# 3D STORYBOARDING FOR MODERN ANIMATION

ALEXANDROS GOUVATSOS

Doctor of Engineering in Digital Media
Media School
Bournemouth University

Centre for Digital Entertainment
University of Bath

September 2017

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

Alexandros Gouvatsos: *3D Storyboarding for Modern Animation*
© September 2017

SUPERVISORS:

Dr. Zhidong Xiao
Prof. Jian J. Zhang
Jerry Hibbert
Neil Marsden

## ABSTRACT

Animation is now a classic medium that has been practiced for over a century. While Disney arguably made it mainstream with some hand-drawn classics, today's industry is focused on Three-Dimensional (3D) animation.

In modern 3D animation productions, there have been significant leaps in terms of optimising, automating and removing manual tasks. This has allowed the artistic vision to be realised within time and budget and empowered artists to do things that in the past would be technically more difficult. However, most existing research is focused on specific tasks or processes rather than the pipeline itself. Moreover, it is mostly focused on elements of the animation production phase, such as modelling, animating and rendering. As a result, pre-production parts like storyboarding are still done in the traditional way, often drawn by hand. Because of this disparity between the old and the new, the transition from storyboarding to 3D is prone to errors.

3D storyboarding is an attempt to adapt the pre-production phase of modern animation productions. By allowing storyboard artists access to simple but scale-accurate 3D models early on, drawing times as well as transition times between pre-production and production can be reduced. However, 3D storyboarding comes with its own shortcomings. By analysing existing pipelines, points of potential improvement are identified. Motivating research from these points, alternative workflows, automated methods and novel ideas that can be combined to make 3D animation pipelines more efficient are presented. The research detailed in this thesis focuses on the area between pre-production and production. A pipeline is presented that consists of a portfolio of projects that aim to:

- Generate place-holder character assets from a drawn character line-up
- Create project files with scene and shot breakdowns using screenplays
- Empower non-experts to pose 3D characters using Microsoft Kinect
- Pose 3D assets automatically by using 2D drawings as input

# CONTENTS


I RESEARCH BACKGROUND 18

1 INTRODUCTION 19
- 1.1 Hibbert Ralph Animation . . . 19
- 1.2 Research focus . . . 20
- 1.3 Impact . . . 21
- 1.4 Research contributions . . . 21
- 1.5 Thesis structure . . . 22

2 DIGITAL MEDIA IN 3D ANIMATION PRE-PRODUCTION 24
- 2.1 An analysis of animation as a medium . . . 24
- 2.2 Animation as media, animation in media and media in animation . . . 27
- 2.3 Pipeline elements . . . 29
- 2.4 Breakdown of different media used in pre-production . . . 32
- 2.5 Animation producers as consumers of media . . . 36
- 2.6 Remediation of the pre-production phase . . . 39

3 STORYBOARDING IN 3D 42
- 3.1 Why storyboard in 3D . . . 42
- 3.2 3D storyboarding in theory . . . 43
- 3.3 3D storyboarding in practice . . . 44
  - 3.3.1 Storyboarding . . . 47
  - 3.3.2 Layout . . . 47
  - 3.3.3 The HRA pipeline . . . 48
- 3.4 Moving forward . . . 48

II PROJECTS PORTFOLIO 51

4 AUTOMATIC CREATION OF PLACE-HOLDER ASSETS 53
- 4.1 Motivation . . . 53
- 4.2 Background . . . 54
- 4.3 Methodology . . . 55
- 4.4 Results . . . 58

4.5 Discussion & future work . . . 58
5 SCREENPLAY ANALYSIS SYSTEM FOR AUTOMATIC ASSET IMPORT 65
5.1 Motivation . . . 65
5.2 Background . . . 66
5.3 Methodology . . . 68
5.3.1 Parsing the screenplay . . . 68
5.3.2 Linking the information . . . 69
5.3.3 Generating Redboard project file . . . 70
5.4 Results . . . 71
5.5 Discussion & future work . . . 73
6 INTEGRATION OF PLACEHOLDER CREATION AND SCREENPLAY ANALYSIS 75
6.1 Motivation . . . 75
6.2 Background . . . 75
6.3 Methodology . . . 76
6.4 Results . . . 77
6.5 Discussion & future work . . . 77
7 CHEAP MOTION-CAPTURE DEVICES FOR HUMANOID POSING IN PRE-PRODUCTION 80
7.1 Motivation . . . 80
7.2 Background . . . 80
7.3 Methodology . . . 83
7.3.1 Mapping of storyboard timeline to Maya timeline . . . 84
7.3.2 Mapping skeleton capture data to 3D rigged assets . . . 85
7.4 Results . . . 89
7.5 Discussion & future work . . . 91
8 AUTOMATIC 3D POSING FROM 2D HAND-DRAWN SKETCHES 96
8.1 Motivation . . . 96
8.2 Background . . . 97
8.2.1 Related work . . . 98
8.2.2 State-of-the-Art work . . . 104
8.2.3 Proposed Approach . . . 109
8.3 Methodology . . . 109
8.3.1 Overview . . . 110
8.3.2 PARAC-LOAPSO . . . 112
8.3.3 Comparing Drawings to Renders . . . 119

8.4 Results . . . . . 122
8.5 Discussion & future work . . . . . 129

III CONCLUDING REMARKS 135
9 DISCUSSION & FUTURE WORK 136
10 CONCLUSION 144

REFERENCES 147
GLOSSARY 162

# LIST OF FIGURES


Figure 1 Example flow of a traditional pipeline with its distinct stages. . . . 29
Figure 2 Example of pre-vis, with overlay of the corresponding storyboard panel at the bottom right. . . . 31
Figure 3 Example screenplay from the movie 'Wolf of Wall Street' (Winter and Belfort 2013). . . . 32
Figure 4 Example storyboard (Glebas 2009). . . . 33
Figure 5 Screenshot of Redboard. . . . 43
Figure 6 Example of 3D storyboarding, with 2D drawing on top of a 3D environment. The 3D models are unposed. 45
Figure 7 Efficient storyboarding workflow example from Tree Fu Tom (FremantleMedia, CBeebies, Blue-Zoo Productions 2012). (©BBC MMXVI) . . . 50
Figure 8 Example of input character drawings (© Hibbert Ralph Animation). . . . 55
Figure 9 Automatically generated 3D pegs as seen in Autodesk Maya. . . . 55
Figure 10 Examples of automatically generated character abstractions. . . . 60
Figure 11 Screenshot of the Peg Creator interface. . . . 61
Figure 12 Step 1 — Segmenting the character from the line-up. Step 2 — Trimming it based on the feet. Step 3 — Averaging the colours into stripes to create a 'mosaic' effect. Step 4 — Generating the rectangular cuboid and adding a plane on top to convey orientation. . . . 61
Figure 13 The same character drawn differently can result in a different bounding box that may affect how the character's width is estimated. . . . 62

Figure 14 Side by side comparison of the original character drawing (left-most) and different mosaic stripe thickness parameters: 10, 40 and 100 respectively from left to right. . . . . 62
Figure 15 Screenshot of Final Draft software. . . . . 67
Figure 16 Automatically populated Redboard project from 'Q Pootle 5' (Blue-Zoo Productions 2013) screenplay. . . . 71
Figure 17 Automatic shot breakdown as seen in Redboard. . . . . 72
Figure 18 Structure of integration between the Peg Creation and the Screenplay Analysis software. . . . . 76
Figure 19 A screenshot of the Peg Creation software, where dropdown boxes allow the artist to choose the character's name. . . . . 79
Figure 20 The created system running within Maya, tracking the pose of an actor while viewing a series of storyboard panels. . . . . 83
Figure 21 The main User Interface (UI) of the system, as an actor is posing in an office environment, according to the selected storyboard panel. . . . . 85
Figure 22 For 25 frames per second, the first panel corresponds to the first frame in Maya's timeline. . . . . 86
Figure 23 For 25 frames per second, the second panel corresponds to the 25th frame in Maya's timeline. . . . . 86
Figure 24 The skeletal structure of the Kinect data. There are 20 joints on the human body tracked from the Kinect sensor (Microsoft 2017). . . . . 87
Figure 25 Kinect data needs to be transformed to be used in Maya. 88
Figure 26 The skeletal structure of an example 3D rigged T-posed asset in Maya. Note that there is not a one-to-one correspondence to the skeletal structure of the Kinect data shown in Figure 24. . . . . 90

Figure 27 A screenshot during practice for the live demo sequence. The top left window shows the actor acting out the pose in front of the Kinect, with the joints overlaid on top. The bottom left window shows the storyboard panel the actor is currently acting out. The Autodesk Maya window is updated whenever the actor captures a new pose. . . . . 90
Figure 28 An example of an actor posing a 3D character in Maya, according to the selected storyboard panel. . . . 92
Figure 29 An example of a character in a Maya scene posed using the proposed pose capture system. The actor is in this case controlling the computer and thus does not appear in front of the camera. . . . . 93
Figure 30 An example of the forward-backward ambiguity issue in inferred 3D pose of a lamp when not using the internal lines for additional information. . . . . 104
Figure 31 Character poses created by sketching lines of action (Guay et al. 2013). . . . . 105
Figure 32 Bone rotations are constrained to the viewing plane. The bones are parameterised as a single axis-angle component. The axis is the camera's viewing direction projected onto the floor plane. From left to right is the initial pose, a side view with the LOA, the result after optimisation, and a frontal view of the final pose (Guay et al. 2013). . . . . 106
Figure 33 The pose descriptor consists of in-the-image plane angles for every limb segment (Jain et al. 2009). . . . . 108
Figure 34 Character poses that match the hand animation much more closely than the mocap animation. 'Our result' refers to the work by Jain et al. (2012). . . . . 108
Figure 35 Different descriptors for comparing poses in 2D. . . . . 110
Figure 36 Flow chart of original Particle Swarm Optimisation algorithm. . . . . 113

Figure 37 The ring local topology of a particle swarm (on the left) allows each particle to communicate with two other particles, while the global topology (on the right) allows particles to exchange information with all other particles. . . . . 116
Figure 38 Example of silhouette comparison to calculate similarity between two poses X and Y. Green colour signifies the silhouette of pose X, red colour the silhouette of pose Y and blue colour the overlap between the two poses. . . . . 122
Figure 39 The 3D assets of the lamp, horse and human shown with their joint hierarchies and controllers in Autodesk Maya. . . . . 123
Figure 40 Side by side comparison of the drawings and less successfully estimated 3D poses (bottom). . . . . 123
Figure 41 Side by side comparison of the drawings (top) and the estimated 3D pose (bottom) for the horse model. . . . . 125
Figure 42 Side by side comparison of a drawing (left) and the estimated 3D pose (right) for the human model. The estimated 3D pose is close to the drawing. . . . . 126
Figure 43 Side by side comparison of the drawings (top) and the estimated 3D pose (bottom) for the lamp model. . . . . 126
Figure 44 From initial storyboard to pre-visualisation. The lamp in these three shots is automatically posed using the storyboard drawing. On the left column, drawings are overlaid on top of the 3D scene with the assets unposed. On the right column, the lamp has been posed automatically using the proposed method. . . . . 127
Figure 45 Side by side comparison of a sequence of drawings (top) with temporal relationships and a sequence from the estimated 3D poses (bottom) for the lamp model. Lighter opacity and colours signify frames earlier in the sequence while darker opacity and colours signify frames later in the sequence. . . . . 128

# LIST OF TABLES


Table 1 Peg Creator software classes. . . . . 56

Table 2 Number of strokes and rotations and time taken using lines of action versus using 3D IK widgets in Maya (Guay et al. 2013). . . . . 107

Table 3 Runtime comparison for the human model, for various PARAC-LOAPSO swarm sizes, on different hardware as well as a non-parallel implementation on the CPU . . . . . 132

# LIST OF CODE SNIPPETS

Code Snippet 1 Peg creation algorithm pseudo-code. Input: image with character drawings (image file), number of characters in input image (integer). Output: 3D peg model for each character (.OBJ file). . . . . 63
Code Snippet 2 Pseudo-code to average colours of a character image in order to create mosaic effect. . . . . 64
Code Snippet 3 Hip rotation, legs correction and A-Pose rotations in Autodesk Maya Python/MEL pseudo-code. . . . . 95

*Making the process better, easier, and cheaper is an important aspiration,*
*something we continually work on—*
*but it is not the goal.*
*Making something great is the goal.*

— Catmull and Wallace (2014)

---

## ACKNOWLEDGEMENTS

---

I would like to thank my academic supervisor, Zhidong Xiao, who apart from guiding me and supporting me in my research, was always there to talk to when I needed it. Thank you for sharing your knowledge, time and experience with me. Many thanks also go to my industrial supervisor, Neil Marsden, who helped me shape a thorough understanding of the animation industry as well as the wonderful Soho community.

A huge thank you to Bournemouth University and the Centre for Digital Entertainment, Hibbert Ralph Animation, NVIDIA and the EPSRC for supporting this research and funding me over the years (with grant EP/G037736/1). However, it is not only the funding and the opportunity I was given I am grateful for: the CDE has managed to build a wonderful family of people and I hope that moving forward it will continue to be a family. I met some incredible people and made many friends through the programme. I would like to thank Daniel Cox for sorting out almost everything I needed, from training to tickets, as well as being there to ensure I'm doing well both professionally and personally. I am also grateful to Mike Board who picked up during my final years and helped me push through the difficult time of writing up.

I would like to thank Hibbert Ralph Animation for allowing me to be part of the team and giving me real insider knowledge of the industry. I am honoured to have worked with Jerry Hibbert, who taught me as well as gave me creative freedom to pursue projects on my own. I am grateful to Keith Pang, my 'neighbour' sitting on the computer next to me, for many insightful conversations, both work-related and not, who amongst other things helped me explore new coffee and lunch spots around London.

Of course, I want to give special thanks to my family and friends, who have supported me, listened to my rants (excitement followed by disappointment and followed by excitement again) and without whom these years would have been very difficult.
Finally, I want to thank my beloved partner Katerina, who was there supporting me in chasing my dreams, despite having her own PhD thesis to tackle. I thank you and I love you.

*When faced with a challenge, get smarter.*

— Ed Catmull

Dedicated to my niece Zoe, born 24th of May 2017.

## AUTHOR'S DECLARATION

I hereby declare that this thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

This thesis contains work that has been published in the form of posters, papers and a book chapter as well as presentations as listed.

*Bournemouth, UK, September 2017*

Alexandros Gouvatsos

## PAPERS

Gouvatsos, A., Z. Xiao, N. Marsden, and J. J. Zhang (2017a). "Posing 3D Models from Drawings." In: *Comput. Entertain.* 15.2, 2:1–2:14. DOI: 10.1145/2729984.

Gouvatsos, A., M. Zezulkova, and Z. Xiao (2017b). "[Ommitted due to double-blind peer-review process.]" In: *Convergence*. In review.

Gouvatsos, A. and Z. Xiao (2015). "Sketch-Based Posing for 3D Animation." In: *Encyclopedia of Computer Graphics and Games*. Ed. by N. Lee. Cham: Springer International Publishing, pp. 1–10. DOI: 10.1007/978-3-319-08234-9_47-1.

Gouvatsos, A., Z. Xiao, N. Marsden, and J. J. Zhang (2014). "Automatic 3D Posing from 2D Hand-Drawn Sketches." In: *Pacific Graphics Short Papers*. Ed. by J. Keyser, Y. J. Kim, and P. Wonka. The Eurographics Association. DOI: 10.2312/pgs.20141264.

