly perform and reuse renders of various types of assets, at different qualities, depending on the specific production needs. In modeling we render quick turntables of the assets; in rigging we daily animation clips on new rig versions to evaluate their proper functioning; in animation we do

quick grayscale previews of characters and more complex dailies with textures and draft lighting.

The reuse of existing renders improves our performance as complex scenes with hundreds of characters can be rendered in minutes if only few assets have changed.

One critical limitation of using separated layers is the lack of inter asset shadowing. This is a big issue when producing dailies for lighting as simple contact shadows (e.g. screen space ambient occlusion) lack essential details. Thus, heavy clustering of assets is currently the only way to obtain complex light interaction (shadows, colour bleeding, etc.).

We are currently working on a new method that will enable us to obtain most of the required shadows as a post-process and take full advantage of the proposed layered approach.

We are also developing a companion system to access the dailies information from the comped images, to enable visual navigation of the rendered layers and query relevant information about their content.

Finally, we are designing a render forecasting framework to allow our system to approximately predict the amount of resources required by every render and intelligently build the assets clusters to properly use the available resources.

d) References

[1] G. Butler, A. Langlands, and H. Ricklefs. A pipeline for 800+ shots. In Proceedings of ACM SIGGRAPH 2008 talks, 2008.

[2] N. A. Dogson. Going to the movies: Lessons from the film industry for 3d libraries. In 3D Research Challenges in Cultural Heritage Lecture Notes in Computer Science, 2014.

[3] G. Meeres-Young, H. Ricklefs, and R. Tovell. Managing thousands of assets for the prince of persia city of alamut. In Proceedings of ACM SIGGRAPH 2010 talks, 2010.

[4] M. Mittring. Finding next gen: Cryengine 2. In Proceedings of ACM SIGGRAPH 2007 courses, 2007.

3.2. Full tracking dailies

In production artists create a large quantity of dailies. These need to be registered in the Production Management Systems so that coordinators can review the progress of production and supervisors can access and evaluate new results. To find the right entry in the DPMS to bind the daily to is a difficult task if the artist has many tasks assigned. Furthermore, once the entry is identified, it is commonly a tedious process to link the daily to the task, having to browse a commonly packed file system in search for the right path and file. Finally, the user also has to fill all the extra data as tags and comments.

While this process is necessary to keep track of the production, it is possible to use the DPMS to build specialized tools for artists to use.

a) RVGun

The RVGun integrates the Shotgun DPMS within the RV media player. Users can load a daily within RV and use the RVGun (figure 3.2.1) to select the right tasks the daily should refer to. The user can also specify custom metadata.

The available options in RVGun are:

- *project*: the name of the production;
- *link*: name of the asset the daily is for;
- *discipline*: the stage (or department) the daily is meant for;
- *task*: name of the task (one discipline can have different tasks);
- *description*: a short description of the daily;
- *tags*: some tags to help the search and categorization of the daily.

RVGun populates all the possible choices reviewing all the projects, assets, disciplines and tasks the user is assigned to. This way it is simpler to select the right information.

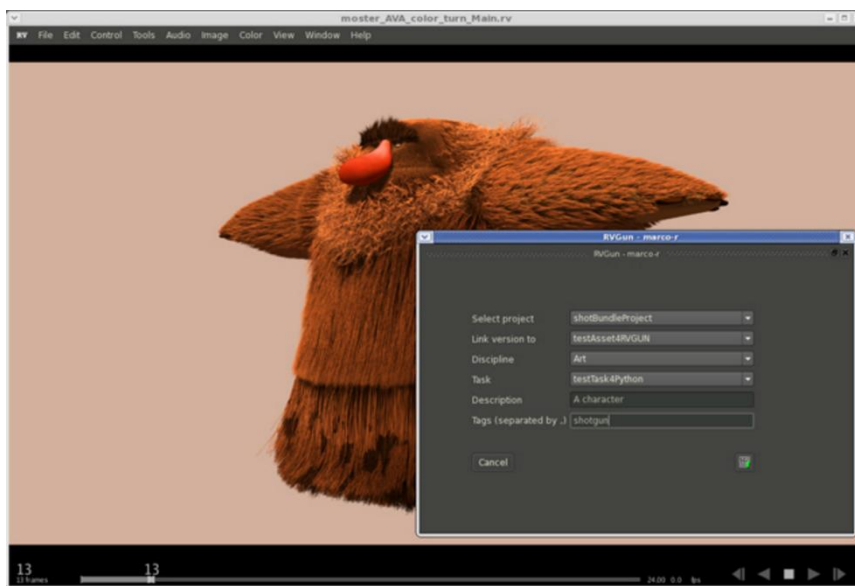


Figure 3.2.1: A screenshot of the RVGun interface (left upper corner). The user is presented with a list of possible choices that are suited to his/her tasks in the production. The available options are appropriate to the user task thanks to the production management system, helping the user reduce time and possible mistakes.

When the user is satisfied with the selected options, RVGun generates a video that fulfills basic production requirements as resolution, codecs and compression quality and has a slate with the relevant information (figure 3.2.2). Once created, the video is automatically registered into the DAMS. All the custom data is also filled into the asset and task entry and all the information, including the daily itself, is now accessible through the DPMS (figure 3.2.3).

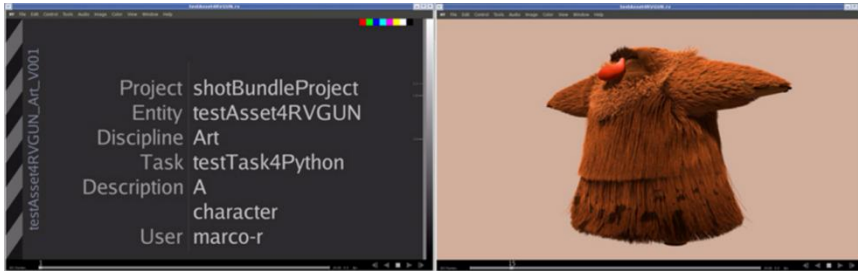


Figure 3.2.2: The quicktime video is created with a slate that summarizes the main information about the daily.


Thumbnail	Version Name	Link	Artist	Description	Status	Path to frames	Date Created
	testAsset4RVGUN_Art_V001	testAsset4RVG...	 Marco R	A character	👍	./mnt/cosmo2/pipe_1	Today 06:15pm

Figure 3.2.3: After all the fields are filled and the user submits the work, the asset management system registers the entry. This is now accessible from the production management system (in this case Shotgun).

b) Evaluation

To understand the possible impact of the RVGun in production, at StudioNest the company selected five senior artists (all had previous experience in feature animation and VFX international companies) to review and evaluate the process. In the test the artists were asked to register two dailies each, first using Shotgun and then using RVGun .

During the test I recorded the time spent for each required step using each system. The time spent dealing with the interface without realizing any useful process is also considered (“others”). The resulting total times are provided, including the time that the process took without considering the time to encode and upload the video (quicktime) on the studio storage. Finally, I collected any comments from the user about the process. The resulting times are collected in table 3.2.1 (Shotgun) and 3.2.3 (RVGun).

After the process was concluded, the production manager reviewed the resulting entry in the DPMS and checked for errors. The errors report can be seen in table 3.2.2 (Shotgun) and table 3.2.4 (RVGun).

Shotgun	U1	U2	U3	U4	U5
Time (seconds)					
locating daily	21	46	30	60	35
locating element to version	6	173	80	115	47
writing description	2	8	3	5	9
writing tags	2	0	19	28	0
uploading thumbnail	28	56	8	11	15
generating quicktime	35	180	62	40	40
uploading quicktime	60	120	60	50	80
others	103	150	90	100	80
total	257	733	352	409	306
total (no quicktime)	162	433	230	319	186

Table 3.2.1: Timing details for the operations required to register a daily on a task with Shotgun. The times varied a lot across the users but it is interesting to notice that they spent similar time (104 seconds in average) in “others” actions which were not required.

Errors (Shotgun)	U1	U2	U3	U4	U5
file location	✘	✓	✓	✘	✓
thumbnail	✘	✓	✓	✓	✓
quicktime	✓	✓	✓	✓	✓
description	✓	✓	✓	✓	✓
tags	✘	✘	✘	✓	✘

Table 3.2.2: Mistakes (denoted by the symbol ✘) occurred when using Shotgun to register daily. Tags were almost omitted and the file location is wrong in 2 cases out of five. Having a wrong file location is a big issue in production as supervisors would review the wrong daily, could generate confusion and will finally require to repeat the process.

The comments gathered through and after the process are mostly related with the difficulty of navigating the file system to search for the right path and file to link:

- *I hate linux*
- *I like to copy paste, not to type-in*
- *You should be able to browse for paths*
- *A file browser for "path to frames" would help a lot*
- *Where did I save the quicktime?*

This fits with the reported errors (table 3.2.2) where we saw that four out of five users mistakenly identified the file to link. This is a major issue as the linked daily is not the correct one, meaning the whole process did not produce the intended result and will have to be repeated. Moreover, if not detected, this issue can cause delays in the production as supervisors would review the wrong daily.

Some comments also gave hints of why the time spent in “others” is so big:

- *There is no thumbnail field*
- *I always forget how to use it*

These comments relate to the user getting lost in the interface of the Shotgun system and trying to remember what to do next or browsing around looking for the right button or field.

RVGun	U1	U2	U3	U4	U5
Time (seconds)					
locating daily	20	60	33	15	24
locating element to version	32	52	45	40	35
writing description	2	27	10	5	11
writing tags	9	0	18	0	3
generate+upload+thumbnail	180	60	120	40	47
extra	0	30	30	20	0
total	243	229	256	120	120
total (no quicktime)	63	169	136	80	73

Table 3.2.3: In RVGun the times spent in the various processes are reduced. Most of the time is spent in generating and uploading the video and the thumbnail, which in this system, is not performed by the user. The time taken to write the tags is very low as the system already provides a set of default tags for the selected task.

Errors (RVGun)	U1	U2	U3	U4	U5
file location	✓	✓	✓	✓	✓
thumbnail	✓	✓	✓	✓	✓
quicktime	✓	✓	✓	✓	✓
description	✓	✓	✓	✓	✓
tags	✓	✓	✓	✓	✓

Table 3.2.4: All processes produced the expected results (✓) with RVGun.

The comments were very few, with the users focusing in using the tool and getting their work done rapidly. Only one user mentioned he would prefer to have less options to choose from: *“it would be cool if there weren't so many options to choose”*. At the same time, another one admitted that *“the progressive, step-by-step, appearance of data, eases the usage”*.

Across the two approaches I found that using RVGun reduced the time required for the task by an average of 61% (162 seconds faster). Considering the amount of critical mistakes made when using Shotgun and the lack of mistakes with RVGun (table 3.2.4) the improvement becomes even more relevant. If we scale these results over a whole company of hundreds of artists working on many tasks, it is possible to understand the impact a tool such as RVGun has on production time and budget.

3.3. The digital production pinboard

As we have seen in the previous chapter, the production pipeline for a film can be very complex and the common softwares designed to set, edit and track the production stages are mainly using spreadsheets. Spreadsheets are very good to rapidly edit information but it gives a poor representation of the complex interactions between different aspects of the production.

One common metaphor to visualize complex interactions between elements in different contexts is the pinboard [Jetter, 2010]. On a pinboard, information can be added rapidly, rearranged, removed and connections across items on the board can be traced. This provides a good overview of the interrelation across heterogeneous bits of information. Even so, physical pinboards suffer from limitations due to their finite space and the impossibility to store the status for later use. These can be overcome in a digital implementation.

I present a Digital Pinboard prototype to edit and track the pre-production workflow. It enables the user to create new data and arrange it in ways that intuitively relate it together. This data is synced with the DAMS and DPMS systems and enables users to

create sequences of shots and collections of art design and references for assets.

The creation of shots happens through the creation of sequential floating panels, similarly to what happens in a storyboard, with the addition of the ability to add metadata and important information as the list of assets to appear in the shot or the dialogues between characters. The art design and references are managed in a similar way.

As users are free to create new empty data on the pinboard, they can easily populate the board with placeholders to brainstorm ideas. In fact the proposed design accounts for a way to play with data without affecting the crucial set of data. Furthermore, the design enables for easily zooming in and out of the pinboard to intuitively change from a broad context (e.g. all shots in a sequence) to a very detailed view of one item (e.g. the list of actions in a shot).

Finally, as multiple users might be interacting with the same data, it provides the syncing logic required to work collaboratively.

a) Design

The data to be displayed and accessible through the Digital Pinboard can vary in quantity and typology. Furthermore, the data also requires to be structured properly to represent the hierarchical layering of the production stages; data is not only connected within the same stage but also across stages, at different levels of granularity.

First of all it is necessary to define how each bit of pipeline information has to be displayed and which properties or attributes require to be exposed. In this prototype, I have identified the following asset types and properties to be made accessible through the Digital Pinboard design:

Sequence	Shot	3dAsset
Code	Code	Code
Name	Name	Name
Version	Version	Version
Preview	Preview	Preview
Notes	Notes	Notes
File	File	File
Shots	3dAssets	Category
Edit	Actions	
	Dialogues	
	Sounds	
	Duration	

Table 3.3.1: Table of relevant properties for different types of assets.

The above information has to be provided in a compact dialog that allows the user to focus only on one kind of data at a time. Each board panel should at least provide information about its name, version and a preview but can also be used to define more detailed information as, for example, the list of assets in a shot (figure 3.3.1).

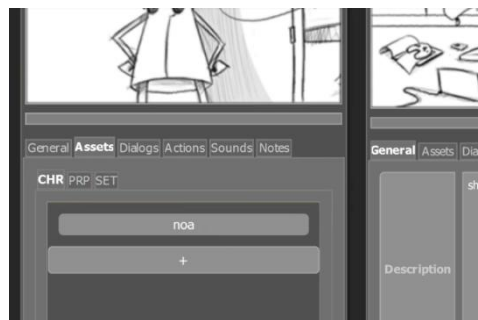


Figure 3.3.1: With the Digital Pinboard each panel in a sequence can describe multiple properties. One key feature is the ability to specify the 3D assets that have to appear in the shot and define their typology: character (CHR), prop (PRP) and set (SET).

By zooming out, it should be possible to easily see all the panels properly arranged together. The tool is designed to automatically arrange panels on a dynamic grid layout if required (figure 3.3.2). In fact, working on a shot sequence, it helps to have a nicely arranged

structure, while for art design and references it might be more interesting to have a free layout to play around with ideas. Anyway, the user can switch between the two solutions at will.



Figure 3.3.2: A complete sequence can be described as a set of panels. Each shot is described by one or multiple panels that contain all the relevant information for the pipeline and the artists. This information includes description of the shot, camera description, the list of assets, sounds, actions and dialogues. The panel also shows the status and the feedback for the shot.

In order to provide the right flexibility to give definitive structure to the data and also to be free to try new solutions, the tool is designed to have an active area and a brainstorm area. Panels can be placed in both areas; the ones within the active area will be assetized as enabled (e.g. actual shots in a sequence), while the others will be assetized but flagged to be disabled (figure 3.3.3). The user can move the panels inside and outside the active area at any moment to modify the active content of the board.

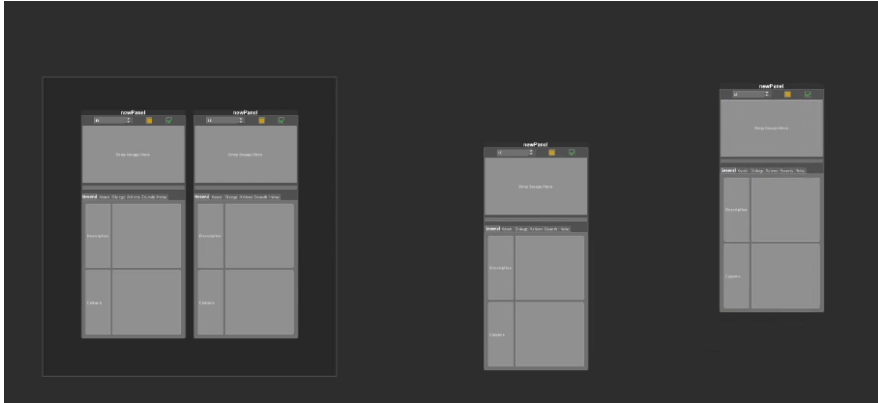


Figure 3.3.3: The active (left) area of the pinboard is the one that contains the active assets. The other region (lighter gray) can be used to experiment with data, waiting for it to be enabled.

The Digital Pinboard has to be considered a collaborative tool. Changes made on certain items or to the relationships between them may affect the work of others, when accessing the same data. It is then important to inform the user of the current sync status of the visible data with the database. The data could be matching on both the Digital Pinboard and the database but it may happen that the information was modified by the user, requiring him to push his changes to the database or by someone else, which would require the user to fetch the new modified information from the database. This cannot happen automatically as it would generate undesired versions of the information (when pushing) and possibly confuse the user or overwrite his/her work (on fetch). Thus, the sync status is displayed through an interactive icon suggesting the action that will be performed by clicking on the icon itself as a button.

b) Evaluation

As indicated earlier, the studio where this work was carried out had to close, and the production evaluation that I had envisaged could not take place. On the other hand its development was driven and informally evaluated by the person who is now Layout lead at MPC. Thus I expect new developments around these ideas, which might be published.

3.4. Render-embedded production tracking

In media production, to ensure quality and creative coherency, directors and supervisors commonly review frames or image sequences and provide feedback. When negative, the feedback might refer to some parts of the image that need to be modified (e.g. modify an object material or modify the animation of a character). Though, providing such feedback may become hard when the reviewed frames are very complex, containing lots of assets that are difficult to distinguish between each other. Moreover, the requested modifications may require a lot of work, thus it is crucial that the feedback is unequivocally provided.

To provide an efficient way to inspect the assets depicted in an image and bind feedback to them, we propose to encode the unique asset IDs from the DAMS in the rendered images. This enables the creation of tools to inspect and annotate the image, the assets used to render it and the production processes involved in the creation of the assets.

We describe a solution (figure 3.4.1) that integrates existing technologies to implement such set of functionalities and provide an advanced production tracking system in rendered images or what I will call render-based tracking. We also present a prototype for a review tool, describe its limitations and discuss possible solutions.

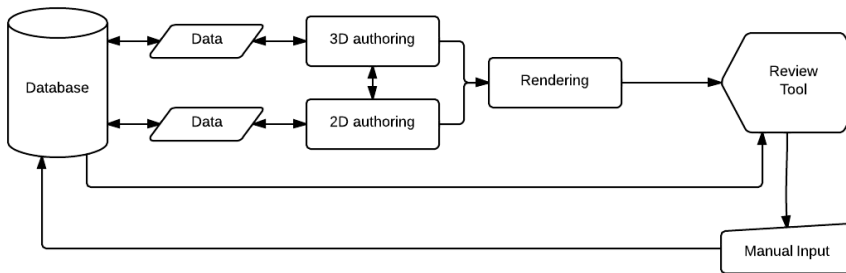


Figure 3.4.1: A schematic view representing the flow of data to embed tracking and reviewing assets in rendered pixels. The tracking data (assets' identification code) is managed from and into the database thanks to the DAMS and DPMS, then it is fed to the three-dimensional and bi-dimensional authoring softwares so that it can be embedded in the rendering process. This embedded data can be accessed through a tool designed to review frames and query the database to gather the information related to the tracking data of selected pixels. Finally, the

information can be reviewed and modified so that the database gets updated with the required changes.

a) Method

Render-based tracking can be obtained by propagating the assets metadata from the DAMS and DPMS into output rendered frames. Assets can vary from simple textures to shading materials and complex three-dimensional scenes.

Basic metadata (i.e. image name and creation date) can be stored into three-dimensional and image files using the standard metadata capabilities of existing file formats as FBX, OpenEXR, jpeg and MPEG-4. Enhanced metadata for rendered images should include per-pixel information of which asset is represented in the pixel, which shader has been used to lit the pixel and which textures were used by such shader. This information may be stored into an external XML file or directly into custom buffers of the resulting OpenEXR from the rendering process. Each entity in the DAMS and DPMS systems has a unique numeric ID that can be encoded in 24 bits colours (8 bits x 3 channels) and then stored in each pixel, creating a unique binding between pixels and database entries.

Once relevant data is set in the DAMS and DPMS through specific tools (figure 3.4.2), the metadata propagates forward thanks to extensions (scripts and plugins) for standard authoring applications (ie. Maya, Photoshop, etc.) that query the DAMS and DPMS systems, feeding the actual workspace with the according data. Figure 3.4.3 shows a Maya scene with assets retrieved from the database following the shot description in the DAMS.

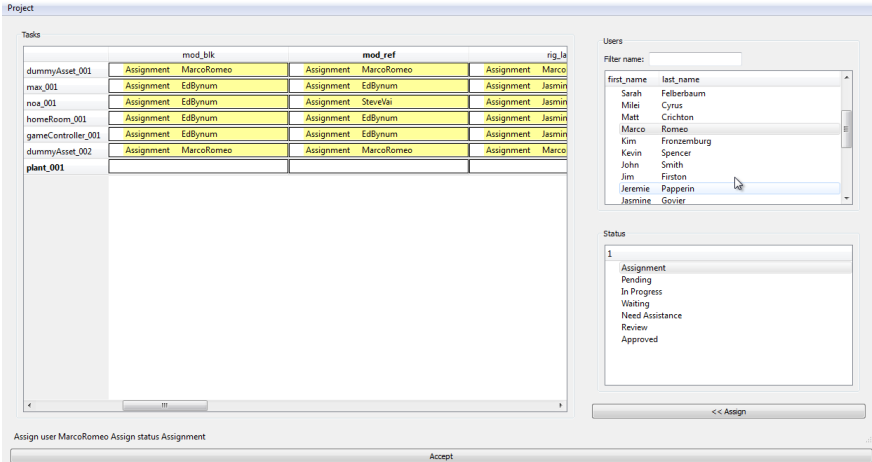


Figure 3.4.2: A custom tool can be used to define the assets in a shot and the required tasks for its completion. This information is stored in the DAMS and DPMS to be used later in the pipeline.



Figure 3.4.3: A shot animated in Autodesk Maya. The contents were loaded through a pipeline that generated a pass per each asset, encoding the asset's DAMS code.

The backward propagation is obtained through a custom review tool that reads OpenEXR images and binds each pixel to the DAMS and DPMS systems. This tool is designed to allow users to select objects directly from the image and annotate changes that will then be sent back into the DAMS and DPMS systems. It is designed to facilitate

the labour of supervisors to spot errors or undesired results and notify the interested departments for retakes. The strong binding with the DAMS and DPMS systems will also automatically create tasks for artists and stop any depending task.

The tool can open OpenEXR image files and, given the required passes (diffuse, specular, reflections, refractions, indirect, scatter, ambient occlusion and shadows) can apply equation 1 from section 2.3.3 to perform a precomp that computes a parameterizable image. This way, the user can modify the rendered images and obtain a quick preview of the required changes in the tool itself, without having to re-render the image (common rendering times for one frame may vary from minutes to hours). Furthermore, as the pixels are univocally associated to assets, changes can be applied only on selected object.



Figure 3.4.4: This figure shows a rendered frame from the shot, open in my prototype review tool. When clicking on a pixel, the tool extracts the code of the asset depicted in the selected pixel and retrieves all the relative production data from the DAMS and DPMS.

Unfortunately, the anti-aliasing applied on the image data produces inconsistent values at the edges between objects. In fact, if one pixel is shared by two assets, the embedded code will be somehow averaged between the values for the two assets. This produces ambiguities that might make the technique not reliable. The solution to this issue are Deep pixels, available in OpenEXR 2.0 [Hillman,

2013]. Deep pixels store the various samples used to compute each pixel in arrays, thus storing the DMS codes in Deep pixels would enable the system to select multiple assets for edge pixels and even select assets through transparent surfaces.

3.5. Conclusions

In this chapter I presented four solutions that build on top of DAM and DPM systems to tie together and help managing different aspects of a production workflow and pipeline.

In one case (section 3.2) we present an early evaluation, showing that a production environment can benefit from a system to help artists to bind dailies to database entries. The improvement in production was considered to be related to a reduction in time (61% faster) and a dramatic reduction of the mistakes made by artists: no errors made with RVGun, while 2 out of 5 users failed to deliver their work properly with standard toolsets. These results were also reinforced by artists' comments suggesting that standard tools were frustrating to them.

For the other tools, I do not present any evaluation yet. The work in section 3.1 is currently being evaluated to understand the benefits through a complex production in terms of reduction of time spent to produce dailies and overall usage of the renderfarm. The solutions presented in sections 3.3 and 3.4 are promising, as demonstrated by the support provided by experts. Their ongoing integration in large-scale productions will show soon results that will constitute their evaluation.

3.6. References

Brutzman, D. & Daly L. (2007). X3D: Extensible 3D Graphics for Web Authors, Morgan Kaufmann Publishers Inc. San Francisco.

Fagnou, D., Cameron C., & Valdez, A. (2013). ReviewTool: a database-driven visual effects editing application. *ACM SIGGRAPH 2013 Talks*, ACM, New York, article 28.

Hillman, P. (2013). The Theory of OpenEXR Deep Samples. Available from: <<http://www.openexr.com/documentation.html>>. [13, December, 2015].

Jetter, H.C., Gerken, J, Zöllner, M. & Reiterer, H. (2010). Model-Based Design and Implementation of Interactive Spaces for Information Interaction. *Human-Centred Software Engineering*. Springer Berlin Heidelberg, London, vol. 6409, pp 22-37.

Romeo, M., Auty, J. & Fagnou, D. (2015). Intelligent rendering of dailies: automation, layering and reuse of rendered assets. *Proceedings of the 12th European Conference on Visual Media Production (CVMP '15)*. ACM, New York, article 15.

RV, computer software. (2015). Available from: <<http://www.tweaksoftware.com/products/rv>>. [13 December 2015].

Shotgun, computer software. (2015). Available from: <<http://www.autodesk.com/products/shotgun>>. [13 December 2015].

SIGGRAPH2013 USD Presentation. (2013). Available from: <http://graphics.pixar.com/usd/s2013_presentation.html>. [13 December 2015].

4. CREATIVE AUTOMATION: MAKE IT MOVE

This chapter deals with part of my research contributions towards finding solutions to aid and automate the animation and rigging of virtual characters, one of the many processes within the context of CG production. I have investigated some issues that are known to be problematic and cause unnecessary delays in production. In this chapter I discuss:

1) *Automated full body animation*

My approach focuses specifically in tweaking walk cycle animations. This research was developed within the context of the i3Media project [Barcelona Media, 2010], and is oriented to find ways to automatically modify and apply walk-cycle animations to various bipedal characters of any proportion to represent their emotional state through motion. The work could be used to animate virtual characters for television programmes, videogames and crowd systems. The process of animating a biped character requires substantial manual intervention, and our research contributes to this field by creating a model for emotional character walking, and defining a series of proportional variables which can be changed to create different emotional walk cycles. As we avoid fixed values and work solely with proportions, the system implicitly retargets the data to any biped body shape, regardless of the size and structure of the skeleton. We used motion capture data to assign real-world values to the proportional variables, which are then used to procedurally create ‘emotional’ walk cycles.

The results were presented in *Articulated Motion and Deformable Objects (AMDO) 2010* [Romeo et al., 2010]. The previous approaches which are most similar to our research, in a sense, are Densley and Willis [Densley & Willis, 1997] and Meredith and Maddock [Meredith & Maddock, 2005]. Our approach combines their philosophies but underpins it with a more formal and focused theoretical background. We tackle the issue of walk-cycle animation inspiring us in traditional character animation techniques, applied within a modern computing and mathematical framework.

In recent research Aristidou et al. [Aristidou et al., 2015] investigate how to analyze and classify the emotion in human body motion by using Laban Movement Analysis features, whose body and shape components resemble the proportional representation of animation features in our work.

2) *Two automated facial rigging and expression contributions*

2.1) The MASKLE, which is an automated facial rigging system. Its main purpose is to provide a simple but functional deformation system for the face; in this way, it is possible to reduce the time spent by artists for the rigging task and ensure that the facial rig is coherent across characters, enabling reuse and automation. The work was developed partly within the SALERO European research project [Salero, 2006].

The work, published in the *International Conference on Computer Graphics Theory and Applications* (GRAPP 2009), is based on the encoding of deformation parameters (e.g. linear skinning weights) on a low resolution face template for future reuse. In fact, thanks to dedicated tools and a fitting algorithm, the template is easy to setup onto diverse faces and the deformation parameters can then be automatically transferred to make them animatable. As proved by the evaluation, the resulting automated skinning weights differ by 2.63% from hand-crafted ones. This idea was gestated as a solution to enable future automation of characters' faces; having a standardized facial rig, makes it possible to define techniques and algorithms for its animation and reuse.

The proposed solution has not been adopted by other recent works. Recent research still focuses on transferring shapes, rather than skeletal structures. However, the most successful approaches [Noh & Neumann, 2001], [Sumner & Popović, 2004], [Saito, 2013] require a large set of landmarks to compute mesh correspondences (between 50 and 100 landmarks are necessary for complex shapes). The process of setting landmarks is time consuming and prone to mistakes, thus an approach similar to the Maskle could be used to select a subset of key feature points and extract the other landmarks once the mask is fitted onto the character's face.

The work was also evaluated in three experimental productions within the SALERO European project, showing productivity benefits [Third Phase Experimental Productions and Evaluation Report SALERO, 2009]. The details with respect to two of them, which combined different technologies are analyzed later in this chapter. The remaining is more concerned with a full animation system. The Maskle was used to produce a video (approximately 3 minutes long) of a virtual character talking on television. The character's body is animated using simple animation clips, while the face of the character is animated by applying a set of poses onto the rig obtained with the Maskle. The system proved well suited to provide a working facial rig for humanoid faces and positions and it was the only available tool of this kind at that moment. Commercial products do not incorporate yet any similar tool/technique and the skinning algorithms have unfortunately not improved either, making the Maskle a still relevant contribution today.

As part of the evaluation, the MASKLE was also compared to the process of creating blendshapes for a facial rig manually in 3DS Max 2010 [3DSMax, 2015]. The estimated productivity increment was 80% and no tangible difference in qualitative comparison.

2.2) A Synthetic Model for Facial Expression. At the time, a lot of the productions of the industrial partners in the research projects undertaken lacked facial expressivity; the main motivations were three:

- Characters are difficult to setup to express emotions through the face;
- Animation of the expressivity of the face is hard to automate;
- The complexity of facial rigs to provide characters with expressivity limits the frame-rate of realtime applications and requires higher processing times for off-line productions.

All these three barriers could be summarized as increased human, software and hardware resources.

In this work we describe how a simple set of five facial shapes, plus the articulated deformation produced by the jaw bone, can be combined together to represent a wide range of emotions. We also propose that these combinations are arranged into a Whissell wheel [Whissell, 1989] [Plutchnick & Kellerman, 2015] (alternatively known as bidimensional valence-arousal emotional model) to easily set and interpolate different emotional facial expressions. This solved the barriers for facial emotional animation indicated above, as it made faces easier to setup (only six shapes to be modelled), simple to animate (thanks to the Whissell wheel interface) and fast to compute (only six blend targets to be interpolated).

The quality of the resulting facial expressions was evaluated through an online survey, which showed that most of the obtained emotions were easy to understand across three different countries. The work was also used and evaluated in the SALERO experimental production Alan01 [Tuomola et al., 2009][Alan01, 2010]. The use of this technique produced a mixed response from the artists working on Alan01, finding it useful and easy to use, but not flexible enough at times. On the other hand, 96% of the spectators visiting the installation who took part in a survey, rated the quality of facial animation as excellent, enlarging the positive results of the online survey mentioned above.

The work was not published at the time, but the new trends in real-time online 3d graphics [Evans et al., 2014] on the web and the renewed interest in modeling emotional expressivity for virtual characters [Hyde et al., 2014], [Faita et al., 2015] may set the ground for a publication.