## POSTERS

Gouvatsos, A., Z. Xiao, K. Pang, N. Marsden, D. Van der Ark, J. Hibbert, and J. J. Zhang (2016). "Efficient Storyboarding in 3D Game Engines." In: *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation: Posters*. SCA '16. Zurich, Switzerland: Eurographics Association, 1:1–1:1.

Gouvatsos, A., Z. Xiao, N. Marsden, and J. J. Zhang (2014). "Automatic 3D Posing from 2D Hand-Drawn Sketches." In: *Pacific Graphics Posters*. The Eurographics Association, 1:1–1:1.

## PRESENTATIONS

Gouvatsos, A., N. Marsden, and J. Hibbert (2014). "Optimising TV series preproduction: Redboard." In: *Annecy International Animated Film Festival 2014 Conference Summaries*. Annecy, France.

Hibbert, J., G. Andrews, and A. Gouvatsos (2014). "Redboard: Pre-Vis Software and Creativity - a dialogue." In: *Creative Skillset*. London, UK.

# Part I

# RESEARCH BACKGROUND

# 1

# INTRODUCTION

## 1.1 HIBBERT RALPH ANIMATION

Hibbert Ralph Animation (HRA) is an award-winning animation production company based in Soho. It is one of the UK's top animation studios producing television and cinema commercials, idents, logos, special effects, website animations and promos for over 35 years.

In the last few years it's been mostly focusing on television commercials as well as television animated series. An important part of the business is the development of a software package called Redboard[1] which is a specialised pre-production management tool revolving around the concept of 3D storyboarding. The company has undertaken a Technology Strategy Board (TSB) funded research project part of which is reported in this thesis.

Working together with the company's founder, Jerry Hibbert, as well as Neil Marsden and Keith Pang, the undertaken research had a strong industrial focus. Work within the company involved research and development, web design, pipeline scripts and tools, organising version control and meeting with clients, to name a few responsibilities. The research was undertaken while based at HRA, in London, from the beginning. Even though the research project took place entirely at HRA and not at the university, the company ensured the necessary space was provided to carry out research in several directions and ensure academic standards were kept. At the same time, this also provided the opportunity to get an insider view of the animation industry and what is truly relevant in a real production environment.

As such, the methodology is a mixture of theoretical work drawn from action research combined with technical research and development.

1 http://redboard.tv/

## 1.2 RESEARCH FOCUS

The main topic of this thesis is the use of media within the animation pipeline (further discussed in Chapter 2 and Chapter 3), with focus in pre-production. Variables of productions change as studios or individuals from different parts of the globe need to collaborate. Moreover, scope and budgets vary. However, since the pre-production phase has not evolved to facilitate these changes, the transition between the pre-production and the production phase can be inefficient. This thesis proposes ways to use digital media and processes in order to reduce costs and increase efficiency of this transition.

Focusing on 3D animation only, 3D storyboarding is proposed as the centrepiece of this transition. Apart from reducing storyboard drawing time due to some elements already being there (e.g. background) and offering a structured approach, 3D storyboarding provides a crucial link between pre-production and production. Allowing artists to see 3D assets in real scale while they are — otherwise as normal — drawing their storyboards, can help mitigate mistakes and therefore reduce costs as well as the stress of finding these mistakes (often not budgeted). Moreover, once 3D storyboarding is in place it can open up potential for various technical improvements.

The overall research is generally focused on pipeline production development, with an emphasis on the pre-production phase. More specifically, the research presented in this thesis concentrates on linking and connecting individual aspects of a pipeline, such as screenplay writing, traditional storyboarding, 3D storyboarding, pre-visualisation and disposable 3D model creation for early prototyping. Moreover, it is not only interesting to investigate how these independent components can be organised in the most optimal way, but also how their combination creates the potential for interesting new questions and solutions.

More specifically and closely related to the work of HRA, the aims of this research fall into the following categories:

- Investigation of the inclusion of 3D storyboarding within already established pipelines
- Investigation of novel pipelines using 3D storyboarding
- Mitigating the disadvantages of 3D storyboarding and/or pipelines which include it

- Creating advantages for 3D storyboarding and/or pipelines which include it

Furthermore, through the pursuing of the above aims, the theory and solutions which are brought to the surface should also be applicable in other context outside of 3D storyboarding and even in other fields outside of animation.

All aspects are really part of the main objective: making 3D animation pipelines more efficient. It is assumed that 3D storyboarding is more efficient for 3D animation productions, partly due to the industrial focus of this research (Hibbert Ralph Animation 2014) and partly because companies have used it for real productions, like 'Fireman Sam' (HIT Entertainment 2012). However, its shortcomings and advantages are analysed in order to build methods around it. Each method has its own hypothesis and evaluation.

## 1.3 IMPACT

Apart from the published works contained in this thesis, the Eng.D. included coursework, open source project contributions, helping young children consider a career in Science, Technology, Engineering or Mathematics as a STEM ambassador, creating and carrying out a creative programming course for the Camden Collective and a Virtual Reality (VR) course for the Decoded company.

Moreover, the research presented in this thesis helped HRA successfully complete a project funded by the Technology Strategy Board.

## 1.4 RESEARCH CONTRIBUTIONS

The main contributions introduced in this thesis are:

- A novel pipeline for 3D animation flowing from screenplay to second pass layout without the need for expert 3D software operators
- The term *resumer* to describe how professional producers are also consumers as media flows down the 3D animation pipeline
- A way to generate low resolution placeholder 3D assets to represent characters suitable for 3D storyboarding

- Exploiting useful information that already exists in screenplays to automatically structure sequences, scenes and shots for 3D animation
- A way for non-expert users to directly pose 3D characters in complex software like Autodesk Maya, using the Microsoft Kinect
- An analysis of methods for comparing pose embedded in 2D images, like storyboard drawings
- A new algorithm for posing 3D assets using 2D drawings, that can have applications in other optimisation problems

## 1.5 THESIS STRUCTURE

This manuscript is a thesis, part of a greater work carried out towards the completion of a Doctor of Engineering (Eng.D.) in Digital Media degree.

The manuscript is structured in three parts, 'Research background', 'Projects portfolio' and 'Concluding remarks'.

The first part consists of three chapters. The first chapter (this one) includes information about the Eng.D., the company and the impact of the work presented in this thesis. The second chapter aims to introduce animation and Three Dimensional (3D) animation in the context of media as a whole. Looking at animation as a medium and examining the whole industry and process from that lense, provides reason for looking at pipelines more broadly. Moreover, the pipeline itself is explained and described in order to provide context for the following chapters. The third chapter introduces 3D storyboarding in more depth and analyses its advantages as well as disadvantages that serve as a starting point for the research projects presented in the second part.

The second part consists of five chapters and is the bulk of this thesis. It details the motivation, background, methodology and evaluation of a series of research projects that were carried out while at HRA, all revolving around 3D animation pipeline improvements. Each chapter in this part has the same structure: a 'Motivation' section which explains the reasons for tackling that project, a 'Background' section which contains information such as related work in the field, a 'Methodology' section which contains the implementation details of each project, a 'Results' section which shows the evaluation

and finally a 'Discussion & future work' section that concludes each chapter with ideas for future work, possible improvements and constructive criticism.

The third and final part of this thesis, consists of two chapters. The first chapter puts together the key points from all the work and discusses the results as well as possible improvements. Future work, research avenues that were not followed and implications for the overall industry can also be found in this chapter. The second chapter summarises and concludes the thesis.

# 2

# DIGITAL MEDIA IN 3D ANIMATION PRE-PRODUCTION

In order to identify problematic areas and to make decisions on how to tackle them, it is important to analyse animation as a medium, as well as the different media used internally in animation. This in turn helps explore how information flows not only through pipeline processes, but also people.

A summary of the current state in mainstream animation pipelines when it comes to the pre-production and production phases, clarifies that the disparity between these two phases is problematic.

Furthermore, analysing how it all fits into the context of consumers as producers and producers as consumers provides insight as to what is important in an animation production, not only in terms of the outcome but also in terms of the process. Drawing from the idea of production by consumers or 'prosumers', the term *resumer* is introduced in order to discuss this relationship and to describe how professional producers are also consumers in meaningful ways.

Based on this insight, transition points are identified. These transition points can be targeted to aid in the pass from the pre-production phase to the production phase and motivate the work presented in the 'Projects Portfolio' part of this thesis.

## 2.1 AN ANALYSIS OF ANIMATION AS A MEDIUM

Animation features in many forms of entertainment, from traditional hand-drawn classics like 'Castle in the Sky' (Studio Ghibli 1986) all the way to blockbuster films with cutting-edge Visual Effects (VFX), like 'Bladerunner 2049' (Warner Bros. Pictures 2017). Many studios have adopted 3D computer animation to push the limits of what is possible to show on screen. Cartoon

characters mixed in together with real actors, impressive explosions and giant monsters destroying famous landmarks all become possible at a lower cost than in the past.

Currently, most methods of 3D animation require knowledge of complicated and expensive software. The field has become highly technical and modern animators need to work with more tools than they did in the past (e.g. 3D software and computers), as productions rely more on computer graphics and Computer Generated Imagery (CGI). This offers them a plethora of ways to bring characters to life.

In such CGI productions, there exists a pre-production phase which includes writing and editing the screenplay, blocking out storyboard panels and creating 3D pre-visualisation (pre-vis). Usually, the word *production* means the entire *creative process* or *pipeline*, including pre-production (Winder and Dowlatabadi 2001). In this thesis the term production also means the phase after pre-production which includes 3D asset generation, animation and compositing. The term *animation* also refers to any sequence, from commercials and television (TV) series to short films and feature films, as long as they include 2D or 3D animated elements. Finally, the term *producer* is defined as someone involved in creating, not the person in control of finances (e.g. executive producer).

The motivation to look at media within animation is drawn from its contemporary importance. Even by only taking live action films into account, over a quarter of film credits belong to the VFX department, based on data for Hollywood movies released between 1994 and 2013 (Follows 2014).

Although the production phase of a CGI sequence often includes state of the art technologies, the pre-production phase is much more basic, relying on more traditional methods for management and communication within the team, while improvements come as positive side-effects from overall pipeline overhauls (Selby et al. 2013). Additionally, almost everything created in the pre-production phase is used internally, without reaching the client or the final audience. Ultimately, the production goal is to make the transition from something as abstract as a screenplay, to something visual like a rendered 3D scene.

Unfortunately, it is not easy to envision a one-fits-all solution. Scope and budgets vary. Variables of productions change, as stakeholders collaborate from different parts of the globe. However, since the pre-production phase

has not evolved to facilitate these changes, the transition between the pre-production and the production phase is inefficient. Based on real production statistics (3D Clic 2014) provided by the Paris-based company 3D Clic during this research project, layout shots can have a retake ratio of almost 50%, meaning that half of the work needs to be done twice. Given that the layout is meant to recreate the already approved storyboards, why would so much extra work be necessary?

To tackle these issues, solutions need to be intuitive and viable for both large and small studios. Big companies can afford their own research and technical teams in addition to artists. This disparity gives them an advantage over the hundreds of smaller studios that have to be careful about each pound spent.

The emerging trend of a stronger relationship between consumer and producer is examined in this thesis in order to motivate future technical solutions. Consumers becoming producers (Ritzer et al. 2012) is one perspective of this emerging trend. In the next chapter of this thesis, this emerging trend is analysed from a different perspective: looking at professional animation producers as consumers.

The issues raised throughout this thesis are based on the experience of companies like London-based Hibbert Ralph Animation (HRA) and Blue Zoo Animation Studio (Blue Zoo), as well as international companies like Prime Focus Ltd. (Prime Focus) and Double Negative (DNeg). Additionally, based on the work of researchers from companies like Dreamworks Animation (Dreamworks) and Pixar Animation Studios (Pixar), a list of common issues emerges. In this thesis, an assumption is made, that more studios or individuals may face these issues.

While the field of animation is rich in academic work concerning its technical aspects, there is little academic research concerning the organisational structures of animation studios. Conferences and journals in the field tend to aim at big productions with big budgets, as these are the ones that usually push the bleeding edge of the industry. However, it would also be valuable to address the entire population of studios that work in 3D animation. There is little research when it comes to the internal workings of creative organisations in the context of the media industry (Malmelin and Villi 2017). This thesis aims to do that by combining an action research methodology with

media theory, computer animation theory, online resources and discussions with industry leaders.

## 2.2 ANIMATION AS MEDIA, ANIMATION IN MEDIA AND MEDIA IN ANIMATION

Animation is a powerful medium, able to tell stories that take place in worlds that defy our reality. It can entrance audiences of all ages and cultures and can range from a pure entertainment medium to communicating complex ideas. It has been used in the form of short films, full-length feature films, VFX, whether telling new stories or refashioning older ones. As animation evolves in this way, the refashioning of old media by new media coined as remediation (Bolter and Grusin 2000) becomes topical. As elaborated on below, there is remediation *through* animation, remediation *of* animation and remediation *in* animation.

An example of the first would be *Tales of the Tribes* (Douglas 2016a): an attempt to get young people in North East and Central India more involved in their heritage, by telling and remediating traditional stories *through* animation. The outcome of this type of remediation was not only a way of preserving culture, but also of helping bring contemporary relevance to it (Douglas 2016b).

An example of the second would be 'My Neighbor Totoro: A Novel' (Kubo 2013): a retelling and remediation *of* the Hayao Miyazaki animated film 'My Neighbor Totoro' (Studio Ghibli 1988). Using illustrations by the original film director but bringing them to a different form can be a way of introducing the story to a wider audience.

As the creative process becomes more technical, remediation *in* animation becomes worth analysing. Within companies, it is examined from an economic and artistic point of view. The people in charge of productions (e.g. directors), need to be convinced that new media provide enough benefits in order to risk adopting them.

According to Bolter and Grusin (2000), media are continually commenting on, reproducing, and replacing each other. Media constantly interact with other media and they need to justify their existence by remediating old media. In disciplines like journalism, new media have been resisted because they are seen as too different from the traditional ways (Usher 2010). In con-

trast, remediation in animation may be resisted because it looks too similar. An artist would have to understand the technical benefits of drawing in an app like Adobe Photoshop compared to using pencil and paper, for example. Because of this, convincing other companies to alter their workflow has been the main challenge for HRA, while working with them to optimise their TV series pre-production. Therefore, new media in the field of animation cannot justify itself by its common properties with the old media but by the new properties it can offer, to justify the risk and short-term costs of adopting it. Common properties can help adopters feel more comfortable with the new media.

Bolter and Grusin (2000) break remediation down into *transparent*, *translucent*, *hypermediated*, *aggressive* and *refashioning within a single medium*, with the last categorisation seen as the most respectful to the medium.

In an industry that relies heavily on budgets, it is worth examining this question in a budget-focused way. Moving from 2D animation to 3D animation, an artist would have to learn to operate 3D software. One may argue that this is natural evolution and whoever does not adapt should stay behind. However, training costs money and time. Added to that, there is the risk that something new is not going to work. However, there are more reasons to 'respect' the medium. Hand-drawn animation, as a process, has properties that would be beneficial in 3D animation too: drawing by hand is tactile and interactive as the lines drawn appear on the paper (or screen) immediately. As such, the result can also be judged and corrected immediately. In contrast, a hand-drawn storyboard's conversion accuracy can only be judged after its transformation to a 3D scene is finished. Bijker (1997) added the idea of a technological frame as a shared cognitive structure that defines a relevant social group. Looking at an animation pipeline as a technological frame, which defines the artists and other stakeholders, is useful as it provides the boundaries which serve as constraints between phases (Ashuri and Frenkel 2017). Looking at the boundaries may in turn may help decrease the number of errors that happen during phase transitions, for example from pre-production to production.