4.1. A reusable model for emotional biped walk-cycle animation with implicit retargeting

This paper presents a reusable model for rapid animation of the walk-cycle of virtual biped characters, with implicit retargeting of motion capture to any character, regardless of dimensions. Despite modern software continuously improving the quality of automatic assistance, the process of animating a biped character still requires substantial manual intervention. Our research contributes to this

field by creating a theoretical model for emotional character walking, defining a series of proportional variables which can be changed to create different emotional walk cycles. We used motion capture data to assign real-world values to these variables, which are then used to procedurally create ‘emotional’ walk cycles. Due to the fact that we avoid fixed values and work solely with proportions, the system implicitly retargets the data to any biped body shape, regardless of the size and structure of the skeleton.

a) Introduction

Since the idea of technological convergence arose in the early 1990s, the media industry has consistently looked at systems that share resources and interact with each other, cooperating in order to create content in a more efficient and cost-effective manner. This search for more efficient systems is caused, in part, by the nature of digital media production, which remains a very labour intensive, high-risk and high-cost industry. One of the reasons for this is that productions are crafted, almost without exception, at very low levels, in order to better satisfy artistic needs. Indeed, in many applications, the existence of more sophisticated digital tools has actually pushed up costs, as more time is spent on complex off-line processes in the quest for quality.

An excellent example of this can be seen in the design and animation of virtual characters or avatars. These are used in many fields of the audiovisual industry, such as television, video games, internet, and mobile phones. Believable animation (both for bodies and facial expressions) is crucial so that such characters can properly express emotions, improving their ability to communicate. While this issue has been solved in the film industry and, increasingly, in the high-end video games industry, the same techniques cannot be applied to lower-budget productions, due to strong time and hardware resources constraints. Typically, time consumption could be due to the difficulties in creating a good animation rig for preparing handcrafted deformations that express a character’s current emotional state. Those difficulties directly affect the hardware resources as complex animation rigs or a high number of different deformations make real-time animations hard to be

replayed while maintaining the desired frame rate. In that sense there is the need for an animation methodology that can rapidly (and in real-time) animate a character by using fewer hardware resources and being sufficiently straightforward for animators to set up.

This paper presents one aspect of our research towards this goal, focusing particularly on the typical human walk-cycle animation. Our approach has been to view the problem from the traditional animator's point of view, attempting to apply well-established philosophies of hand-drawn animation within a modern computing framework. Thus, by studying the relationship between a character's apparent centre of gravity and the curvature of its spine, we present a new model for walk-cycle animation which classifies the possible walk-cycles into one of four separate gaits.

By studying this model, we define a series of proportional variables of body movement (e.g length of stride in proportion to length of leg). Motion capture data was used to provide real values for the defined variables. As the all the data is stored as proportions, this leads to the creation of an algorithm that allows the implicit retargeting of 'emotional' walk-cycle data to any biped with the correct skeletal structure. The result is real-time modification of a basic walk-cycle animation to allow the character to express a wide variety of emotions while walking. Crucially, our system modifies the walk-cycle while *maintains the underlying animation*, thus if a character is limping, the limp is maintained even though the system may tweak aspects of the animation to change the characters expressed emotion.

We demonstrate the system working both as a plugin for Autodesk 3DS Max (for use by animators) and as a self-contained C++ API, for use in custom real-time graphics engines.

b) Related work

Early efforts to control the animation of walk cycles were made by both Badler et al.[1] and Hodgins et al.[2], who use similar parameter and goal based techniques that are solved by the animation system. This idea of scripting character animation was extended by Perlin and Goldberg[3], who parameterized human ‘actions’ and blend the motion data to achieve combinations of movement based on different classes (gestures, stances etc.). A logic system prevents clashes between classes such that a character will not sit and stand at the same time.

The major achievement of these studies was to introduce the idea of abstract descriptions of motions, yet they made little effort to generate those motions (either via motion data or manually created animation). This problem was tackled from a more mathematical point of view by several studies[4][5] that attempted to generate movement based on interpreting the movement curves of the joints of the body. These concepts were further extended by the use of multiresolution filters to modify the ‘signals’ of the movement curves[6].

Grunvogel et al.[7] introduced the concept of *Dynamic Motion Models*, where the combination of abstraction of movement and procedural generation allows complex animations sequences to be rapidly created. A more general version of this is used by Abadia et al. [8] who use a dynamic timeline to cue animation clips, and provide the framework for automatic creation of such clips based on an overall emotional theme.

The issue of retargeting of motion captured data was tackled by Meredith and Maddock[9], who use weighted inverse kinematics to adapt motion capture data. This results in individual walk-cycle animation for each character, and enables rapid application of unique characteristics, such as a limp.

Research into procedural walk-cycle animation has differed notably from equivalent work into facial animation, in the sense that it has focused less on the use of emotions. In the facial animation field, perhaps the most cited work is that of Ekman[10], who specified the six basic emotions that can be deduced from facial expressions,

independent from cultural background. Densley[11] et al did introduce emotion to the characters appearance, by constraining joint angles based on the emotional state of the character.

In a sense, of all the previous research in this field, it is the work of both Densley et al[11] and Meredith and Maddock[9] that bears the most similarity to the work in this paper. In a sense, our approach combines their philosophies but underpins it with a more formal and focused theoretical background.

As mentioned above, our approach has been to tackle this issue using traditional character animation techniques, applied within a modern computing and mathematical framework. For an introduction to animation techniques and their evolution over the years, both Williams[12], and Johnston and Thomas[13] present the subject in detail.

c) Overview of approach

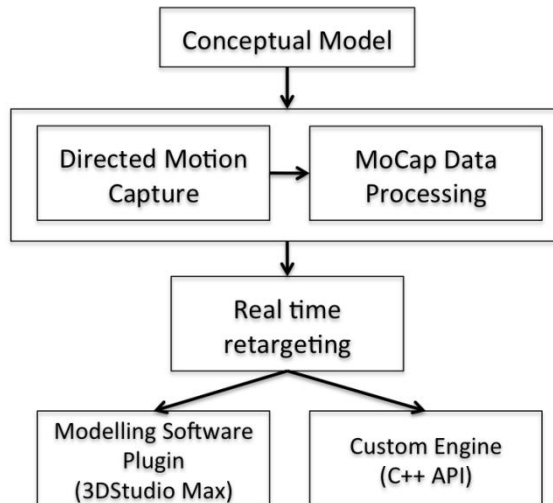


Figure 4.1.1: Diagrammatical overview of the system architecture.

Figure 4.1.1 is intended as an overview of our approach. A Conceptual Model (introduced in Section 4.1.4), based on traditional theory of animation, was used to guide and direct motion capture sessions involving male and female actors (see Section 4.1.5). The motion data from these sessions was processed (as described in Section 4.1.6) and used as an input for our novel retargeting algorithm. We have created two implementations of this algorithm, one as a plugin for the popular modeling and animation software, Autodesk 3D Studio Max, and the other as separate C++ API than can be used to retarget animation in custom 3D animation or games engine.

d) Conceptual model

In this section we present our conceptual model that describes how a character's walk cycle may express the character's emotion. Extending some of the principals of the basic walk cycle animation Williams[12], our model is focused on modeling how a character's general posture changes according to the current emotional state. The typical distribution of body mass situates the character's centre of gravity in the lower abdomen. Yet we base our model on the characters *apparent* centre of gravity – the area of the body that, in effect, guides the remainder of the body according to the principals of 'follow-through' animation and 'overlapping action' that are described by Johnston[13]. We use the phrase 'apparent' centre of gravity because we are not physically modeling any changes in the character's actual centre of gravity. In contrast, we are following the traditional animation route of abstracting the concept to a different level in order to allow the animation to better convey the required message.

Figure 4.1.2 shows the four basic poses (or gaits) that we define in our model, based on the location of the characters centre of gravity (COG). Each of these poses associates a particular curve of the spine with a centre of gravity position, which accordingly represents a different manner of walking. The model structures the poses to show the apparent COG dropping in height, from its highest point in the head, to its lowest point in the hips.

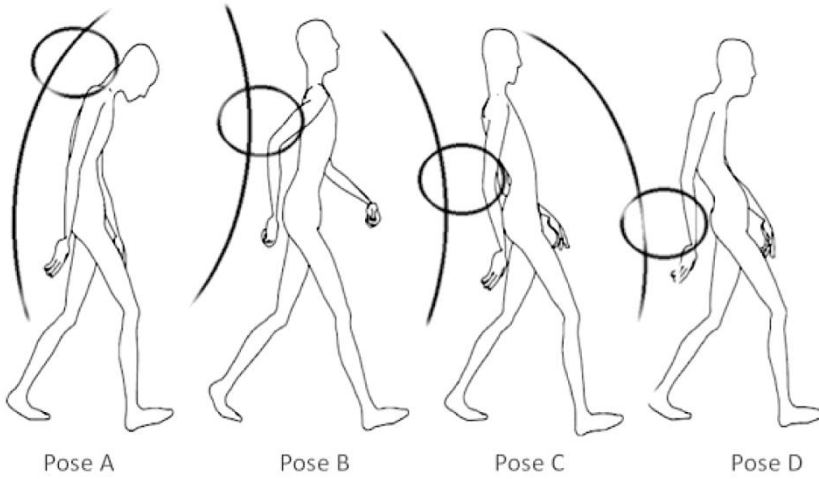


Figure 4.1.2: The four poses of our model. The circle represents the *apparent centre of gravity* – the area of the body that is leading the character while walking. The associated curve shows how the spine is curved in each pose.

Pose	Characteristics	Associated emotions
A	Apparent COG is head or neck. Spine curved into a 'C' shape.	Sadness, Depression Concentration/Worry Anger, Hurry, Fear
B	Apparent COG at chest height Spine opposite to Pose A	Happiness, Joy, Pride
C	Apparent COG at abdomen Spine curved as B	Satisfaction Relaxation, Serenity
D	Apparent COG dropped to Pelvis Spine curved back further than C#	Sensuality, Arrogance, Fear, Hurry

Table 4.1.1: The four poses of the model, and examples of associated emotions.

So that we could use this model practically in a computer animation system, we translated it into a **series of equations** that describe the movement of the body at each key-frame of the walk cycle[12]. Crucially, all the parameters of these equations are expressed as proportions relating joint position and rotation to the individual dimensions of each skeleton. The list of equations is as follows:

Wrist relative position:

- $wrist.x = wt.x - (ht.x + hw/2)/sw$
- $wrist.y = (wt.y - ht.y)/al$
- $wrist.z = (wt.z - ht.z)/ll$

Ankle relative position:

- $ankle.x = (at.x - (ht.x + hw/2))/hw$
- $ankle.y = at.y/al$
- $ankle.z = (at.z - ht.z)/ll$

Hip relative position:

- $hip.x = ht.x/hw$
- $hip.y = ht.y/ll$
- $hip.z = ht.z$

where **wrist**, **ankle** and **hip** are location vectors;

wt = wrist translation vector; **at** = ankle translation vector; **ht** = hip translation vector;

sw = Shoulder width (Euclidean distance between left and right shoulder);

hw = Hip width (Euclidean distance between left and right joints);

ll = Leg length (Euclidean distance between leg hip joint and ankle);

al = Arm length (Euclidean distance between the shoulder and the wrist).

For joint **rotations**, wrist, spine, pelvis and neck are taken into account. Neck and spine rotations are considered as a concatenation of local rotations.

These equations then form the core of the model, as by assigning different values to the parameters of the equations we can procedurally generate a variety of different animated walk-cycles.

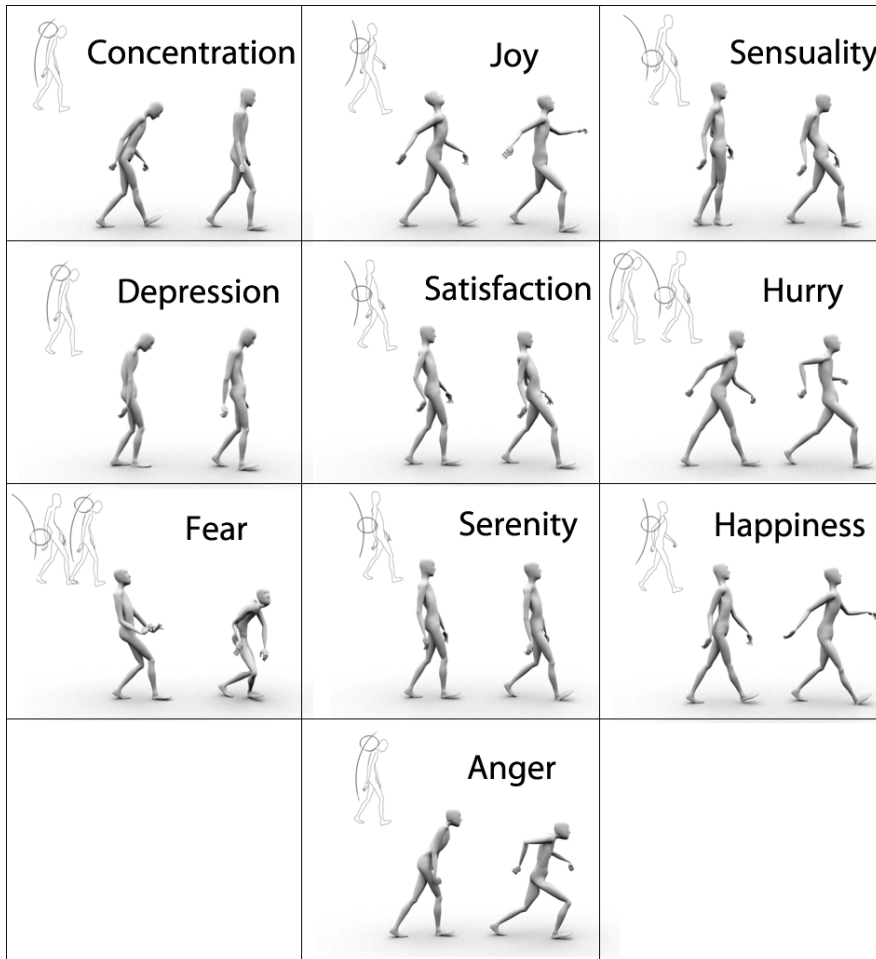


Figure 4.1.3: Still images of real motion capture data, grouped by emotion. Of the two shaded characters in each cell of the table, the one on the right is representing data from the female actress while the character on the left is the male actor. In the upper left corner of each cell is the equivalent pose from the conceptual model (see Figure 4.1.2).

While it is possible to fill the equations of Section 4.1.4 with randomly selected values, in order to generate meaningful procedural animation, it is necessary to base the values on some form of real data. To achieve this, we carried out a series of motion capture sessions, where male and female professional actors were carefully directed to walk in a manner that expressed the emotions identified from the different poses of the model (see Table 4.1.1). However, the motion capture director was careful not to actually instruct the actors the change their performance based on the model. At the time of recording the motion data, the actors were completely unaware of the model, and were merely told to attempt to express the requested emotion through their gait alone. Figure 4.1.3 shows example of the emotions recorded during the capture sessions.

e) Implicit retargeting of data

Following the motion capture sessions, the recorded data was analysed and the values for the proportional variables for the different emotions were extracted for the five key-frame poses of the typical walk-cycle animation, identified by Williams[12] (“Contact”, “Down”, “Pass”, “Up”, “Second Contact”), using the set of equations presented in Section 4.1.4.

This resulted in a matrix of proportional values that enable us to mathematically describe the emotions expressed by the actor during the motion capture i.e. a set of values for each emotion. By blending these proportional values into an existing virtual character walk-animation, we are able to transfer the aspects that make the motion captured data “emotional” to the virtual character. It is possible to proportionally blend the data, such that a character can be made to be “a little bit” sad, by blending in only a small percentage of the *Sad* emotion variable set.

Furthermore, there are two very important aspects that separate our system from anything that has been previously produced:

1. **The existing underlying animation of the character is maintained.** The system does not *create* the walk-cycle animation, but modifies the existing animation. This ensures that characters

maintain their personality, and any individual quirks added by the animator are not removed.

2. **The dimensions of the character are unimportant.** Because the system is based on the relationship between the proportions of the body, it works on a wide variety of body shapes, so long as the basic skeletal structure is that of a biped.

The equations presented in Section 4.1.4 allow values to be *extracted* from the motion capture data. To take those values and apply them to another character (thus carrying out the retargeting) we merely need to invert the equations to calculate the offsets to the existing animation. As our system only directly controls ankles, hips and wrists, standard inverse kinematics are used to ensure the remainder of the bones of the skeleton move in a believable manner. The side-advantage of this (other than convenience) is that retargeting is *independent of joint-chain length*.

Finally, the system implements time stretching to ensure that the resulting animation matches the average speed of the target emotions (i.e. a sad character should walk slower than a happy one).

We have created two software implementations for the system, a plugin for 3D Studio Max, and an independent C+ API. Screenshots of our plugin for 3D Studio Max are shown in Figure 4.1.4. The plugin allows an animator to easily incorporate the system into an existing walk cycle, using a slider-based GUI to proportionally add or remove aspects of each of the emotions. 10 emotions are explicitly mapped to sliders (those in Figure 4.1.3, derived from studying the conceptual model and from studying the literature[10][11][12][13]), by combining the sliders in different proportions, the animator can experiment until the walk-cycle is modified to their satisfaction.

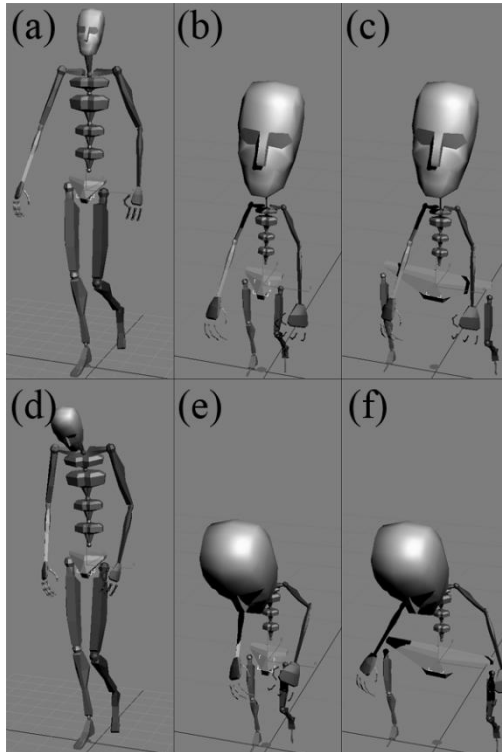


Figure 4.1.4: Screenshots of the results of the plugin developed for Autodesk 3D Studio Max. (a), (b) and (b) show a standard walk animation applied to characters of different dimensions. (d), (e) and (f) show the same characters with the ‘Sad’ emotion blended in. In (c) and (f), the characters hip joint has been scaled artificially to three times the width, yet the retargeting still works perfectly.

We have also extended the plugin into an independent C++ API that can be used to modify walk-cycles in real time. The API is designed to be used to be used with a variety of graphics and game engines, and we have tested it successfully with two such engines[8][14]. The API gives the developer the option to build in real-time changes to a character’s walk-cycle, thus allowing the character to respond directly to actions occurring within the specific application (be they from user input, or from internal factors that may affect the character).

Both the 3D Studio Max plugin and the C++ API are currently being used by our industrial partners in the creation of real-time animated productions.

f) Discussion and future work

The contributions of this paper are twofold. First, we present a conceptual model that has enabled us to define a series of proportional variables that describe the biped walk-cycle. Motion capture data was used to creating a matrix of these values for a series of different emotions. Second, this matrix enabled us to programmatically retarget the emotion-based capture data to any suitable biped walk-cycle, irrespective of character size or shape, and maintaining the underlying animation.

This approach of this system is different to those taken by previous authors. We are not attempting to adapt motion capture mathematically, such as Meredith and Maddick[9]. Neither are we attempting to create a global dynamic motion model such as Grünvogel et al.[7] Rather, we have approached the problem from the practical position of the traditional animator/artist, created a conceptual model, and then used applied modern mathematical and programming solutions to create the final system. This said, the manner in which it has been developed means that it could be easily combined with the weighted inverse kinematic approach of Meredith and Maddock, given that both approaches produce results that *extend* (or tweak) the existing animation, rather than completely replace it.

There are two typical applications of the work, reflected in the two implementations that we have developed. The first is in computer animation for film or television, where our system allows the animator to rapidly add, remove or combine specific emotions to a character, while maintaining the personality that they have created for the that character.

The second, and possibly more powerful, application is for real time situations such as in video games, or ‘serious’ games for developed for training or educational purposes. Currently in such applications, character emotional state is frequently heavily pre-scripted. Our system opens the door for dynamic linking of emotional walk-cycle animation to non-scripted factors that may affect the ‘emotional state’ of the character. This enables the character to behave in a more believable way, and reduces the amount of scripted animation work required for the application.

Our future work lies in integrating the system into with other methods in which a character can express emotion, such as static posture, gestures, and facial animation. While systems for both static postures and gestures can be generated using a very similar system that has been presented in this paper, facial animation requires a different approach, and this is the focus of current research.

g) References

1. Badler, N, Phillips, C.,Lynn, B.: *Simulating Humans: Computer Graphics and control*. Oxford University Press (1993).
2. Hodgins, J., Wooten, W., Brogan, D., O'Brian, J.: *Animating human Athletics*. *Computer Graphics* 29, 71—78 (1995).
3. Perlin, K., Goldberg, G.: *Improv: A system for scripting interactive actors in virtual worlds*. *Computer Graphics* 30, 205—218 (1996).
4. Witkin, A., Popovic, Z.: *Motion Warping*. *Computer Graphics* 29, 105—108 (1995).
5. Unama, M., Takeuchi.: *Fourier Principals for Emotion-based Human Figure Animation*. In: *International Conference on Computer Graphics and Interactive Techniques*, pp.91—96. ACM Press, New York (1995).
6. Bruderlin. A., Williams, L.: *Motion Signal Processing*. *Computer Graphics* 29, 97—104 (1995).
7. Grünvogel, S., Lange, T., Piesk, J.: *Dynamic Motion Models*. In: *Eurographics* (2002).
8. Abadia, J., Evans, A., Ganzales, E., Gonzales, S., Soto, D., Fort, S., Romeo, M., Blat, J. *Assisted Animated Production Creation and Programme Generation*. In: *ACM Advances in Computer Entertainment Technology*, pp207—214. (2009). ACM Press, New York (2009).

9. Meredith, M., Maddock, S.: Adapting Motion Capture Data Using Weighted Real-Time Inverse Kinematics. *ACM Computer in Entertainment* 3, 1 (2005).
10. Ekman, P., Friesen, W., Ellsworth, P.: What emotion categories or dimensions can observers judge from facial behaviour? In: *Emotion in the Human Face* (Ed. Ekman, P) (1982).
11. Densley, D., Willis, P., Emotional Posturing: A Method Towards Achieving Emotional Figure Animation. In: *IEEE Computer Animation 1997* pp. 8—14. IEEE Computer Society, Washington DC (1997).
12. Williams, R.: *The Animator's Survival Toolkit*. Faber and Faber (2001).
13. Johnston, O., Thomas, F.: *The Illusion of Life: Disney Animation*. Disney Editions (1995).
14. OGRE – Open Source 3D Graphics Engine, <http://www.ogre3d.org>.

4.2. The MASKLE: automatic weighting for facial animation

Facial animation of 3D characters is frequently a time-consuming and repetitive process that involves either skeleton-rigging or pose-setting for morph targets. A major issue of concern is the necessity to repeat similar tasks for different models, re-creating the same animation system for several faces. Thus there is a need for reusable methods and tools that allow the introduction of automation into these processes. In this paper we present such a method to assist in the process of facial rigging: the Maskle. Based upon the standard bone-weight linear skinning animation technique, the desired distribution of vertex-movement weights for facial animation is pre-programmed into a low-resolution, generic facial mask. This mask, or 'Maskle', is then semi-automatically overlaid onto a newly created face model, before the animation-weight distribution is automatically transferred from the Maskle to the model. The result

is a weight-painted model, created semi-automatically, and available for the artist to use for animation. We present results comparing Maskle-weighted faces to those weighted manually by an artist, which were treated as the gold standard.

The results show that the Maskle is capable of automatically weight-painting a face to within 1.58% of a manually weighted face, with a maximum error of 3.82%. Comparison with standard professional automatic weighting algorithms shows that the Maskle is over three times more accurate.

a) Introduction

The goal of facial animation for 3D models is to enable the representation of emotions and expressions in a plausible manner. Since pioneering work in the field was first published over 25 years ago (Parke, 1982), a large amount of research has been carried out on the development of computational models of the human face. The ultimate goal for this research is a system that 1) creates convincing animation, 2) operates in real time, 3) is automated as much as possible and 4) adapts easily to individual faces. While there has been significant progress towards solving each of these four matters individually, there has been relatively little progress in developing techniques that succeed in solving all four of the problems simultaneously. It is with this goal in mind that we present the first results of a novel method for automatic bone-weight facial rigging, called the Maskle. The motivation for the development of the Maskle has two sources. The first is the amount of time it can take an experienced artist to create a simple animation on a 3D face model, even using a powerful tool such as Autodesk's Maya or 3DS Max. The second is the multimedia industry's increasing need for reusable tools to assist in production work, reflected by current research being carried out by several EU-funded projects, for example SALERO (SALERO, 2006). The concept of the Maskle is a result of direct contact of academic researchers with multimedia production companies, and the results of the research are being directly funnelled into the professional sector.

The main contribution of this paper is in the area of facial animation, specifically towards automation of the facial animation process. We show the results of our initial tests which are designed to ascertain whether the Maskle is a viable concept for use within a professional production. The results obtained show a mean error of only 2.63%, and have led to the Maskle system being used already by our production partners.

b) Related work

The techniques used for the rigging of 3D characters to create convincing facial animation can be broadly divided into two distinct areas. The first are those that focus on mimicking the movement of the face surface only, attempting only to replicate facial poses using surface deformations (Guenter et al., 1998; Kalra et al., 1992). The second are those that model the anatomy of the face, attempting to replicate the movement of bones and muscles within a virtual framework (Lee et al., 1995; Platt and Badler, 1981; Waters and Frisbie, 1995). Some of the earliest work in facial animation represents this split, with Parke's work on the parameterisation of faces balanced by Waters' (Waters, 1987) attempt to replicate facial movement by modelling the movement of muscles. In the decades since this pioneering work there has been considerable research effort put into generating realistic facial animation, much of it reviewed in detail by Noh and Neumann (1998) and Ersotelos (2008).

Of particular relevance are Lee et al.'s (1995) efforts at digitising facial geometries and automatically animating them through the dynamic simulation of facial tissues and muscles. Their approach was to construct functional models of the heads of human subjects from laser-scanned range and reflectance data. These models were then extended with contractile muscles embedded within a dynamic skin model. The result was automatic animation of a human face from a scanned subject.

Noh and Neumann (Noh and Neumann, 2001) made considerable advances within the field of automatic character animation with their work on the cloning of expressions. Their technique was one of the first to directly address the problem of reusing existing animations, and transferring them to newly created virtual characters. After letting users select a set of points of the surface of a model, their method was able to transfer vertex motion vectors from a source character to a target character, with the aid of an automated heuristic correspondence search.

Orvalho et al. (Orvalho et al., 2006) extend this concept by attempting to adapt a generic facial rig to different facial models. The technique required considerable labelling effort yet was able to find corresponding points between source and target faces. This point matching was then used as a basis for the transfer of a complex, muscle-based facial rig system, to enable the target face to replicate the expressions provided by the base rig. Although this technique is of some interest, the authors do not present quantitative results, and only a few qualitative images to prove the validity of their method.

Despite these efforts, there is still a substantial gap in the state of the art that remains to be filled before we reach a facial modelling system that fulfils all four points mentioned above in section 4.2.1. In this paper we present our efforts at addressing this gap, with a highly automated system that is capable of enabling artists to quickly create suitable facial animation for a wide variety of face models.

c) Maskle overview

The concept of the Maskle is to allow artists to create a facial animation system once, yet be able to re-apply it to as many characters as they desire. In this sense, the keyword is that the system is *reusable*. Unlike some of the related research presented above, the goal of the system is not to transfer an animation system from one face to another; neither is it to automatically animate a face based on a scan or photograph. Rather, it is designed such that an artist can develop their own system of facial animation and, once

designed, quickly apply this system to any number of characters that they create.

The type of facial animation system that the Maskle is based around is a standard bone-weight system, where the deformation of the set of vertices that form the skin of a model is controlled by the movement of an underlying skeleton; the exact movement of each vertex (proportional to the bones of the skeleton) is represented by a set of numeric proportions, or *weights*, assigned to each vertex. The justification for basing our algorithm on a bone-weight system, as opposed to other animation systems such as blend shapes, is that the bone-weight system can be abstracted to a number of control points (representing the locations of the bones); this abstraction facilitates the organisation of weight-transfer algorithm presented below, and allows rapid testing to ensure the results are satisfactory.

Once an artist has created a character, bone-weight animation of a face usually requires extensive effort in accurately assigning, or *painting*, the weights for each vertex and for each bone. This can be done automatically, and many 3D design packages such Autodesk Maya and 3DS Max have such functionality, frequently based around using envelope systems (Autodesk, 2007). However, despite being generally successful at automatically painting body areas where the skin can be tightly bound to the bone (such as the arms or legs), these existing systems are less suitable for facial rigs, where the ‘bones’ of the face are rarely modelled on the real-life bones of a skull. Thus, weight-painting a newly designed character to fit an existing facial rig becomes a labour intensive and time consuming process, as there is little existing automation that can be used.

The Maskle system directly addresses this problem by using a pre-painted, low resolution mask to automatically weight-paint a newly created face. The initial step is for an artist to create a facial rig, (according to individual requirements) for the Maskle itself. Given this rig, the overall process occurs as follows. After the user has marked a total of ten specific marker points around the areas of the lips and the eyes, the structure of the Maskle is loaded. This structure consists of a low-resolution facial ‘mask’, designed so that it will be able to wrap around areas of a 3D face that are commonly used for animation. The Maskle is then adjusted semi-automatically so that it shrinks around the shape of the face; a collision detection

algorithm detects when the Maskle has contacted the face and prevents its further movement. Once the Maskle has wrapped around the face, ray-face collision algorithms find, for each vertex of the face model, the nearest point on the surface of the mask, which is then used to calculate the animation weight for each vertex of the face mesh. The Maskle can then be deleted, leaving the face surface bound to the bone rig, ready for animation.

The mesh used to represent the Maskle can vary in terms of its actual structure. However, for the work presented in this paper, the Maskle structure consists of a 90-vertex, 82-face, symmetrical triangle/quad mesh. While it is designed to cover the areas surrounding the lips and eyebrows; the precise details of topology and dimensions are less relevant due to the changes that it undergoes during the process of fitting it to a target face (as explained in section 4.2.3 below). The techniques presented in section 4.2.3 were developed using C++ and Maya Scripting Language (MEL) to create a plug-in for Autodesk Maya, due to it being one of the most popular 3D modelling packages available. It has also been ported to Autodesk 3DS Max, and the design of the system is such that it would be straightforward to transfer it to other modelling and animation tools such as Blender, or proprietary software. The facial models used for testing and evaluation are triangle/quad vertex-face models, consisting of between 757 and 6603 vertices. Each facial model must have a small gap between the upper and lower lips to ensure correct weighting in these areas (see below).

d) Automatic fitting

To ensure that the animation weights for each vertex are transferred as accurately as possible, it is important that the Maskle fits closely over the face model. More importantly, it is vital that the equivalent areas of the mask and face model are in close proximity; for example, the upper lip of the Maskle must be in close proximity to the upper lip of the face model. Failure to ensure this proximity frequently results in errors in the transferring of weights, for example, the upper lip of the face model acquiring the weights from the lower lip of the mask.

Initial placement. Due to the wide variety of facial shapes and sizes that can exist, for each face it is necessary to pre-programme the Maskle with some figures that relate to the scale of the model in question. Thus, the fitting of the Maskle to the face is initialised by the user manually marking ten vertices of the face: two at the lateral boundaries of the lips; four along the central axis of the face, marking the highest and lowest vertices of both the upper and lower lip, and four marking the inner and outer lateral points of both eyes. While it is unfortunate that the user should have to manually mark locations, the number is substantially less than that required by similar techniques (Orvalho et al. 2006).

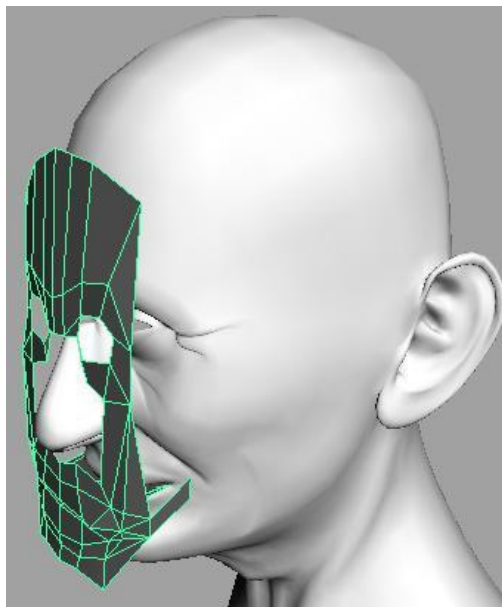


Figure 4.2.1: View of the Maskle system post-initialisation step.

Once the vertices have been marked, the respective distances between them are used to scale the basic Maskle shape so that it has the same dimensions (e.g. the distance between nose and mouth) as the face. The Maskle mesh is then loaded and placed in front of the face model. It is almost entirely flattened to a 2D plane, with only a ‘tongue’ extending into the mouth. This ‘tongue’ is present to ensure that the Maskle will correctly cover the lips of the face model; if it is not present the probability of either lip being assigned the animation weight of the other is much higher. The final

placement of Maskle is decided by the user, to ensure that the ‘tongue’ of the Maskle enters the mouth at the correct angle. Figure 4.2.1 shows the position of Maskle post-initialisation.

Automatic shrinking. Following the initial placement, the Maskle undergoes an automatic movement/shrinking process which allows the shape of the Maskle mesh to hug closely the shape of the target face mesh. The basic system of movement is a curtailed version of the dynamic force formulation for active surface models (Sonka and Fitzpatrick, 2000). An iterative process applies a combination of forces to each vertex, the direction and magnitude of each force contributing to the overall movement. This method was preferred over direct correspondence techniques as it allows the structure of the Maskle to change dynamically as it is laid over the face model; also it is very fast.

The location, \mathbf{x}_i , of each vertex, v_i , moving through time t can be calculated as

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{F}_i(t) \quad (1)$$

The total force, \mathbf{F} , applied to each vertex at each iteration is dependent on two forces; $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

$$\mathbf{F}_i(t) = a\boldsymbol{\alpha}_i(t) + b\boldsymbol{\beta}_i(t) \quad (2)$$

a and b are factors used to control the influence of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively. $\boldsymbol{\alpha}_i(t)$ is calculated according to:

$$\boldsymbol{\alpha}_i(t) = c_i - \mathbf{x}_i(t) \quad (3)$$

where c_i is the average coordinate location of the set of vertices adjacent to v_i (i.e. the set of vertices that are connected to v_i by a single edge). $\boldsymbol{\beta}_i(t)$ is equivalent to the normalised vector representing the initial (i.e. when $t=0$) direction of the ‘tongue’ of the Maskle (as mentioned above in 3.2.1) and is calculated using the relative locations of the relevant vertices of the Maskle mesh. The effect of iteratively applying equation (1) to the vertices of the Maskle is that each vertex moves (in Euclidean space) according to the direction and magnitude of the resulting vector. Thus, the mesh

moves towards the face model (due to the influence of β) and wraps around it (due to the influence of α). The termination condition for the movement is partially applied when the Maskle mesh collides with the face model mesh (alternatively, the user may choose to manually terminate the movement phase). Specifically, if a Maskle triangle/quad collides with the face model, the constituent vertices of the Maskle are flagged and prevented from further movement. Once all the vertices of the Maskle have been flagged, the algorithm stops. Collision detection is carried out by Moller's triangle-triangle intersection test algorithm (Moller, 1997), optimised by a standard axis-aligned octree (Ericson, 2005), and run at each movement iteration. Quad faces are split into two constituent triangles for the purposes of collision detection. Figure 4.2.2 shows an example of the result of the completed movement.

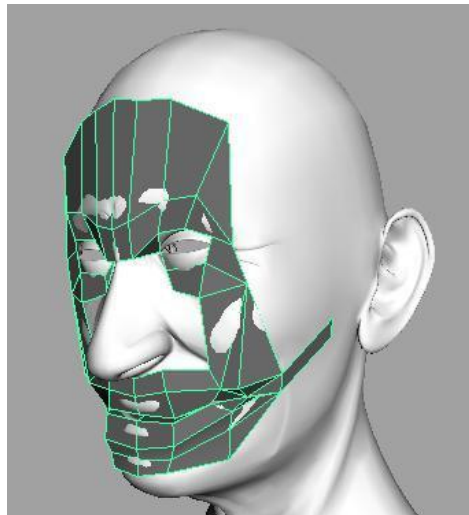


Figure 4.2.2: Final position of the Maskle for weight transfer.

It is important to note that the results of the automatic shrinking algorithm are dependent on the accuracy of the initial placement (described above in section 4.2.3). If the initial placement is incorrect, the overall dimensions of the Maskle structure will be incorrect, and the automatic shrinking algorithm will not hug the face model correctly.

e) Weight transfer

Now that the Maskle mesh has been placed adjacent to the face model, its location can be used to recreate an existing rig on the model (as mentioned in section 4.2.3 above, such a rig will have been already created by the artist and applied to the Maskle). Firstly the system of bones of the facial rig is re-created. Given the location of a manually created ‘base’ bone, a simple script recreates the bones of the rig, using the locations of the vertices of the Maskle to ensure correct placement. The exact form of this script will depend on the number of bones that form part of the facial rig which the Maskle is applying. The animation weights for each vertex and for each bone can now be set for the face model. For each vertex of the face model, an infinite bi-directional ray is projected along the axis of the Normal vector of the model surface at that vertex. An intersection test (Ericson, 2005) is carried out between this ray and the faces of the Maskle. If successful, this test defines an intersection point, p_I , which lies on the plane of a quad/triangle face of the Maskle. This face is labelled f_I . Recalling that the Maskle vertices have been already weighted by the artist during the creation of the original facial rig, the coordinates of p_I on the surface of the Maskle are then used to interpolate the animation weights associated with the constituent vertices of f_I . The method used for this interpolation is the Inverse Distance Weighting, or “Shepard”, method, where the interpolated value is a weighted average of the values of the surrounding points (Amidror, 2002). The interpolated weight values for p_I are then assigned the vertex of the face model from which p_I was created. To increase computational speed, the process is carried out for only those vertices of the face model that lie within the largest possible circumsphere that can be created using the vertices of the Maskle mesh. The result is that the weights for each bone are interpolated from the vertices of the Maskle to the vertices of the face model. The final step is to run a smoothing algorithm (Autodesk, 2007) that ensures even distribution of weights over the area of the face. The artist is now free to animate the face as desired.

f) Evaluation

An evaluation of the Maskle was carried out to prove that the concept of the weight-transfer system had validity. The difficulty in designing such a test is increased due to the very nature of the work that the Maskle is designed to facilitate i.e. animating a face. Such work can be highly subjective, and it is important to try and remove or negate any influences that may introduce such subjectivity. For example, the Maskle is designed to work with a variety of facial rig designs, and any evaluation procedure should be as independent as possible from these or other variable elements that exist in the animation pipeline.

To this extent we designed an evaluation procedure that tests *only* the capability of the Maskle to transfer the weights for the correct regions of the face. An artist created a very simple, seven-bone facial rig, and applied it manually to the Maskle and to four different face models. We then commenced the evaluation process by using the Maskle to automatically transfer its associated rig to unrigged copies of same four face models. The values assigned to variables a and b in equation (1) were deduced according to visual inspection of the movement path of the Maskle mesh, and were set to 0.005 and 0.05 respectively. They were kept constant for all tests. Such low values ensure slow movement of the Maskle mesh and thus more accurate collision detection.

The success of the Maskle weighting was measured by comparing the animation weight distribution of manually weighted face-model (the gold standard) and the Maskle-weighted face model (the test candidate). However, doing this for every single vertex in the face model could have introduced bias into the procedure. This is because neither artist nor Maskle will have applied animation weights to immovable facial features (e.g. the ear) and so these vertices should not be allowed to bias any mean comparison towards smaller error. Thus, we isolated the set of vertices of the face model whose Normal vector rays intersect with the Maskle (as described in section 4.2.3) and label this set M , where $M = \{m_i\}$, $i = 1, \dots, I$. The difference between the manually weighted set, M_{Manual} and the Maskle-weighted set M_{Maskle} , for each of the seven bones in the rig, can now be calculated thus:

$$diff_{bone} = \frac{\sum_0^I abs(m_{i Manual} - m_{i Maskle})}{I}$$

where $m_{i Manual}$ is the weight of the i^{th} vertex of set M_{Manual} and $m_{i Maskle}$ is the weight of the i^{th} vertex of set M_{Maskle} .

The total average difference between M_{Manual} and M_{Maskle} is then calculated by averaging $diff_{bone}$ over each of the seven bones in the test rigs. As the goal was to test the weight distribution (i.e. the spread of weights for each joint across the mesh), the differences between manual and automatic were normalised according to the difference in maximum weight spread.

As a comparison, an industry standard envelope-based automatic skinning algorithm (Autodesk, 2007) was applied to the same four faces, and the differences calculated in the same way (using the same set of vertices). Figure 6 shows a table with the results as percentage of the total possible weight (weight values range from 0 – 1, thus an average weight difference of 0.5 means there is a gap between the sets of 50% of the possible weight). The face models have been labelled according to their appearance.

Table 4.2.1 shows that the Maskle weighted face achieves lower error rates in weighting all the test models. When considering the mean figures, the table shows that the Maskle is over three times more accurate than the standard envelope algorithm. Visual analysis of the results for each face shows an even greater difference that the data in the table illustrates, as the envelope based method deals very poorly with the area of the lips, incorrectly associating vertices of the upper lip to the bone of the lower lip, and vice versa. The error can be seen visually in Figure 4.2.4(c). Due to the nature of the Maskle (specifically, the ‘tongue’ that divides the upper and lower lips, this error is completely removed.

Model	Maskle difference as % of possible weight	Envelope difference as % of possible weight
Realistic Human	1.58	8.43
Cartoon Human	2.49	7.40
Cartoon Devil	2.61	7.17
Venetian Mask	3.82	11.92
Average	2.63	8.73

Table 4.2.1: The mean differences between manually weighted faces and automatically weighted faces. A sample of each model is shown in Figures 4.2.4, 4.2.5 and 4.2.6.

To illustrate this, Figure 4.2.3 consists of two graphs that show the weight distribution profile for the weights of the lower lip bone of one of the models used in the study (the ‘Realistic Human’ model in Table 4.2.1 and illustrated in Figure 4.2.4). Figure 4.2.3 (a) shows a comparison between the manually weighted face and the Maskle weighted face. The graph shows that there the two sets of data overlap and that there are very few vertices that are weighted in only one of the sets. Figure 4.2.3 (b) shows a comparison between the manually weighted face and the envelope-algorithm weighted face. In this graph there is less correlation between the datasets, and the envelope algorithm has incorrectly weighted several vertices which remain unweighted by the manual operator. Further examples of animated faces generated successfully using the Maskle can be seen in Figures 4.2.4 (b), 4.2.5 and 4.2.6.

g) Discussion and future work

In this paper we have presented the concept of the Maskle system and the results of a study on the validity of using such a system. The novel contribution is in the area of automation of facial animation, with specific contribution in the automated animation of areas of the face that are time-consuming to animate manually, such as the lips. The results show that the system used for this study is capable of creating a facial rig that is, on average, within 2.63% of being identical to that created manually. This error rate is over three times less than that obtained by carrying out the same tests on a standard envelope-based weighting algorithm.

The Maskle is a novel idea of automatically recreating a facial animation system on a variety of face models, and addresses this issue in a way that is different to work previously published; thus, it is difficult to compare the obtained error rates with the results published by other researchers. Perhaps the most similar previous work is Orvalho et al.'s (2006) efforts in transferring a facial animation system, yet their work focuses strongly on transferring a rather complex generic facial rig, and involves the user manually marking 44 points landmark points. The Maskle system is more flexible in that it can be used with a variety of facial rigs, and only requires the marking of ten landmark points. Unfortunately, Orvalho et al. do not provide numerical results for their technique, showing their results only in graphical format, thus it is difficult to directly compare the accuracy of the Maskle with their results. Noh and Neumann (Noh and Neumann, 2001) presented statistics comparing the results of the motion vectors for cloned expressions (i.e. expression that are copied from a 'source' face to a 'destination' face). Again, this is difficult to compare to the results in this paper, as the Maskle does not attempt to directly transfer expressions, rather it is directly transferring movement weights to allow artists to animate as they desire.

Due to the low error rates, analysis of graphs of the type shown in Figure 4.2.3, and visual analysis of the face models, our interpretation of the results obtained in the study is that the Maskle is a tool that can greatly aid the process of creating facial animation for 3D characters. This being said, the technique does have several limitations. The first is that in its current state, the Maskle does not cover certain areas of the face that are usually used in animation e.g. the chin. The reason for this is that, in practice, such areas are straightforward for even a non-experienced artist to animate, and thus there is little need for an automated tool to assist in this area. By contrast the area around the lips is difficult and time-consuming to animate, and it is this and other similar areas in which the Maskle is of most use. Nevertheless, in the interest of creating a more comprehensive tool, our future work will involve expanding the Maskle to cover other areas of the face. A further limitation is in situations where the face shape is very different to the structure of the Maskle. The system works well with humanoid faces, but has not yet been tested with animal, fantasy, or highly abstract faces. There is also scope for technical improvement by investigating

different methods of Maskle-face correspondence. While the current method is adequate and fast, it does require manual fine adjustment. Several other correspondence techniques exist, and a study could be carried out to see if using any of these improves the system. Our immediate future work is to conduct a more comprehensive evaluation study to measure the impact of the use of the Maskle in real-life animation situations, possibly recording the time that it takes several artists to create a facial rig on several characters, with and without the Maskle.

Finally we also intend to combine the Maskle with other facial animation work that is being currently being conducted within our group, which involves automatic creation of facial emotions across a wide range of facial models. It is expected that this further work will lead to a considerable breakthrough in the field of automatic facial animation, in which the Maskle system will play a major role.

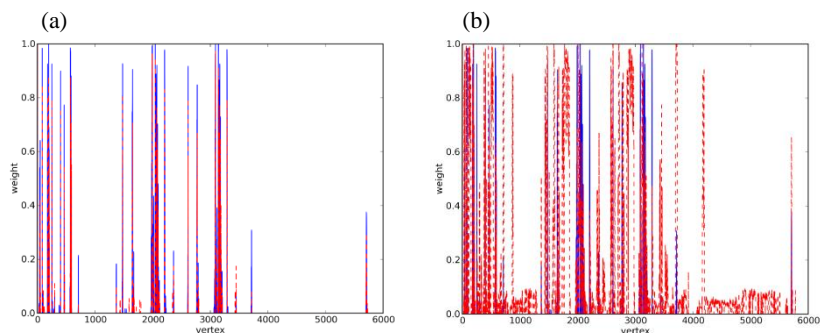


Figure 4.2.3: Sample weight profiles for one bone of a manually-weighted face (blue line) and automatically weighted face (red dashed line). Figure (a) shows the results for the Maskle, Figure (b) for the envelope algorithm. The envelope algorithm shows a much greater level of noise and erroneous calculations.

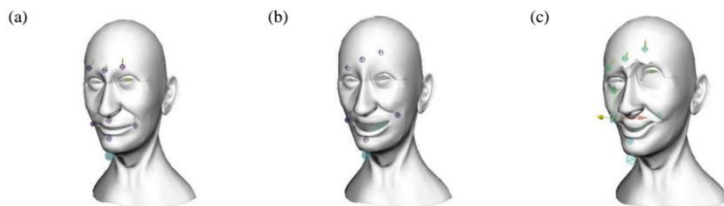


Figure 4.2.4: Visual differences between Maskle-weighted and envelope-weighted faces, comparing a simple expression created by moving the bones of the rig. (a) shows the unanimated face; (b) shows the expression applied to the Maskle-weighted face; (c) shows the same expression applied to the envelope-weighted face.

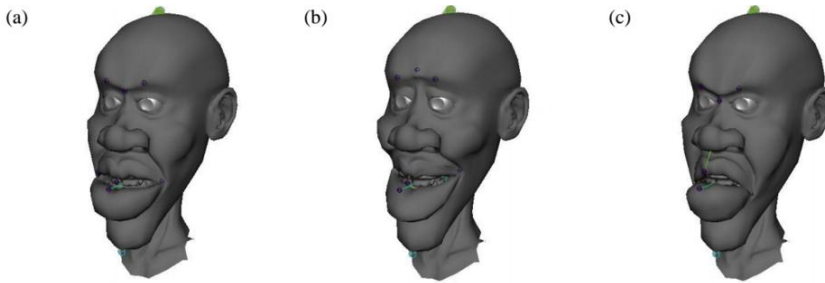


Figure 4.2.5: Example of expressions successfully created *using the Maskle* on a cartoon character with pronounced features. (a) is the unanimated face, (b) and (c) are expressions created after the Maskle has weighted the face.



Figure 4.2.6: Examples of expressions created on 3D models animated using the Maskle (images © Merja Nieminen, Crucible Studio / University of Art and Design Helsinki 2008).

h) References

Amidror, I. 2002. Scattered data interpolation methods for electronic imaging systems, a survey, *Journal of Electronic Imaging*, 11, 2, 157-176.

Autodesk Maya Press. 2007. *Learning Autodesk Maya 2008: The Modeling & Animation Handbook*. Sybex.

Ericson, C. 2005. *Real-time Collision Detection*. Morgan Kaufmann.

Ersotelos, N. and Dong, F. 2008. Building highly realistic facial modeling and animation: a survey, *The Visual Computer*, 24, 13-30.

Guenter, B., Grimm, C., Wood, D., Malvar, H., and Pighin, F. 1998. Making Faces, In *Proceedings of ACM SIGGRAPH 1998*, 55-66.

- Kalra, P., Mangili, A., Thalmann, N., and Thalmann, D. 1992. Simulation of Facial Muscle Actions Based on Rational Free From Deformation. *Eurographics*, 11, 3, 59-69.
- Lee, Y., Terzopoulos, D., and Waters, K. 1995. Realistic Modeling for Facial Animation. In *Proceedings of ACM SIGGRAPH 1995*, 55-62.
- Lee, Y., Terzopoulos, D., and Waters, K. 2002. Realistic Face Modelling for Animation. In *Proceedings of ACM SIGGRAPH 1995*, 55-62.
- Möller, T. 1997. A fast triangle-triangle intersection test, *Journal of Graphics Tools*, 2, 2, 25-30.
- Noh, J. and Neumann, U. 2001. Expression Cloning, In *Proceedings of ACM SIGGRAPH 2001*, 277-288.
- Noh, J. and Neumann, U. 1998. A survey of facial modeling and animation techniques, USC Technical Report.
- Orvalho, V., Zacur, E., and Susin, A. 2006. Transferring a Labeled Generic Rig to Animate Face Models, *Lecture Notes in Computer Science 4069*, 223-233.
- Parke, F. 1982. A parameterized Model for Facial Animation, *IEEE Computer Graphics and Applications*, 2, 9, 61-68.
- Platt, S., and Badler, N. 1981. Animating Facial Expressions. *Computer Graphics*, 15, 3, 245-252.
- SALERO. 2006. <http://www.salero.info/>.
- Sonka, M and Fitzpatrick, J. 2000. *Handbook of Medical Imaging, Vol. 2 Medical Image Processing and Analysis*. SPIE Press.
- Waters, K. 1987. A Muscle Model for Animating Three-Dimensional Facial Expression, *Computer Graphics*, 21, 4, 17-24.
- Waters, K., and Frisbie, J. 1995. A coordinated Muscle Model for Speech Animation, *Graphics Interface*, 163-170.

4.3. A synthetic model for automatic and real-time facial expression animation

With the advent of mobile and web 3D technology, there is the need for rapid and automatic techniques to provide virtual characters of facial expressivity. This can be achieved only if the facial rigs of the characters can produce convincing expressions and if its computation requires low hardware resources.

This paper presents a synthetic model of facial shapes that defines a reduced number of facial shapes whose interpolation produces a set of believable facial expressions. By studying 2D animation and classic literature with reference to anatomy and representation of facial expression, our reach for believability is carried out through synthesis rather than realism. In this way our model can reconstruct facial expressions by linearly interpolating only five facial shapes. Furthermore, the values used for the interpolation can be retargeted from one face to another, obtaining the same expressions over different characters. The paper describes two different evaluation processes, which are aimed to determine both whether it fits the needs of the animators and whether it is believable for the final viewers. The analysis of the data gathered during the evaluation process is then reported, showing that 72% of users were satisfied with the method, and 86% of viewers were able to perceive the correct emotions.

a) Introduction and background

Automatic and real-time virtual characters are used in many fields of the audiovisual industry such as television, video games, internet, and mobile phones (e.g. game bots, real-time tv presenters and crowd agents for visual effects). Facial expressions are crucial to allow virtual characters to properly express emotions, improving their ability to communicate. While this issue has been solved in the film industry, the same techniques cannot be applied to the fields mentioned before due to strong time and hardware resources constraints. Typically time consumption could be due either to the difficulties in creating a good animation rig for facial expression or preparing handcrafted deformations that would be used for

generating the facial expressions. Those difficulties directly affect the hardware resources as complex animation rigs or a high number of different facial deformations make real-time animations hard to be replayed while maintaining the desired frame rate. In that sense there is the need for an animation methodology that can represent several facial expressions by using fewer hardware resources and being sufficiently straightforward for animators to set up.

In “Art, perception and reality” E.H. Gombrich defines *Topffer’s Law* as: “whatever we can interpret as a face, has, ipso facto, expression and individuality”. In other words, everything that can be identified as a face must have a facial expression, and this expression is a metaphor with which an emotion is associated. This issue of perception of facial representations is unavoidable within the field of virtual character animation.

During the discussion and analysis of facial expression, two groups can be distinguished:

- 1) Static- referring to the permanent shape and configuration of the face. The shape is generated by biological and biographical factors.
- 2) Dynamic – referring to mobile configuration of the face, the *mimic behaviour*. In contrast to (1) above, the variable expression is defined more by the movement of muscles, and less by the shape of the face.

Note that this distinction between movement and shape is not exclusive, but interdependent. Repeated use of the expressions generated by (2) gradually affects the structure of (1), modelling the face according to lines of expression. Due to it being centred on animation, our work is currently focused on the second group introduced above.

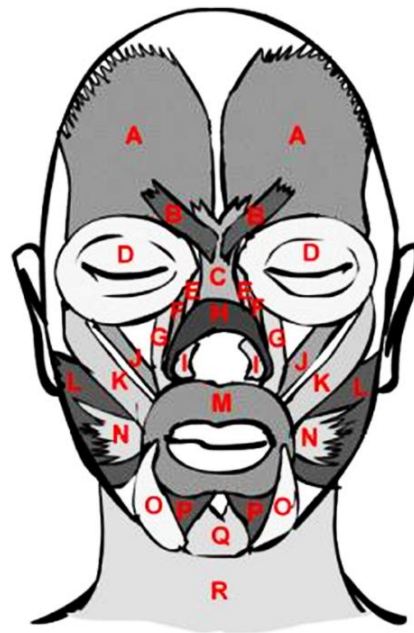
The study of facial expression in the audiovisual and fine arts fields uses several standard terms that arise from other fields. *Representation* is a concept commonly used in semiotics. It is used as a reference to every image or figure that substitutes, imitates or reflects reality (for instance, a painting can represent a landscape, a map can represent a city and a drawing can represent a person performing a facial expression). The concept of *Representational*

Culture stresses the importance of *representation* within a culture or society. Every culture has his own way to represent reality, and for that reason everyone, as part of a determined culture and historical period, perceives reality that is filtered by a representational background. The term *mimic gesture* refers to the shape of a face created by contraction of subcutaneous muscles, and is commonly associated with the term *Expression*. Finally, *Emotions* can be associated with particular *mimic gestures* or *expressions* – although there is not necessarily a one-to-one correlation between them, as the same or similar *expressions* may be interpreted as different *emotions*, as affected by the context and the *representational culture*.

This concept of emotion and its connection to the cultural context is similar to what can be found in the literature of Paul Ekman². Starting from the study of nature and real faces' expressions, he studied how expressions are perceived across different cultures. He defined a set of basic emotions, each one with a unique associated facial expression but he also stated that there are many other emotions that can be represented through different expressions and vice versa.

Finally, it is worth presenting a brief overview of the main muscles of the face, as knowledge of their location and function will aid in the understanding of the methodology employed in our work, and of facial animation in general. The muscles of the head can be divided into two main groups, masticators and subcutaneous. The former manage the skeletal structure; the latter manage the facial gesture, contracting and producing wrinkles perpendicularly to the direction of the muscle fibre. Face muscles aid both expression and the functionality of the senses (such as touch and smell). There are eighteen sub-cutaneous muscles (*fig. 4.3.1*) that are involved in creating a facial gesture. Almost all possible gestures involve the contraction of several of these muscles at once. Not all contractions of these muscles achieve the same level of potential expressiveness, and some gestures only appear in very unusual situations. Our research has attempted to ignore these uncommon situations, as our primary intention was to create a system of high expressive ability, using a low number of associated movements. In other words, the goal was to create convincing animation by including *only the*

gestures that are absolutely necessary to maintain high expressive ability.



- A** Frontalis **B** Corrugator **C** Procerus
D Ocularis Oculi **E** Quadratus Labii Superioris
F Own Elevator **G** Caninus **H** Nasalis Pars Transversa
I Nose Expander **J** Zygomaticus Minor
K Zygomaticus Major **L** Buccinatorius
M Orbicularis Oris **N** Risorius **O** Triangularis
P Quadratus Labii Inferioris **Q** Mentalis **R** Platysma

Figure 4.3.1: Sub-cutaneous muscles

b) Related work

Much of the literature regarding facial animation has studied how the morphology, deformation and movement of the face produce emotional expressions. The work in this research typically implements tracking of markers or feature points on a real actor's face, gathering the motion data and trying to reproduce them on a virtual actor's face by directly deforming the geometry³ or mapping the motion data to some kind of handcrafted deformation⁴. These

studies tend to produce complex geometry and motion data as output.

Recent work has studied how facial expressions, represented on a synthetic character, are perceived by the viewers⁵. These studies are commonly related to reality as they use captured data about motion, 3D geometry and 4D geometry deformations to generate the animations. Captured or scanned motion data is just a way of producing animation and cannot be considered as the only form of computer animation.

When studying the interaction between the types of expression, static and dynamic, a schematic representation is very useful, due to its ability to graphically represent several abstract facial aspects. In 1827, Superville proposed three schemes in order to combine several aspects of the static human physiognomic appearance [Superville 1827].

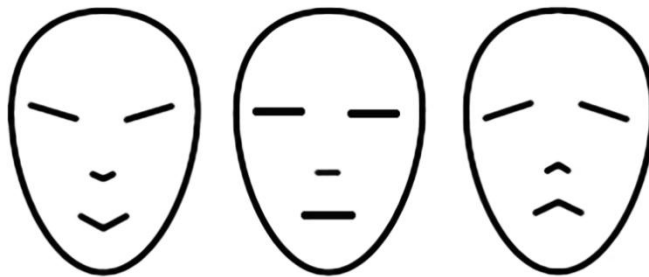


Figure 4.3.2: Superville's schemes

The schemes start from an oval, defining the shape of the face, to which is added several straight lines, drawn to represent areas of the face associated with expression. The default configuration of these lines is for them to all to be horizontal. In this state the face is said to represent *calm*, and it is demonstrated diagrammatically in Figure 4.3.2 (the scheme in the middle). Superville then created two further schemes as extremes of the first, following the axis of the face and adjusting the angle of the lines, from the horizontal to the oblique. The resulting schemes are defined as the *oblique expansive lines scheme* and *oblique convergent lines scheme*. They are said to represent *happiness* (left scheme in Figure 4.3.2) and *sadness* (right

scheme in Figure 4.3.2) respectively. These basic schemes used by Superville to describe permanent aspects of the face, can also be used to create a schematization of the dynamic facial expressions. Plasencia⁶ addressed this issue: "...Some muscular actions (frontal and orbicular) create a directional movement that is more vertical than oblique, creating tensions (across the surface of the skin) that are represented by facial wrinkles....The type of mimic event associated to expressions generated in this manner are related to *astonishment*, *panic*, and more generally, *surprise*....Thus, a fourth state can be added to Superville's scheme, not only as an intermediate between the two extremes⁷, but as an opposite to the base state." Plasencia's schemes are summarised in Figure 4.4.3.

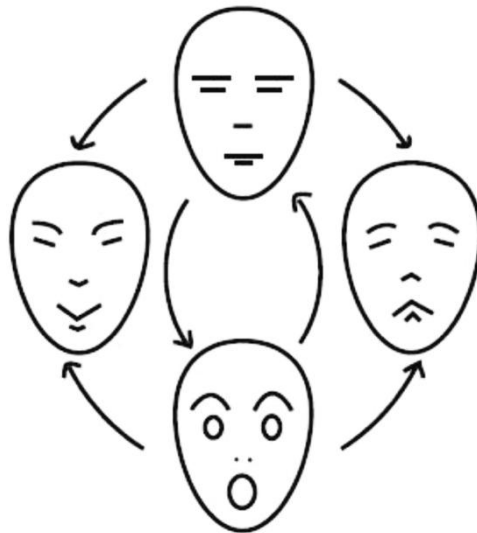


Figure 4.3.3: Plasencia's schemes.

In the proposed schemes the upper part and lower part of the face participates in the recognition of the facial expression with differentiated roles between each other and in relation to the configuration of the whole face⁸. The implementation of the deformations synthesized in the schemes presented above can be reproduced on a virtual character using blend shapes. This is a common approach to facial animation in many fields of the audiovisual industry and it still a main point of discussion in the scientific community⁹.

c) Contribution

In this paper, we address the problem of reducing the complexity of three-dimensional facial rigs to enable automation and real-time animation of facial expressions. We present a conceptual model that permits the representation of a wide range of mimic behaviour through facial expressions. This model is built to constitute a conceptual background on which automatic and real-time algorithms can be designed. The model is based on the idea of interpolating a neutral facial pose together with four ‘extreme’ poses, the latter being created by independent movement of the face in zones of the eyes and mouth. The four extreme poses are generated by considering the maximum possible believable movement of facial features, considering independent displacement vectors of the face’s surface. The model is formulated via a process that starts from the study of faces’ representation in art and audiovisual productions and, in that sense, it concentrates more on the *believability* than on the realism. For that reason the study does not take advantage of techniques such as motion capture or 3D scanners that have strong constraints with reality rather than how this reality can be represented or perceived. Finally the paper presents two different experiments that aim to evaluate both whether the model is flexible enough for artists to produce animated characters, and whether the animated expressions produced with the model are correctly perceived by viewers.

d) A synthetic model for expression representation

All artistic representations involve an inherent intellectual process and a synthesis of reality. This reality, and its perception, is affected by the underlying culture, in that one person’s interpretation of the expression of another is filtered by the current representational culture. Throughout the course of history, both artists and mechanical media have used this knowledge to express emotions in the faces of characters. Yet it is important to remember that a fictional character with an artificially created facial expression is very different to a human being with the same expression. The former, despite its analogy with reality, still remains a two-dimensional representation. For this reason the underlying

representational culture is of fundamental importance to any image or animation, as it affects the perception of a character's expression. In other words, the underlying culture is supplying the information that is inherently lacking in the 2D image.