Before continuing, it is important to explain the meaning of new media in context. During the pre-production phase alone, there is a significant amount of media being used internally between artists, directors and other stakeholders. This thesis refers to this as *media in animation*. The aim of media in

animation is to communicate information internally, such as characters in a shot, but also ideas, such as the feel and look of a specific scene, or the emotion that must be communicated to the audience. Examples of media used for this reason during pre-production are the screenplay, storyboards and 3D layout.

## 2.3 PIPELINE ELEMENTS

CGI elements are used in film, series, games, adverts etc. and how a project's problems are tackled may depend on the size of the production, the complexity of the project, money, experience and methodology. It is the pipeline that defines the workflow, in order to outline how the goals will be achieved and to manage the complexity of the project.

Traditionally, pipeline flow for animation can be broken down into assets and processes. Assets are audio, artistic 3D elements (characters, sets), animations etc. The processes are the actions used to produce the assets (Patel 2009). Assets tend to flow from one process to another. For example, in the pre-production of a CGI feature film, the storyboard panels are usually assets that are created through the process of drawing by hand on a tablet.

Of course, based on project scope and company size, the pipeline may differ, but overall the aim is to remain flexible, avoid bottlenecks and mitigate costs (Patel 2009). Good planning and structure are important in reducing the frequency of experienced teams going back to previous pipeline stages for corrections. The traditional pipeline pre-production phase tends to remain less technical, with most media created being 2D, such as drawings and annotations (Figure 1).

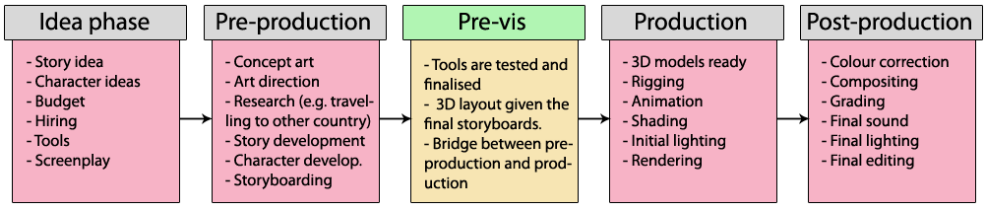



Figure 1: Example flow of a traditional pipeline with its distinct stages.

Pre-vis is a technique used to ensure that everything created and approved in pre-production is accurate, before the more expensive production phase

begins. It is a complex and expensive process and there are separate companies like London-based The Third Floor Inc. that focus exclusively on providing it. Smaller studios may not have the budget or time to do it.

Other researchers have recognised the value of pre-vis as a medium. It can mitigate the cost of correcting mistakes (Jhala et al. 2008), improve communication between different people (3D artists, animators, directors etc.) (Jung et al. 2010) as well as automate manual tasks (Mao et al. 2006). The level of detail of pre-vis depends on the specific requirements of each production. It is not only reserved for pre-production purposes, as it has been used in many creative ways, where feedback to the artist is important. For example, allowing pre-vis of the camera (Ichikari et al. 2006) can help the director determine the quality of shots, even if they contain a lot of CG content. A pre-vis camera system was used in Star Wars: Rogue One, which allowed Gareth Edwards, the film director, to show rather than say what kind of shot he was looking for (Anderton 2017). These mixed-reality techniques can be extended to stereoscopic filmmaking and to help actors visualise interactions with CG elements on-set (Mori et al. 2011).

It is worth noting that this research into on-set pre-vis solutions has benefited by the proliferation and accessibility of modern game engines like Unity3D and Unreal, allowing researchers to focus on developing frameworks (De Goussencourt et al. 2015) or combining motion capture solutions (Northam et al. 2011). Weta in the production of 'The Hobbit: The Battle of the Five Armies' (New Line Cinema 2014) used fire simulations as lightweight previews to ensure consistency of effects between sequences (Weta 2015).

By combining methods such as the ones proposed in this thesis (Part II: Projects Portfolio), pre-vis creation can become easier and cheaper. Specifically, by leveraging the power of 3D storyboarding, storyboarding *is* pre-vis. Once all the information from the screenplay and storyboards is brought together, pre-vis itself can be remediated to provide an information-rich transition from pre-production to production. For example, drawn storyboard panels can be displayed at the bottom right corner of the pre-vis video for quick reference (Figure 2). This process can help lower the costs by requiring less drawing by the storyboard artist, less time spent importing assets and fewer revisions once a shot has been approved. This could help pre-vis become more accessible to all types of production.

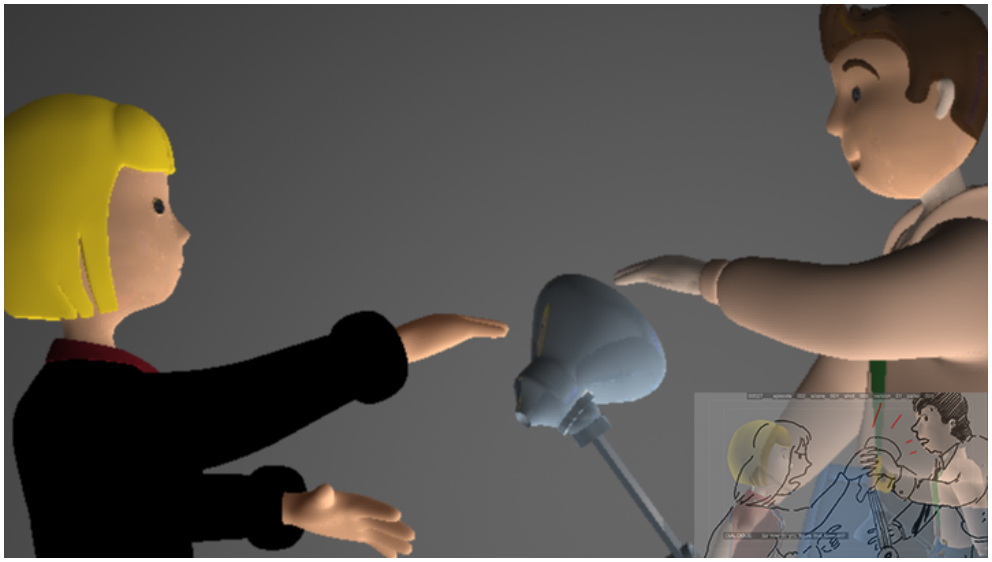


Figure 2: Example of pre-vis, with overlay of the corresponding storyboard panel at the bottom right.

Specifically, the pre-vis stage is vital in keeping costs down. Moving from pre-production to production is expensive as all the assets up until that point, that may be in textual or 2D form, need to be transformed to 3D. This includes set and character positioning as well as camera positioning and settings according to the story. If this transformation is done incorrectly, the storyboard artists, directors and 3D layout team need to get involved to solve the discrepancies. In some cases it might be as simple as slightly moving a 3D character. In other cases though, it might mean redrawing entire storyboard panels because the story being told does not match the real assets at hand. It may also lead to shot specific solutions which can still lead to problems later. For example, a drawn storyboard might dictate that a car needs to drive through a specific environment location, but in the real 3D environment there might be a building blocking the way. This happens as the 2D storyboard was drawn without knowledge of the real 3D assets.

Little research exists for using technical processes from other phases or even fields (e.g. computer vision) to improve the transition from pre-production to production through pre-vis (Hoshino and Hoshino 2001). There seems to be more of a trend for using artistic processes from pre-production to aid in other fields, like for example storyboarding for user requirements analysis (Gregor et al. 2002). Although companies recognise

1 **INSERT - TV COMMERCIAL - DAY** 1

Over jungle sound effects, the CAMERA is low, moving through brush from the POV of a stalking animal. As the brush parts, revealing Wall Street and the New York Stock Exchange, we HEAR the resonant voice of GENE HACKMAN.

**GENE HACKMAN (V.O.)**
The world of investing can be a jungle.

1A WE SEE a charging, snorting BULL. 1A

**GENE HACKMAN (V.O.)**
Bulls.

1B WE SEE a ferocious, growling BEAR. 1B

**GENE HACKMAN (V.O.)**
Bears. Danger at every turn.

Pretentious CLASSICAL MUSIC kicks in.

**GENE HACKMAN (V.O.)**
That's why we at Stratton Oakmont pride ourselves on being the best.

Figure 3: Example screenplay from the movie 'Wolf of Wall Street' (Winter and Belfort 2013).

the importance of pre-vis, it is still very much a manual process. If done correctly, it saves money and effort for the whole pipeline, by reducing the cost and as such risk of visualising an idea.

For other phases of the pipeline, such as lighting and rendering during production and compositing during post-production, there are tools like Nuke (The Foundry Visionmongers Ltd. 2014) that allow users to seamlessly move back and forth between the two. Since pre-vis is the centre of the pipeline and plays the important role of bridging pre-production and production, it becomes clear that this level of continuous and seamless integration is important at that stage too.

## 2.4 BREAKDOWN OF DIFFERENT MEDIA USED IN PRE-PRODUCTION

The screenplay (Figure 3) contains the overall story in textual form. It also contains descriptions of scenes, characters and the dialogue between them.

Figure 4: Example storyboard (Glebas 2009).

Since animation is a visual medium, it is necessary to transform the screenplay's text. Storyboards (Figure 4) attempt to visualise textual information from the screenplay as pictures. They include framing information (camera orientation and properties), visual representations of scenes, characters and overall visual mood (e.g. film noire). Moreover, the storyboards create a link with the screenplay, by including the necessary text together with each panel.

Finally, the 3D layout completes the transition from textual to visual form by constructing everything in a 3D set. This contains a 3D environment like buildings and roads, 3D characters like pedestrians and 3D models like cars. Since storyboards are hand-drawn and exist earlier in pre-production, they are less accurate in terms of framing, position and the scale of assets (e.g. height of a building). Framing discrepancies occur because even the best storyboard artist is not able to draw a 100% accurate representation of a camera lens, e.g. a 50 millimetre lens, even though that is what is used later in the production phase. Position discrepancies occur because without knowledge of the 3D assets, an artist might frame a character to have a certain distance from the camera, even though in the 3D set this might mean a wall blocking the shot. Scale discrepancies occur because an artist might draw something

to be of a certain size even though the modelled 3D asset is of a different size.

In standard pipelines for feature films and animated series, this is usually one of the last steps of the pre-production phase, because the populated 3D set contains everything that the animators, render artists and compositors need to create the final movie. Throughout this whole process there are short videos, concept art, music and other media that are used to communicate ideas and vision e.g. as seen in the behind the scenes of the DVD versions of 'Snow White and the Seven Dwarfs' (2009) or 'Star Wars' (2013). Voice-overs can be combined by an editor (a person who edits the panels and the audio together) with storyboard panels to form an animatic: a visual but unanimated story with sound. Once the animatic is approved by the director, it can be viewed as a 2D sequence telling the story from start to finish.

The pipeline used for 3D productions over the past few years by companies like Pixar[1] is explained in the following paragraphs in order to provide the context of the aforementioned *media in animation*. Variations of this pipeline exist depending on the production or studio.

At the earliest stage there is the screenplay. The screenplay undergoes multiple revisions by a team of writers while concept art, photographs and other reference material is used to build the story and world. However, it is not always easy organising this information. A problem that HRA encountered early on because of this, is that while a team is reading through the screenplay, keeping the story as the focus, another piece of media (e.g. a photograph of a location) is brought up and the focus shifts from the screenplay, which should be the focus, to that new piece of media. Furthermore, it is easy for these new pieces of media to lose their context if they do not maintain their link to the screenplay.

Once the screenplay is approved, storyboard artists start drawing 2D panels for each shot. It is important to note that for each panel, the artist has to draw the background (e.g. an urban landscape) as well as the foreground and characters (e.g. two androids on a balcony). In parallel, 3D artists model the necessary assets for the story, such as props and characters. Once the storyboards are approved by the director, 3D layout artists re-create the hand-drawn storyboards in 3D, using the assets modelled by the 3D artists.

1 Personal communication at SIGGRAPH 2015.

Before the 3D layout is passed on to the animators for the production phase to begin, it needs to be approved by the director to ensure that it still looks like the approved 2D storyboards. This is where unseen costs are created, as the director, storyboard artists and 3D layout team need to communicate back and forth to fix errors that happen during the transformation from storyboards to 3D.

One reason mistakes happen is that a storyboard artist draws an approximation of a shot. Even if the artist is incredibly capable, it is difficult to accurately draw a scene as it would look through a specific camera lens. It is also difficult to accurately draw the scale of objects in the scene without a 3D reference. Since storyboard artists work separately from 3D artists, this reference does not exist. In TV series that HRA has worked on, such as 'Fireman Sam', the director could identify errors in how the 3D shots look compared to the storyboard panels only once the 3D layout pass was complete, meaning several iterations of 3D layout passes were necessary to address all mistakes.

Some common mistakes include:

- 3D character being larger or smaller than the drawn character, changing the framing of the shot
- The shot looking different through a real camera lens compared to what the artist drew
- The shot being impossible to frame, e.g. because the camera has to go through a 3D wall to match the drawing
- Items like props that the artist drew do not have existing 3D assets to represent them

Furthermore, modern productions require multi-site collaboration, be it smaller studios, individuals or one large company with studios distributed across the world. Companies like Prime Focus and DNeg have studios in almost every continent. Therefore, the need for accurate communication of information and ideas becomes apparent. Re-imagining the media used during pre-production can help reduce costs, facilitate communication, reduce stress and optimise the creative process.

As Galloway (2013) states, the differences between human-centred approaches and user-centred approaches 'come down to whether researchers

privilege the technology or the people who use it' (p. 55). Instead, one can shift their thinking 'to the assessment of how particular practices create specific relations amongst all the players — human and non-human' (Galloway 2013, p. 56). Understanding the different media used internally in animation pipelines, the next step is to look at how and why it is consumed. In fact, media creation is becoming more interactive with a wider range of stakeholders (Deuze 2007).

In the words of Umberto Eco, 'it is not true that works are created by their authors. Works are created by works, texts are created by texts, all together they speak to each other independently of the intentions of their authors' (Haberer 2007, p. 57). In other words, works are influenced by the environment of their creator, including the professional environment. Thus, understanding animation industry workers both as consumers and producers, as discussed in the following section, can offer guidance on how to change the existing processes, as 'the management of media work is increasingly about the management of co-creation and partnerships' (Malmelin and Villi 2017, p. 183).

## 2.5 ANIMATION PRODUCERS AS CONSUMERS OF MEDIA

A trend identified as emergent 37 years ago, is that of participatory structures and producer-consumers or 'prosumers' (Toffler 1980). The rise of the prosumer has arguably been confirmed, as humans are increasingly involved as both producers and consumers of media, with consumers writing content for blogs and news and publishing their own music and photographs (Deuze 2011).

Inexpensive digital production tools, built-in best practices and accessible online publishing have contributed to the emergence of the prosumer. However, while accessible online publishing is possible in animation via platforms such as Vimeo and Youtube, production tools are still expensive for non-professionals. The software package Autodesk Maya cost GBP 3,200 in July 2014. This software package is not all-inclusive and a prosumer wanting to create 3D animation would need more software packages, quickly ramping up the cost. Having said that, companies like Autodesk (Autodesk 2018) and Adobe (Adobe Systems 2017) have recognised the prosumer market and attempt to win it over with flexible subscription plans. Similarly, in May 2014

Pixar announced that their rendering software RenderMan would be free for non-commercial use and would cost US$495 for commercial use, bringing it down from a flat rate of US$5000 per license. This shows that Pixar has also identified the emergence of the trend Toffler (1980) first discussed and finds value in winning over the prosumer.