In traditional animation the facial actions are mainly described through the animation of the mouth and the eyes, while wrinkles and nose movements are typically considered a less important layer. In Japanese animation, for example, which tends to use standard facial masks, there is an absence of lines that represent the wrinkles produced by the contractions of the frontal muscles, there are no changes in the nose base, and the wrinkle between nose and the upper lip is not marked. This suggests that synthetic schemes can be applied to animated characters.

The model developed by our research is based on the study of the history of facial expression representation and how it can be applied to virtual character animation. Our research uses Plasencia's four schemes⁶ as a model of the movements of the eyebrows and the mouth, along with the transitions between them. The core of our model is based on the interpolation between five facial poses, one of which being the default, 'neutral', pose. One of the key aspects of our system is that the four non-neutral poses are related to facial deformations, *not* directly to emotions. In Superville's work, both the expansive lines scheme and converging lines scheme are extreme poses based around the same axis of movement, yet restricted to two dimensions.

In order to adapt this approach to 3D, we included the third dimension, resulting in the *projection* and the *contraction* of the lips. Figure 4.3.4 shows a diagrammatical representation of the model after the inclusion of these schemes.

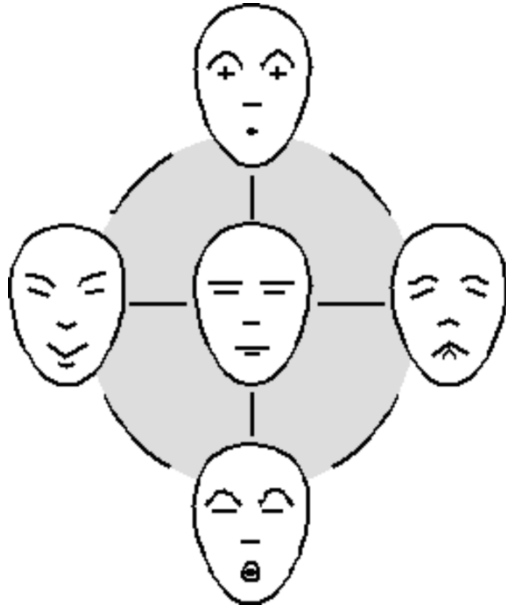


Figure 4.3.4: Our model's schemes.

It is now possible to dismantle these poses by splitting the face in half, separating the area around the eyebrows from the one around the mouth. By splitting the schemes in this way, we obtain three additional poses, bringing the total to eight: one with the eyelids closed, three for different configurations of the eyebrows, and four for different configurations of the mouth. These eight poses, along with the 'neutral' pose, can now be interpolated to create complete expressions. Table 4.3.1 shows the eight extreme poses applied to a character's face; the labels assigned to these poses are: *mouth down*, *mouth up*, *mouth projection*, *mouth contraction*, *eyebrows down*, *eyebrows up*, *eyebrows curved*, *eyelids closed*. The scheme is complemented by the deformation produced by the jaw bone.





















			
Neutral		Mouth Down	
			
Mouth Up		Mouth Projection	
			
Mouth Contraction		Eyebrows Down	
			
Eyebrows Up		Eyebrows Curved	
			
Eyelids Closed		Mouth Open	

Table 4.3.1: The poses proposed, modelled on two different character's faces.

By controlling the interpolation weights, it is now possible to recreate various expressions that can be associated to different emotions. In Table 4.3.3 we present some interpolation values that we have used to create a set of basic expressions (although these can change in relation with the needs of each production).















			
Surprise	Fear	Attention	Furious
			
Doubt	Determination	Sadness	Shout/Yawn
			
Crying	Smile	Irony	Pleasure
			
Laugh	Pain		

Table 4.3.2: Some proposed facial expressions.

	Mouth Down	Mouth Up	Mouth Projection	Mouth Contraction	Eyebrows Down	Eyebrows Up	Eyebrows Curved	Eyelids Closed	Mouth Open
Surprise	0.129	0.323	0.000	0.301	0.172	0.000	0.957	0.000	15.172
Terrified	0.312	1.000	0.000	0.097	1.000	0.989	0.753	0.000	6.61
Attention	0.000	0.065	0.344	0.344	0.000	0.000	1.000	0.000	0.81
Doubt	0.000	0.194	0.900	0.398	0.000	0.500	0.989	0.237	-1.53
Determination	0.000	0.194	0.258	0.806	1.000	0.007	0.054	0.473	-0.762
Shout/Yawn	0.570	0.391	0.591	0.462	0.981	0.000	0.077	0.342	28
To Cry	0.000	1.000	0.000	0.000	0.000	1.000	0.000	0.710	9
Sadness	0.000	0.995	0.225	0.000	0.000	0.996	0.996	0.194	-0.2
Irony	0.652	0.000	0.000	0.545	1.000	0.462	0.395	0.720	-3.562
Pleasure	0.075	0.138	0.675	0.000	0.380	0.056	0.214	1.000	8.1
Smile	0.700	0.000	0.000	0.000	0.000	0.527	0.763	0.452	-3.247
Laugh	1.000	0.000	0.000	0.000	0.732	0.346	0.779	0.720	2.122
Pain	0.381	0.957	0.000	0.376	0.957	0.430	0.000	1.000	5.324
Furious	0.527	0.237	0.000	0.151	1.000	0.000	0.000	0.000	23.324

Table 4.3.3: Set of experimental interpolation values between facial poses.

e) Muscle groups of the model

The discrimination between the areas of the mouth and the eyes simplifies the process of facial animation by having to represent the action of only a few muscles. In the region of the eyes the scheme copies the action of the *Frontalis* (only for the action of the eyebrows, not for producing forehead wrinkles), the *Corrugator*, the *Procerus* and the *Orbicularis Oculi* (only for the action of opening and closing the eyes). A greater number of muscles are represented around the mouth: the *Orbicularis Oris*, the *Zygomaticus Minor*, the *Zygomaticus Major*, the *Risorius*, the *Quadratus Labii Inferioris*, the *Triangularis* and the *Mentalis*.

A crucial feature of the scheme is that it does not attempt to directly model the action of these muscles, rather it groups them together into clusters that are, in effect, new ‘pseudo-muscles’. The focal point of these muscles can be represented by points, as demonstrated in Figure 4.3.5.

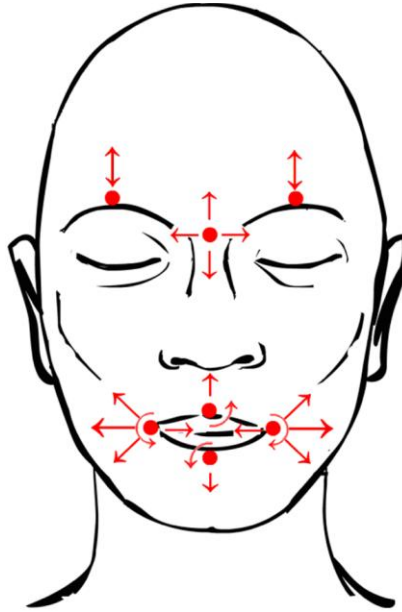


Figure 4.3.5: The points of action and movements proposed by our model.

f) Interpolation formula

The implementation of the synthetic model is done by using a blend shape system, similar to what is commonly used in several commercial animation software; details of the precise formula we have used are given here for completeness. Given a neutral pose for the face, the other poses used by the synthetic model are considered as blend targets. The distance vector from every vertex v_b in the neutral pose and the same vertex in a blend target v_i is calculated as an offset. Then, those offsets are applied on the neutral face modulating them by the weights of blend assignment v_i . The sum of the offsets for a vertex is then normalized by dividing it by the number of vertices with weight greater than zero, W . The result is a set of vectors v_r that are located to produce a facial expression.

The formula can be written as follows:

$$v_r = \frac{\sum_{i=1}^N (v_b - v_i) * w_i}{W}$$

where i is the number that identifies a blend target and N is the total number of blends.

g) Evaluation

The model was designed in response to the need to create a methodology that would be able to quickly control and modify facial expressions for a variety of animated characters. To evaluate our methodology, we developed a system that was tested internally on eleven different facial meshes to ensure that no problems existed with regards to mesh blending. Three of these characters were selected for use in wider studies to evaluate the function of the model; this choice was influenced solely by issues of copyright and of character quality (measured by mesh resolution, texture and shading quality).

The goal of this evaluation is to ascertain whether a virtual character's facial expression (generated using the new schemes presented in this paper) is identifiable as a believable emotion. The test was carried out in relation to the work of Gombrich¹², who suggested that it is possible to interpret the same facial expression in several different ways. Furthermore, the test attempted to validate whether the model can generate believable facial expressions.

Using the new facial mimic scheme developed in our research, several animations were created by an artist, each four seconds long. Each of these animations featured a 3D virtual character, animated to present a different configuration of our schemes, and each configuration was set with the intention of representing an emotion. The character used for each animation was identical.

The animations contained rotations for the head and the pupils, obtained by applying some basic concepts¹³. The character gaze was always aiming at a point off the scene, suggesting some form of hypothetical observer or emotional trigger. Furthermore the pupil

and head rotations were always related, either by following each other or by moving in opposite directions. The model presented in this paper is focused on the action of subcutaneous and maxillary muscles and, for that reason, does not describe how pupils and head rotations have to be managed.

To test whether these facial expressions correctly represented the intended emotion, a user-based study was conducted via a web page interface. Upon visiting the page, a user was twice shown one of the generated animations. Immediately after the second viewing, the user was asked to select, from a fixed list, which emotion they thought was being represented in the animation. The user was requested to repeat this process, or *round*, a minimum of three times. The choice of the animation for each round was based on a semi-random algorithm that ensured two things: that the rounds were not repeated in a fixed sequence, and yet that each round was displayed an equal number of times (over the course of the entire evaluation procedure). It is important to note that there was no outside context associated with the presentation of the 3D model.

Nine facial mimics were used for the test, each designed to correspond to a specific expression that can represent an emotion. Moreover the chosen emotions and expressions are not related to the synthetic model itself but are necessary only for its validation process and they are not the only ones that can be achieved.

Ideally, there should be an unambiguous relationship between an expression and an emotion. However, a given emotion can be associated with more than one expression. For example, in the list below, the *expression* can be associated with the corresponding emotions:

- *Blissful*: fullness
- *Delighted*: delighted and pleased
- *Exhilarated*: happiness and humour
- *Furious*: anger and fury
- *Terrified*: fear and terror
- *Disgusted*: disgust, rejection and bad temper
- *Desperate*: hopelessness and anguish
- *Sad*: sadness and depression
- *Serene*: serenity

425 responses were gathered from at least 50 different people (anonymous responses were included in the results but were grouped as a single user). The tests were carried out in English, Spanish and Italian, by users in Britain, Spain, Italy and Argentina. Figure 4.3.7 presents the results of the test in tabular form. The rows correspond to the expressions presented in the animation, while the columns are the responses given by the users. The values in the cells represent the number of user-responses given for each expression.

The relative proximity of the emotions in the table is defined by the Whissell's wheel activation-evaluation space¹⁴ (see Figure 4.3.6). This has been used due to the proliferation of studies that uses it in relation with the expressivity of animated virtual characters^{15, 16, 17}.

The cells are shaded according to these definitions; red cells are the same expressions, orange are very closely related expressions, yellow represents loosely associated expressions, while cyan represents distantly associated expressions. Figure 4.3.8 is a dispersion graph that shows the results of the test, using the same colour scheme.

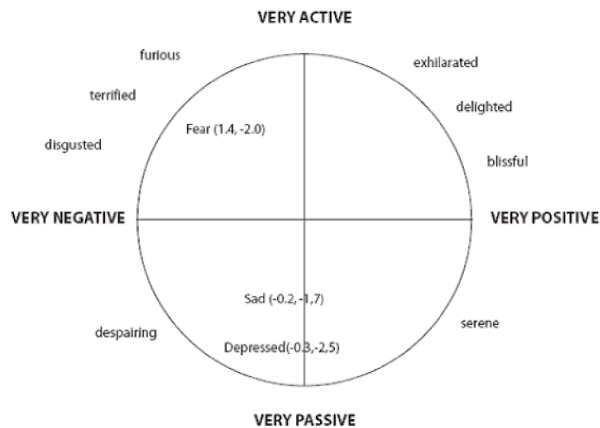


Figure 4.3.6: Whissell's activation(vertical axis) - evaluation(horizontal) space.

serene	19	6	0	0	0	1	0	2	27
sad	0	0	2	0	1	2	32	39	1
desperate	2	0	0	0	0	0	6	0	0
disgusted	0	1	0	16	4	40	5	7	1
terrified	0	0	0	1	38	0	0	0	0
furious	0	0	0	26	0	5	0	0	0
exhilarated	3	16	31	0	0	0	0	0	4
delighted	8	14	16	0	1	0	0	0	6
blissful	19	11	0	0	0	0	0	0	12
	blissful	delighted	exhilarated	furious	terrified	disgusted	desperate	sad	serene

Figure 4.3.7: Table of viewer’s test results.

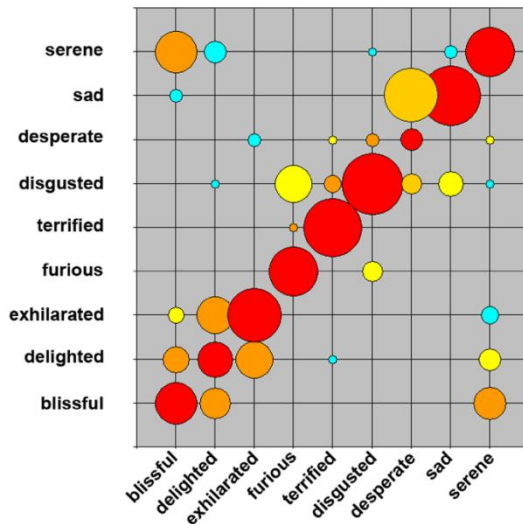


Figure 4.3.8: Viewer’s test results dispersion graph.

56.34% of the responses perfectly matched the presented emotions. However, when the closely associated emotions (shaded orange) are considered, this percentage rises to 85.92%, figure 4.3.9 demonstrates both these results graphically; for each presented expression, it demonstrates the proportion of correct/closely associated responses. The graph shows a high correspondence for expressions such as *exhilarated*, *furious*, *terrified*, *disgusted* and *sad*; while lower correspondence for *blissful*, *delighted*, *serene* and *desperate*. This result is partially explained by the fact that *blissful*, *delighted* and *serene* expressions have similar connotations (according to their activation-evaluation definitions). However, these definitions do not adequately explain the error with *desperate*, which was frequently confused with *sadness*. While common understanding of these emotions may consider them to be similar, they are relatively distant from each other in the activation-evaluation space.

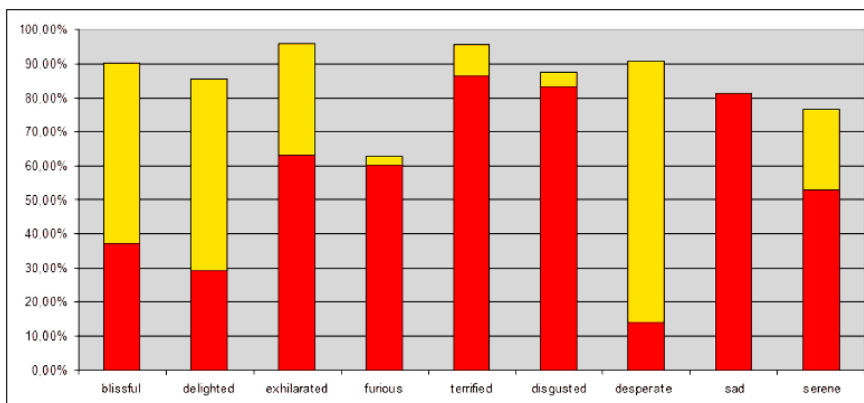


Figure 4.3.9: Per-expression results obtained with the viewer test evaluation. The bars in the chart represent ratios of matching emotional expressions recognized. Red colour indicates recognition rate for exact expression, while yellow represents the ratio of recognition as the closest emotion in the whissell wheel.

The results of this evaluation can also be viewed in relation to the work of Wallraven⁵ that present a study of perception of static and varying facial expressions obtained with 3D and 4D scanners and a facial motion capture system. Their work presents some evaluation results that are obtained using a set of data that can be compared to the one used in our viewer test. The data are relative to a human face scanned in different poses associated to different expressions

(neutral, confusion, disgust, fear, happy, sad, surprise and thinking) and the animation produced is obtained by linearly interpolating the neutral expression with the others as done in our test. The following table shows a direct comparison with our evaluation of the recognition percentages.

	Our Model	Scanned Data
Disgust	83.33	85
Fear	86.36	45
Happy*	79.38	71
Sad*	81.25	95

Table 4.3.4: Comparison of our synthetic model with the animated 3D scans of Wallraven *et al.*, measured in percentage of viewer recognition. * indicates a mean value of corresponding emotions from our model. The table shows that the expressions obtained with our synthetic model are as believable as the ones obtained with a 3D scan of a real human’s face and expression.

h) Conclusions and future work

The proposed model, using eight different facial shapes and interpolating them, represents an abstract system of the representation of different emotions across a wide variety of facial shapes. The model was internally tested on nine 3D models, aesthetically very different to each other, demonstrating that by interchanging the interpolation values between them we can create the same mimic behaviour that suggests an emotion. The results obtained from more extensive tests using three of these models (two for the animator test, one for the viewer test) show that the functionality of the scheme facilitates the set-up of effective facial animation rigs. Because of the simplicity of the facial rig required, the model represents a novel method to reduce the time taken to prepare a character for animation. Furthermore, it would be straightforward to implement the model in a procedural animation system.

The results of the animator-user test show that the model is useful to animation professionals. Considering this result and the final purpose of the proposed model, it reveals that the use of the model in automatic and real-time productions can enhance the expressive ability of virtual characters, making the whole production more believable and evocative. The results of the viewer test show that it is possible to create an animated piece using only these shapes. Furthermore the results of the evaluation show that the Pose Space Deformation proposed by Lewis¹⁷ can be applied to facial expression animation, obtaining a continuous emotional space by managing only two parameters.

The immediate future work on the scheme is to modify it to permit asymmetric facial shapes. Further work will focus on procedural processes that can help to automate both the creation and management of facial expressions. One possible implementation would be an interface to manage the facial expressions mapped into a coordinate system, such as the activation-evaluation model first described by Whissel¹⁴.

Finally the research can be extended to consider the static form of the face, making it possible to define aspects of a character's personality. Summing such personality with the variable form should make it possible to give to each different character his unique expressivity. It would be also interesting to model the artistic processes used by animators to relate the pupils and head rotations with the desired emotional status of the character. In this way it would be possible to define a new layer of automatic animation that, working with the model in its current state, could enhance richness and believability.

i) References

1. Gombrich EH. The Mask and the Face: the perception of physiognomic likeness in life and in art. In Gombrich EH, Hochberg J, Black M. *Art, perception and reality*. The Johns Hopkins University Press, Baltimore and London, 1972, pp. 1-46.
2. Ekman, P. Facial Expressions. In Dalglish T, Power T, *The Handbook of Cognition and Emotion*, John Wiley & Sons Ltd., Sussex, 1999.
3. Goto T, Escher M, Zanardi C, Magnenat-thlamann N. MPEG-4 based Animation with Face Feature Tracking, In *Proceedings Eurographics Workshop on Computer Animation and Simulation 1999*; pp. 89-98.
4. Kouadio C, Poulin P, Lachapelle P. Real-Time Facial Animation based upon a Bank of 3D Facial Expressions. In *Proceedings of Computer Animation '98 Conference 1998*; pp. 128.
5. Wallraven C, Breidt M, Cunningham DW, Bülthoff HH. Psychophysical evaluation of animated facial expressions. In *Proceedings of the 2nd Symposium on Applied Perception in Graphics and Visualization 1999*; pp. 259-269.
6. Plasencia C. *El rostro humano, Observación expresiva de la representación facial*. Publications service, Universidad Politecnica de Valencia, Spain, 1993.
7. Cuyet E. *La mimique*. Publications Service O. Doin, Paris, 1902.
8. Costantini E, Pianesi F, Prete M. *Recognising emotions in human and synthetic faces: the role of the upper and lower parts of the face*. In *Proceedings of the 10th international conference on Intelligent user interfaces*, 2005; pp. 20-27.

9. Pushkar J, Wen CT, Mathieu D. Frédéric, P. Learning controls for blend shape based realistic facial animation. In *Proceedings International Conference on Computer Graphics and Interactive Techniques*, ACM SIGGRAPH 2005.
10. Simon M. *A visual reference for artists*. Watson-Guptill Publications, 2005.
11. Faigin G. *The Artist's Complete Guide to Facial Expression*. Watson-Guptill Publications, 1990.
12. Gombrich, EH. Invention and Discovery. In *Art and Illusion: A Study in the Psychology of Pictorial Representation*. Trustees of the National Gallery of Art, Washington D.C. 1972.
13. Plachaud C. *Communication and Coarticulation in Facial Animation*. PhD thesis, University of Pennsylvania, 1991; pp. 86-90.
14. Whissell CM. The dictionary of affect in language. In Plutchik R. Kellerman H. *EMOTION: Theory, Research and Experience: Vol. 4, The Measurement of Emotions*. New York Academic, 1989.
15. García-Rojas A, Vexo F, Thalmann D, Raouzaïou A, Karpouzis K, Kollias S, Mocozet L, Magnenat-Thalmann, N., 2006. Emotional face expression profiles supported by virtual human ontology. In 2006. *Computer Animation and Virtual Worlds*. John Wiley and Sons Ltd. Vol. 17, pp. 259-269.
16. Rausaiou A, Karpouzis K, Kollias S. Online Gaming and Emotion Representation. In *Proceedings of the International Workshop on Very Low Bitrate Video Coding (VLBV) 2003*; pp. 295-320.

17. Lewis JP, Cordner M, Fong N.. Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. In *Proceedings of 27th annual conference on Computer graphics and interactive techniques 2000*; pp. 165-172.

j) An emotional interface for the synthetic model

As introduced in section 4.3.7, it is possible to arrange the facial expressions obtained with our synthetic model on an activation/evaluation space as the one described by Whissell. Here we describe an experiment carried out to determine the feasibility of animating emotional facial expressivity through a user interface inspired by the Whissell wheel.

Activation/Evaluation. *Activation* represents the strength of the current expression, while *Evaluation* specifies the positivity or negativity of the expression. In facial expression terms, the concepts are related, yet separate. For example, an open mouth is indicative of a highly active face, yet it is independent of whether the expression is one of joy or sadness. Put in colloquial terms, evaluation can be thought of as the ‘happiness’ factor, while activation determines the expressivity of the current evaluation level.

Both activation and evaluation can thus each be mapped onto perpendicular axes, and assigned numerical values ranging from -1 to +1. On the activation axis, a value of +1 is characterized, for example, by an open mouth, open eyes and raised eyebrows (i.e. an active face); while a value of -1 is characterized by a closed yet slightly sagging jaw, and low eyebrows and eyelids (i.e. a passive face). On the evaluation axis, a value of +1 is characterized by an expression that could be interpreted as ‘happiness’ (e.g. upturned lips), while a value of -1 is characterized by an expression that could be interpreted as ‘disgust’.

The application of numerical activation-evaluation values to facial expressions has several advantages. The first one is that it decouples expressions from emotions, so that expressions that are mapped to the two axes are not necessarily related directly to emotions. This

distinction is important as it allows the definition of an emotional space within which the character’s face can move. The second advantage is that this emotional space is analogue rather than discrete. There is no strict definition of what “a happy face” or “an angry face” consist of. Instead there are clusters of activation-evaluation values that result in expressions that confer an emotion. The third advantage is that this lack of discrete mapping between expression and emotion allows the gradual modification of an expression as a real-time response to an external input (via, for example, an artificial intelligence algorithm controlling character mood). For example, a slight improvement in character mood could be represented by a slight increase in the evaluation value (rather than changing the facial expression directly to “happy”). This allows for an increased layer of subtlety when designing virtual character response.

Interface design. Based on activation/evaluation concepts, we have designed an emotional wheel to represent the possible combination of the two values (see Figure 4.3.10). The wheel uses a coloured gradient scheme to represent the axes; red/cyan for +/- activation, and white/black for +/- evaluation. Figure 4.3.10 also contains overlaid some expected emotions that should be deduced from different combinations of activation-evaluation values. For example, characters with high activation and high evaluation can be expected to appear ‘delighted’, or perhaps ‘exhilarated’; whereas characters with low activation and low evaluation should appear more ‘hopeless’ or ‘depressed’. The current activation-evaluation level is indicated by a marker, which doubles as a tool which the user can move to change the values (as described below).

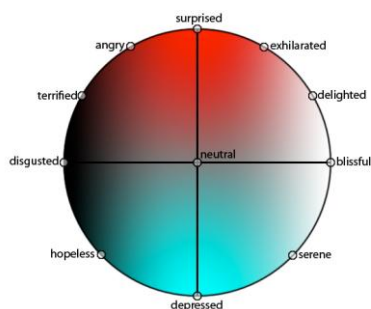


Figure 4.3.10: Our defined wheel for Activation (vertical axis) and Evaluation (horizontal).

Implementation. As mentioned before, the main advantage of the activation-evaluation system is that it decouples expressions from emotions. Thus our implementation assigned a series of extreme facial poses to different areas of the wheel. Table 4.3.5 shows a summary of these poses:









	
1- mouth up	2- mouth down
	
3- lips pulled in	4- lips puckered
	
5- eyebrows down	6- eyebrows up
	
7- eyebrows curved	8- jaw open

Table 4.3.5: Extreme poses of the face, and the Activation-Evaluation values assigned to them. See Table 4.3.4 for interpolation values.

These poses are created by a modeler/ animator and exported as a series of morph targets before insertion into the activation evaluation space. Table 4.3.6, at the end of the paper, shows the precise mapping of the interpolation of the poses (along with an “eyelids shut” pose). Any combination of activation-evaluation values will proportionally blend the morph targets of the various

poses (according to Table 4.3.4), resulting in a unique expression for every combination. Alternatively, if the animator is using a bone-rig animation system, the poses are stored by recording the transformations required (for each bone within the rig) to recreate the pose. The advantage of this second method is that it can be combined with automatic rigging systems such as the Maskle (section 4.2).

act.	eva.	1	2	3	4	5	6	7	8	eyes
0	0	0	0	0	0	0	0	0	0	0
0	1	0.138	0.075	0	0.675	0.3	0.056	0.2	0.216	0.3
0.5	0.866	0	0.7	0	0	0	0.53	0	0.763	0
0.866	0.5	0	1	0	0	0.6	0.346	0.732	0.779	0
1	0	0.065	0	0.344	0.344	0.7	0	0	1	-0.3
0.866	-0.5	0.391	0.57	0.591	0.462	1	0	0.981	0.077	0
0.5	-0.86	0.92	0.527	0	0.757	0.25	0.989	0	0.366	-0.6
0	-1	0.527	0	0.441	0.531	0	0	1	0	0.6
-0.71	-0.71	1	0	0	0	0.5	1	0	0	0.6
-1	0	0.995	0	0.225	0	0	0.996	0	0.996	0.2
-0.71	0.707	0.138	0.075	0	0.675	0.3	0.38	0.05	0.216	0.3

Table 4.3.6: Interpolation values for the poses described in table 4.3.5.

The next figures show screenshots of the implementation working in separate environments. Figure 4.3.11 shows the interface working within Autodesk Maya, and Figure 4.3.12 shows it implemented as a standalone software program. We also implemented this system as a plug-in for Autodesk 3D Studio Max. Once the morph targets have been set, the animator is able to move the control point within the ‘emotional wheel’ defined by the model. This allows the animator to rapidly prototype, view, and test various different expressions for use in an animated clip. Figure

4.3.13 demonstrates the capability of the interface to change according to real-time input, and how it can be used by people not trained in the use of computer animation tools. The control point location can be changed either via a mouse or via a USB control pad, thus changing the character expression in real-time, and without the need for any further use of animation software.

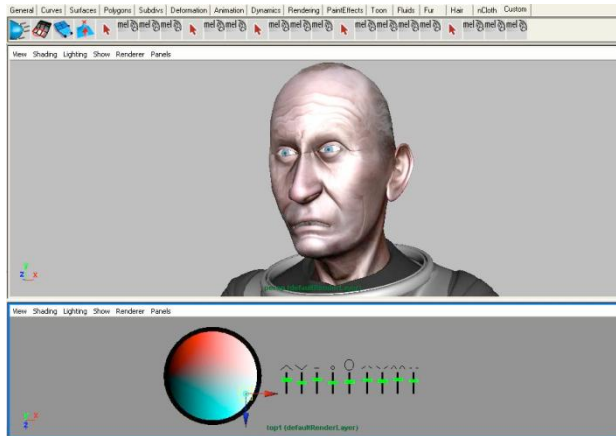


Figure 4.3.11: Screenshot of interface as a Maya plug-in.



Figure 4.3.12: Screenshot of a standalone interface controlling a character's face in real-time.

k) Evaluation of the interface

The interface described in 4.3.10 was designed in response to the need to create a technique that would be able to quickly control and modify facial expressions for a variety of animated characters. The system was tested by the GTI team internally on eleven different facial meshes to ensure that no problems existed with regards to mesh blending. Two of these characters were selected to be used in a wider study to evaluate the functionality of the model. The choice of characters was influenced solely by issues of copyright and of character quality (measured by mesh resolution, texture and shading quality).

The evaluation was carried out with 18 animation Masters students as users, and took place during a class focused on facial animation. At the time of the evaluation, the students were already trained in the use of standard facial animation practices [Osipa 2010]. Two 3D characters were provided to the students, each setup for use with the interface. The students were asked to choose a character and were given approximately two hours to create a small animated sequence of a close-up face using the interface. The system was presented to the students as a plug-in for Autodesk Maya. After creating their animation, they were asked to fill-in a questionnaire, which consisted of two questions:

1 - “Were you able to recreate the facial animations that you desired?”

2 - “If not, name three facial expressions that you would have liked to achieve but you could not due to the limitations of the model.”

Of the 18 users, 13 were satisfied with the result of creating a short clip using only the expressions generated using the activation-evaluation interface. However, five of the users highlighted limitations of the interface. The main criticism was focused on the issue of asymmetric poses. Four users explicitly stressed the problems caused by the absence of a way to create an asymmetric facial pose; while two pointed out the impossibility of producing poses that suggest both ‘irony’ and ‘suspicious’, expressions that are frequently represented by an asymmetric facial expression. Further feedback described certain expressions (“open-mouth

smiling”, “blinking”, “disgust”, “silly happiness”, “neutral” and “happiness”) as being impossible to achieve. Yet our internal tests have successfully demonstrated the possibility of replicating these expressions. For graphical representation of the results, the answers of the questionnaires were collated and sorted into three values: bad, sufficient and good. A bad response was one that specifically criticized the model; a good result specifically praised the model, while responses that made no comment either way were classified as sufficient. While this classification does not represent an accurate statistical representation of the data, it is useful to enable an overall graphical view of the results (see Figure 4.3.13). The figure shows that the majority of the testers (72%) found the system to be at least adequate for the purposes of creating an animation sequence.

The results of this small evaluation enable several conclusions to be drawn. In general, the interface was considered intuitive and facilitated the facial animation of a character in a short clip. However, the fact that over a quarter of the users highlighted the lack of asymmetric facial expressions highlights a gap in the technique, although one which should be possible to address in the future. Of more interest were results that indicated an inability to create desired expressions emotions using the interface, even though our tests suggest that the majority of the problematic expressions were indeed achievable. This is perhaps to be expected, given that the aim of the interface is to treat expressions and being independent of emotions. Yet what is undeniable is that animators wish to animate emotions, and thus perhaps the interface needs more visual guides to facilitate this.

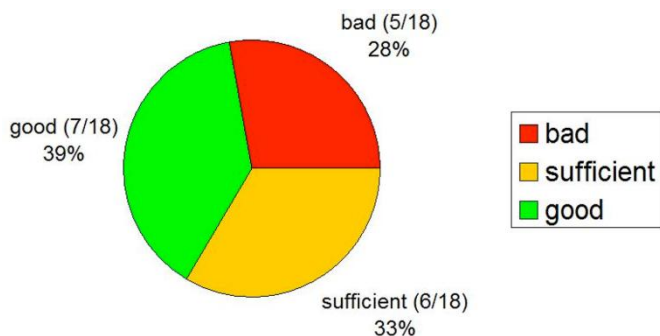


Figure 4.3.13: Chart showing overview of evaluation results.