Furthermore, Bruns (2007) analyses situations where the very consumption or usage of a medium is also productive: for example, the app Instagram provides content that is exclusively user-generated. This production through consumption, or produsage, is also characteristic in massively multiplayer games where by existing in the virtual world, the user is essentially also producing — in this case, an experience for the players around them.

Recognising the value that audiences generate gave rise to a subject of interest to both academics and industry workers: the subject of co-creation (Prahalad and Ramaswamy 2000). Co-creation has the potential to 'become an eminent future practice in the media industry' (Malmelin and Villi 2017, p. 183), as understanding the dynamics between organisations and audiences can improve collaboration. If industry workers are seen as the audience for internally used media, consequently the same theoretical framework could be used to understand their dynamics.

This thesis proposes another emergent trend which goes hand-in-hand with that of the prosumer: the professional consumer-producer. Just like in produsage, consumption necessitates production, in the field of animation, production necessitates consumption. As outlined in Section 2.4, various media are produced to guide animation production. *Media in animation* is purely for internal consumption. The fact that media is so ubiquitous in the animation process is what makes its identity as media invisible (Bolter and Grusin 2000). Nonetheless, a producer in animation is also a consumer in at least three ways.

Firstly, producers in animations are also consumers by consuming media for inspiration, research reasons or for pure entertainment. Secondly, they are consumers by making use of media, like storyboards, within the creative process and continuing their work based on their interpretation of that media. After all, the meaning of media is co-created rather than only transmitted. For example, a storyboard artist draws a panel based on their interpretation of the screenplay. Thirdly, an artist is always a consumer of their own work, as they engage in a creative feedback loop with themselves.

For example, an animator adds motion to a 3D character, plays back the animation and judges the result, iteratively. This dialogue with oneself can become more efficient by introducing interactive technologies (Cowley 2016). In an attempt to address this third way in which a producer is a consumer, Pixar proposed RIS (Pixar Animation Studios 2015) and Autodesk proposed IPR (Autodesk 2016) in order to facilitate the internal dialogue of artists through interactivity. Nonetheless, these interactive solutions are related to the production phase, leaving pre-production less evolved, as argued in the introduction section. From this, three main trends can be extracted which, combined, support users in picking up a new technology and using it to improve their work: accessibility, iterative workflow and real-time interactivity.

This use of real-time interactivity shows 'cooperation between multiple media industries' (Jenkins 2006, p. 2), in this case the video game and the animation/film industries. The idea of using game engines and assets to create cinematic stories, also called Machinima, is one way this cooperation manifests itself. Another example is the 'demoscene', where computer graphics enthusiasts create videos that are rendered and played back in real-time, with Pouet (Pouet 2000) being one of the more active communities. Through this observed phenomenon of convergence (Jenkins 2006), these ideas and methodologies are becoming important in fields like VFX for film, despite the fact that initially they were developed for the medium of video games. As computers become more powerful, the opposite becomes apparent too: more accurate and physically-based methods that were traditionally used for film are now available to use in video games. Following the discussion earlier on in this thesis about the impact game engine accessibility has had on pre-vis research, it is worth looking at examples where entire productions were developed using real-time tools. For example, the short 'A Boy & his Kite' was produced in Unreal (Unreal Engine 2015) and the TV series 'Mr Carton' was produced in Unity3D (Muller 2017).

One thing in common between those three ways of consuming that a producer can engage in, is that the medium consumed is being interpreted and then incorporated in the producer's work. As such, the media consumed by a producer is interpreted and then continued or *resumed* into the producer's own media. Therefore, this thesis proposes the term *resumer* to describe the professional consumer-producer, due to the continuity of the process. For

example, a 3D animator *consumes* a storyboard artist's drawings of the story, then *resumes* that in the form of 3D animation.

Companies from the field of video games show a preference for producers who are also consumers. In other words, not just people who have experience in making games but people who have experience in playing games, two very different things. This becomes evident after reviewing job descriptions from the gaming industry. The Creative Assembly (2017) lists 'broad gaming experience' as a bonus for a programmer candidate, while Sony Interactive Entertainment (2017) explicitly lists 'enthusiasm for playing games' under the desirable skills of an engineer. Blizzard Entertainment (2017) goes a step further and requests the candidates list games they are currently playing as well as their experience playing a specific game, to be included together with their cover letter. Evidently then, resumers are considered valuable. This is not trivial but it is intuitive; if a producer of media is supposed to understand their audience, becoming the audience of that media is a way of empathising. Moreover, consuming media can help a producer stay up-to-date — the equivalent of an academic literature review. Based on the action research undertaken with HRA for this thesis, job descriptions in the animation industry do not include the need for a producer to be a consumer yet.

Even if the animation industry does not consider the benefits of a resumer the same way the gaming industry does, it does not lower the value of looking at producers as consumers. The consuming of *media in animation* means the resumer must constantly shift between their role as a producer and their role as a consumer. As such, producers involved in the animation creative process are in essence resumers, by default. For example, 3D layout artists first consume the storyboards and then resume them from 2D drawings into a 3D environment.

## 2.6 REMEDIATION OF THE PRE-PRODUCTION PHASE

Having formulated the term resumer, identified why producers in animation are resumers and located problematic areas of 3D pre-production, it is important to explore possible solutions. By looking at producers in animation as resumers, it is possible to improve media in animation by understanding this internal audience.

It is through software that pre-vis can be possible. With it comes the potential for better data collection and by extension better understanding of the process, the roles of people involved and the future steps for improvement.

This thesis proposes focusing on 3D pre-vis as the transition point between the pre-production and the production phase. Three transition points are identified, which can be linked through the use of digital media.

The first transition point is from world creation and screenplay writing to storyboarding, which is a transformation from text to a visual medium. The second transition point is from 2D storyboards to 3D layout. The third transition point is from pre-production to the production phase (i.e. unposed and posed 3D layout). It is important that once production begins, there will not be any need to go back and correct mistakes.

The Foundry Visionmongers Ltd. (2017) also recognises the need to break down barriers between the different phases of production through consistent version control with Flix. Flix is a collaboration tool for story creation that manages versioning in a shot-based workflow, in order to facilitate people working from different parts of the world. While it focuses on aggregating the moving pieces into one app without altering the way those individual pieces are created (e.g. frames of a sequence), a 3D storyboarding app would focus on actually creating the pieces.

When introducing new media, one should consider how the consumer interacts with it. Consequently, the question of immediacy versus hypermediacy is raised. As explained by Bolter and Grusin (2000), immediacy is when '[...] the user is no longer aware of confronting a medium, but instead stands in an immediate relationship to the contents of that medium' (p. 24), while 'the logic of hypermediacy acknowledges multiple acts of representation and makes them visible' (p. 34). The choice between the two can then be determined by how familiar the remediated medium will be.

Professional tools do require immersion, and therefore hypermediacy is preferred to ease the transition from the old media to the new. Ibrus and Scolari (2012) provide a summary on the space of media and suggest that translating from one medium to another (or from one sign system to another) is impossible without altering the meaning. In animation this is seen as the creative input of each person or team across the pipeline. This theory is consistent with the experience of HRA and Blue Zoo when transitioning from screenplays to storyboards and then to 3D layout. This transition means

work flows from writers, who create the screenplay, to storyboard artists whose drawings create a visual story, to 3D layout artists who transfer the drawings into a 3D set, within the constraints of the built assets (created by 3D modellers). Therefore, it is important to provide a framework which can help identify alterations early, through tighter feedback and better communication between all steps of the pipeline. Whether the alterations are ultimately kept is a different matter.

In the next chapter, 3D storyboarding and HRA's pipeline are discussed further.

# 3

# STORYBOARDING IN 3D

## 3.1 WHY STORYBOARD IN 3D

Getting an accurate match between initial storyboards and the 3D layout is difficult, especially when they are performed by different people, in different locations, given today's reality of multi-site productions. While big studios have the luxury of re-iterating over the 3D layout of a shot, smaller studios with smaller budgets have limited attempts to get it right. Going back to make corrections may be the difference between delivering on time or going over budget. Studios can avoid mistakes and deliver on time by storyboarding directly in 3D, with the help of game engines.

HRA developed 3D storyboarding software to use for their own shows. As the studio (based in the UK) collaborated with layout artists in China, they realised there was a very time consuming problem: the layout almost never matched the storyboards, unless multiple iterations were carried out. Since the two studios were not even on the same continent, iterations took a long time and the whole process was expensive. HRA built Redboard, a 3D storyboarding tool, to address this issue (Figure 5). It started off as a simple game engine environment with props that allowed artists to draw on.

As Redboard evolved internally over the years, HRA reached a stage where they started licensing it for other studios to use. This software is the initial motivation around which the research presented in this thesis revolves.

Redboard provides storyboard artists access to low resolution versions of the final 3D environment for them to stage their shots. The artists can explore the location or set in real time, since Redboard runs interactively. As such, the process is similar to framing a live action shot. The storyboard artists can also add 3D characters, vehicles or props to the shot. Once the camera

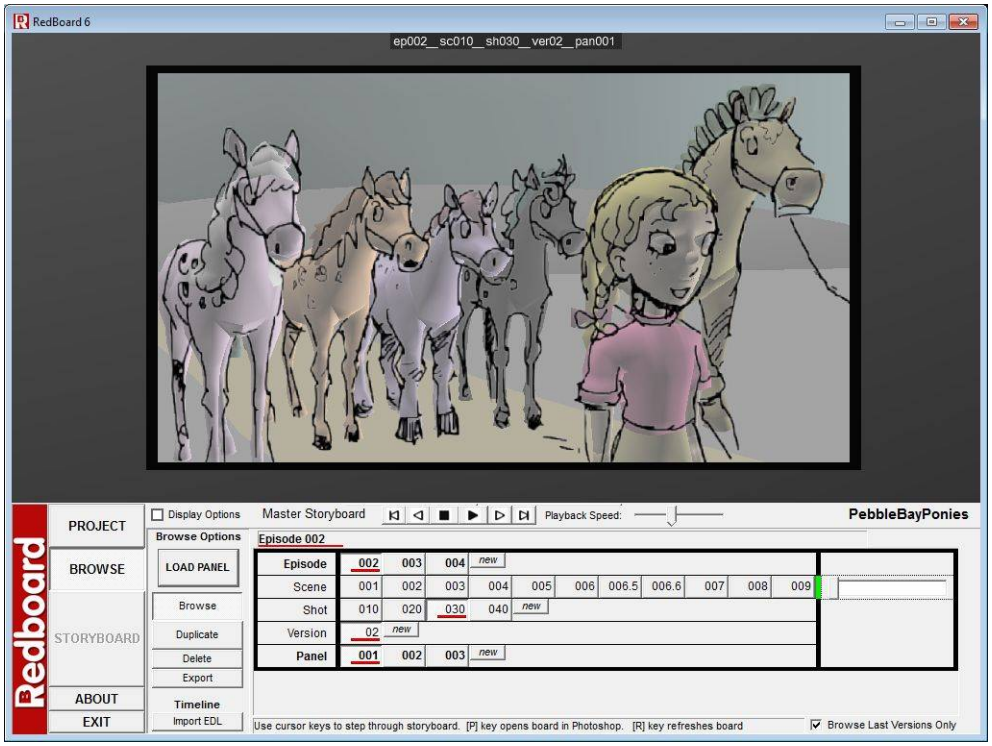



Figure 5: Screenshot of Redboard.

framing for the shot is chosen, expressions and poses are drawn over the top of the 3D background to convey the action. This is done like traditional storyboard drawing, but the artists do not need to draw aspects or details provided by the underlying 3D assets.

## 3.2 3D STORYBOARDING IN THEORY

Currently, there is great disparity between the skills an artist is required to have for 2D and 3D productions, with each school of thought having its own advantages and disadvantages. For example, 2D drawings are intuitive because they have been the primary method for decades and can be used for quick prototyping. In contrast, 3D requires knowledge of new tools and non-intuitive mechanisms like Inverse Kinematics (IK) (Zhao and Badler 1994) and takes longer to set up. However, it can greatly reduce costs in the long term, as changing cameras and lights or adding simulations can be automated.

Other researchers have noted this disparity and made attempts to bridge the gap between the two worlds, be it for 3D modelling (Igarashi et al. 1999), 3D layout (Eitz et al. 2012), body (Jain et al. 2009) or facial animation (Sucontphunt et al. 2008) and even simulations of interacting objects (Jain et al. 2012). However, these attempts only look at specific problem spaces rather than the overall pipeline.

The standard pipeline requires a manual transition from storyboards to 3D layout, which can be problematic: inconsistent scale between the drawings and the real 3D assets, drawing something that does not exist as an asset, or obstructing the camera view with real set elements. Using game engines, storyboard artists can access 3D assets in an intuitive way, controlling what appears on the screen. For tasks requiring quick feedback (e.g. animation) interactive solutions are preferred (Lin et al. 2015). Game engines offer real-time response with full production quality (Moran 2015), making excellent back-end technology for pre-vis and as such storyboarding in 3D (Nitsche 2008).

In traditional 2D animation, storyboard artists were also what is today known as layout artists, as what they drew is what appeared on screen. However, as 3D animation became prominent, 3D software operators were employed to translate storyboard drawings into 3D scenes. This is still today's common practice.

Performing the storyboarding process in 3D can help resolve the issue of transforming 2D storyboards into 3D layout, by providing interactive pre-vis of the storyboard (Gouvatsos et al. 2014c). It can bring the storyboard artist closer to the traditional role, by using a more modern approach. Moreover, it reduces the need for large layout teams, bringing costs down significantly, as the teams might only need to carry out a clean-up iteration.

## 3.3 3D STORYBOARDING IN PRACTICE

This thesis proposes that the storyboarding application should provide storyboard artists with simple controls, so that they can use it without knowledge of more complex 3D packages like Autodesk Maya. This application should allow artists to move the camera in 3D space and place characters in a set. Everything should run in real-time. The reason for this is to allow for maximum interactivity, and thus quicker iterations. However, it should

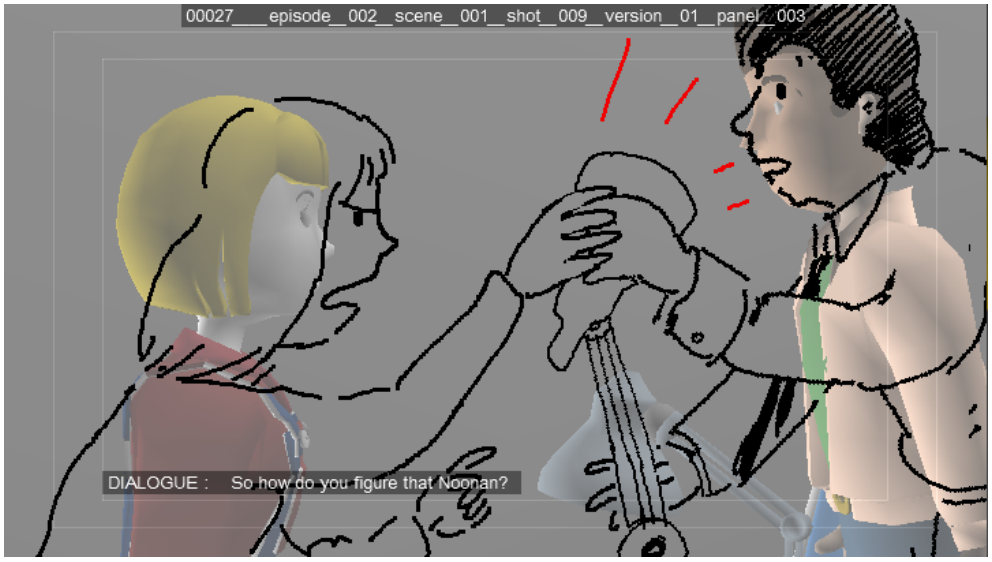



Figure 6: Example of 3D storyboarding, with 2D drawing on top of a 3D environment. The 3D models are unposed.

also ensure that the core of storyboarding is there, by allowing artists to draw on top of the 3D scene (Figure 6). 3D elements can provide accurate information about the scale, camera position and lens. The overlaid drawings add information such as character pose, facial expressions and other story-related details.

The main advantage of such a storyboarding application is that it is adapted to 3D from the beginning. As such, when the director approves the storyboard, they would also be approving the 3D layout. Additionally, storyboard artists no longer have to draw all elements in a scene, like background and props, which saves time. Furthermore, it creates an inherent connection between the 2D drawings and the 3D assets. Jerry Hibbert from HRA has a good example of the power of pre-vis in pre-production, even for writers: in an episodic production where all the assets were pre-determined, the writers wanted two key characters to walk to the school principal's office undetected. By being able to easily navigate the real 3D set, the writers found a plausible route for the characters to take, as well as possible emergent stories that would make sense[1].

This connection can be exploited to introduce methods that are otherwise non-trivial to include in the pipeline (Gouvatsos and Xiao 2015). For exam-

1 Based on personal communication with Jerry Hibbert.

ple, it allows for a system that automatically poses 3D assets from storyboard drawings (Gouvatsos et al. 2017) and provides intelligent suggestions to the artists, e.g. if they are deviating from cinematic composition guidelines (McDermott et al. 2002), or if extreme camera movements or lenses are used (Sullivan et al. 2008). This means that 3D layout and even 3D posed layout (similar to an animatic, but 3D) can emerge from this process without additional effort. Combined with audio and interpolated motion between key-frames, this process would create production-standard pre-vis at a lower cost.

Game engines require low polygon assets. A model can either be optimised from a pre-existing high polygon asset, or an early version can be built, as an iteration of the design process, once the shape and size have been approved. It is important to note that the 3D assets used in the app do not need to be the final 3D assets used in the production.

A disadvantage of 3D storyboarding is that it is not as suitable for quick prototyping as traditional storyboarding is, because it requires initial imports of 3D sets, characters and props. To mitigate this disadvantage, one can leverage the link between the different steps offered by new media. Assuming a screenplay analysis system like the one proposed in this thesis (Chapter 5) is used, the annotations can be used to automatically populate the storyboarding app with scene and shot breakdowns, text information from the screenplay, such as descriptions or dialogues, and 3D assets. This integration between screenplays and storyboards can speed-up the initialisation process, allowing storyboard artists more time to be creative and less time performing tedious tasks like importing 3D assets. It is these observations that motivate the work presented in Chapter 5. To summarise (Gouvatsos et al. 2016):

> Leveraging 3D game engines for storyboarding, directors are empowered as they are able to control exact details such as camera lenses. Artists are empowered too, as they are confident that the final output will match their drawings. (para. 5)

### 3.3.1 *Storyboarding*

Storyboard artists can explore the 3D set with a camera to find the best framing for their shot. After they have chosen a camera framing, they can place characters into the shot. Once these panels are approved by a director, the artists begin drawing details on top. The drawings provide information such as character poses, facial expressions and effects like fire. The final panels are exported and timed into an animatic by an editor. Storyboarding in this way is more accurate because storyboard artists create panels using assets that are a close representation of the ones used in the production, with a camera that matches the one used in the final render. Knowing where the set ends means avoiding pointing the camera towards emptiness; knowing where set elements such as trees are means avoiding framing an obscured shot. It is also faster, as it is no longer necessary to draw backgrounds, characters or props that can be represented by the 3D assets (Figure 7a).

Storyboarding in this way is more accurate because 3D positions are exact, while the artist views their shot through the same camera used in the final render. It is even possible to automatically sketch the basic lines of any character or other object, given that the 3D model exists.

### 3.3.2 *Layout*

The 3D assets (in the game engine), the 2D drawings (drawn by the artists) and the timings (set by an editor) are the three elements needed to automatically create the first pass of 3D layout. Automatically exported to software like Autodesk Maya, camera information (cuts, pans and lenses) is accurate and the scale is correct. Creating 3D layout in this way means directors and artists can move on to the next scene or episode without looking back. Keyframes are created based on the timing data from the animatic and the low-poly 3D assets are automatically replaced by the high-poly final assets (Figure 7b).

### 3.3.3 *The HRA pipeline*

At HRA 3D storyboarding is done using Redboard. The pipeline does not treat 3D storyboarding as an afterthought but rather takes it into consideration from the beginning. A process as described earlier in this chapter is preferred as part of HRA's pipeline.

During the first years that this pipeline was tried, using Redboard had a large overhead as artists needed to be trained to use it. While the software was simple to use, getting used to a new pipeline and achieving maximum efficiency required time. Over the years, however, HRA built a network of trained Redboard storyboard artists that they used for their own productions, or for the productions they consulted on.

In episodic productions like TV series, which is what HRA has been working on the most, it is easy to get started as it is common to have existing 3D sets and characters from previous seasons and episodes. Having said that, 3D assets may exist that are too heavy for the requirements of the game engine environment of Redboard. In this case, a 3D artist simplifies them to reduce their polygon count. It is common to use a mix of traditional storyboard drawings and 3D storyboarding, all done within Redboard, depending on the availability of assets and the stage of pre-production.

While a lot of the story creation took place in the UK, the actual 3D layout and later on, animation, was usually outsourced to countries in very different timezones e.g. China. The common workflow started with camera positioning and framing done first, followed by character placement and finally drawn details. Assuming a ten minute episode, each episode's storyboard had three weeks allocated to it, with an approval point during the second week. Following that, there were three 3D layout approval points during a six week period. The final 3D layout was done in Autodesk Maya.

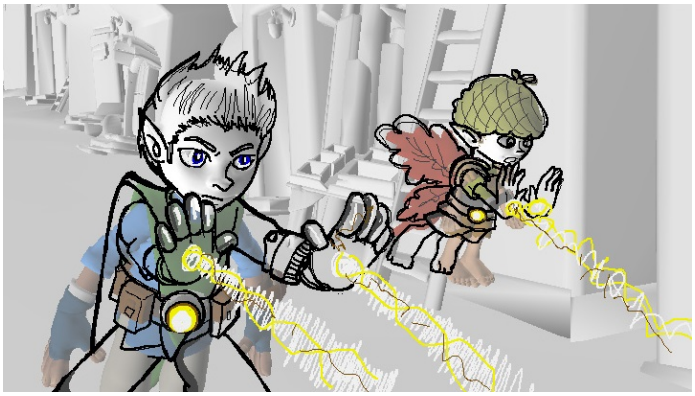
## 3.4 MOVING FORWARD

3D storyboarding is a way to modernise the otherwise still traditional preproduction to match today's animation pipelines. It can help accurately visualise the scenes early on, mitigate transition errors from storyboarding to layout as well as scale and position mistakes.

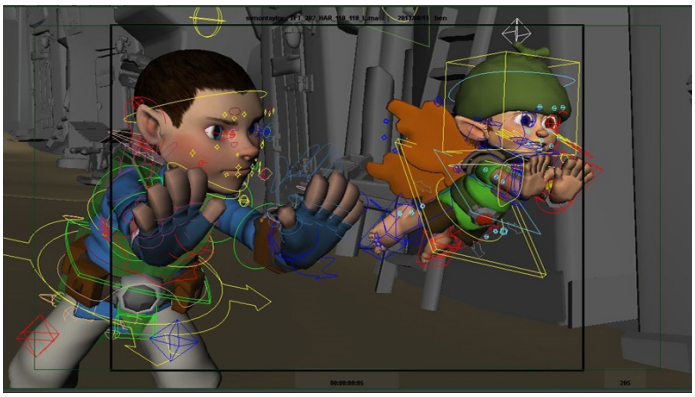
Thousands of minutes of animation have been created in this way, for productions such as 'Bob the Builder' (HIT Entertainment 2015) and 'Tree Fu Tom' (FremantleMedia, CBeebies, Blue-Zoo Productions 2012). HRA has used 3D storyboarding internally as well as by consulting and licensing their software to external studios, like HIT Entertainment.

Having said that, 3D storyboarding by itself also has some drawbacks. Working in a 3D environment this early requires 3D assets that might not exist yet. Moreover, storyboard artists that are used to working in a certain way, would have to deal with non-creative tasks like importing 3D sets and characters into the software. Apart from the limitations, there are also new opportunities: the relationship between 2D drawings and 3D assets can have interesting applications that are explored, like automatic posing of models using storyboard drawings (Figure 7c), knowing that the drawings are correctly laid on top of the 3D assets.

The aim of this chapter was to solidify what 3D storyboarding is and why companies like HRA prefer it. Other companies using this workflow following collaboration with HRA are Blue Zoo and HIT Entertainment. The second part of this thesis contains a portfolio of projects developed for the purpose of improving pipelines, specifically around 3D storyboarding.



(a) Storyboard panel with drawing on top of the assets, in a 3D game engine.



(b) 3D layout in Autodesk Maya. The low detail models are replaced by the high detail rigged models.

(c) The final render of the shot, faithful to the original storyboard.

Figure 7: Efficient storyboarding workflow example from Tree Fu Tom. (© BBC MMXVI)

Part II

# PROJECTS PORTFOLIO

In this second part of the thesis, motivated by the three transition points identified in the first 'Research background' part, five different projects and their implementations are detailed, focused around a 3D storyboarding pipeline.

As a way to mitigate the 3D storyboarding issue of requiring 3D assets earlier, the first project (Chapter 4) describes a system that can create simple 3D assets from character line-up drawings, to be used as reference during storyboarding.

As a way to make the process of 3D storyboarding more immediate by requiring less set up time by the artist, the second project (Chapter 5) describes a system that can structure a Redboard project file with a scene, shot and action breakdown automatically carried out based on a screenplay.

Furthermore, this system can use the simple 3D assets generated by the first system in order to populate the structured scenes with the correct character models automatically, as the third project describes (Chapter 6).

While 3D storyboarding aids in the transition from storyboarding to unposed 3D layout, the next step of going from unposed to posed layout is also explored.

Cheap motion-capture devices can facilitate in this transition, as the fourth project (Chapter 7) describes a system that uses Microsoft Kinect to allow for posing of characters that can be carried out by non-expert 3D software users.

Alternatively, the fifth project (Chapter 8) describes how this step from unposed to posed 3D layout can be performed automatically, by using 2D drawings (e.g. from storyboards) to infer the 3D pose.

The combination of these projects results in a pipeline that transitions from screenplay, to storyboarding, to unposed layout and finally to posed layout in a semi-automated way, modernising the pre-production phase in the process.

# 4

# AUTOMATIC CREATION OF PLACE-HOLDER ASSETS

## 4.1 MOTIVATION

A pipeline that does storyboarding in a 3D environment can have great benefits in terms of performing the storyboarding and the initial layout phases simultaneously, but also has some shortcomings.

The first shortcoming of this alternative pipeline is that 3D models are needed earlier than usual. It is important to note that this is mostly a problem in a situation where new models are needed (e.g. in the case of a completely new production) and it would not be as prevalent in episodic productions where the models have already been crafted (e.g. a follow-up TV series). Assuming that new models are indeed a prerequisite, it suggests that the modelling teams would have to increase their efforts: either to create the final models earlier during pre-production or to create temporary models which will then be replaced. The first option puts a greater emphasis on 3D models being available earlier in the pre-production, which may not be possible due to the cost and time required. The second option is viable because for the purposes of storyboarding the artists do not require a complete visual representation, since they draw on top, using the 3D models merely as a reference.

Traditionally, 3D models are created later in the pipeline while character drawings are created early on. This observation is leveraged in order to propose a system that takes drawn character line-ups as input and generates representations as simple rectangular cuboids. These models can be used as placeholder assets in 3D storyboarding, until the real 3D models are completed.

## 4.2 BACKGROUND

During storyboarding with Redboard an artist draws in 2D over the 3D scene. This allows for a more abstract representation of the characters in 3D. Because of this it is assumed that only position, orientation and scale are needed for the drawing process. Moreover, such an approach allows the artist to be unconstrained by the model and draw more freely (due to the abstraction of the character). More specifically, a scale-accurate abstract shape can help ensure that the artist does not make errors when it comes to positioning and size of the drawn character, while allowing them to be free to experiment with new expressions and poses.

Vaillancourt and Egli (2011) define placeholder assets as 'temporary resources used ... in place of final resources that haven't been created yet'. At the same time, it is important that placeholder assets correctly signal they are placeholders that will be replaced by the final assets, be it internally or when communicating with external clients (Zagal and Altizer 2015). A simple solution like applying a character drawing on a 3D plane can cause confusion in terms of whether the artist is dealing with a high or low resolution placeholder asset[1]. It is worth noting that placeholders are not the same as low level of detail (Luebke et al. 2002) assets: levels of detail are different versions of the final asset while placeholders are meant to be discarded the moment any version of the final asset exists (Zagal and Altizer 2015).

Inspiration for this approach is drawn from an advertising campaign (*Minimalist Lego Cartoon Characters* 2012). Lego has experimented with the idea of using simple rectangles with stripes of colour to represent complex characters in a recognisable way, during their Lego Imagine advertising campaign. The equivalent in 3D can be re-created using geometric primitives such as rectangular cuboids. As long as the scale is representative and can guide the storyboard artist's drawings, this method is effective in conveying a character with very little detail.

1 Based on personal communication with HRA artists.

## 4.3 METHODOLOGY

The proposed system takes character drawings in the form of character line-ups as input (Figure 8) and outputs 3D rectangular models (Figure 10). The generated models are in the OBJ file format. The general algorithm in pseudo-code is given in Code Snippets 1 and 2.

Figure 8: Example of input character drawings (© Hibbert Ralph Animation).

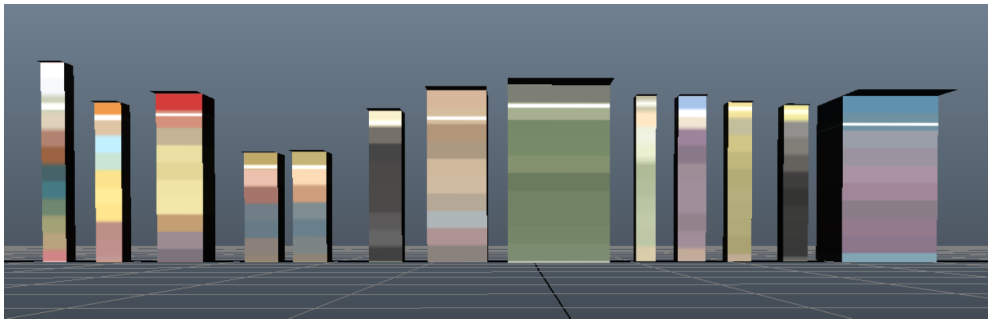


Figure 9: Automatically generated 3D pegs as seen in Autodesk Maya.

An object-oriented approach is taken, allowing for a modular and easy-to-maintain system. The Model-View-ViewModel (MVVM) architectural pattern is used in order to facilitate design and data presentation between the UI and the actual logic of the system, using data-binding techniques. It also helps uphold Don't Repeat Yourself (DRY). The algorithm is broken down and structured into several classes as seen in Table 1. The classes of the MVVM pattern, such as the MainViewModel, are not included.