Our further work now has two foci: improving the implementation and improving the evaluation. The implementational work will be to extend the model to include asymmetric poses, though the exact details of how these will be mapped within the emotional space are yet to be made explicit. Furthermore we will look into creating more visual guides for the interface design to improve user familiarity. However we are also interested in extending the evaluation of the interface in an attempt to measure the efficacy of its other main feature, the ability to gradually modify the activation-evaluation state of a character through time. Ideally the evaluation for this feature should centre on linking the interface with artificial intelligence algorithms that control a virtual character's mood, with the goal of better conveying subtle changes of mood to the viewer.

4.4. Industrial evaluation of the Maskle (4.2) and the facial expression animation system (4.3)

As indicated in the introduction of this chapter, section 4.1, the results of 4.2 and 4.3 were evaluated in industrial environments. In this section we discuss the evaluation with more detail.

a) Industrial evaluation of the Maskle

The *Maskle* was evaluated in three experimental productions within the SALERO European project [Salero 2006], showing productivity benefits. In one of them the Maskle was used to produce a 3-minute video of a virtual character to appear on television, whose face was animated by applying a set of poses onto the rig obtained with the Maskle. Commercial products do not incorporate yet any similar tool/technique. The following table is extracted from the D9.6.3 SALERO deliverable [Third Phase Experimental Productions and Evaluation Report SALERO 2009], and reflects the industrial evaluation, broadly in terms of a SWOT diagram.

Halley sequence on TV	
Strengths <u>Usefulness of the Technologies</u> Maskle works very well on broadly human face geometry, although less effective with highly stylized characters (which is not surprising).	Weaknesses <u>Interoperability and Integration</u> Struggles somewhat with non-human characters and when we last tried it seemed to have some difficulties with very dense geometry. (this may have been resolved).
Opportunities Plug-in for Maya and 3DS max – the technology seemed very close to the point at which marketing this commercially was possible.	Threats Any sudden dramatic improvement in facial skinning in Max and Maya (not anticipated).
Similar technology Unable to find any obviously similar technologies	

Table 4.4.1: Evaluation results of the use of the MASKLE in a professional production context (extracted from SALERO deliverable *D9.6.3: Third Phase Experimental Productions and Evaluation Report*, p. 35).

The Opportunities and Threats presented in the table were meant for a possible commercial product. Since then, the commercial offer has not changed, and there have not been industrial derivatives of further skinning research.

The main concern shown in the evaluation was that the system would not be suitable for non-human characters. The tool was provided with just one mask template, for human-like faces and it would be easy to produce other templates to fit specific production needs. In the evaluation it is also mentioned that the system was not performing well for very dense meshes, but unfortunately this issue was never properly reported and was never mentioned in later discussions, which might suggest (as indicated in the table) that was related to an early prototype of the system.

The Maskle was compared to manually creating blendshapes for a facial rig in 3DS Max 2010 [3DSMax 2015], with productivity

increment estimated at 80% without visible qualitative difference. These high productivity gains were probably motivated mainly by the immediacy of using the Maskle.

In the other two experimental productions, the system was used but the evaluation was focused on the use of its output, rather than the use of the system itself. In section 4.4.2 I discuss the evaluation of the Facial Expression Animation System that was applied on top of the resulting rig from the Maskle. In this case some weaknesses are mentioned as the impossibility to perfectly match the desired facial expressions and the lack of flexibility; in both cases this can be attributed to the deliberately simple setup. The third experimental production also indirectly addresses the system, as the resulting facial rig from the Maskle, was used to create a set of facial expressions that were controlled by the Facial Expression Animation System through the Programme Editor, which we introduce and discuss in the next chapter.

b) Industrial evaluation of the facial expression animation system

The Facial Expression Animation System using a sort of Whissel wheel discussed in 4.3 was used and evaluated in the SALERO experimental production *Alan01* [Tuomola et al. 2009][Alan01 2010], which consisted of an interactive installation about Alan Turing. The installation enabled visitors to interact with a virtual Alan Turing through a set of symbols which triggered sequences of video projections of the Alan Turing face. Some videos were shootings of an actor (Hannu Kivioja), while others were renderings of a three dimensional character, whose face was animated using our Facial Expression Animation System.

Table 4.4.2 below is taken from the same SALERO deliverable mentioned in the previous section. It takes a SWOT approach for the evaluation. The table shows that the use of this technique produced a mixed response from artists working on the production. They found it useful and easy to use, but not flexible enough at times. The lack of flexibility was mostly related to the limited range of motion provided by the underlying Maskle rig but also relates to

the impossibility to manually modify the deformed mesh (this could have been overcome by the artist creating a further geometry to apply the tweaks). A more complex Maskle template would have made the used rig more flexible but it would have made the system more complex.

Alan01 Installation	
<p>Strengths</p> <p><u>Quality of resulting media</u> We were able to create emotional states of the faces with the tool.</p> <p><u>Usefulness of the technologies</u> The tool enabled us to create the machine-like emotional animations for the installation.</p> <p><u>Usability of the technologies</u> The tool was very useful in creating the animations and it saved time in our workflow.</p> <p><u>Interoperability and integration</u> The tool offered an integrated interface inside 3dsmax, it was possible to use it as a plugin. We were fully able to use the tool in creating our assets for the installation.</p>	<p>Weaknesses</p> <p><u>Quality of resulting media</u> The resulting emotional states created with the tool did not match exactly with the storyboard. For some parts it was impossible to move the vertices to desired positions. This problem could be solved by improving the part of the tool that is used to map the vertices.</p> <p><u>Usability of the technologies</u> Creating a variety of emotional states would require more points in the face that one could animate. Interoperability and integration The installation process inside 3dsmax was quite intuitive, but in the future it could be fully automated.</p>
<p>Similar technology There is no similar technology available. It would be very useful for the industry to have Facial Animation Toolset finished.</p>	

Table 4.4.2: Feedback received from artists using the Synthetic model for facial expression to create the installation Alan01 (Extracted from SALERO deliverable D9.6.3: *Third Phase Experimental Productions and Evaluation Report*, p. 56).

Finally, 96% of the enquired spectators visiting the installation, rated the quality of facial animation as excellent, confirming the results of the online enquiry described in 4.3.7. This shows that, even if the artists could not exactly achieve the desired expressions, the obtained ones were convincing for the audience.

4.5. References

3DS Max, computer software. (2015). Available from: <<http://www.autodesk.com/products/3ds-max>>. [13 December 2015].

Alan01. (2010). Available from: <<http://mlab.taik.fi/alanonline/#>>. [13 December 2015].

Aristidou, A., Charalambous, P. & Chrisanthou, Y. (2015). Emotion Analysis and Classification: Understanding the Performers' Emotions Using the LMA Entities. *Computer Graphics Forum*, vol. 36, no. 6, pp. 262-276.

Barcelona Media, 2010. Available from: <http://www.barcelonamedia.org/report/the-i3media-project-reaches-successfully-its-first-year-research-objectives_231>. [13 December 2015].

Densley, D. & Willis, P. (1997). Emotional Posturing: A Method Towards Achieving Emotional Figure Animation. *IEEE Computer Animation 1997*. IEEE Computer Society, Washington DC, pp 8-14.

Evans, A., Romeo, M., Bahrehmand, A., Agenjo, J. & Blat, J. (2014). 3D graphics on the web: A survey. *Computers and Graphics*. vol. 41, no. 0, pp. 43-61.

Faita, C., Vanni, F., Lorenzini, C., Carrozzino, M., Tanca, C. & Bergamasco, M. (2015). Perception of Basic Emotions from Facial Expressions of Dynamic Virtual Avatars, *Lecture Notes in Computer Science*. Springer International Publishing, Switzerland, vol. 9254, pp. 409-419.

Hyde, J., Kiesler, S., Hodgins, J. K. & Carter E. J. (2014). Conversing with Children: Cartoon and Video People Elicit Similar Conversational Behaviors. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, pp. 1787-1786.

Meredith, M. & Maddock, S. (2005). Adapting Motion Capture Data Using Weighted Real-Time Inverse Kinematics. *ACM Computer in Entertainment*, vol. 3, no. 1 pp. 5-5.

Noh, J.Y. & Neumann, U. (2001). Expression cloning. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, New York, pp. 277-288.

Plutchik R. & Kellerman H. (2015), The Measurement of Emotions, *Emotion: Theory, Research and Experience*, vol. 4, pp. 113-129.

Romeo, M., Dematei, M., Bonequi, J., Evans, A. & Blat, J. (2010). A reusable model for emotional biped walk-cycle animation with implicit retargeting. *Proceedings of the 6th international conference on Articulated motion and deformable objects*. Springer-Verlang, Berlin, pp 270-279.

Saito J. (2013). Smooth contact-aware facial blendshapes transfer. *DigiPro '13 Proceedings of the Symposium on Digital Production*. ACM, New York, pp. 7-12.

Salero: Semantic Audiovisual Entertainment Reusable Objects. (2006). Available from: <<http://www.salero.eu>>. [13 December 2015].

Sumner, R.W. & Popović, J. (2004). Deformation transfer for triangle meshes. *ACM Transactions on Graphics*. vol. 23, no. 3 (Aug.), pp. 399–405.

Third Phase Experimental Productions and Evaluation Report SALERO. (2009). Available from: <<http://salero.eu/en/resources/deliverables.html>>. [13 December 2015].

Tuomola, M. L., Korpilahti, T., Pesonen, J., Singh, A., Villa, R., Punitha, P., Feng, Y. & Jose, J. M. (2009). Concept, content and the convict. *Proceedings of the 17th ACM international conference on Multimedia*, ACM, New York, pp. 1063-1072.

Whissell C. M. (1989). *The Dictionary of Affect in Language*.

5. FULLY AUTOMATED ANIMATION PRODUCTION: THE QUEST FOR THE MAGIC BUTTON!

The research in this chapter was undertaken at the verge of web video contents, multimedia messages in 3G mobile phones and the dawn of Digital Terrestrial Television. These technologies required more contents to be produced and delivered faster, with reasonable quality, but perhaps not the same as that of high-end cinema production. This chapter focuses on automating the complete production process of short animation TV and web programs by making use of pre-made clips and templates.

Templates are used in a lot of areas, from word processing to software engineering. In our case, templates express a pattern for a (short) television-like programme as weather forecast and sport news, which are detailed in the following sections. This pattern is then brought to life by means of an automatically animated 3D character. A simplified template could be: *Hello, Block_1, Block_2, Block_3, Goodbye*. Pre-made clips are used to represent a few variants of the different elements of the template, obtained by importing motion captured data or handcrafted animations. In this chapter I will describe how this approach was used to generate web content, multimedia messages, and short TV programmes for a period of more than five years. The templates can be designed to be dynamic, by adding variables to it. Thus, it is possible to use external data (e.g. weather forecast data) to drive parts of the template. A system using these templates could then be automated to generate a set of videos at selected times, with varying results depending on the data injected into the variables.

As it is demonstrated in this chapter, the automation can be extended from the creation of the contents, to the creation of the rendered result. In the chapter I present two research works that have successfully been used to automatically produce media contents for the web, for multimedia messages and television:

1) *Automated template-based production*

Automating the complete production process had been seldom addressed in CG production, although more work exists on video editing, as discussed in 5.1.

In CG media, the key factors to reduce the production time are the simplification of the three-dimensional assets and animation creation, plus the acceleration of the rendering and editing processes. The solution proposed in this chapter addresses these issues through a system inspired by videogame technology, i.e. creating real-time 3D graphics based on GPU use. The system can blend animation clips together, apply appropriate transformations on 3D geometries, shade and render them in real-time. The system can also apply audio on rendered videos, which can be processed to automatically synchronize the facial animation and the audio. These features were all integrated within the GTI (Grup de Tecnologies Interactives, Universitat Pompeu Fabra) framework, a collection of Open Source libraries for different aspects of real-time interactive 3D graphics.

Another aspect to be taken into account is how the animations and other contents have to be selected and combined to provide a narrative context to the produced media. In fact, a lot of time can be spent in defining the narrative of a video and deciding which assets better communicate the intended message. In this case we provided a template-based system that enabled production companies to prepare templates that can be fed with live data to properly choose the contents which are suitable, to generate and display the 3D animations. These templates could be prototyped through an interactive system branded *Programme Editor* that enables non-technical users to access all the features of the GTI framework. The system can preview videos of the contents selected by the user, set the variables that drive the logic behind their automated selection and finally render a video or store a template. The system uses PVML, our own variant of SMIL, as discussed later.

This work was published in *Advances in Computer Entertainment (ACE)*, 2009 and is the result of a few years of development. Through these years, I contributed in many ways by driving the changes to its overall design and interface to accommodate real

production requirements, iteratively testing the system, and championing its adoption in different projects. After the publication, I added features to the underlying GTI framework, integrating the contributions described in sections 4.1 and 4.3.

Following the publication of our work, several systems have appeared to ease the creation of high-quality interactive and real-time CG media. Systems like Unity [Unity, 2015] and Unreal Engine [Unreal Engine, 2015] have rapidly conquered the market and are now the platforms of choice. In theory, it would be possible to re-implement the functionalities of our system, using any of the aforementioned engines, as briefly demonstrated in the work of Bídía et al. [Bídía et al, 2011]. In this work, the authors integrate a system to build narratives (focusing on the creation of learning material) within Unreal Engine.

2) Automatic generation of Sign Language

The second research I present is related to automated animation production in a specific area which has attracted a lot of research, that of Sign Language (such as the different European projects ViSiCAST, eSIGN, CogViSys, HamNoSys, Dicta-Sign). This work mainly focuses on the automatic generation of sign language videos with virtual characters. As a system, its approach is similar to that of the Programme Editor, using templates. The domain is specific as the system generates signed sport news, based on heterogeneous raw data collected from multiple sources. This system was created to provide accessible news to the deaf audience of the Barcelona World Race sailing regatta around the world and the resulting videos were video streamed in the Spanish National Radio (RNE) website during the approximately 4 months of the regatta duration. There are different levels of evaluation of its results. Two of them are relevant for the system evaluation:

- 1) The system was successful in producing automatically animation clips, being able to run publicly during the event.
- 2) The textual transcript of the automatically generated news was checked by RNE journalists who did not suggest any changes, proving the soundness of the news generator.

More importantly, in terms of sign language understandability, and complex animation:

3) There was an indirect proof that the sign language was understandable in terms of the popularity of the videos in the web site (as shown below).

The research, which is the content of 5.2, was presented in *GRAPP 2014*.

As indicated earlier, Sign Language performed by avatars has attracted a significant amount of research. Evaluation of the understandability is complex, and the different systems that have been researched have received mixed ratings [Kennaway et al. 2007]. In section 5.3 we present a more direct, detailed evaluation of the Sign Language which our systems were able to generate, specifically in terms of the understandability related to the facial expressivity. In the same section we discuss the limitations of this evaluation, and plans for a further round of evaluation which we were unable to carry out for a number of reasons. The section has not been published, waiting for this round of evaluation to be performed and analysed.

Recent work from McDonald et al. [McDonald et al., 2015] focuses on improving the quality of signing virtual characters. It confirms our belief that hand-crafted animation of signs by an animator with experience in sign language (with sign language expert supervision in our case) produces the best results. However, it also confirms that it is a time-expensive process to build a library of signs with this method and suggests a way to improve simple animation by introducing procedural tweaking. In this way it should be possible to reduce the amount of work required to animate the signed clips and let the procedural system add details. The paper also presents an extensive evaluation work which would be interesting to reproduce with our animations, to test how both compare to each other.

5.1. Assisted animated production creation and programme generation

The creation of animated productions is a labour intensive process. Whether the end result is a large-budget motion picture, or a small-scale internet production, there is invariably a large amount of time spent in creating the timeline, arranging assets, previewing and editing. This iterative process is necessary in large-scale productions but can become repetitive and frustrating when the end result is a small production that may have similar elements to previous work. We present a workflow system and framework that are able to both greatly facilitate animated programme production and introduce an element of procedural generation. We further present the Programme Editor, an application designed to be a powerful front end for the framework. The principal contribution of this work is the creation of an XML-based scripting engine that can be used to create an animated production. This permits several techniques, tools and workflows to interchange information, allows rapid incorporation of further tools, and furthermore facilitates the complete automatisisation of the production process.

a) Introduction

Since the idea of convergence arose in the early 1990s, the media industry has looked at cross-platform exploitation methods as a way of producing more exciting content, more cost-effectively. Yet while technology has helped to produce better quality sounds and images, the costs continue to rise. Digital media production remains very labour intensive, making it a very high-risk, high-cost industry. One of the reasons is that productions are crafted, almost without exception, at very low levels, in order to better satisfy artistic needs. Indeed, in many applications, the existence of more sophisticated digital tools has actually pushed up costs, as more time is spent on complex off-line processes in the quest for quality.

This leads to the concept of “re-usable” media, content that can be repurposed and used again for a different production. A classic example of this has existed for years, with films and television productions being dubbed into different languages, but what about

situations where the actual *content* of the production should change? This is the situation that is addressed in the paper, with the presentation of a framework for assisted creation of animated productions and the automatic generation of programmes that vary in visual and audio content.

The work presented can be split broadly in two aspects: the first is an introduction of the multimedia production scripting *framework* which allows fine control of the creation of an animated production or clip. The principal contribution of this work is enabling aspects of animated production to be *automated* and *re-used* according to external data. The framework also allows multiple reuse of assets and semi-automatic programme creation, and the complete separation of content from rendering.

The second aspect deals with the creation of an advanced editing application, the ‘Programme Editor’ for rapid prototyping and programme setup, and designed specifically for use with the framework. The Programme Editor is capable of rapidly adjusting different aspects/sections of a programme (for example, camera type, actor locations, audio, and provide on-the-fly previewing of the results. While essentially acting as a front end for the programme scripting and generation aspects of the framework, the Programme Editor is nevertheless a powerful application in its own right, using a custom 3D graphics engine to display results in real time (see Figure 5.1.1).

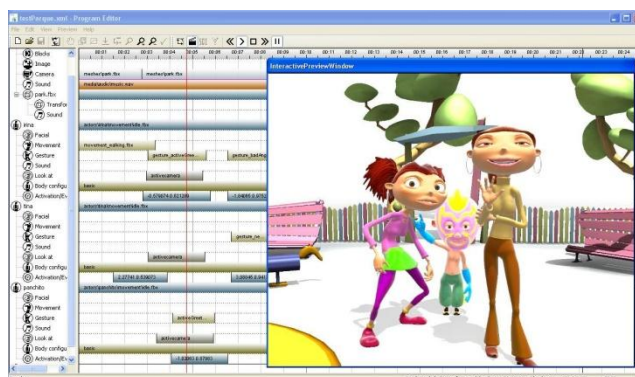


Figure 5.1.1: The Programme Editor and real-time preview window.

The paper is structured in the following order. Following a discussion of related work we present in detail the framework for assisted programme generation. We then present the Programme Editor application, before providing several example of the use of our work in both the commercial and academic fields. Finally we conclude that the presented work is a novel and powerful methodology for assisted animated production creation and programme generation. The professional use that the framework has already seen suggests that it provides real benefit to production workflow, and provides a strong platform for the integration of novel multimedia technology.

b) Related work

Perhaps the most similar work to our framework is the Synchronized Multimedia Integration Language (SMIL) [1]. SMIL is a W3C recommended markup language for describing multimedia presentations. It is written directly in XML, and defines markup for layout, animation, timing, visual transitions and embedded media such as text, audio, images and video. SMIL is the basis for the more famous Multimedia Messaging Service (MMS), which is now ubiquitous within mobile telephony, and was one of the underlying technologies for the now defunct HD-DVD format. SMIL, and the possibilities of its use, were a strong inspiration for the work in this paper, which can be thought of extending SMIL to provide more complete cover for animated production.

In terms of video editing applications, there are several companies that provide similar instant editing capabilities. Redboard [2] is storyboarding and prototyping technology, designed to enhance existing workflows by integrating the traditional skills of the storyboard artist into a CGI workflow. Their system consists of a powerful computerised storyboarding tool coupled to a CGI renderer, allowing artists to transfer their ideas directly to the screen. Redboard estimate that the increased communication between directors, producers and artists permitted by their system leads to a reduction of up to 40% in storyboarding costs. Redboard has already underpinned in-house productions, and it is now launching for licensed use.

A slightly different approach is taken by French company Xtranormal [3]. Instead of targeting high end users, they have developed an online system that allows novice users to “make movies”, using several preset combinations of backgrounds and characters. The browser-based technology allows users to create simple 3D animated clips with multiple characters, employing text to speech software to provide audio files for the characters to utter. Once configured, the clip is rendered and downloaded to the browser for viewing and sharing on online video sites such as YouTube.

Despite the technological accomplishments of commercial software, there is a lack of automation and possibilities for connection to external data. Indeed, the task of automatic animated production generation is not that has seen that much previous work in the academic field either. Assisted video editing, of course, is a field that has seen much work, with Girgensohn *et al.* [4] providing a recent example of advances in the field, backed up by powerful commercial applications such as iMovie [5] and Premiere [6]. For a more theoretical overview of usability of visual programming environments, the reader is directed to Green and Petre [7].

c) Workflow overview

5.1.3.1 Programme scripting via XML

The XML scripting framework, while designed from scratch, was inspired by the scripting philosophy behind SMIL [1], particularly with regards to self-reference of contents with respect to a time-line. The main difference is that the scripting XML explained in this section makes use of the concept of different “content tracks” that allow relative referencing (i.e. not direct referencing in time). This allows the system to link any content without restriction. In SMIL these concepts do not exist and are approximated with production flows which lack the same flexibility.

The introduction of this relative referencing means that our scripting framework is much more suitable for programme scripting than SMIL. In this section, we describe in detail the different elements that are used within the framework.

The timeline. The workflow scheme we have defined for the generation of programmes describes such programmes as a set of clips and relations between clips. Each of these clips and relations are defined using XML, and placed with reference to a common base timeline. The timeline is the basis of the programme and used to tie the programme together into a coherent whole.

Programme components. Each clip represents a component of the programme, examples of which are:

- camera location and direction
- virtual character location and orientation
- animation clip (for any component, including cameras)
- audio file
- background image

Each component can therefore be regarded as its own individual entity; it can be placed at any location along the timeline and can be moved forwards or backwards in time, simply by changing one variable within the XML template. Furthermore, due to the fact that each clip is merely an XML file, once generated it can be shared and used among different programmes. Thus it is possible to build up a variety of generic clips that can be re-used for several purposes.

Time dependency and relations. The relations between the components are also defined using XML (with reference to the base timeline). The relations create a series of time dependencies that indicate time constraints between the components. This ensures that there are no clashes between similar types of component, while allowing components that do not affect each other exist at the same point along the timeline. For example, one clip may specify that, between the time $t=10s$ and $t=20s$, the virtual camera should look at a certain point in the scene. This relation for this component specifies that, during the same period of time, it is perfectly possible

for an audio component to exist at the same point on the timeline; however, another camera may not overlap.

Independence. The main advantage of the scripting schema that we have developed is that it is completely independent of the rendering of the programme. This allows several important benefits when programme rendering is considered

A programme description can be rendered to different resolutions and qualities, yet still remain the same programme. The programme description specifies what is happening, not how.

The time dependencies indicate time constraints between actions.

It is possible to change the representation of the components and obtain the same programme with different assets. For example, a programme description might indicate that there is a component of type actor, which at time $t=10s$ raises a hand. The current programme might also specify the actor is called 'Actor1', and thus use meshes, textures and animation from 'Actor1'. Yet if we wish to swap to a different actor, there is no need to re-script the programme. It is possible to change the name of the actor to 'Actor2' and still keep the fact that at time $t=10s$ s/he will raise her/his hand. The visual representation of the actor and how each actor raises the hand is kept independent of the programme design stage, and moved to the rendering process.

Block template. As described above, the work of facilitating the process of programme generation is based around the concept of defining a set of components and the time dependencies between them. Block templates assist further with this facilitation by grouping components together and allowing for modification according to some external input. For example, a block template for a programme showing today's weather might consist of an actor, background and camera, linked on the timeline along with the weatherman character's associated gestures and audio files to explain the weather.

The principal advantage of saving configurations of components in this manner is that the individual components can be changed according to external factors, thus creating a different programme based on data that may come from database, or from user input. In the example of the weatherman, typical components that could be changed according to the current weather forecast would be the background image of the scene and the audio file that the actor would recite. A further example is that of a user interaction – users could change the appearance of a character, or write a sentence for the character to utter (via a text-to-speech engine). Further possibilities include the ability to dynamically change actor gestures or facial expressions depending on the input (although this would require artificial intelligence beyond the what the framework currently offers). The framework also supports the use of changing variables within block templates, for example for positioning objects within the scene. These variables can be set to be constant, or vary along certain paths, or even to be completely random. An example may be a bird flying through a scene, and taking a different route each time the scene is replayed.

In summary the grouping of components into block templates allows the output of the framework to be dynamic. In this sense it is a dynamic scripting framework, allowing the possibility for programmes to be scripted while retaining (and encouraging) the change in variables and components according to external factors. The fact that the framework is based around a well-defined XML schema allows such changes in scripting to be carried very easily, as is described in Section 5.3, and demonstrated in several of the use cases described in Section 5.5.

5.1.3.2. Programme creation and editing

There are four main methods of creating a programme or scene using the framework. The most basic method is manual creation. In this sense the framework can be thought of as an extremely high level scripting language, capable of creating scenes, animations and interactions along a timeline. While manual creation offers precise control over a scene, the disadvantage is of course the time involved in writing code by hand.

A more powerful option is the use of software designed specifically to replace the need for manual coding. The requirements for this software are that it should be intuitive enough to be used without the need for detailed training, yet be powerful enough to be able to modify scenes to a high level of detail. To this extent we have developed the Programme Editor, a powerful yet straightforward visual timeline editor. The Programme Editor is explained in detail in Section d.

An extension to the Programme Editor is the possibility of creating a web-portal for modification of a scene. For example, once a block template is created using the Programme Editor, the variables within it can be changed by users, via a web-portal, and the results encoded as a video and downloaded by the user. An example of this has been mentioned above, that of using a text-to-speech engine to allow users to specify an utterance for a character (see Section g).

Finally, the framework allows easy integration with more powerful visual editors such as 3D Studio Max (see Figure 5.1.2), allowing the exporting and use of assets and textures in a straightforward manner.

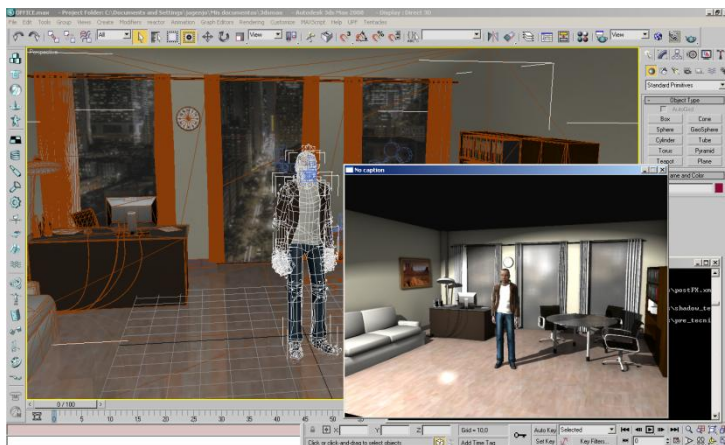


Figure 5.1.2: 3D Studio Max direct export to the framework.

Once the programme description is generated, it is then passed to the rendering software for visualisation. Visualisation takes place either in real-time or as a video for various devices.

Real time. The framework comprises a powerful shader-based rendering engine with sophisticated culling and lighting effects. The engine supports real-time update of a scene, for example if a component is added to a scene, or an existing component modified, the scene description is seamlessly reloaded and the component appears within the scene. This is particularly useful when prototyping how a scene will appear, as it enables rapid addition/modification of components without the need for the engine to reload each time. For example, character appearance or gestures can be modified quickly in order to quickly deduce which are more appropriate for the scene; or random changes of variables can be quickly tested to ensure that they are coherent. Furthermore, this visualisation allows real-time interaction with the scene, in the form of users being able to move objects and the camera (although this is not so relevant to the generation of programmes).

The real-time renderer works via OpenGL. It is available both as a direct executable and as a plug-in for internet browsers (such Firefox and Internet Explorer), and runs on a variety of platforms (included handheld platforms such as the iPhone).

Video generation. Using the graphics engine, the framework can generate videos based on the programme description. As mentioned above, this area houses one of the most powerful capabilities of the programme generation framework, as it is completely separate from any decisions regarding content. This allows videos to be generated for a variety of different platforms and delivery methods. Videos can be encoded using different codecs, using different compression algorithms, and at different resolutions. This capability is of high importance given the variety of the use cases presented in Section 5.2.5 (from television broadcast quality, to downloadable flash video).

d) The programme editor

This section intends to elaborate on the ‘front end’ of the framework – the application that has been developed specifically to drive assisted programme production using the framework. The *Programme Editor* is a Windows application that provides an

intuitive drag-and-drop style interface to the placement of programme components in the relation to both themselves and the timeline.

Basic track/time structure. The structure of the Programme Editor is representative of the structure required by the framework to create a programme description. A programme consists of a series of base *tracks* along the timeline, to which components can be added. Each track is responsible for containing components of a particular type: the background image, background objects and terrain, camera position and animation, and any ambient audio. To the basic set of programme tracks, *actors* can be added. Each actor has associated with it its own set of tracks that are independent from the main programmes tracks. The list of tracks is displayed on the left hand side of the software, with the timeline running from left to right, and displayed above. Figure 5.1.3 shows a screenshot of the structure of a programme without any components added. The left pane shows the list of tracks, while the right pane shows the timeline and contents of the tracks.

Component addition and movement. Addition of components to a track is simply a case of right-clicking anywhere on the track and using a drop down menu to place components. The menu will only show assets and components that are compatible with that track, regardless of the current data directory (see Section 4.5). Once the component has been selected, it will appear as a coloured rectangular indicator on the track, overlaid with the name of its source component. The location and lateral length of the indicator on the track, with reference to the timeline, defines the start and end point of its effect within the programme. For example, if the indicator for component “Camera1” lies on the track between 00.10” and 00.15” on the timeline, between these times the camera used by the programme will be as specified by component ‘Camera1’. By holding the left mouse button on the indicator, it can be dragged and dropped to any point along the timeline, although it cannot overlap with another camera. By click-dragging either end of the indicator left or right it is possible to change the duration of the effect, and by right-clicking on the indicator it is possible to rapidly align it to the boundary of the previous and/or following clip (or the start of the scene). Furthermore it is possible to alter the ‘fade in/out’ property of certain components so that their effect does not

appear/disappear immediately (this is more relevant to animation and audio components).

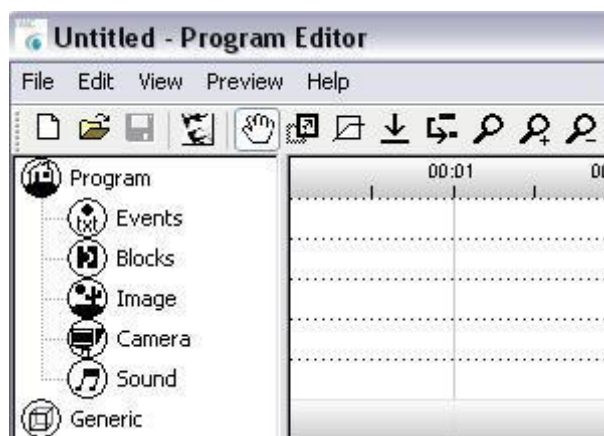


Figure 5.1.3: Basic programme structure.



Figure 5.1.4: Camera and audio components added to programme description.

Figure 5.1.4 shows an example of a camera component added to the timeline, and an audio file added below it. During this region of the programme, the output will be viewed using this particular camera, and the background audio will be played from this particular audio file. Furthermore a background scene, complete with light sources, can be loaded into the left pane to provide context for the programme.

Actors. Virtual characters located within a scene are a prominent feature of the framework and Programme Editor. Within the left pane of the editor, a sub-menu allows the addition of an actor to the scene. The number of actors added is limited only by the rendering capabilities of the machine that is running the software (which itself is determined by the visual complexity, for example the number of polygons, that each character has). Once a character is added, the left pane displays the tracks for that character, as shown in Figure 5.1.5. The basic tracks of an actor consist of:

- Default – specifies the default whole-body animation of actor, usually an idle pose or animation.
- Facial animation – indicates the current facial animation of the character
- Movement – specifies the current whole-body movement animation of the character (i.e. animations that will change the location of the character in the scene).
- Gesture – defines any gesture animation (i.e. an animation that will not alter the location of the character) e.g. a waving hand.
- Sound – indicates the current speech file that is uttered by the character. The software uses simple babble loop animation to produce basic but effective lip-synch.
- Look-at – specifies where the character should be looking. Can be the active camera, another object within the scene, or another character.
- Body configuration – specifies different configuration of changeable items on the character e.g. clothes.

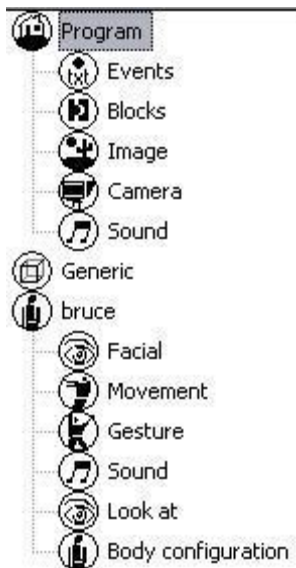


Figure 5.1.5: Addition of character "Bruce" to the scene.

Figure 5.1.6 shows a typical screenshot of two characters with several components assigned to their tracks. Both characters are looking at the current active camera; one ("irina") is currently walking forward, the other ("panchito") is making a greeting gesture while uttering a text file. The drag and drop nature of the components means that it is straightforward to adjust the location and/or duration of any of these components, and once the programme setup is saved in XML, other software could modify the code to change the data source for each component (for example, to change the source for Panchito's utterance to a different audio file).

There are several important parameters for the actors' appearance and behaviour that can be changed via context menus. An actor's position and orientation can be changed, and the loop/stretch status of individual animations can be changed. For example, if a character is to act a 'wave' animation for five seconds, yet the actual duration of the animation is only three seconds, the user can choose whether the animation is looped or stretched (slowed) to accommodate the extra time. Importantly, the fade in/out tool (mentioned above) can be used on all animations to ensure that actors do not suddenly jerk unnaturally into movement.

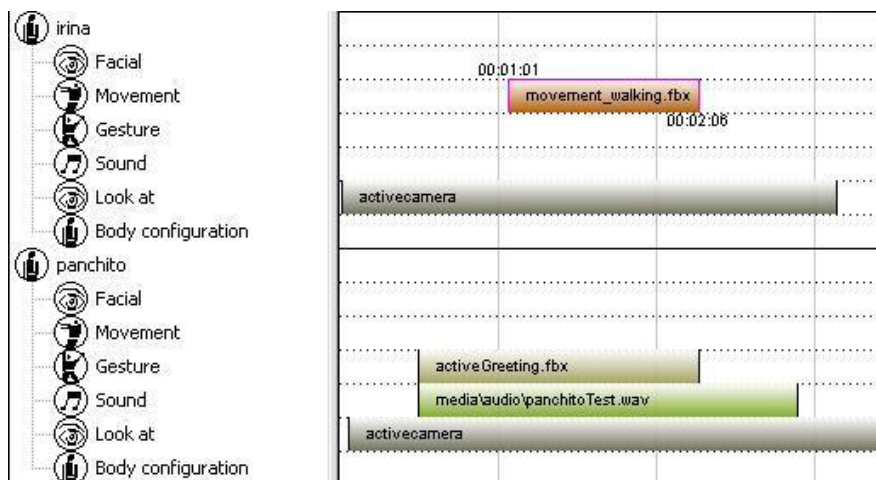


Figure 5.1.6: Tracks with two characters and several components added to the scene.

e) Output

The output of the Programme Editor follows the schema set out in Section 3.3 above. The real-time preview is a powerful feature of the editor as it loads the graphical engine in a sub-window to preview the programme. In the main window, a vertical red bar travels along the timeline to indicate exactly which section of the programme is currently being previewed. Standard video control buttons (Play, Pause, Stop, Forward, Reverse) allow control over the output, while clicking on the timeline will skip instantly to that moment in the programme. The viewer supports real-time addition of components to the programme: without pausing the action, a user can add a component to the scene and it will be displayed instantly. Figure 5.1.1 shows an example of the real-time preview window laid in front of the Programme Editor.

The other output option is the direct creation of a video clip, accessed through the menu bar of the main window. This is in fact a GUI for a separate command line programme that generates the video. The GUI provides a variety of options as the resolution and file types required for the video (as explained in Section 3.3) and allows the user to specify a name and location for the output video

file. When creating videos with dynamic content, this final step is frequently unnecessary, as the videos will be generated separately with different content (as demonstrated in Section f).

Formats and technical requirements. The output of the Programme Editor, and in fact the whole framework, is obviously highly dependent on the quality of the visual assets that are used. The file format supported by the framework is FBX, chosen as it allows assets (objects, cameras, scenes, characters, animations) to be exported from several major art packages (such as both Autodesk Maya and 3D Studio Max). However FBX is still developing as a format and so we have developed guidelines for the best methods to develop and export assets. For use with the Programme Editor, assets must be placed in a directory which is specified by the user as the ‘source’ directory; all assets within here will be accessible through the context menus of the software.

The technical requirements for the Programme Editor vary according to the complexity of the scenes being designed and characters being used. However, minimum requirements for the real-time visualisation are generally a Windows-based machine with a GPU capable of running OpenGL Shader Model 2.

f) Use cases

This section lists several cases in which the automatic programme generation framework has been used. None of the use cases present a comprehensive evaluation of the technology, however the mix of commercial and research based applications demonstrates that the framework and applications have been used across a broad spectrum of applications.

Virtual Weatherman. The most prominent use case of our framework is shown with “the world’s first virtual weatherman”, Sam [7]. Sam is a virtual character capable of presenting the weather for hundreds of locations around the world, and was created by Aactiva Multimedia Digital[1], in collaboration with our group and La Salle Engineering at Universitat Ramon Llull[2]. Figure 5.1.7 and Figure 5.1.8 show screenshots of Sam in action.

The development of Sam used the Programme Editor to create the block template, and then took advantage of the ability to modify the programme description separately according to external input data – in this case, the weather forecast. Three times a day, the latest weather 48-hour weather forecast for hundreds of locations is accessed from an online weather database. This data is then used to generate a video, using the visualisation technology presented in this paper, for each individual forecast for every location that is covered by the system. Multiple versions of these videos are then made available to download on a variety of platforms, including television and mobile platforms.

Virtual stockmarket report. A sample project in collaboration with a multinational media company demonstrates further use of the technology in the creation of programmes. The workflow for this use case was similar to that used for the virtual weatherman. Characters were designed and animated, while the Programme Editor was used to locate them within a scene. Up to date stock market information was used as text input for the character’s speech, and videos generated periodically to present the information. A screenshot of one of these videos is presented in Figure 5.1.9.



Figure 5.1.7: Sam the virtual weatherman

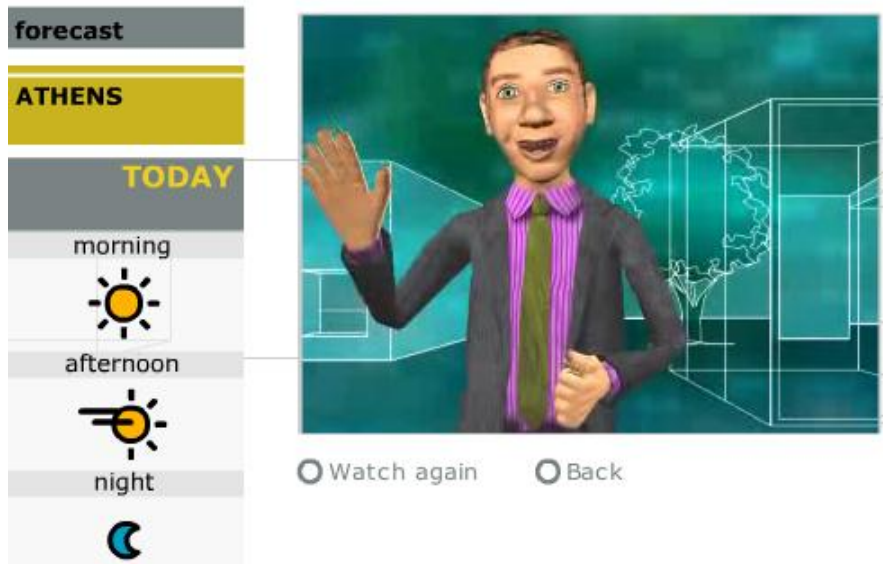


Figure 5.1.8: Sam explaining today's weather



Figure 5.1.9: Virtual stock market presenter

Virtual bartender. “Manolo” is a virtual bar-tender who proffers opinions on the latest sports stories in a stereotypical manner. This use case differs from the previous virtual characters in that the audio comes is supplied by a recording of a real human voice (specifically from recordings of a sport-radio talk-show host). In that sense the technology is enabling a virtual view on non-visual events that have occurred previously. Figure 5.1.10 shows a screenshot of Manolo in action.



Figure 5.1.10: Manolo the virtual bar-tender.

SALERO (Semantic Audiovisual Entertainment Reusable Objects) [8] is an FP6 European Union Integrated Project which aims at making cross media-production for games, movies and broadcast faster, better and cheaper by combining computer graphics, language technology, semantic web technologies as well as content based search and retrieval. The technology presented in the paper has been the basis of a considerable amount of integration within the project.

Programme generator server. The automatic programme generation technology has been adapted for use with server technology, in a similar manner to Xtranormal [3]. Typical applications of this include the ability for users to specify different configurations within a scene (for example, choice of character,

clothes, speech etc.) and then download a video of the generated scene (an Adobe Flash video in our implementation). Figure 5.1.11 shows a basic example. The limitations of what is possible with such a web-portal are due largely to the design of the interface page. It would be perfectly possible to create a comprehensive web-portal that mimicked the majority of the functions of the Programme Editor. Possible use case for this technology are in the ever growing field of user-generated content, allowing companies to let users create custom animations or videos according to their marketing plans.

Voice generation and transformation. Web based access to text-to-speech [9] and voice transformation [10] systems allows the creation of web-portals that combine these technologies into one application. Within the SALERO project such collaborations are resulting in proof-of-concept web-based mash-ups that show how such integration can rapidly create results (see Figure 5.1.11).

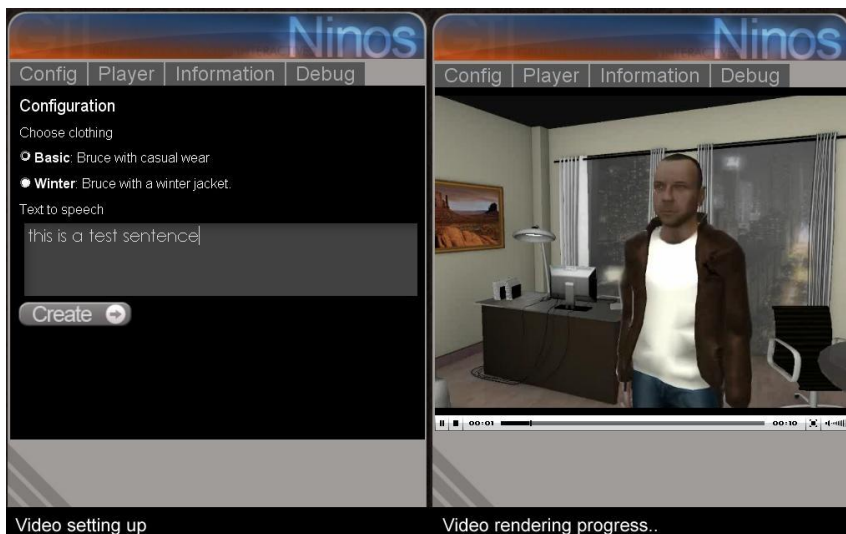


Figure 5.1.11: Programme Generator Server. On the left the user can specify some characteristics, including text to utter, on the right the resulting video is displayed.

Automatic gesture suggestion from audio. Speech tagging systems are able to extract time-correlated speech stress levels from a given input audio file. It is possible to use the output of such algorithms to drive automatic gesture suggestion for virtual characters within our framework. This cuts down on the time required for the user to create a programme description, adding a further layer of automation to generation process.

g) Conclusions and future work

This paper has presented a framework for the assisted creation of computer-generated audiovisual productions. The use cases presented cover both commercial applications and more academic-based integrations. The commercial use cases demonstrate how the framework and applications are viable for use in a commercial environment, complete with the economy based time-pressures that are usual in such situations. The more academic use cases show that the framework forms a solid base upon which several interesting integrations and research collaborations can take place.

Yet further evaluation is required to paint a more accurate picture of the worth of the framework. While it is difficult to compare the framework directly with related work, it certainly is possible to perform user evaluation as to whether the framework and associated applications do genuinely facilitate the creation of audiovisual clips (although the existing commercial use seems to suggest already a positive answer to this question). This further evaluation will then be our immediate future task.

Further work lies in the correction and modification of several technical issues. A major improvement will be the introduction of animation blending within the same track. Animation blending is already supported for animations that lie on separate tracks (e.g. a walking animation can run concurrently, and is blended, with a waving gesture), yet the framework does not support animations overlapping on the same track. This means that any gesture, for example, must finish before a new one starts. Further ideas for improvements are the capability to use the visual preview screen to move objects with the scene, in a drag-and-drop manner. Yet going

down this path may lead to several pitfalls, as 3D scene modification is something that is already covered comprehensively by several large modelling software programmes.

In conclusion, the framework and application presented in this paper presents a novel and powerful methodology for assisted animated production creation and programme generation. The heavy use that the framework has already seen suggests that it provides real benefit to production workflow, and provides a strong platform for the integration of multimedia technology.

h) References

- [1] Bulterman, D. and Rutledge, L. 2008. SMIL 3.0 - Interactive Multimedia for the Web, Mobile Devices and Daisy Talking Books. X.media.publishing. ISBN: 978-3-540-78546-0.
- [2] Redboard. <http://www.redboard.tv/>.
- [3] Xtranormal. <http://www.xtranormal.com/>.
- [4] Girgensohn, A., Boreczky, J., Chio, P., Doherty, J., Foote, F., Golovchinsky, G., Uchihashi, S., Wilcox, L. 2008. A semi-automatic approach to home-video editing. Proc. ACM User interface software and technology (San Diego, USA, 2008). 81-89.
- [5] Apple. "iMovie". <http://www.apple.com/imovie/>.
- [6] Adobe. "Premiere". <http://www.adobe.com/products/premiere/>
- [7] Meteo Sam. <http://www.meteosam.com/>.
- [8] SALERO (Semantic Audiovisual Entertainment Reusable Objects). <http://www.salero.info>
- [9] Alías, F., Iriondo, I., Formiga, L., Gonzalva, X., Monzo, C. and Sevillano, X. 2005. High quality Spanish restricted-domain TTS orientated to a weather forecast application. Proc. 9th European Conference on Speech Communication and Technology (Interspeech) (Lisbon, Portugal, 2005). 2573-2576.

[10] Mayor, O., Bonada, J. and Janer, J. 2009. Voice Transformation from interactive installations to Video-Games. Proc. 25th AES Conference: Audio for Games (London, UK, 2009).

i) Integrating emotional work

Following the results presented in chapter 4 I have integrated the synthetic model for facial animation and the emotional interface in the GTI Framework. The emotional interface was then accessible through the Programme Editor. This way, it is possible to easily modify the facial expressivity of any character through time, to match the desired mood.

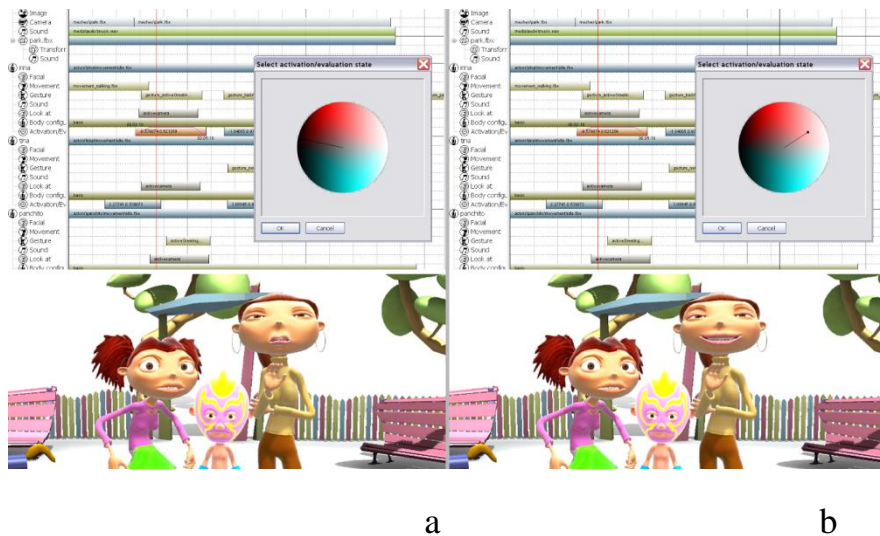


Figure 5.1.12: Screenshots of the emotional interface (section 4.3.10) working in the Programme Editor. The emotional interface is interpolating the shapes described in the synthetic model for facial expression (section 4.3). (a) shows disgust (approximately activation -0.9 , evaluation 0.1). (b) shows happiness (approximately activation 0.6 , evaluation 0.3).

5.2. Domain specific sign language animation for virtual characters

This paper describes the design, implementation and results of system to animate a virtual character to sign in International Sign Language. By automatically parsing the input data, the system blends animations smoothly together in order to create a coherent and understandable presentation in sign language, with equal importance assigned to both hand and facial animation. Following the blending step, a video is rendered and output in a variety of formats suitable for distribution. The system was created in collaboration with groups of people with impaired hearing who were fluent in international sign language, and who were able to validate the results. We present a test case of the system in the shape of 'Borja', a virtual character who signed biweekly daily updates of the Barcelona World Race sailing regatta 2010/2011.

a) Introduction

The internet has transformed the way we access news. Be it a sports event, a major news story, or a conference or trade-show presentation, there are frequently several internet portals dedicated to providing live and interactive reports at a moment's notice. Indeed, this trend was visible as far back as 2004, where a survey of American internet users indicated that 53% checked sport news daily, and 55% of them did so through internet (Fallows 2004). This increase in news consumption has driven an equal rise in the number of services (Miesenberger et al. 2010)(Othman et al. 2010) that aim at creating news (and sports news) automatically from numerical data; generating stories which appear attractive to readers, while requiring minimal manual effort to maintain and update. Furthermore, this approach is particularly appealing for coverage of minority sports events, which typically do not receive the attention of more mainstream sports. One disadvantage of this increased focus on automated news is that there is frequently no effort to make the news accessible for those users requiring more specialised services, such as those from the deaf community. According to the World Federation of Deaf, only 20% of the 70 million Deaf people in the world receive an education in spoken

language (World Federation for the Deaf 2013), so the remaining 80% is not able to easily read and understand text written in spoken language (which is quite different from written sign language). While some effort is occasionally made on television to show expert signers providing translation of the news for deaf viewers, the limited resources available mean that this is rarely, if ever, duplicated across to internet-based media outlets.

In this paper, we present a virtual signing avatar designed to update users on the status of live events in a specific domain, such as sports events. The initial data sources are XML-based, featuring information such as the current score, competitor position in the event, or current prevailing conditions such as the weather. These data are sorted to fill a series of templates, which are input into our animation system. The system then automatically creates a video clip, with the data translated into animated sign language, and structured into logical sentences and phrases. Although the application of the system is domain-restricted, our work forms an important contribution to the on-going challenge of providing a fully automatic text-to-sign translation service.

Our system was tested live for a major sporting event, the Barcelona World Race sailing regatta 2010/2011. For the regatta, we designed an avatar and created a series of animation templates in IS. A fully automatic system parsed data from the race and, twice a week, rendered a video which was displayed on the website of the Spanish National broadcasting company, Radio Televisión Española (RTVE). Results showed the pages featuring the animated clips to be the most popular in the Section focusing on coverage of the Barcelona World Race.

Our paper is organised as follows. Section 2 presents related work in the field of automated signing avatars. Section 3 details the methodology used to create our system. Section 4 presents Borja, the signing avatar designed for the Barcelona World Race. Finally, Section 5 summarises the paper and draws conclusions.

b) Related work

Several studies (Dehn & Van Mulken 2000; Johnson & Rickel 1997; Moundridou & Virvou 2002) found that rendering agents with lifelike features, such as facial expressions, deictic gestures and body movements may rise the so called persona effect. A persona effect is a result of anthropomorphism derived from believing that the agent is real and authentic (Van Mulken et al. 1998; Baylor & Ebbers 2003).

During the last decade there has been extensive research and development carried out with the goal of automating the animation of sign language for virtual characters. The ViSiCast and eSIGN projects (Elliott et al. 2007) focused on the development of a comprehensive pipeline for text-to-sign translation. Using text written in English and German as input, the system translates it into written text in sign language (English - ESL or German - DGS, respectively). The translation, though, is very literal and liable to produce unnatural results. Written sign language is translated into signs using the HamNoSys (Hanke 2004) notation which describes signs as positions and movements of both manual (hands) and non-manual (upper-torso, head and face) parts of the body. This notation enables signs to be performed by a virtual character procedurally, using inverse kinematic techniques. In this sense the animation system is procedural and, thus, flexible and reusable compared to others that use motion captured data or handcrafted animation. The HamNoSys notation, however does not include any reference to speed and timing, and ignores prosody. This has been considered one of the reasons of low comprehensibility (71%) of signed sentences using HamNoSys (Kennaway et al. 2007). This is a particularly serious disadvantage given that recent studies have demonstrated that prosody is as important in spoken languages as in signed ones, activating brain regions in similar ways (Newman et al. 2010), and has a crucial role in understanding the syntax of a signed message.

Automatic Program Generation is a field of natural interest to the commercial domain, but which has seen little academic research (Abadia et al. 2009). The most recent integrated attempt to disseminate sport news using a virtual signer has been the SportSign platform (Othman et al. 2010). It is a partially automatic system that

needs an operator to choose the kind of sport and specify other relevant data, such as the teams playing match, the number of goals or points scored etc. Then the system generates a written version of the news in sign language (specifically, American ASL) that a human operator has to validate.

Finally, a server-based service generates a video with an animated character that is then published on a web page. The workflow needs extensive interaction by a human user, in order to guide and validate the results of the system, meaning that it would not meet the important goal of reducing the costs to make signed sport news feasible.

c) Methodology

System overview. Our animation system is designed to build a complex signed animation with a virtual character by concatenating, blending and merging animated clips previously prepared to digitally mimic the gestures of expert signers. The system relies on a series of XML-based templates indicating which signs should be used to build certain content, and which type of data is needed for the relevant information (numbers, text, etc.). Figure 5.2.1 shows a schematic overview of how the system constructs an animation.

The parser receives the data and decides whether is a known phrase, a number or a custom word. If it is a known phrase, it is retrieved directly from a database that stores the animation data for that entire phrase. If it is not a phrase, the system falls back and looks for a number or word. In this case, the sign elements that are used to compose that number or word then are taken individually and merged together to build a single new sign. This fall back is very useful when dealing with names of people or cities, or even with numbers of more than one digit.

Finally, each of the clips is merged using animation blending. How each clip has to be blended is specified in the metadata description associated with that clip in the database. Each clip has its own length, start time, end time and in-out trim points which specify the boundaries for the blending.

Lexical structure. Sign languages are complete languages with their own grammar and vocabulary. Communication occurs via a complex interrelation between manual, body and face gestures. These three elements cooperate as a whole to provide terms, grammar structure and intention (through prosody), and thus meaning. Even when written, sign languages are different from spoken languages. In fact, it can be difficult for pre-lingually deaf people to understand written English or other languages, as the grammatical structure of the spoken language is different to the structure of the sign language.

As many other languages, vocabulary and grammar also change depending of the country (ASL is different from Spanish Sign Language LSE) and it may also adapt to the characteristics of a certain zone inside a country (non coastal regions of a country may lack the term "boat"). In an attempt to provide a common standard of communication for the world deaf community, in 1975 the World Federation of Deaf proposed a unified system which included the most common terms used in the majority of different sign language. It also included signs that are easy to understand. Since then, International Sign has evolved and is now used in conferences around the world and for sport reviews (as in the 2010 Fifa World Cup). In this sense, the lexicon of International Sign System is quite poor but its grammar is as rich and complex as for any other language (Newman et al. 2010).

Data structure. The goal of any automated news system is that it should be able to take data from automatically updated sources, and transform that data into a manner such that it can be broadcast and easily understood by users. Our system parses XML data with a specific format (see Figure 5.2.1 below) and uses it to fill in a series of data templates. These templates are structured in a similar way to the lexical structure of IS as mentioned above. The data is now in suitable format to be converted to animation.

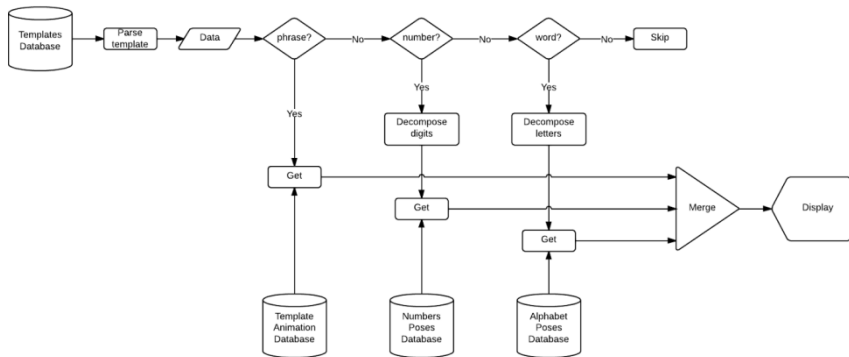


Figure 5.2.1: The animation system receives an XML file that contains the description of the clips that has to be concatenated in order to build the complete animation. Depending on whether it's a pre-made phrase, a number or a word, the system retrieves and merges the appropriate animated clips to build the signed contents.

Animation database. There are three possible sources for the animation data suitable for our system:

- Motion Capture Systems, where an actor's performance is digitally captured
- Procedural Systems: where the actual animation is generated by a fully automatic system
- Manual recreation from video, where a human animator manually creates animations using a video of an actor as a reference.

Standard Motion capture systems do not provide detailed information on the movements of the fingers and of the facial muscles: both essential for accurate representation of sign language. Sign language makes use of incredibly subtle movements of the arms, fingers and facial muscles, which requires considerable effort to capture accurately.

Although full performance capture technology is being developed (Weise et al. 2011), combining facial and finger systems to the required standard is beyond the scope of this work. Furthermore, raw motion capture data contains many small errors and rarely can it be used in its native form. The effort required to manually 'clean'

the capture data, adapting it to a virtual character and enhancing it with the missing pieces (commonly face and fingers) is a highly time-consuming process. Thus, we discarded motion capture as an animation source. Procedural animation has also been recently used for automated sign language generation (Elliott et al. 2007). It is a flexible solution as it does not require an expert to sign the contents, and any sign can be generated using a sign notation like HamNoSys (Hanke 2004). Unfortunately, as mentioned above, animations generated with sign notations suffer from a notable lack of prosody, a very important factor in sign languages.

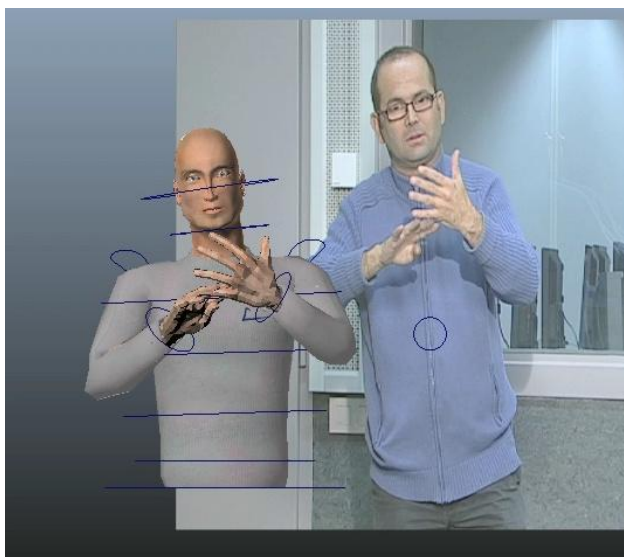


Figure 5.2.2: Screenshot of a signing expert (prelingually deaf) with virtual character overlay.

This means that every animation clip used by the system has to be animated by hand by following video references of signing experts recorded on video. We created a skeletal animation rig for the character body which uses both forward and inverse kinematics. We use blend shape animation for the the facial expression as this gives us greater control on the precise shape of the face – an essential component for the understanding of sign language. While this requires initial manual effort, the main motivation behind the choice of this technique is the freedom and accuracy that hand-made

animation can provide. Furthermore, as the ‘library’ of animations is created, adding to it becomes progressively easier, as many previous animations can be reused. Figure 2 shows a screen capture of the animation process. It is important to note that, from a technical point of view, our system is capable of working with both manually created animations, motion-captured performances, and procedural animation.

For animation purposes, we define four different semantic levels:

1. Phrases: where the actor signs a complete or partial phrase e.g. “My name is...”
2. Words: where the actor signs an entire word e.g. “weather”
3. Letters: individual letters of the alphabet
4. Numbers: single digit numbers only

We distinguish between words and phrases because, frequently, sign language will employ a single sign to encapsulate an entire phrase of spoken language. The advantage of these distinctions is that they are designed to work very effectively with the ‘trickle down’ animation composing/blending.

Animation clips are designed to unequivocally communicate a single chunk of information, which can be understood in isolation, which is essential for composition and blending (see below).

Once recorded and animated, each clip is stored in an Animation Database along with its associated metadata (mainly the spoken language equivalent of the animation).

Composing and blending. To compose a sign language clip from spoken language input, our system is split into three components (the Classifier, Blending Queue and Composer) which process any input ready to be passed to the renderer (see Figure 5.2.3). The Classifier first analyses the input template according to the four semantic levels defined above. From this classification, the system searches the Animation Database in a trickle down manner, (as shown in Figure 5.2.1 above). Let us take the example “My name is John”. The classifier first parses the entire template to search for a

known phrase in the database. It finds the phrase “My name is” and adds it to the Blending Queue, but does not find the phrase “John”. It then drops down a semantic level to look for any numbers. In this case, there are none, so it drops down another level to search for the word “John”. If it does not find the word, then it drops down a final level and adds the individual letters ‘J’, ‘O’, ‘H’ and ‘N’ to the Blending Queue.

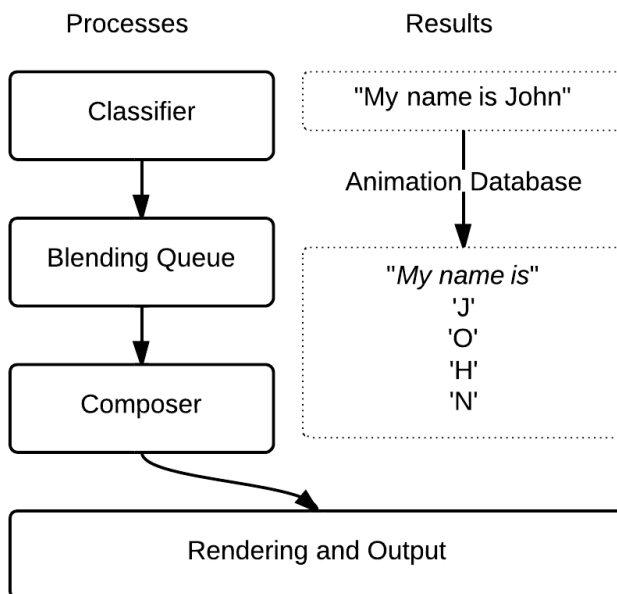


Figure 5.2.3: The flow of information through the different processes that create an animation clip from input.