Characters are segmented from the overall image using the Connected Components Labeling algorithm (CCL) (Samet and Tamminen 1988) algorithm. As this may result in multiple segmented parts e.g. if there is noise in the image, only the N largest parts returned by CCL are considered as characters, where N is the number of characters in the image, provided as user input.

| CLASS | DESCRIPTION |
|---|---|
| Character | The Character class is a custom data-structure for storing information of the segmented characters (e.g. character image). |
| Segmenter | The class which takes care of segmenting the initial input image into several Character objects. |
| Trimmer | This class is used for locating the eyes as well as performing the third width heuristic as shown later in this thesis. |
| Mosaic | The Mosaic class performs the colour-averaging which creates the character image abstraction. |
| ObjCreator | This class generates a 3D rectangular model of a character, using the mosaic image as a material. |

Table 1: Peg Creator software classes.

In order to maintain the eyeline of each character while avoiding performing some complex face recognition algorithm, the eye colour of each character is manually set to $\mathrm{RGB}(0, 255, 0)$. This colour is chosen because it is uncommon to appear in character drawings that HRA had access to. It could be any different colour, as long as it uniquelly used for this and does not appear in the drawn character colours.

The height of the resulting 3D rectangular cuboid depends on the maximum pixel height of the character in the drawing. Width can be calculated using various heuristics. The first width heuristic is to calculate the width of the resulting 3D rectangular cuboid as the maximum pixel width of the character in the drawing. This essentially considers the bounding box of each character in the drawing.

However, alternative heuristics were examined too. More specifically, the bounding box of a character which is generally thin can have a large maximum width if some aspect of the character extends far beyong the body e.g. a character with their arms extended (Figure 13). To deal with this problem, collecting the width of the character in each pixel row and calculating the average width can be used as a second heuristic. These pixel-based heuristics are based on the concept of using scan-lines at every row of the image and counting the coloured pixels, assuming that colours always belong to the character. Apart from these two pixel-based width heuristics, a third

heuristic was developed which is based on the assumption that the feet of the character are always at the bottom of the image.

For the third heuristic, each character image is segmented as to remove any limbs and keep only the core (Figure 12). Using a scan-line at a height of the image where the feet are assumed to be (near the bottom), the left-most and right-most points are found, called botX1 and botX2 respectively. Then, knowing the location of the eyes it is possible to similarly get the left-most and right-most eye points topX1 and topX2. The removal of the limbs is achieved by using a rectangle: the left edge is the min(botX1, topX1) and the right edge is max(botX2, topX2) (Figure 12).

Given the mosaic image which serves as an abstraction of the character, an OBJ file is created representing a rectangular cuboid. Its dimensions are directly proportional to the dimensions of the mosaic image in terms of height and to a value given by a width heuristic for the width. These values are normalized to be between -1 and 1. This allows for the center of rotation to be the middle of the model and absolute scaling to be decided later by the 3D artist depending on the scene dimensions, while relative scale in terms of the other characters in the line-up is maintained (Figure 9). All faces of the cuboid, apart the top and bottom, are textured with the input bitmap of the mosaic. The top face of the rectangle, which tends to be the most visible, has a protruding plane with an arrow texture, in order to add orientation to the character model.

The mosaic creation is left for artists to control, by increasing or decreasing the pixel thickness of stripes (Figure 14).

The system is developed in C#, using the Windows Presentation Foundation (WPF) for the UI, as it was aimed to be used on Windows machines. The system is developed to work independently and the UI aims to allow a non-expert user to use it comfortably (Figure 11). The function to binarise an image (Szeliski 2010) and CCL were initially implemented in C#. However, in the final implementation of the system the AForge.NET library (Kirillov 2011) is used. The reason for this is to decrease development time as other contained functions are also employed, while reducing the need for in-house maintenance.

## 4.4 RESULTS

Output examples of the PegCreator system can be seen in Figures 9 and 10. Examples of the output of each individual step can be seen in Figure 12. For all of these examples, the third width heuristic (which makes the assumption about feet positioning) was used, as it deals with most cases for humanoid characters. The mosaic stripe thickness for the presented results was set to 40 pixels which worked for the chosen character line-up.

## 4.5 DISCUSSION & FUTURE WORK

Using the Lego advertising campaign as inspiration and knowing from HRA's previous productions that very simple 3D models with the correct scale can be used as placeholders, a system was developed that takes a character line-up as input and generates simple rectangular cuboids. The cuboids have different dimensions depending on the character's size in the line-up, while a coloured texture is created by averaging the colours of the character.

It is worth noting that the artist does not have to draw the characters as a line-up. This input was chosen for two reasons. Firstly, character line-up drawings can exist early on in the pipeline. Secondly, a character line-up contains information about the scale of characters relative to each other. Individual character drawings could be used too, as long as they all use the same scale.

The limits of this approach are mostly related to the pixel-specific heuristics, be it for width estimation or mosaic stripe thickness. Depending on the types of character, the width heuristics need to be changed to improve results. Even so, 3D assets can be created that are too wide for the character they are meant to represent. It is worth discussing how the three presented heuristics can fail depending on the drawn character. Using the maximum width heuristic, depending on how the character is drawn, can result in a much wider bounding box e.g. depending on if the character has their arms extended or not (Figure 13). The method works with non-humanoid characters too, but the same issue could arise for other situations, like for example an animal that has a tail extending out of its body. In those cases, using the third width heuristic that makes an assumption about where the feet are

located can help. However, it would fail in other situations, like for example if a character has a cape that increases their width around the feet area (Figure 10).

The mosaic thickness can be adjusted depending on how much colour detail the artist wants in the final result. Depending on the actual pixel dimensions of the drawn characters, these values would also need to change as they represent mosaic stripe thickness in terms of pixels (Figure 14). As such, the proposed method could be improved if mosaic stripe thickness would be automatically picked, removing this part of user input.

This is a step in making 3D storyboarding more approachable for people that prefer a traditional pipeline, by removing the need for 3D artists in pre-production. Following evaluation by HRA's in-house artists, it was determined that even though further improvements could be gained by increasing the accuracy of the character representation, the generated placeholder assets would be useful early on in pre-production. However, storyboarders still have to spend time importing models into software, instead of quickly getting started on their creative process, as they would with a pencil-and-paper-like medium.

Figure 10: Examples of automatically generated character abstractions.

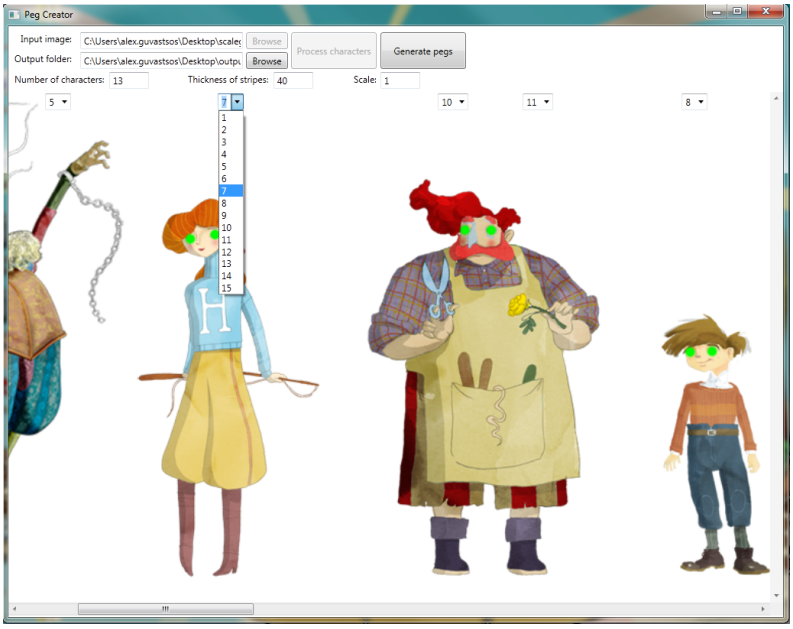



Figure 11: Screenshot of the Peg Creator interface.

Figure 12: Step 1 — Segmenting the character from the line-up. Step 2 — Trimming it based on the feet. Step 3 — Averaging the colours into stripes to create a 'mosaic' effect. Step 4 — Generating the rectangular cuboid and adding a plane on top to convey orientation.

Figure 13: The same character drawn differently can result in a different bounding box that may affect how the character's width is estimated.

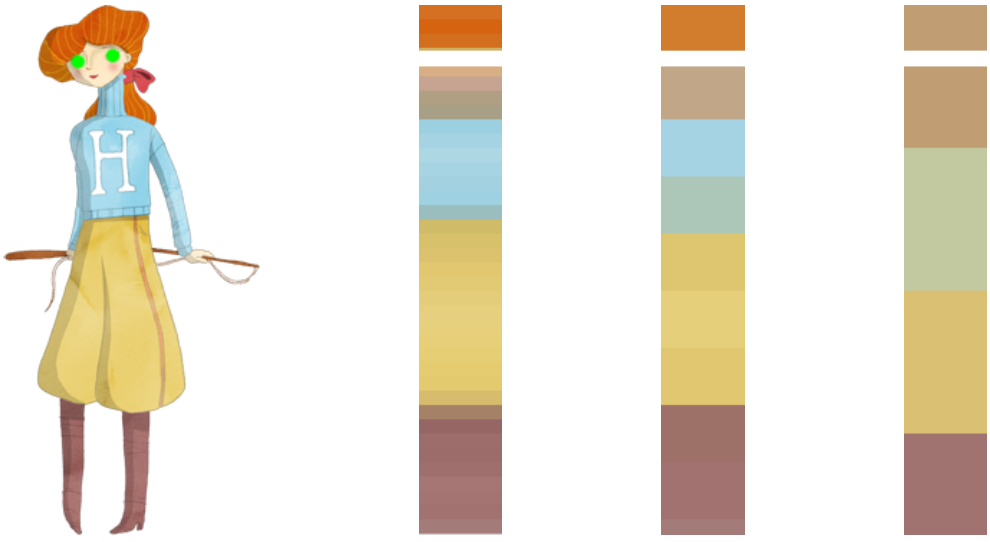


Figure 14: Side by side comparison of the original character drawing (left-most) and different mosaic stripe thickness parameters: 10, 40 and 100 respectively from left to right.

Code Snippet 1: Peg creation algorithm pseudo-code. Input: image with character drawings (image file), number of characters in input image (integer). Output: 3D peg model for each character (.OBJ file).

```
class ModelCreator
{
    public void CreateModel()
    {
        int N = numberOfCharacters;
        var binImage = Binarize(inputImage);

        // Perform CCL algorithm to segment characters
        var Characters[] =
            ConnectedComponentsLabelling(binImage);

        // Sort characters by area
        Characters.Sort(character => character.Area);

        // Take N characters with largest area
        Characters[] = Characters.Take(N);

        foreach (var character in Characters)
        {
            // Transform character image into
            // rectangular abstraction
            var mosaic = Mosaic(character);

            // Output OBJ file with a rectangle as high and
                wide as the character image
            // Use the abstraction from the mosaic as the
                texture
            Generate3DPeg(mosaic, character.Width);
        }
    }
}
```

Code Snippet 2: Pseudo-code to average colours of a character image in order to create mosaic effect.

```
1
        private image Mosaic(character)
        {
            // Find location of Chroma green eyes in the
               character image
            var eyeLocation = LocateEyes(character);
6
            // Average colours in the image per stripe
            // Stripe-size X=image width, Y=k
            // Where k is a fine-tuned constant
            // to create the wanted artistic result
11          // This works similarly to a sliding average function
            // The background colour is ignored, as well as
            // the Chroma green (eye) colour
            var averaged = ColorAveraging(Character, stripeSize);

16          // Having the eye location, overlay a white
            // stripe at the correct position
            image mosaic = OverlayEyes(averaged, eyeLocation);

            return mosaic;
21      }
```

# 5

# SCREENPLAY ANALYSIS SYSTEM FOR AUTOMATIC ASSET IMPORT

## 5.1 MOTIVATION

By using 3D storyboarding in a pipeline, storyboard artists are required to operate 3D software. When it comes to setting up their scenes to begin working, either the artists themselves or other human operators have to spend time importing the correct 3D sets, props and characters. Moreover, they have to structure their work into scenes and shots and manually copy over text from the screenplay in each shot in order to help show what they will be referring to. This is a tedious task that is not part of the creative process of storyboarding. Similarly to how character drawings like character line-ups exist early on in the pipeline, it is also a fact that the screenplay —the text describing the story— is also a vital part of pre-production even in the most traditional pipelines (Chapter 2). In fact, the screenplay is often the first asset that exists in a production.

In Redboard, HRA's 3D storyboarding software, all the information is saved in a binary format called a Redboard project file. This contains information of the scenes and shots, camera positioning, drawings and positioning of 3D assets.

In this chapter, the observation that the screenplay is rich in information is leveraged in order to propose a system that can break down the screenplay into sequences, scenes, shots and actions, identify which characters and which sets are linked with them and generate a Redboard project file with all the set up work completed automatically.

## 5.2 BACKGROUND

Ideally, a screenplay writing application for 3D animation purposes should be able to track multiple revisions, in order to handle multiple writers working together from different locations. The application should be available on mobile devices; the medium loses its power if the writer has to print it out on paper to bring it to a meeting. Writers should be able to attach audio, as well as images such as concept art or photographs next to the text. This can ensure that everything is stored together and is easily accessed by all people involved. It also means that even when looking at outside material such as photographs, the screenplay will not be put aside; it will remain in focus. Furthermore, it is important to provide a structured method for writers to annotate character names, scene locations, time of the day etc. Having a structured and labelled screenplay can help further down the pipeline. For example, quick character and location drag and drop functions, diagrams showing transitions between scenes as well as which characters appear in which location. Once annotated, dialogue can also be easier to manage as it can be reviewed as a whole, per character or per group of characters. This can be particularly useful when there is a special requirement for a character to appear in, e.g. at least 50% of the scenes. There are cases where characters appearing in a TV series are also sold as merchandise e.g. toys. In these situations, it is possible to ensure already from the storyboarding phase if the marketing goal of a character appearing in a specific number of shots has been met.

The current state of the art in screenplay writing is Final Draft (Figure 15) (Hillin et al. 2013). This is also the case for HRA and the companies they collaborate with. While it does not have all the above features, it offers options such as quickly finding characters and scene locations in the text, while providing a 'story map'.

Final Draft has structure features too, but writers do not always use them if they do not understand how they can be useful after they hand off the script. For example, screenplays that HRA has received were often missing structural information despite Final Draft supporting it.

The use of 3D storyboarding tools is often avoided due to the large overhead which follows most 3D approaches compared to their 2D counterparts. However, once the initial stage of loading assets and organizing them by

populating scenes is finished, a 3D approach can offer significant advantages, such as immediate feedback in terms of how the existing assets can fit and create the actual storyboard panel.

The approach presented in this chapter to automate the process of importing and organizing all the assets is to parse a Final Draft screenplay and extract relevant information such as character names, in which scene each character name is mentioned, set descriptions etc.