In order to concatenate each clip to conform a complete discourse, The Composer must blend the clips in the Blending Queue together to perform smooth transition between different animations. Blending can be performed in two different ways: Additive blending, where an animation clip is added on top of an underlying clip, offsetting it by means of differences between each other; and Override blending, which substitutes a given animation clip with the content of another one. Furthermore, the blending can be performed either between two clips or across two clips (see Figure 5.2.4). When working with sign language, it is important to consider that even minor modification of the sign could change the meaning for

the viewer (potentially drastically). To avoid this, we decided to concatenate phrases performing override blending between clips. In this way we guarantee that the signed content is 100% accurate and that no undesired signs are produced in the transition between one clip and the other. On the other hand, words and numbers are built by blending together multiple short clips of letters and digits. In this case, we blended across clips, which produced smooth and continuous (not robotic) signs.



Figure 5.2.4: Figure shows how blending can be made between two clips. Left: interpolating from last frame of Clip A to first frame of Clip B; or Right: cross fading from Clip A to Clip B.

Rendering output. The system is designed to render a video output using standard algorithms from Autodesk Maya. Background can be static or animated in a pre-rendering step. This approach provides acceptable quality results with a relatively low rendering times, which is crucial for providing large amount of content in a reasonable time.

d) Borja: sports reporter

Our system was tested in a real environment for the Barcelona World Race sailing regatta, 2010/2011. The Barcelona World Race is a non-stop, round-the-world regatta, where crews of two people circumnavigate the globe in an IMOCA 60 racing boat, competing to be the first to return to Barcelona. Apart from being a challenging regatta, the event also has at its heart a commitment to research and development, education, and science and the environment. In that sense it was an ideal opportunity for us to test our system in a real-world context. The race was forecast to last 3 months, and consisted of 14 boats. Our goals, therefore, were to:

§ Design and animate a virtual character suitable for presenting news from the regatta, in sign language;

§ Adapt our system to be used for automatic creation of news from a sailing regatta;

§ Create an automatic, server-based system that would output a video of the current animated news, twice a week, and make it available for public dissemination.

Character design. Any virtual character has to be designed in a way that makes it believable and, for a virtual newsreader, in a way that helps convey information to the audience. The character design should match the characteristics of the sport and adapt to different context, for instance changing clothing, depending on external factors such as the weather.

Another aspect, particularly important for virtual signers, is the believability of the movements and gestures of the character. It has to look convincing (and not necessarily realistic), in order to encourage the audience to engage with the provided news. This insistence on believability was a major factor in our deciding to create personalized hand-crafted animations for the character, rather than procedurally animated signs that lack prosody and life.

Another crucial aspect is the size and proportions of the hands. The automatic contents generated with our method were going to be viewed mainly through a video streamed on a web page, meaning that the playback area would not be large enough to ensure that the hands and fingers will be clearly visible all the time. On the other hand, larger hands assist in driving attention toward such an important communicative item. Thus, we ran several tests with experienced signers in order to decide on the correct proportions for all aspects of the body. After several iterations, we settled on a blond male, dressed in typical sailor garb (varying depending on current weather), and christened him Borja.



Figure 5.2.5: Art design for Borja. Design includes different clothing for different weather conditions. The design shows a sporty style for the character to make it closer to the sailing world, bigger eyes to empathize with audience and big hands to make signs more readable through streaming videos.

Input data and template creation. There were three main data sources used for this application, all provided under license by the race organisers:

- § Boat XML information, with direct information about each competitor boat and race info per boat, such as GPS location, speed, course, ranking place, distance covered etc;
- § The GRIB weather data, updated every 30 minutes;
- § Tertiary information regarding the race e.g. historical information regarding previous races, skipper biographies etc.

The system groups the data needed for each news and ranks the ‘relevance’ of each set of data. The relevance is calculated according to criteria pre-set by expert news writers focusing on sailing regattas. For example, a change of leader is considered highly relevant, while a change at the bottom of the ranking is less relevant. Once these criteria are set, no further expert input is required, and the system generates rankings automatically. Based on the calculated ranking, the system picks the three most relevant items of news, and adds opening and closure sections to build the complete template for animation (see Figure 5.2.6). In the case

where the third less relevant news is below a relevance threshold, the system provides information about the skipper, the boat or the geography.

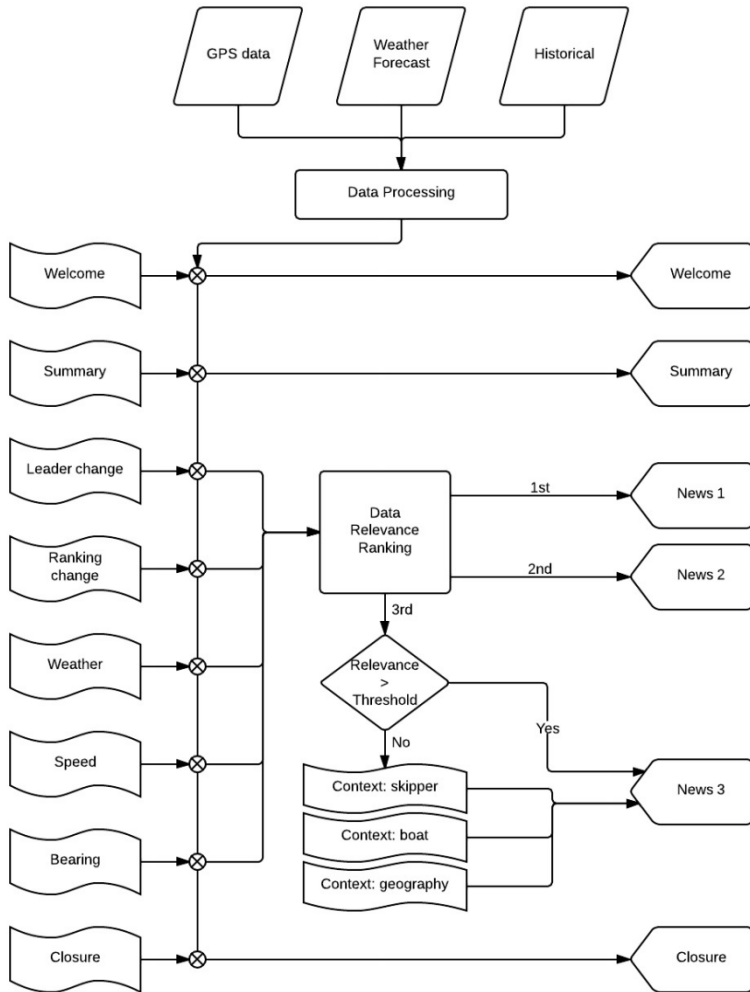


Figure 5.2.6: Scheme of the news generation system.

Scheduled output. During the course of the regatta, the system compiled the data twice a week, every week during a period of three months (the length of the regatta). From this data, the system reads the prominent weather condition and configures Borja to wear different clothes (wet jacket, t-shirt, etc.) depending on it. Then, the

system concatenates a sequence of animations to represent the desired news in IS and launches the rendering. All the information about how to build the animation sequences and the database of available animations is stored in XML files.

Once the rendering is completed, the video (in Adobe Flash .flv format) and the equivalent written news in Spanish is sent to the server of RTVE to be displayed on their webpage (see Figure 5.2.7). Figure 5.2.8 shows a larger screenshot of Borja in action.



Figure 5.2.7: Webpage of the Spanish National Radio showing Borja's videos for the Barcelona World Race.



Figure 5.2.8: Screenshot of Borja in action.

e) Results and conclusions

In this paper, we present the design and implementation of a system capable of automatically transforming input data into sign language, given a specific, restricted domain. The system is designed to automatically create reports for live events such as sports events, and we present a successfully prototype, implemented to create automatic signed news for the Barcelona World Race 2010/2011.

Títol ▼	Tipus	Durada	Popularitat ▼
Avatar: Crònica Jornada 81	Completo	2:19	<input type="range"/>
Crònica 21 de març	Completo	13:00	<input type="range"/>
Resum onzena setmana	Completo	29:59	<input type="range"/>
Crònica 18 de març	Completo	12:33	<input type="range"/>
Avatar: Crònica Jornada 77	Completo	1:17	<input type="range"/>
Crònica 17 de març	Completo	12:05	<input type="range"/>
Crònica 16 de març	Completo	13:05	<input type="range"/>
Crònica 15 de març	Completo	13:18	<input type="range"/>
Avatar: crònica Jornada 74	Completo	3:07	<input type="range"/>
Crònica 14 de març	Completo	12:47	<input type="range"/>
Resum desena setmana	Completo	30:04	<input type="range"/>
Crònica 11 de març	Completo	13:50	<input type="range"/>
Crònica 10 de març	Completo	13:27	<input type="range"/>
Avatar: Crònica Jornada 70	Completo	1:17	<input type="range"/>
Crònica 9 de març	Completo	14:04	<input type="range"/>

Figure 5.2.9: Screenshot of RTVE ranking page showing Borja popularity.

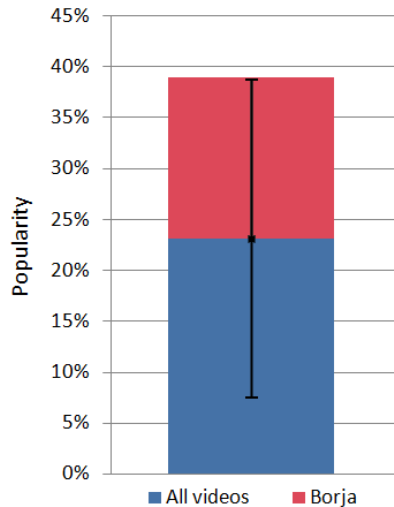


Figure 5.2.10: Chart showing the 15.8% greater in popularity of Borja’s videos over all other Barcelona World Race multimedia. Also shown in the Standard Deviation bar.

25 videos featuring Borja were created during the Barcelona World Race and featured on the RTVE website. These 25 formed part of a total of 91 multimedia entries concerning the Barcelona World Race 2010/2011. The ‘popularity’ of the multimedia entries is habitually measured by RTVE by measuring web traffic on each page. The results show that the mean popularity rating for the 25 Borja videos was 38.96%, 15.8% greater than the mean popularity of all 91 multimedia videos. This increased in popularity is marginally greater than the dataset standard deviation of 15.6%.

The output of the system was validated as “consistently correct” by the expert signers involved in the project, but future work is now focused on obtaining more quantitative results as the quality of the animated output. A study featuring a statistically valid sample of deaf users is being carried out in order to ascertain statistically whether the animated sign language produced by the system is capable of being understood naturally in different countries around the world. If the results of this analysis are positive, we plan to develop our system and use for different scenarios such as rolling news and other live events.

f) References

- Abadia, J. et al., 2009. Assisted animated production creation and programme generation. In Proceedings of the International Conference on Advances in Computer Entertainment Technology ACE 09. ACM Press, p. 207.
- Baylor, A. & Ebbers, S., 2003. The Pedagogical Agent Split-Persona Effect: When Two Agents are Better than One. In World Conference on Educational Multimedia, Hypermedia and Telecommunications. pp. 459–462.
- Dehn, D.M. & Van Mulken, S., 2000. The impact of animated interface agents: a review of empirical research. *International Journal of Human-Computer Studies*, 52(1), pp.1–22.
- Elliott, R. et al., 2007. Linguistic modelling and languageprocessing technologies for Avatar-based sign language presentation. *Universal Access in the Information Society*, 6(4), pp.375–391.
- Fallows, D., 2004. The Internet and Daily Life. Pew Research Center's Internet & American Life Project. Available from: <http://www.pewinternet.org/Reports/2004/The-Internet-and-Daily-Life.aspx>. [Accessed July 24, 2013]
- Hanke, T., 2004. HamNoSys - representing sign language data in language resources and language processing contexts. In LREC 2004, Workshop proceedings : Representation and processing of sign languages. pp. 1–6.
- Johnson, W.L. & Rickel, J., 1997. Steve: an animated pedagogical agent for procedural training in virtual environments. *ACM SIGART Bulletin*, 8(1-4), pp.16–21.
- Kennaway, J.R., Glauert, J.R.W. & Zwitterlood, I., 2007. Providing signed content on the Internet by synthesized animation. *ACM Transactions on Computer-Human Interaction*, 14(3), p.15–es.
- Miesenberger, K. et al. eds., 2010. *Computers Helping People with Special Needs*, Berlin, Heidelberg: Springer Berlin Heidelberg.

Moundridou, M. & Virvou, M., 2002. Evaluating the persona effect of an interface agent in a tutoring system. *Journal of Computer Assisted Learning*, 18(3), pp.253–261.

Van Mulken, S., André, E. & Müller, J., 1998. The Persona Effect: How Substantial Is It? *People and Computers*, XIII, pp.53–66.

Newman, A.J. et al., 2010. Prosodic and narrative processing in American Sign Language: an fMRI study. *NeuroImage*, 52(2), pp.669–76.

Othman, A., El Ghouli, O. & Jemni, M., 2010. SportSign: A Service to Make Sports News Accessible to Deaf Persons in Sign Languages. *Lecture Notes in Computer Science*, 6180, pp.169–176.

Weise, T. et al., 2011. Realtime performance-based facial animation. In *ACM SIGGRAPH 2011 papers on - SIGGRAPH '11*. New York, New York, USA: ACM Press, p. 1. Williams, R., 1957. *The Animator's Survival Kit*, Faber and Faber Inc.

World Federation for the Deaf, 2013. Human Rights. Available from: <http://wfdeaf.org/human-rights>. [Accessed July 24, 2013].

5.3. Measuring the facial expressivity impact in virtual characters for sign language communication

This paper presents an experiment that aims to evaluate and measure the impact that facial expression has on virtual characters communicating using sign language. The study is based on basic grammar rules for sign language that distinguish between two main kinds of facial expressions assigned to “yes-no” type questions and “wh-?” type questions. An experiment was run to test how human signers discriminate between two questions, in four separate cases, where a real person and a virtual character sign while performing with coherent and incoherent facial expressions. The analysis of the data gathered during the experimental process is reported, suggesting that facial expression are not a crucial factor when

perceiving signed questions in both real signers and virtual characters. This finding differs from the sign language literature.

a) Introduction

The study of the perception of sign language is a key point towards the development of new interfaces for improving the possibilities of communication between deaf and non-deaf people [1]. Sign Language can be analyzed starting from its basic components: the movements of the hands, whole body and head; and the facial expression. The latter can be divided into mouth, and eyes\forehead movements. In our research we focused on the eyes\forehead expressions. Asking questions is a case in which facial expressions do not differ between countries. The standard literature distinguishes between two main kinds of questions: “yes\no” questions and “wh?” (why, when, who, etc.) ones [2].

The study was carried out via a web-based test where the participants were asked to watch eight videos and discriminate between the two types of questions mentioned above. The data collected during the experiment is then statistically analysed. The work is described in the following four sections. The first one presents a brief explanation on the distinction between the two types of questions in sign language. The second contains the description of the experiment. Then we present the results and, in the final section we discuss the output of the study.

b) Facial expressions in question types

In general it is possible to distinguish between two main types of questions, “yes-no” and “wh-?” questions. The first type refers to those questions whose answer comes in the form of a “yes” or a “no” while the questions of the second type are the ones that require an open or detailed answer. In sign language grammar there are facial expressions assigned to each of those types [2]. “Yes/no” questions are signed raising the eyebrows and opening the eyelids in a surprised-like facial expression (Figure 5.3.1), whereas “wh-?”

questions are signed with eyebrows lowered and eyes narrowed in a puzzled-like facial expression.

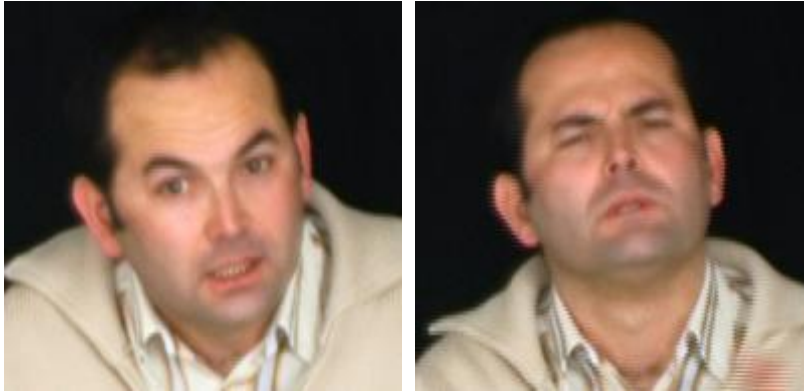


Figure 5.3.1: Two stills of a real person signing two types of questions, “yes/no” (left) and “wh-?” (right).

c) Experiment

Design. The experiment was designed to evaluate the hypothesis that in sign language, virtual character's facial expression plays an important role in the perception of the message, even if the characters are synthetic and less detailed than real human faces. In order to demonstrate this hypothesis, first we must demonstrate that facial expressions affect the understanding of signed questions. To accomplish this, the study compares the data retrieved by testing a real signer (RS) and a virtual signer (VS), with coherent and incoherent facial expression. We chose four “yes/no” questions and four “wh-?” questions, with the help of professional signers. The questions were formulated so that the context was not important for discriminating the type of question. Moreover the questions were chosen so that the content of a “yes/no” question was the same of the one of a “wh-?” question (table 5.3.1).

Yes/No questions	Wh-? questions
Have you gone there?	Where have you been?
Do you have a car?	Which car do you have?
Have you eaten meat?	What have you eaten?
Do you like meat?	Which food do you like?

Table 5.3.1: List of questions used in the experiment (translated to English from Catalan). The questions are divided in two groups, depending on the type of question.

These questions were filmed, and four different sets of video assets were digitally created, as described below.

Each participant saw 4 “yes/no” and 4 “wh-?”, randomly chosen, questions. The videos were shown in two different ways, with coherent or incoherent facial expression, in order to test how the perception of the viewer changed. The test was introduced by an explanation in LSC (Catalan Sign Language) that shows the difference between the two question categories. In the context of the experiment, the coherence or incoherence of the facial expression and the fact that the character is real or virtual, are independent variables, while the correctness and time spent in discriminating between the two different kinds of questions are dependent variables.

Setup and procedure. The experiment is set up on a web-based interface. The test begins with an introductory video explanation about the procedure, signed by an expert in LSC, together with a graphic explanation of the two buttons that are available during the test. It makes clear the distinction between the two different kinds of questions and underlines that the answering speed is an important factor, specifying that there is no need to wait till the end of the movie before answering. The interface gives the opportunity to watch the explanatory video again.

Once the participant presses the "Continue" button at the end of the explanation video, the test session begins. The participant watches eight short movies each one asking a question through LSC. All the

videos are preloaded in order to avoid distortions due to bandwidth or slow internet connections.

For each question the participants have to distinguish whether the question is a "yes/no" question or a "wh-?" one and accordingly click on one of the two available buttons (Figure 5.3.2). Time was measured in milliseconds. Each video is introduced by a 1.5 seconds countdown video and is repeated until an answer is given.



Figure 5.3.2: Snapshot of the web interface (based on Flash). The buttons are placed below. “Abierta” and “Si/No”, translated to English, mean “Wh-?” and “Yes/No”.

Assets creation. As mentioned above, the experiment uses 4 sets of video assets. The videos of the RS were produced filming an expert LSC signer asking the 4 "yes/no" questions and the 4 "open answer" ones. This set of 8 videos was then digitally manipulated, interchanging the portion of facial expression in the upper zone of Figure 5.3.4: A real signer and an animated character signing.

the face between “yes/no” and “wh-?” videos (Figure 5.3.3). Using the filmed videos as a reference, the performance of the signer was

converted in 3D animation by an expert animator using Autodesk Maya 2008. The animation of the body and the face was made using handcrafted key-frame animation and, moreover, the facial expressions were animated using blend shape techniques [3]. This animation was applied onto a virtual human-like 3D virtual character (Figure 5.3.4). The 8 animated questions were also altered by inverting the animation key-frames of the eyebrows and forehead. The animations were finally rendered out using Mental-Ray (a standard render engine).

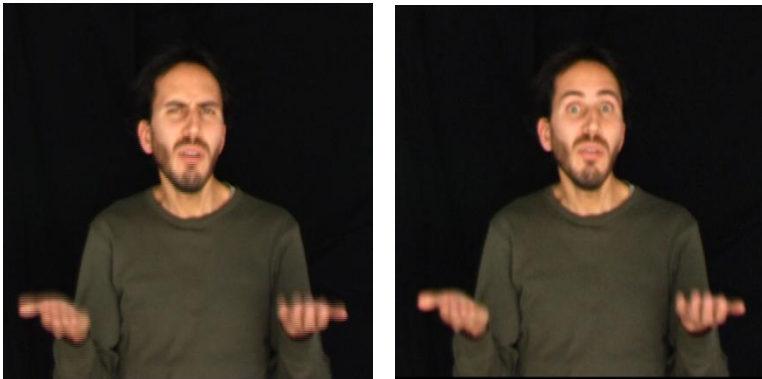


Figure 5.3.3: Two stills of a RS signing “what” in LSC, one with coherent facial expression (left) and another with incoherent facial expression (right).



Figure 5.3.4: A real signer and an animated character signing.

d) Data processing and results

Data time normalization. First of all, the time measures were corrected, comparing them with the actual start of each signing movement. In addition, each time value was divided by the total duration of the signing movement as there was a slight difference between videos. The result was a time variable as a ratio (a value of 1 means that the answer was given just at the end of the signing movement). For each participant the first two answers were discarded, considered as habituation questions, in order to be sure that the time variable was not dependent on the interface. This reduced the sample size a little bit, since there were fewer participants that experienced all the four conditions.

Results in real and virtual signers. For 36 participants, the average of coherent facial expression time ratio for correct answers was 1.61 ± 1.11 (SD) for RS (Figure 5.3.5a) and 1.41 ± 0.97 (SD) for VS (Figure 5.3.5b), while the average of incoherent facial expression time ratio was 1.53 ± 0.99 (SD) for RS (Figure 5.3.5a) and 1.22 ± 0.80 (SD) for VS (Figure 5.2.15b).

Having checked the normality with Shapiro-Wilk test ($p < 0.001$), we performed a repeated measures t-test, since the same participant was subjected to the same measurements for all the two different situations (coherent and incoherent facial expression). The result was not significant in both cases ($t = 0.435$, $df = 35$, $p = 0.666$ for RS and $t = 1.349$, $df = 37$, $p = 0.186$ for VS).

For 51 participants, the average of correct answers with coherent facial expression was 0.7188 ± 0.4021 (SD) for RS (Figure 5.3.5c) and 0.7269 ± 0.3855 (SD) for VS (Figure 5.2.15d), while the average for incoherent facial expression was 0.6761 ± 0.3807 (SD) for RS (Figure 5.3.5c) and 0.6695 ± 0.3910 (SD) for VS (Figure 5.3.5d). After having checked normality with Kolmogorov-Smirnov test ($p < 0.001$), we performed a repeated measures t-test. The result, again, was not significant in both cases ($t = 0.613$, $df = 50$, $p = 0.542$ for RS and $t = 0.950$, $df = 54$, $p = 0.347$ for VS).

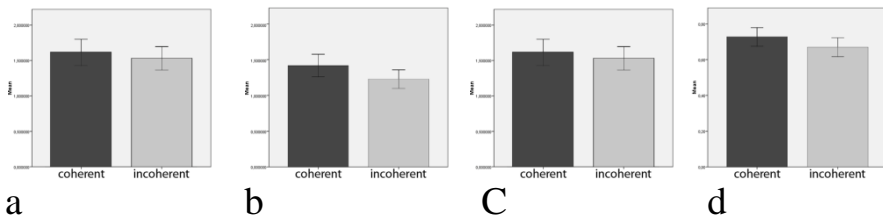


Figure 5.3.5: Time ratio averages and standard error for correct distinction of questions in coherent and incoherent cases for RS(a) and VS(b). Distinction correctness of questions in coherent and incoherent cases for RS(c) and VS(d).

A last analysis was performed using the time ratio for both the correct and incorrect answers. A repeated measures MANOVA was applied, comparing the time ratio for the four groups (Real coherent, Real incoherent, Virtual coherent, Virtual incoherent), on 23 participants (Figure 5.3.6). The difference was not significant ($F(3,20)=0.617$ $p=0.612$).

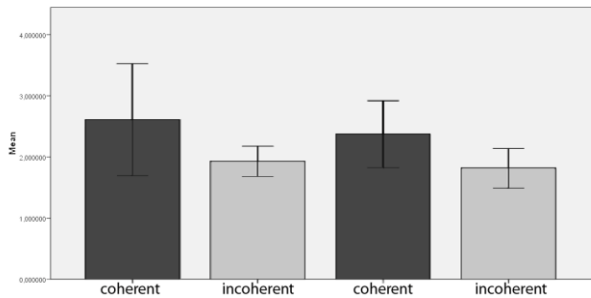


Figure 5.3.6: Time ratio averages' comparison between RS with coherent facial expression, RS with incoherent facial expression, VS with coherent facial expression and VS with incoherent facial expression.

e) Conclusions

The experiment shows no statistically significant difference between real and virtual signers when communicating with deaf people. The synthesis process that is performed while animating using a real reference does not affect the comprehension of the signs. The experiment raises some issues related to official sign language grammars as it suggests that facial expression in understanding of questions through sign language may not be as

crucial as previously thought. Both in real and virtual signers, the results show lack of significant difference in time and correctness between coherent and incoherent expression. This could be due to the fact that the attention is mainly focused on the hands and the body. Alternatively it could be due to a more complex use of facial expression that relates puzzled and surprised expressions not only to question types, but also some other communication aspects from the signer, more related to intention.

f) References

[1] Ryokai, K., Vaucelle, C. and Cassell, J. (2003). Virtual peers as partners in storytelling and literacy learning. In Blackwell Publishing Ltd, *Journal of Computer Assisted Learning*, 19(2), pages 195-208, 2003.

[2] Woll, B. (1983). The semantics of British Sign Language signs. In J. Kile & B. Woll (Eds.), *Language in Sign*, London: Croom-Helm, pages 41-45, 1983.

[3] Joshi P., Tien W. C., Desbrun M., Pighin F. (2003). Learning controls for blend shape based realistic facial animation, In Eurographics Association, *Symposium on Computer Animation Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 187-192, 2003.

g) Evaluation design for the international sign system virtual signer

The Sign Language evaluation presented in the previous subsections of 5.3 is very partial in terms of understandability of the Sign Language generated animating avatars. In the context of the Borja project, discussed in 5.2, a deeper evaluation of the degree to which the Sign Language was understood seemed worth. Another motivation was that the most comprehensive evaluations we had found then were quite limited. [Kennaway et al. 2007] deals with the evaluation of single sign and sentences, in which signed

sentences were evaluated by repetition, with success rates ranging from 35% to 58%. Another experiment [Vink & Shermer, 2005 *Report of research on the use of an avatar compared with drawings or films of gestures*] where 71% of correct answers related to filling in forms, instructed by Sign Language generated by animation, was achieved, was tested with 7 users.

We designed an experiment related to Borja where:

- More users than 7 should be considered
- Evaluation of sentences or paragraphs.
- The experiment should run like this:
 - The user (native deaf) is introduced to the experiment.
 - The user is shown 3 news paragraphs.
 - After watching each paragraph, the user is asked about some crucial data explained in the news.
 - Only answers to the third news are considered, while the first two are discarded as adaptation.
- Repetition is not viable as signed paragraphs would be too long to remember and sign back. Our news also include names (skippers, boats, places, etc.) that would definitely lead to mistakes on repetition.

The thorough evaluation of works related to sign language require the assistance and guidance of people that is well known and trusted by the deaf community, to be able to get a significant number of users, which are committed to the evaluation. Unfortunately, we could not manage enough support to carry out this deep evaluation.

5.4. References

Bídia, M., Brom, C., Popelová, M. and Kadlec, R. (2011). StoryFactory – A Tool for Scripting Machinimas in Unreal Engine 2 and UDK. *Interactive Storytelling, Lecture Notes in Computer Science*. Springer Berlin Heidelberg, vol. 7069, pp. 334-337.

Kennaway, J.R., Glauert, J.R.W. & Zwitterlood, I. (2007), Providing signed content on the Internet by synthesized animation.

ACM Transactions on Computer-Human Interaction (TOCHI).
ACM, New York, vol. 14, no. 3, art. 15, pp 1-29.

McDonald, J., Wolfe, R., Schnepf, J., Hochgesang, J., Jamrozik, D.G., Stumbo, M., Berke, L., Bialek, M. and Thomas, F. (2015). An automated technique for real-time production of lifelike animations of American Sign Language. *Universal Access in the Information Society*. Springer Berlin Heidelberg, pp. 1-16.

Unity. (2015). Available from: <<https://unity3d.com>>. [22 December 2015].

Unreal Engine. (2015). Available from:
<<https://www.unrealengine.com>>. [22 December 2015].

6. CONCLUSIONS

The work in this dissertation represents my efforts to automate computer generated media production processes. Throughout the years I have been involved in the automation of different aspects, where I made research contributions:

- specific processes as animation and rigging;
- the complete automation of the production based on templates and pre-made animation clips;
- the workflow management and pipeline data flow.

I started with the automation of the production using templates and animation clips. The results obtained were adequate for some specific serious fields (e.g., weather forecast), but unfortunately were lacking crucial features, such as expressivity and emotions, to empathize with the audience. Thus, I turned my focus towards finding ways to automate the rigging and animation of characters' faces in a way that would not require a lot of resources and could easily deliver understandable expressivity to characters. Once this was achieved and integrated in the previous work, I started researching ways to achieve similar results for the body of characters. The work aimed to enable production companies to reuse animation walk cycles and easily modify the gait to express emotions. In the search for broader usage that would lead to further evaluation, I started working with different companies to integrate and expand the systems developed. This attempt showed that it was necessary first to formalize the production workflow and pipeline and to properly integrate asset and production management systems. Indeed, this would improve artists' productivity in itself and other tools would benefit from this formalization. I started collecting a lot of information through several meetings with industry experts, coming to the formulation of the proposal for formalization contained in chapter two. Through this process, I investigated and developed several tools and systems that help dealing with the complexity of a production pipeline and ease the navigation of the workflow for artists. This helped me in refining my ideas through practical implementation. I present four of these tools and systems, as a representation of the possible benefits and features enabled by my work. During this period, I also had the opportunity to go back

to my initial research steps to develop an automated system to provide sign language video news about the Barcelona World Race regatta around the world, which allowed me to further elaborate over my first research results and test the integration of my later research. Following I summarize the conclusions for each work.

In section 4.1 I presented a method to tweak biped walk-cycle animation on different characters using a dataset of motion captured walk clips. The approach is based on the traditional animation principle of representing mood and emotions of a character through poses. Thus, we show that it is possible to alter an existing walk animation by layering tweak poses along relevant frames of the animation. Our approach is to store and apply the pose information from motion-captured data not as conventional transform matrices of joints rotations and translations, but as a relative transformation related to the body proportions and configuration of the character. This way it is possible tweak the walk animation of any character with a forward kinematic chain for the spine and the head and an inverse kinematic rig for the arms and the legs. The work is still relevant today as it may apply to different production scenarios as videogames characters or crowd visual effects. In line with the more recent orientation of my research, I think that a qualitative or quantitative meaningful evaluation would need to be carried out through extensive use in production. This could be possible for me now, while it was not possible at that stage, due to the instability of the industry in Spain.

The Maskle, presented in section 4.2, is a semi-automatic facial rigging tool that enables users to create joint-based skin weights by simply selecting 10 landmarks on the character's face. The experimental results obtained in a comparative evaluation showed that the result obtained with the Maskle differ by an average (over four faces) of 2.63% against a hand-crafted animation-ready facial rig. It also proved to be useful through use in an experimental production, with a productivity increment of 80%, estimated by animation industry professionals. The use of the Maskle was further evaluated in conjunction with the synthetic model and the emotional interface for expressive facial animation (section 4.3), showing that the professional users were satisfied by the set of tools and that 96% of a surveyed audience considered the obtained facial animation as excellent. This evaluation matches the previous evaluation of these

research works. The synthetic model for facial animation evaluation proved that it is possible to recreate nine basic emotional expressions by interpolating eight target shapes. An internet-based study with 50 people from four different countries (Italy, Spain, Argentina and United Kingdom) showed that the expressions can be associated to a coherent emotional state with an accuracy of 85.92% (over 425 samples). The emotional interface was designed to take advantage of the synthetic model and drive the interpolation of the required shapes through a 2D circularly expressed emotional model that describes emotions by means of activation and evaluation. A test carried on with 18 animation students already trained in standard facial animations techniques indicated that 72% of the users were able to obtain the desired facial expressions and that they were able to satisfactorily interpolate between different expressions. On one hand, these results showed that the synthetic model is flexible enough to obtain more than 9 recognizable facial expressions and that the emotional interface is simple to learn and use; on the other hand, it exposed the need to take asymmetry in consideration as required for expressions of irony or doubt. As the synthetic model for facial expression adopts facial shapes that can be obtained by using the rig created by the Maskle, the integration of the different researches would make the asymmetry possible to integrate by simply adding extra logic to the emotional interface.

With the increasing use of digital doubles and actors in visual effects and the ever improving quality of videogames, the idea of automating the facial rigging process is more important than ever. I will focus future work to adapt the Maskle system to ease the process of transferring shapes across characters. Current shape transfer approaches require the selection of more than 50 landmarks which I believe the Maskle should help identifying, and it requires just 10 landmarks.

In chapter five I present two works that are tightly related to each other, aiming at fully automate the production of short animations for television and web contents. The idea, in both cases, is to use templates (described using xml or json schemes) that indicate how to build the required animation to communicate a certain content. The template relies on pre-made animation clips and other procedural animation algorithms to build the custom data represented in the template as variables. The first system (section

5.1) was extensively used in production; thanks to its user interface (*Programme Editor*), Aactiva Multimedia Digital created three web programmes that would automatically generate weather forecasts during the day for different contexts as location or languages. These contents were available for 10 years, with Sam, the virtual weatherman, being the most advertised of the pack. Sam animated weather forecasts were created by generating different templates with a welcome and goodbye block and a series of weather variable information blocks between them. After collecting the weather forecast data from a service, the template would tell the system how to concatenate animations to communicate it. For each location, the system would generate a rendered animation, accessible through the web. Finally, the system was also fed with text-to-speech audio produced by a software from a third-party provider. The overall system was tested in production and delivered results that satisfied the client, but no formal academic evaluation was performed; it was a rather unique software, and it was difficult to finding suitable methodology and metrics which could be shared as well with the client.

In section 5.2 a similar work is presented, targeted to produce contents for a certain audience: the research focuses on automatically generate animated videos of virtual characters communicating using sign language (International Sign System in our case) for sport events. The work was carried out to give access the Barcelona World Race regatta around the world to the deaf community. Again, the system was designed to rely on templates, but the templates were defined on two layers. A first layer would be used to define simple news, while the system would then merge several simple news templates to create a more complex template with *welcome*, *multiple news* and *goodbye* sections. The news were chosen by crossing the information of the position of the racing boats and the weather forecast. For each template, the animation clips of the required signs were made by hand-animating a digital character using videos of signing experts as a reference. Each clip is stored by specifying the blending properties (duration, fade-in and fade-out) for proper interpolation of signs animations. For custom data (names of boats and skippers or numbers), the system implemented a text-to-sign solution to concatenate letters or numbers based on a character string. The results produced by the system were accessible through the Spanish national broadcasting

website and were the most accessed contents of the many videos about the Barcelona World Race. While no formal evaluation was performed, before broadcasting, all the videos underwent a review process by native signing experts and a review of the news by expert reporters; all generated video news were approved for broadcasting without changes. However, the research around generation of understandable sign language surveyed shows that a proper formal evaluation with the deaf community would contribute to this research and I have included the outline for the evaluation of the system.

Given the importance of understandability of face and body gesture in sign language, I also present an experiment aimed to understand the importance of facial expression in understanding computer generated sign language animation. The general understanding is that facial expressivity is a crucial aspect of sign language, thus we evaluated how mismatched facial expression in both real and virtual signers alter the understandability of signed messages. The experiment is based on the grammar rule stating that open questions (e.g. “who are you?”) are to be made with a surprised-like expression, while yes-no questions (e.g. “are you Marco?”) require a puzzled expression. With a set of eight questions (four for each type) being video recorded and animated, the facial expression was altered using computer animation techniques and a final set of 32 videos was produced. The videos were presented to a group of 36 native signers through a graphic interface and asked to identify which kind of question they were asked. Surprisingly, the results showed that there is no difference in recognising the type of question under the different conditions. In other words, no matter whether the signed question was made by the real signer with coherent or incoherent facial expression, or the virtual signer with coherent or incoherent facial expression, the question type was recognised with similar (non significant difference) precision ratios. The same happened for the time taken to provide the answer, which was also tracked in the experiment. This results are extremely intriguing, surprised the sign language experts that tutored the experiment and deserve further investigation.

The results presented in chapters four and five are related to systems that were developed in conjunction with industrial partners that drove the requirements and, as a direct result, had the system

tailored to be integrated in their (rather simplistic) pipeline. With the goal of integrating my research into a broader production scope, I began working with animation and visual effects companies, but it was clear that the the real bottleneck was not caused by lack of creative tools. In fact, while new algorithms and tools were welcomed as nice ideas, workflow and pipeline design were identified as the most critical and urgent aspects to improve productivity. In chapter 2 I present the result of years of work with different companies to define the core aspects of a workflow and how it relates to pipeline. Furthermore, I define how workflows and pipelines interact with asset and production management systems to enable automation of production processes. Finally, I describe existing tools and reference an ever growing interest from companies and conference towards pipeline work. In chapter 3 I present four tools that take advantage of the contents from chapter 2 to ease and automate aspects of the workflow and the pipeline by taking advantage of asset and production management.

Renderflow (section 3.1) is a system that uses different production assets to provide a rapid rendering of high quality dailies. By automating the workflow through layout (which assets are in shot to be rendered), animation (which animations have to be assigned to animatable assets), lighting (how things have to look) and compositing (how each rendered element has to be layered), *Renderflow* can automatically produce a near-to-final render that satisfies the need of supervisors and clients to review contents that look as close as possible to the final product. Furthermore, by assetizing the separate renders for each three-dimensional asset in shot it saves render time, as existing renders are reused if the properties (animation, shaders, etc.) have not changed. The system is currently used in production at the Moving Picture Company and further work is in progress to improve how the system optimizes the layering of rendered assets, shadows and statistics. This new work is intended to be published as as a SIGGRAPH talk and a journal paper in 2016.

RVGun (section 3.2) is a system that binds production management (Shotgun) into a media playback software (RV) to enable artists to register videos, single frames and sequences into the assets management system. The tool guides users through the process and takes care of encoding the delivery video and metadata properly,

according to the studios standards. Through an initial evaluation with 5 professional artists, the tool proved to be beneficial and increased productivity. With the tool artists made no mistakes in registering the delivery while, with standard processes, all users made some sort of error and two out of five made a serious error that caused a wrong delivery. Furthermore, RVGun reduced the time taken for the delivery by 61% (162 seconds faster).

In section 3.3 I present a production pinboard prototype. This can be used to visually create, edit and arrange production data by using a virtual pinboard. The prototype deals with the processes of defining sequences and shots (including the list of assets, dialogues, actions, etc.), as well as three-dimensional assets design (references and art design) and synchronizes all the information with the production and assets management systems. The tool was designed with the help of an experienced layout and storyboard artist, but was not evaluated. The tool is promising and will hopefully solve one of the issues with modern media production: the large amount of assets makes it difficult to review contents in contexts using lists and tables. Thus, future work should deal with evaluating the existing prototype and then extend it to other production aspects. Another interesting aspect to improve would be to introduce graphical resource management as a feature that enables coordinators to assign artists to production tasks and visually understand how work is shared across the floor.

Section 3.4 describes a system that integrates asset management data into rendered images. A recurrent issue in computer generated media production is the review of dailies and the communication of changes to be made. Due to the large number of assets that can appear in one frame of a daily, it can be difficult to properly identify which assets need to be changed and there is no way to unequivocally signal these assets to the artist. Typically, this happens by digitally marking the problematic assets with a digital paint over one or more frames of the daily. With the presented system, it is possible to identify the asset involved in the rendering of each pixel as it encodes the database id of the asset in the pixel. Again, no evaluation was performed and it would be necessary to exactly identify what to evaluate and which metrics to apply. In terms of future work, this idea should be integrated with

Renderflow to ease the review of the produced dailies and design further automation and optimizations.

All in all, I believe that strengthening the evaluation of the presented works in the industrial context would be very valuable. This would help to better understand the benefits of the proposed solutions and define further improvements. Nevertheless, it is necessary to remark that media production is an extremely competitive and performance-driven environment and it is extremely rare to have the resources (testers and time) to reliably evaluate new solutions. Thanks to evaluation currently being carried out on the *RenderFlow*, presented in section 3.1, a paper is being written to be submitted for publication on the IEEE's *MultiMedia* journal [MultiMedia, 2015].

A particularly interesting and delicate issue is the evaluation of workflow and pipeline work, as well as innovation in the field of asset and production management. There are no established metrics to measure the increase (or decrease) of productivity caused by these factors and the consequent tools built around them. The main problem is that it is not possible to properly test two different production environments on a similar set of assets and a controlled evaluation setup might not represent the fuzzy reality and the scale of a real production, returning unreliable or non significant data. Thus, future work should address the need for an evaluation framework and set of procedures to validate the effectiveness of innovative solutions in this ever-growing field. Finally, it is important to keep in mind the changing nature of the industry that is now geographically delocalized and has to address the issue of keeping assets and communication synchronized. Nevertheless, it is important to remark that the vast majority of CG media production studios are still relying on intricate file structures, file naming conventions and a lot of care by artists to understand where things are and how to access them. This undermines any future development within such a demanding industry.

To conclude, beyond thoroughly investigating and evaluating workflow and pipeline solutions, I now have the opportunity to also investigate further my research on facial animation and extend the work towards hyper-realistic digital actors, which are becoming a defining product for visual effects companies. While the results

presented in this thesis aim at delivering a simple, functional facial expressivity with little or no effort, in film industry I can elaborate on my findings and improve the techniques to scale them to detailed characters and physical simulation of human tissues.

BIBLIOGRAPHY

3DS Max, computer software. (2015). Available from: <<http://www.autodesk.com/products/3ds-max>>. [13 December 2015].

Abadia, J., Evans, A., Ganzales, E., Gonzales, S., Soto, D., Fort, S., Romeo, M. & Blat, J. (2009). Assisted Animated Production Creation and Programme Generation. *ACM Advances in Computer Entertainment Technology*. ACM, New York, pp. 207-214.

Adobe Premiere. (2009). Available from: <<http://www.adobe.com/products/premiere/>>. [31 December 2015].

Alan01. (2010). Available from: <<http://mlab.taik.fi/alanonline/#>>. [13 December 2015].

Alías, F., Iriondo, I., Formiga, L., Gonzalva, X., Monzo, C. & Sevillano, X. (2005). High quality Spanish restricted-domain TTS orientated to a weather forecast application. *Proc. 9th European Conference on Speech Communication and Technology (Interspeech)*. pp. 2573-2576.

Amidror, I. (2002). Scattered data interpolation methods for electronic imaging systems, a survey. *Journal of Electronic Imaging*. vol. 11, no. 2, pp. 157-176.

Apple iMovie. (2009). Available from: <<http://www.apple.com/imovie/>>. [31 December 2015].

Aristidou, A., Charalambous, P. & Chrisanthou, Y. (2015). Emotion Analysis and Classification: Understanding the Performers' Emotions Using the LMA Entities. *Computer Graphics Forum*, vol. 36, no. 6, pp. 262-276.

Arnold. (2015). Available from: <<https://www.solidangle.com/arnold/>>. [18 December 2015].

Autodesk Maya Press. (2007). Learning Autodesk Maya 2008: The Modeling & Animation Handbook. Sybex.

Avid. (2013). Available from: <<http://connect.avid.com/MAM-ROI-Whitepaper.html>>. [23 December 2015].

Badler, N., Phillips, C. & Lynn, B. (1993) Simulating Humans: Computer Graphics and control. Oxford University Press, New York.

Barcelona Media, 2010. Available from: <http://www.barcelonamedia.org/report/the-i3media-project-reaches-successfully-its-first-year-research-objectives_231>. [13 December 2015].

Baylor, A. & Ebbers, S. (2003). The Pedagogical Agent Split-Persona Effect: When Two Agents are Better than One. *World Conference on Educational Multimedia, Hypermedia and Telecommunications*. pp. 459–462.

Bídia, M., Brom, C., Popelová, M. and Kadlec, R. (2011). StoryFactory – A Tool for Scripting Machinimas in Unreal Engine 2 and UDK. *Interactive Storytelling, Lecture Notes in Computer Science*. Springer Berlin Heidelberg, vol. 7069, pp. 334-337.

Bruderlin, A. & Williams, L. (1995). Motion Signal Processing. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, New York, pp. 97-104.

Brutzman, D. & Daly L. (2007). X3D: Extensible 3D Graphics for Web Authors, Morgan Kaufmann Publishers Inc. San Francisco.

Bulterman, D. & Rutledge, L. (2008). SMIL 3.0 - Interactive Multimedia for the Web, Mobile Devices and Daisy Talking Books. X.media.publishing.

Butler, G., Langlands, A. & Ricklefs, H. (2008). A pipeline for 800+ shots. *Proceedings of ACM SIGGRAPH 2008 talks*. ACM, New York, article. 72.

Burley, B. & Lacewell, D. (2008). Ptex: Per-Face Texture Mapping for Production Rendering. *Proceedings of the Nineteenth*

Eurographics conference on Rendering. Eurographics Association, Aire-la-Ville, Switzerland, pp. 1155-1164.

Chung, H. J. (2011). 'Global Visual Effects Pipelines: An Interview with Hannes Ricklefs', *Media Fields Journal*, no. 10.

Costantini, E., Pianesi, F. & Prete, M. (2005). Recognising emotions in human and synthetic faces: the role of the upper and lower parts of the face. *Proceedings of the 10th international conference on Intelligent user interfaces*. pp. 20-27.

Cuyer, E. (1902). *La mimique*. Publications Service O. Doin, Paris.

Dehn, D.M. & Van Mulken, S. (2000). The impact of animated interface agents: a review of empirical research. *International Journal of Human-Computer Studies*. vol. 52, no. 1, pp.1–22.

Densley, D. & Willis, P. (1997). Emotional Posturing: A Method Towards Achieving Emotional Figure Animation. *IEEE Computer Animation 1997* . IEEE Computer Society, Washington DC, pp 8-14.

Dittmer, J. (2012). Bringing order to the creative department using metadata-driven workflows. *Journal of Digital Media Management* vol. 1, no. 2, pp. 176-182. Henry Stewart Publications LLP, London.

Dogson, N.A. (2014). Going to the movies: Lessons from the film industry for 3d libraries. *3D Research Challenges in Cultural Heritage Lecture Notes in Computer Science*. Springer Berlin Heidelberg, vol. 8355, pp. 93-103.

Dunlop, R. (2014). *Production Pipeline Fundamentals for Film and Games*, Focal Press, New York.

Ekman, P., Friesen, W. & Ellsworth, P. (1982). What emotion categories or dimensions can observers judge from facial behaviour? *Emotion in the Human Face*. Ed. Ekman, P.

Ekman, P. (1999) Facial Expressions. *The Handbook of Cognition and Emotion*, John Wiley & Sons Ltd., Sussex.

Elliott, R., Glauert, J.R.W., Kennaway, J.R., Marshall, I. & Safar, E. (2007). Linguistic modelling and languageprocessing technologies for Avatar-based sign language presentation. *Universal Access in the Information Society*. Springer-Verlag, vol. 6, no. 4, pp.375–391.

Ericson, C. (2005). *Real-time Collision Detection*. Morgan Kaufmann.

Ersotelos, N. & Dong, F. 2008. Building highly realistic facial modeling and animation: a survey. *The Visual Computer*. Springer-Verlag, vol. 24, no. 1, pp. 13-30.

Evans, A., Romeo, M., Dematei, M. & Blat, J. (2009). The Maskle: Automatic Weighting for Facial Animation. International Conference on Computer Graphics Theory and Applications (GRAPP).

Evans, A., Romeo, M., Bahrehmand, A., Agenjo, J. & Blat, J. (2014). 3D graphics on the web: A survey. *Computers and Graphics*. vol. 41, no. 0, pp. 43-61.

Fagnou, D., Cameron C., & Valdez, A. (2013). ReviewTool: a database-driven visual effects editing application. *ACM SIGGRAPH 2013 Talks*, ACM, New York, article 28.

Faigin G. (1990) *The Artist's Complete Guide to Facial Expression*. Watson-Guptill Publications.

Faita, C., Vanni, F., Lorenzini, C., Carrozzino, M., Tanca, C. & Bergamasco, M. (2015). Perception of Basic Emotions from Facial Expressions of Dynamic Virtual Avatars, *Lecture Notes in Computer Science*. Springer International Publishing, Switzerland, vol. 9254, pp. 409-419.

Fallows, D. (2004). *The Internet and Daily Life*. Pew Research Center's Internet & American Life Project. Available from: <http://www.pewinternet.org/Reports/2004/The-Internet-and-Daily-Life.aspx>. [24 July 2013].

García-Rojas A., Vexo, F., Thalmann, D., Raouzaïou, A., Karpouzis, K., Kollias, S., Moccozet, L. & Magnenat-Thalmann, N., (2006). Emotional face expression profiles supported by virtual human ontology. *Computer Animation and Virtual Worlds*. John Wiley and Sons Ltd., vol. 17, pp. 259-269.

Girgensohn, A., Boreczky, J., Chio, P., Doherty, J., Foote, F., Golovchinsky, G., Uchihashi, S. & Wilcox, L. (2008). A semi-automatic approach to home-video editing. *Proc. ACM User interface software and technology*. ACM, New York, pp. 81-89.

Gombrich, E.H. (1972). *The Mask and the Face: the perception of physiognomic likeness in life and in art. Art, perception and reality*. The Johns Hopkins University Press, Baltimore and London, pp. 1-46.

Gombrich, E.H. (1972). *Invention and Discovery. Art and Illusion: A Study in the Psychology of Pictorial Representation*. Trustees of the National Gallery of Art, Washington D.C.

Goto, T., Escher, M., Zanardi, C. & Magnenat-thlamann N. (1999). MPEG-4 based Animation with Face Feature Tracking. *Proceedings Eurographics Workshop on Computer Animation and Simulation*. pp. 89-98.

Grünvogel, S.M., Lange, T. & Piesk, J. (2002) Dynamic Motion Models. *Eurographics 2002 - Short Presentations*. Eurographics Association.

Guenther, B., Grimm, C., Wood, D., Malvar, H., & Pighin, F. (1998). Making Faces, *Proceedings of ACM SIGGRAPH*. ACM, New York, pp. 55-66.

Hanke, T. (2004). HamNoSys - representing sign language data in language resources and language processing contexts. *LREC 2004, Workshop proceedings : Representation and processing of sign languages*. pp. 1–6.

Hillman, P. (2013). *The Theory of OpenEXR Deep Samples*. Available from: <<http://www.openexr.com/documentation.html>>. [13, December, 2015].

Hyde, J., Kiesler, S., Hodgins, J. K. & Carter E. J. (2014). Conversing with Children: Cartoon and Video People Elicit Similar Conversational Behaviors. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, pp. 1787-1786.

Hodgins, J., Wooten, W., Brogan, D. & O'Brian, J. (1995). Animating human Athletics. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, New York, pp. 71-78.

Jetter, H.C., Gerken, J, Zöllner, M. & Reiterer, H. (2010). Model-Based Design and Implementation of Interactive Spaces for Information Interaction. *Human-Centred Software Engineering*. Springer Berlin Heidelberg, London, vol. 6409, pp 22-37.

Johnson, C., Tobiska, J., Tomlinson, J., Van de Bosh, N. & Whaley, W. (2014). A framework for global visual effects production pipelines. ACM SIGGRAPH 2014 Talks, ACM, New York, article 57.

Johnston, O. & Thomas, F. (1995). *The Illusion of Life: Disney Animation*. Disney Editions.

Johnson, W.L. & Rickel, J. (1997). Steve: an animated pedagogical agent for procedural training in virtual environments. *ACM SIGART Bulletin*. vol. 8, no. 1-4, pp. 16–21.

Joshi, P., Tien, W.C., Desbrun, M. & Pighin, F. (2003). Learning controls for blend shape based realistic facial animation. *Symposium on Computer Animation Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. pp. 187-192.

Journal of Digital Media Management. (2015). Available from: <<http://www.henrystewartpublications.com/jdmm>>. [13, December, 2015].

Kalra, P., Mangili, A., Thalmann, N., & Thalmann, D. (1992). Simulation of Facial Muscle Actions Based on Rational Free From Deformation. *Eurographics*, vol. 11, no. 3, pp. 59-69.

Kennaway, J. R., Glauert, J. R. W., Zwitserlood, I. (2007). Providing signed content on the Internet by synthesized animation. *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 14, no. 3, Art. 15, pp 1-29.

Kouadio, C., Poulin, P. & Lachapelle, P. (1998). Real-Time Facial Animation based upon a Bank of 3D Facial Expressions. *Proceedings of Computer Animation '98 Conference*. pp. 128.

Lee, Y., Terzopoulos, D., & Waters, K. (1995). Realistic Modeling for Facial Animation. *Proceedings of ACM SIGGRAPH*. ACM, New York, pp. 55-62.

Lewis, J.P., Cordner, M., & Fong, N. (2000). Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. *Proceedings of 27th annual conference on Computer graphics and interactive techniques*. ACM, New York, pp. 165-172.

Mayor, O., Bonada, J. & Janer, J. (2009). Voice Transformation from interactive installations to Video-Games. *Proc. 25th AES Conference: Audio for Games*.

McDonald, J., Wolfe, R., Schnepf, J., Hochgesang, J., Jamrozik, D.G., Stumbo, M., Berke, L., Bialek, M. and Thomas, F. (2015). An automated technique for real-time production of lifelike animations of American Sign Language. *Universal Access in the Information Society*. Springer Berlin Heidelberg, pp. 1-16.

Meeres-Young, G., Ricklefs, H. & Tovell, R. (2010). Managing thousands of assets for the prince of persia city of Alamut. *Proceedings of ACM SIGGRAPH 2010 talks*. ACM, New York, article 30.

Meredith, M. & Maddock, S. (2005). Adapting Motion Capture Data Using Weighted Real-Time Inverse Kinematics. *ACM Computer in Entertainment*. vol. 3, no. 1 pp. 5-5.

Meteo Sam. (2009). Not available.

Mittring, M. (2007). Finding next gen: Cryengine 2. *Proceedings of ACM SIGGRAPH 2007 courses*. ACM, New York, pp. 97-121.

- Möller, T. (1997). A fast triangle-triangle intersection test. *Journal of Graphics Tools*. vol. 2, no. 2, pp. 25-30.
- Moundridou, M. & Virvou, M. (2002). Evaluating the persona effect of an interface agent in a tutoring system. *Journal of Computer Assisted Learning*. vol. 18, no.3, pp.253-261.
- MultiMedia. (2015). MultiMedia IEEE. Available from: <<http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=93>>. [4 January 2016].
- Newman, A.J., Supalla, T., Hauser, P.C., Newport, E.L. & Bavelier, D. (2010). Prosodic and narrative processing in American Sign Language: an fMRI study. *NeuroImage*. vol. 52, no. 2, pp.669–76.
- Noa&Max. (2013). Available from: <<http://www.noamax.tv/>>. [27 December 2015].
- Noh, J. and Neumann, U. (1998). A survey of facial modeling and animation techniques. *USC Technical Report*.
- Noh, J. & Neumann, U. (2001). Expression Cloning. *Proceedings of ACM SIGGRAPH 2001*. ACM, New York, pp. 277-288.
- Noteboom, R. (2013). Configurable-WorkflowManagement. Available from: <<http://www.southpawtech.com/wp-content/uploads/2013/05/configurable-workflow-management.pdf>>. [27 December 2015].
- OGRE – Open Source 3D Graphics Engine (2015). Available from: <<http://www.ogre3d.org>> [31 December 2015].
- Open Shading Language. (2015). Available from: <<http://opensource.imageworks.com/?p=osl>>. [18 December 2015].
- Orvalho, V., Zacur, E., & Susin, A. (2006). Transferring a Labeled Generic Rig to Animate Face Models. *Lecture Notes in Computer Science*. vol. 4069, pp. 223-233.

- Othman, A., El Ghouli, O. & Jemni, M. (2010). SportSign: A Service to Make Sports News Accessible to Deaf Persons in Sign Languages. *Lecture Notes in Computer Science*. vol. 6180, pp.169–176.
- Parke, F. (1982). A parameterized Model for Facial Animation. *IEEE Computer Graphics and Applications*. vol. 2, no. 9, pp. 61-68.
- Plachaud, C. (1991). Communication and Coarticulation in Facial Animation. *PhD thesis*. University of Pennsylvania, pp. 86-90.
- Platt, S., & Badler, N. (1981). Animating Facial Expressions. *Computer Graphics*. vol. 15, no. 3, pp. 245-252.
- Plutchik R. & Kellerman H. (2015), The Measurement of Emotions, *Emotion: Theory, Research and Experience*, vol. 4, pp. 113-129.
- Perlin, K. & Goldberg, G. (1996). Improv: A system for scripting interactive actors in virtual worlds. Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. pp. 205-218.
- Plasencia, C. (1993). El rostro humano, Observación expresiva de la representación facial. Publications service, Universidad Politecnica de Valencia, Valencia.
- Polson B. (2014). CG pipeline design patterns. *Proceedings of the Fourth Symposium on Digital Production*, pp. 29-29.
- PostgreSQL. (2015). Available from: <<http://www.postgresql.org/>>. [27 December 2015].
- Pushkar, J., Wen, C.T., Mathieu, D. & Frédéric, P. (2005). Learning controls for blend shape based realistic facial animation. *Proceedings International Conference on Computer Graphics and Interactive Techniques*. ACM, New York.
- Rausaiou, A., Karpouzis, K., Kollias, S. (2003). Online Gaming and Emotion Representation. *Proceedings of the International Workshop on Very Low Bitrate Video Coding (VLBV)*. pp. 295-320.

- Redboard. (2009). Available from: <<http://www.redboard.tv/>>. [31 December 2015].
- Ricklefs, H. (2013). Pronto: scheduling the un-schedulable. *ACM SIGGRAPH 2013 Talks*, ACM, New York, article 29.
- Romeo, M., Dematei, M., Bonequi, J., Evans, A. & Blat, J. (2010). A reusable model for emotional biped walk-cycle animation with implicit retargeting. *Proceedings of the 6th international conference on Articulated motion and deformable objects*. Springer-Verlang, Berlin, pp 270-279.
- Romeo, M., Evans, A., Pacheco, D. & Blat, J. (2014). Specific Sign Language Animation for Virtual Characters. *GRAPP 2014 - Proceedings of the 9th International Conference on Computer Graphics Theory and Applications*. SciTePress, pp. 487-494.
- Romeo, M., Auty, J. & Fagnou, D. (2015). Intelligent rendering of dailies: automation, layering and reuse of rendered assets. *Proceedings of the 12th European Conference on Visual Media Production (CVMP '15)*. ACM, New York, article 15.
- RV, computer software. (2015). Available from: <<http://www.tweaksoftware.com/products/rv>>. [13 December 2015].
- Ryokai, K., Vaucelle, C. & Cassell, J. (2003). Virtual peers as partners in storytelling and literacy learning. *Journal of Computer Assisted Learning*. Blackwell Publishing Ltd, vol. 19, no. 2, pp. 195-208.
- Saito J. (2013). Smooth contact-aware facial blendshapes transfer. *DigiPro '13 Proceedings of the Symposium on Digital Production*. ACM, New York, pp. 7-12.
- Salero: Semantic Audiovisual Entertainment Reusable Objects. (2006). Available from: <<http://www.salero.eu>>. [13 December 2015].
- Shotgun, computer software. (2015). Available from: <<http://www.autodesk.com/products/shotgun>>. [13 December 2015].

SIGGRAPH2013 USD Presentation. (2013). Available from: <http://graphics.pixar.com/usd/s2013_presentation.html>. [13 December 2015].

SIGGRAPH 2015, Global VFX Pipelines. (2015). Available from: <<http://s2015.siggraph.org/attendees/birds-feather/sessions/global-vfx-pipelines>>. [13, December, 2015].

Simon M. (2005). A visual reference for artists. Watson-Guptill Publications.

Sonka, M & Fitzpatrick, J. (2000). Handbook of Medical Imaging. *Medical Image Processing and Analysis*. SPIE Press. vol. 2.

Sumner, R.W. & Popović, J. (2004). Deformation transfer for triangle meshes. *ACM Transactions on Graphics*. vol. 23, no. 3 (Aug.), pp. 399–405.

Third Phase Experimental Productions and Evaluation Report SALERO. (2009). Available from: <<http://salero.eu/en/resources/deliverables.html>>. [13 December 2015].

Tuomola, M. L., Korpilahti, T., Pesonen, J., Singh, A., Villa, R., Punitha, P., Feng, Y. & Jose, J. M. (2009). Concept, content and the convict. *Proceedings of the 17th ACM international conference on Multimedia*, ACM, New York, pp. 1063-1072.

Unama, M., Anjyo, K., Takeuchi, R. (1995). Fourier Principals for Emotion-based Human Figure Animation. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, New York, pp. 91-96.

Unity. (2015). Available from: <<https://unity3d.com>>. [22 December 2015].

Unreal Engine. (2015). Available from: <<https://www.unrealengine.com>>. [22 December 2015].

Van Mulken, S., André, E. & Müller, J. (1998). The Persona Effect: How Substantial Is It? *People and Computers*. vol. 13, pp.53-66.

- Xtranormal. (2009). Available from: <<http://www.xtranormal.com/>>. [31 December 2015].
- Yasur, J.R. (2013). Metadata management in the entertainment industry: A case study from the studio perspective. *Journal of Digital Media Management*, vol. 2, no. 2, pp. 111-119. Henry Stewart Publications LLP, London.
- Wallraven, C., Breidt, M., Cunningham, D.W. & Bülthoff, H.H. (1999). Psychophysical evaluation of animated facial expressions. *Proceedings of the 2nd Symposium on Applied Perception in Graphics and Visualization*. pp. 259-269.
- Waters, K. (1987). A Muscle Model for Animating Three-Dimensional Facial Expression. *Computer Graphics*. vol. 21, no. 4, pp. 17-24.
- Waters, K., & Frisbie, J. (1995). A coordinated Muscle Model for Speech Animation. *Graphics Interface*. pp. 163-170.
- Weise, T., Bouazis, S., Li, H. & Pauly, M. (2011). Realtime performance-based facial animation. *ACM SIGGRAPH 2011 papers on - SIGGRAPH '11*. ACM, New York, Press, pp. 1.
- Whissell C. M. (1989). *The Dictionary of Affect in Language*.
- Williams, R. (2001). *The Animator's Survival Toolkit*. Faber and Faber.
- Witkin, A. & Popovic, Z. (1995). Motion Warping. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, New York, pp. 105-108.
- Woll, B. (1983). The semantics of British Sign Language signs, *Language in Sign*. J. Kile & B. Woll (Eds.), London, pp. 41-45.
- World Federation for the Deaf. (2013). Human Rights. Available from: <<http://wfdeaf.org/human-rights>>. [24 July 2013].