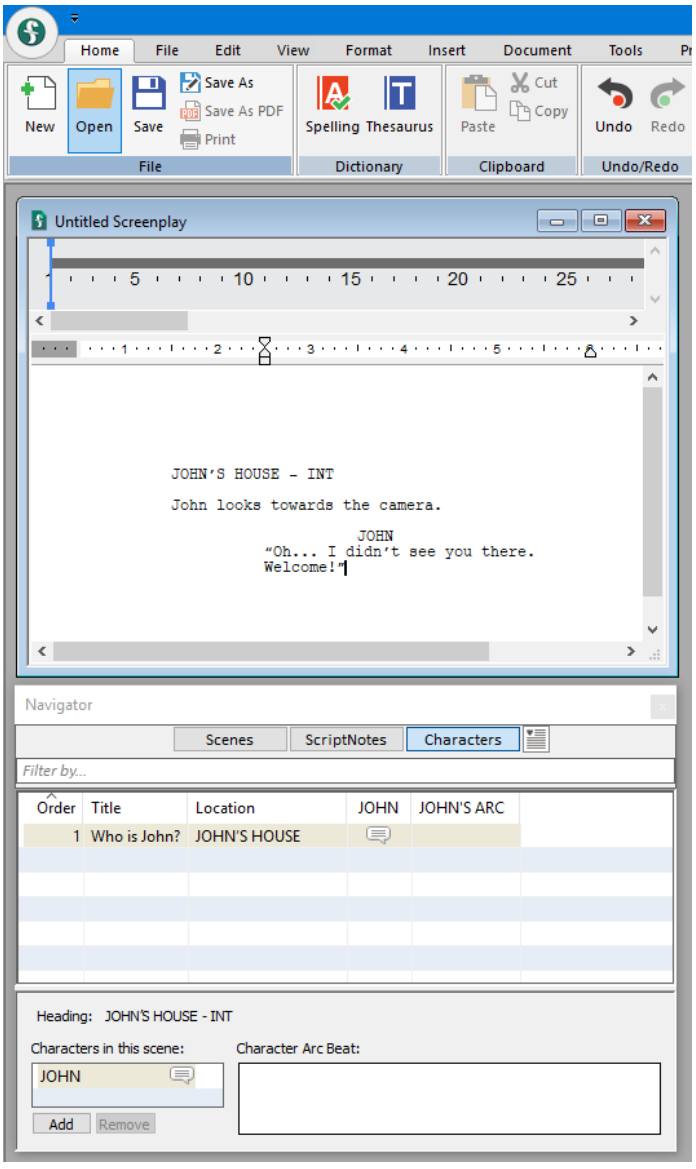



Figure 15: Screenshot of Final Draft software.

## 5.3 METHODOLOGY

The system proposed works with Final Draft files and successfully extracts data such as sequences, scenes in each sequence, shots in each scene and characters in each shot as well as their actions. If the actions are dialogue, all the necessary text is extracted too.

The problem is broken down into three parts:

- Parsing the screenplay
- Linking the parsed information with the 3D assets
- Generating Redboard project file

The system is developed in separate Python modules. This allows for maintainability as one can edit individual parts of the system without affecting the overall behavior. The reason Python is chosen is due to the text-parsing nature of the problem and its popularity in the animation industry. For example, the Python parsing system could be used to setup information extracted from the screenplay, directly in pipeline asset management systems such as Shotgun[1].

### 5.3.1 *Parsing the screenplay*

Final Draft screenplays are XML files. There is useful metadata in the XML that is leveraged for the parsing step. For example character names have an XML attribute appended to them named 'Type' equal to 'Character'.

Screenplays are broken down into sequences, scenes, shots and actions. Sequences describe things like episodes, which contain a series of scenes. Most of the time, the screenplays that HRA has worked with contain one sequence at a time. Nonetheless, there are cases when more than one sequence is included in the same screenplay. Scenes contain a series of shots while shots contain a series of actions. Actions are acted out by characters and describe activities like talking, running etc. as per the screenplay.

Final Draft, as mentioned earlier, has some smart features that allow it to categorise sets separate from the time of day. However, it was often the case

1 https://www.shotgunsoftware.com/

with screenplays used by HRA that these smart features were not used by the writer. As such, manual parsing was necessary.

In Final Draft screenplays, set information is contained in headings. It is important to separate the set name from information like time of day, or whether it is an inside or outside set. Other information, like the time of day (e.g. 'night') also appears in the same heading. Whether action takes place during the day or during the night, the set name is the same, so it is important to separate it from the rest of the information. One of the difficulties in doing so is that there is no standard order in which this information is structured. A scene that takes place during the day in the 'terrace' set, which is outside, can appear in the screenplay as:

- EXT. TERRACE — DAY
- DAY — TERRACE EXT.
- TERRATE (EXT) — DAY

and in many other variations, both in terms of the order and in terms of the capitalisation of the characters. The system proposed in this chapter uses a map of common words such as 'ext' for action that takes place outside and 'day' for time of day. A list of common separators, such as '-' is also used. These mappings were built based on common screenplay conventions extracted from real-world productions, such as Q Pootle 5 (Blue-Zoo Productions 2013). They are parsed with regular expressions to find the words in the map, e.g. 'int' or 'ext'. In this way, the final set name is extracted from the heading.

### 5.3.2 *Linking the information*

Apart from parsing the screenplay, it is important to link information like actions to characters consistently. In the current implementation, conventions are created and kept as separate input to the system in the form of lists, e.g. a list of character names. For example, a character named 'Kat' can appear in the screenplay as 'Kat' but also as 'Katherine', as well as 'Kat (continued)' or 'Kat (cntnd)'. This mapping is specified as a pairing in a separate list, rather than renaming the screenplay directly. This allows the screenplay to

remain intact as the writers originally created it, but to handle all these different instances as the same character. Words appended to a name, such as 'continued' are included by default as they were identified to be common occurences in screenplays. On the other hand, a mapping to state that 'Kat' and 'Katherine' are actually the same character has to be provided as user input.

The system links each character found in the screenplay to their actions. For each action, the text associated with it e.g. 'Kat turns off her computer' is stored. This is useful later on when characters are included together with the screenplay text in the final Redboard project file.

### 5.3.3 *Generating Redboard project file*

The current implementation generates a Redboard project file, as seen in Figure 16. The link between models (characters, sets etc.) and their appearance in the screenplay was done by the exact names extracted from the screenplay. For example, a character named 'Kat' would be linked to a file named 'Kat.obj'.

The current implementation automatically structures the scenes and then for each scene it creates shots, which are based on actions as seen in the screenplay (Figure 17). Actions are used to create separate storyboard panels in the 3D storyboarding software.

If there are multiple sequences in the same screenplay, each sequence is output in a separate project file to make loading them into Redboard easier and faster. It is possible to combine them into one project file later on through Redboard.

The final result is a Redboard project file which has scenes, shots and actions already created (Figure 16). For each of these, the correct set and character models are automatically imported. Textual information from the screenplay is also brought in and overlaid in the UI, e.g. 'TIME: DAY' (Figure 17). At this point, the storyboard artist can begin their creative work, like positioning the imported characters, moving the camera and telling the story, without the need to do this initial project setup manually.

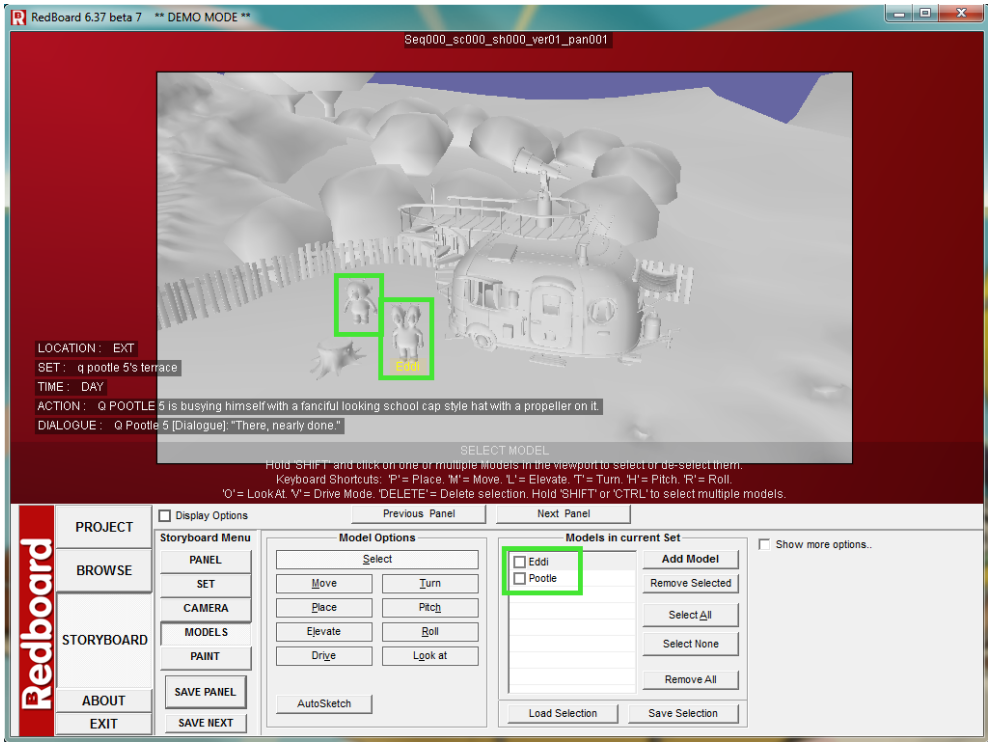



Figure 16: Automatically populated Redboard project from 'Q Pootle 5' (Blue-Zoo Productions 2013) screenplay.

## 5.4 RESULTS

The system's output was evaluated cooperatively by two users, both experts in 2D, 3D and storyboarding. One user was an expert in the use of the specific software (Redboard) too. In this section, the words scene and set are used interchangeably.

The overall feedback was positive, as the users believed the output would save them significant amounts of time. More specifically, they confirmed that the process of populating the sets with the 3D assets is something that storyboard artists do not consider as a part of their job but rather a delay to their creative process and as such should be automated.

The idea of having all the scenes structured with all the models that appear in each scene already imported received particularly good feedback. One of the main problems that can be the result of human error when importing the assets is to neglect adding all the necessary character models for a scene. This automatic process presents the users with a list of already imported

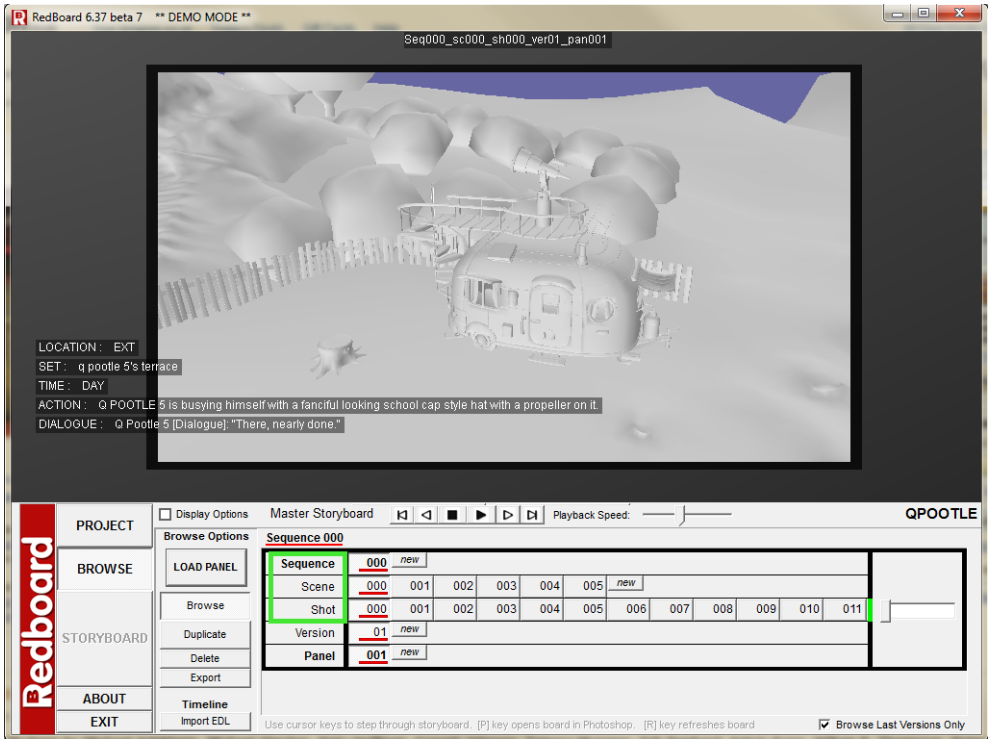



Figure 17: Automatic shot breakdown as seen in Redboard.

characters. Any characters wrongly imported can be removed. This helps ease the burden on the user's short-term memory, as the alternative would be remembering which characters must be imported and bringing them in one by one.

Apart from sequences and scenes, the screenplay analysis system breaks the screenplay down into shots. Shots contain actions such as a dialogue between two characters, a character walking etc. In contrast to the rest of the functionality of the program, feedback on this part was less positive. More specifically, the users consider the process of creating the shots based on the actions to be part of their creative input. Storyboard artists often go through the screenplay and take the liberty of changing the order of certain actions, grouping actions together into a shot while separating others. This process is considered to be creative and therefore — according to the received feedback — should not be automated.

## 5.5 DISCUSSION & FUTURE WORK

Even with available 3D models to load into Redboard or another 3D storyboarding software, it is required for the end-user to setup all of the scenes and then load the corresponding models for each shot. This can potentially take a long time and discourage storyboard artists from following this route. Moreover, it can be a distraction to the actual creative process.

The screenplay is one of the first elements created in a production, containing much information such as the number of sequences, the breakdown of scenes, the events and characters as well as their specific actions. The general idea is to use the screenplay in order to extract such data automatically. This can then be used to automatically populate 3D scenes with the correct characters and allow the storyboard artists to immediately start being creative.

Although the screenplay analysis system has been tested with several different Final Draft screenplay files and it successfully extracts the different parts (sequence, scene, shot, characters and their actions), due to the wide variation that screenplay writers have in the style of writing, there can be potential problems. For example, extracting a character name might depend on the convention with which the writer chooses to annotate that name. For example, 'John (continued)' is a common way to denote continuous dialogue. Therefore, either research on the most common styles of screenplay writing is necessary or a clear set of conventions within studios. In both cases, the system can be transformed easily, due to its modular nature, to include the different conventions of each studio.

Moreover, linking between set names or character names and their corresponding 3D assets is done based on the name. For example, if a character named 'Tom' is found in the screenplay, their corresponding 3D model would be 'Tom.obj. Future work could look into better ways of linking the two, by first establishing a consensual pipeline and a set of conventions.

More specifically, a set of rules or conventions is important to ensure that teams of creative and technical people are thinking in the same terms. Although sets of tools like the ones discussed in this thesis have the potential to reduce the durations of certain pipeline phases by being well integrated into the pre-existing workflow, it is also crucial for a common consensual pipeline process to exist around these tools. Therefore future work can also

focus on ways to develop or extend the existing processes and integrating them into the same pipeline.

# 6

# INTEGRATION OF PLACEHOLDER CREATION AND SCREENPLAY ANALYSIS

## 6.1 MOTIVATION

With Redboard already existing as a 3D storyboarding system and with some of its shortcomings identified and presented in this thesis, some new but disconnected systems were developed. A system for automatic creation of place-holder assets (Chapter 4) was developed in order to mitigate the early requirement of 3D assets when storyboarding in 3D, while a system that can extract scene, shot and character text data from screenplays (Chapter 5) was developed to mitigate importing and setting up of a 3D scene when storyboarding in 3D.

As long as the systems remain disconnected, however, the workflow does not yet form a pipeline.

The aim of the system presented in this chapter is to bring all the work together into a coherent pipeline by connecting and integrating the placeholder asset creation system with the screenplay analysis system.

## 6.2 BACKGROUND

Integration is an important part of the research presented in this thesis. After all, the aim is not to produce individual prototypes of ideas but to actually create solutions integrated into the company's existing workflow. Tools need to be sufficiently decoupled to allow for easy switching or modification of individual components, yet they should form a pipeline flowing from one step to the next.

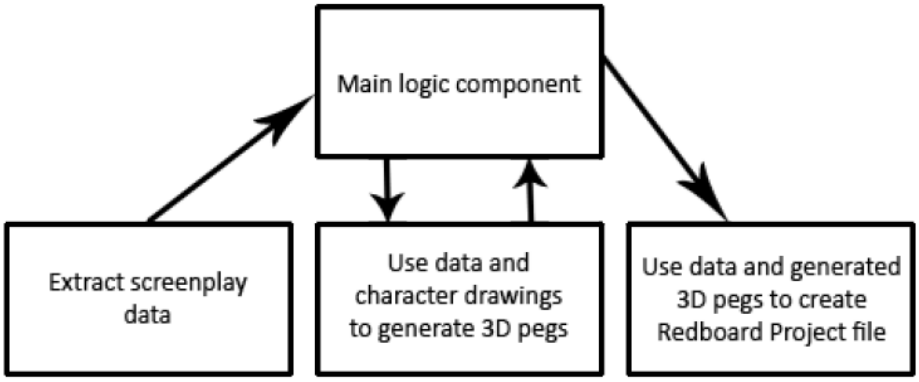



Figure 18: Structure of integration between the Peg Creation and the Screenplay Analysis software.

## 6.3 METHODOLOGY

To combine the screenplay analysis with the placeholder asset creation system, a simplified pipes and filters pattern is used as shown in Figure 18. Since these systems would not work in parallel for each episode or scene, this pattern is suitable. The final output is a Redboard Project file which has all the necessary initial information laid out for the end-user. Redboard project files are binary files, therefore scene data extracted from screenplays are transformed to the correct format using an in-between software interface. Integrating the two tools means that the placeholder asset creation tool UI no longer needed, as arguments are passed via command line.

The proposed solution includes two separate systems, the screenplay analysis and the placeholder asset creation, to address some of the already discussed issues of 3D storyboarding. The systems are fine-tuned and integrated into the Hibbert Ralph environment, however can be easily customised to satisfy other requirements.

For this implementation, Python is used to link the two systems together as it is a popular language in the field of animation and VFX and available in most digital content creation tools like Autodesk Maya. The input list of the placeholder asset creation system allows for a names list to be passed. This indirectly provides the placeholder asset creation system the number of characters in the drawing, used for the segmentation process and is also used to help the user name the generated placeholder 3D assets.

The names list would normally have to be created manually, but in this case it is collected during the screenplay analysis process. By getting the character names from the screenplay, it is no longer necessary to manually provide a character count or list of names to the placeholder asset creation system. Moreover, a Redboard project file is generated breaking up the screenplay into scenes, shots etc. but also automatically linking the generated placeholder assets in the scenes they act in. Because the placeholder assets are created as part of this, they are used to represent the screenplay characters generated by the screenplay analysis system. It is still possible to manually override the names e.g. to specify when multiple names are used to describe the same character.

## 6.4 RESULTS

Figure 19 shows how the character names identified through the screenplay analysis are fed to the placeholder asset creation system in order to name the 3D models appropriately.

Apart from feeding the screenplay analysis names into the placeholder asset creation system to segment and name the characters correctly, the created 3D placeholder assets created by the placeholder asset creation system are linked into the Redboard project file generated by the screenplay analysis system.

## 6.5 DISCUSSION & FUTURE WORK

This research portfolio item serves to show how independent novel ideas can be combined to create a consistent workflow. More specifically, by assisting 3D storyboarding with these new tools, storyboarding in 3D can be integrated with pre-existing media from the traditional pipeline, contrary to the views of some clients HRA discussed with.

Furthermore, by linking all the projects together, it becomes easier to focus on the next problem to tackle. Namely, combining 3D storyboarding with automatic place-holder asset creation and screenplay analysis, a 3D animation production is able flow from screenplay, to storyboarding, to first-pass 3D layout without the need for expert 3D software operators. It is also possible

to establish a better link with existing shot and asset management industry standards, such as Shotgun (Shotgun Software Inc. 2018), to pull set and character assets automatically. As such, achieving second-pass 3D layout in a similar fashion is explored in the next two chapters.

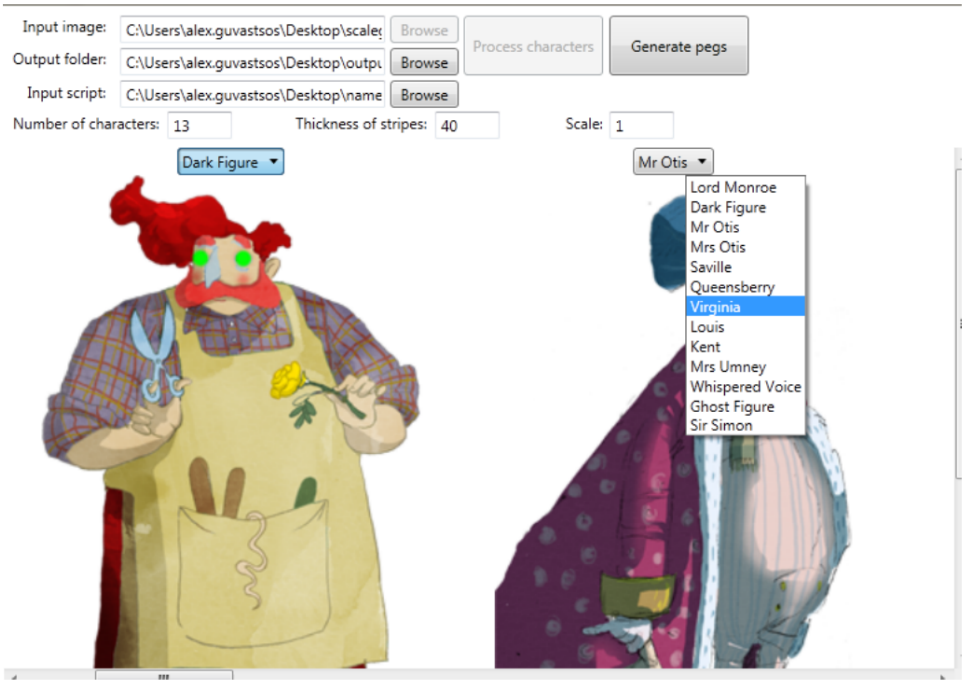



Figure 19: A screenshot of the Peg Creation software, where dropdown boxes allow the artist to choose the character's name.

# 7

# CHEAP MOTION-CAPTURE DEVICES FOR HUMANOID POSING IN PRE-PRODUCTION

## 7.1 MOTIVATION

The character posing stages before going into animation are a time consuming and labour intensive process. For the user performing the process, it is usually required to possess a solid understanding of 3D graphics and to be trained in using a complex software suite such as Autodesk Maya.

Through the effective use of 3D storyboarding and Redboard, storyboarding and first layout pass can be performed in one step. This allows storyboard artists and directors to ensure that each shot looks as intended in 3D, in terms of scale, framing and positioning. However, the second layout pass still requires an expert operator of 3D software (e.g. Maya).

In this chapter, an alternative workflow is proposed with emphasis on making the initial posing stages accessible to less trained users through the use of Microsoft Kinect.

## 7.2 BACKGROUND

Capturing real world information such as character poses is a wide area of research, with fields like surface reconstruction and Motion Capture (mocap).

Surface reconstruction approaches, whether for objects or characters, vary from 3D laser scanning (Wang et al. 2003), to image and video based methods (Zhu et al. 2017). More complex rigs like the former produce better results compared to simple camera approaches, but are more expensive and difficult to setup (Xia et al. 2017).

Pose and motion capture approaches include solutions that range from magnetic sensors (O'Brien et al. 2000) which are less invasive, to wearable inertial trackers (Lintmeijer et al. 2018), to optical tracking methods using multi-view (Vlasic et al. 2008) or monocular (Wei and Chai 2010) camera data. Using multiple views is a way to deal with the most common issues of noise and occlusion. Noise can occur due to lighting conditions, calibration issues and overall sensor errors. Occlusion can occur due to other objects hiding the actor, or the actor self-occluding themselves: for example placing their hands behind their back. For optical methods, markers can be positioned on the captured subject in order to track specific points such as an actor's joints. For blockbuster films, where multiple cameras are available and the captured data must be as accurate as possible, actors can wear suits with optical markers that emit their own light to improve their visibility to the camera (Failes 2017). Depth sensors capture information that has the potential to be more robust than colour information when it comes to using only one device (Baak et al. 2011).

Irrespective of the capture device, results of the captured motion can be improved by constraining the problem space. One example of this is constraining the possible inferred poses of the capture to be physically possible based (Andrews et al. 2016). The fact that researchers began building databases of human poses, for benchmarking and reference (Andriluka et al. 2014) has helped data-driven approaches. Apart from the aforementioned physically-based solvers, databases are another way to constrain the problem space, as they can be used as a pose knowledge-base, e.g. to train convolutional networks (Wei et al. 2016).

Another example of a data-driven approach is the Microsoft Kinect SDK, which was trained on a database of frames consisting of a variety of actions such as driving, dancing, and running (Shotton et al. 2011). The emergence of Kinect in the field of computer vision has provided a way of collecting 3D point cloud data cheaper than in the past. It is a successful example of using multiple data inputs (combining depth and colour information) to provide more robust results. Nonetheless, although a single Kinect is easy to setup, it can still suffer from noise, occlusion and overall depth-data inaccuracy issues (Xia et al. 2017). As an alternative to improving accuracy through the use of multiple Kinect sensors (Ye et al. 2013), it is also possible to collect

more accurate depth information by capturing multiple angles with the same Kinect device, in real-time (Izadi et al. 2011).

Related work in mocap can have applications in other areas of entertainment apart from animation, like rotoscoping (Agarwala et al. 2004). Having said that, it is not only the field of entertainment that benefits from these research breakthroughs. Fernandez-Baena et al. (2012) propose using the Microsoft Kinect as a way to validate rehabilitation treatments, due to its price, portability and no need for markers. Lange et al. (2012) take it a step further and apply gamification to the rehabilitation process.

The field of animation deals with non-humanoid characters too. Even though this project focuses on capturing the static pose of humanoid characters, it is worth noting that related work extends the use of performance capture to a wider range of characters (Seol et al. 2013) and even arbitrary physical objects (Chen et al. 2012). Although the Kinect has been used to track objects such as toys (Held et al. 2012), mocap data is usually collected from human actors; applying it to non-human characters is a non-trivial problem due to topological differences between the source and the target of the pose. Yamane et al. (2010) propose that the non-human character would still convey the same emotion through body language. They approach the problem of mapping the human pose to the character pose by building a statistical mapping function using Gaussian process latent variable models combined with physically-based restrictions.

Assuming that at this stage of the pipeline there are already available 3D models, this chapter looks at capturing human pose information in a way that is accessible to both bigger and smaller studios. Moreover, the project proposed in this chapter aims to leverage the low price and accessibility of the Kinect sensor.

The process of acting out the initial posing can help create more natural poses, and requires no technical skills. After the characters have been positioned in the scene and the storyboarding process is complete, a method is proposed for posing the characters without the need to know about Forward Kinematics (FK) or IK, manipulating joints or other technical skills. The user can act out the various poses by using a Kinect device or similar cheap mocap devices. Through this process, a single user can go through an entire scene and pose all the characters one by one.

## 7.3 METHODOLOGY

This section summarises an overall work flow methodology of going from storyboard panels directly to a 3D layout phase in Autodesk Maya. The methodology relies on a system which makes use of the Microsoft Kinect sensor to track a human pose through skeleton capture and correctly map it to three dimensional rigged assets. In order to make the solution as cheap and simple to setup as possible, only one Kinect sensor is used. Because of the use of pose capture, the system currently focuses on the posing of humanoid creatures.

The proposed work flow can be broken down into the following two steps:

1. Finish the storyboarding process.

2. Using the created system, go through each storyboard panel. For each character in the panel, act out the wanted pose in front of the Kinect sensor to automatically pose the 3D model in Maya (Figure 20).

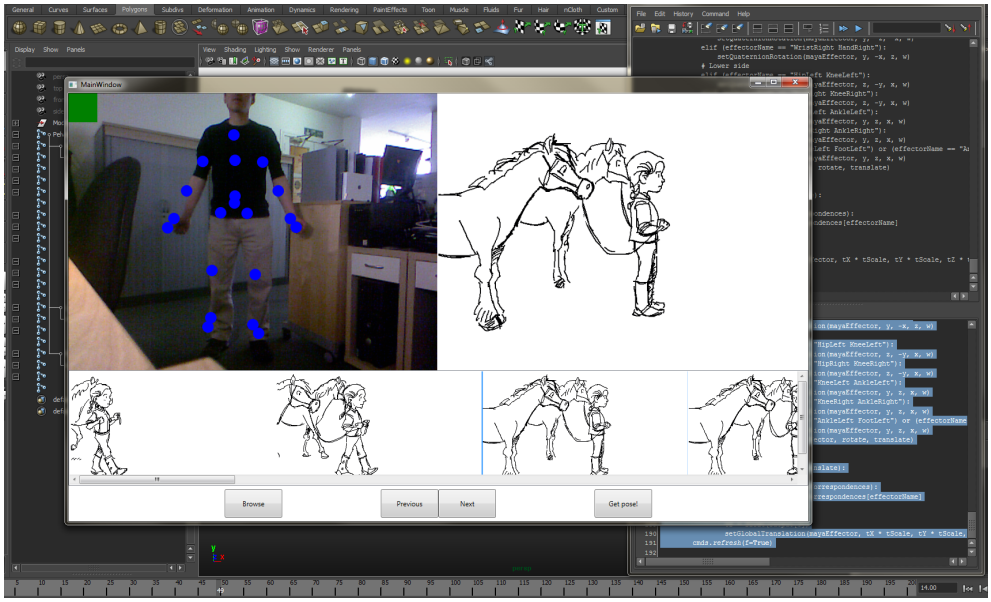



Figure 20: The created system running within Maya, tracking the pose of an actor while viewing a series of storyboard panels.

The created system is written entirely in the C# programming language using WPF for creating the UI. The official Microsoft Kinect drivers as well

as the official Kinect for Windows Software Development Kit (SDK) are used to interface with the Kinect sensor. The first version of the Kinect for Windows sensor was used. Version 1.6.0 of the Kinect for Windows SDK is used, which provides the logic for detecting the joints. These choices were made in order to have a stand-alone system which is robust and whose individual components are well tested. It interacts with Maya through a client-server interface. Essentially, Maya runs a python script turning it into a server that listens to a specific port and exposing a set of functions as an Application Programming Interface (API). Moreover, the Coding4Fun Kinect Toolkit 1.5.0 library is used to help scale joint positions and draw them accurately over the actor as the window size of the UI is resized or drawn on screens of different resolutions. The Microsoft XNA framework 4.0 is used as a library for matrix multiplication, used for concatenating joint rotations.

The UI contains three main components. The first component is a rectangle containing the camera data from Kinect, with the tracked joints overlaid on top of the person who acts out the pose. The second component is a rectangle containing the selected storyboard panel in a large view. The third component is a horizontal slider, showing thumbnails of all the storyboard panels (Figure 21). It is important to note that it is easy to navigate the interface while standing in front of the sensor to act out the pose, by using a device like a clicker which allows wireless control of the mouse (Gouvatsos et al. 2014a).

In order to get a collection of storyboard panels one can press the 'Browse' button and browse to a folder containing the storyboard panel images. To switch between storyboard panels, one can either press the 'Previous' and 'Next' buttons, or select a thumbnail from the slider. Finally, to map the tracked Kinect pose to the 3D rigged asset in Maya, one can press the 'Get pose' button.

### 7.3.1 *Mapping of storyboard timeline to Maya timeline*

As the user selects a different storyboard panel in the UI, the system automatically organises a keyframe structure in Maya. More specifically, based on the order of the storyboard panels, each panel is mapped to a keyframe in Maya, panned out to match a user-specified frame rate. For example, if the wanted frame rate is 25 frames per second, the pose from the first story-

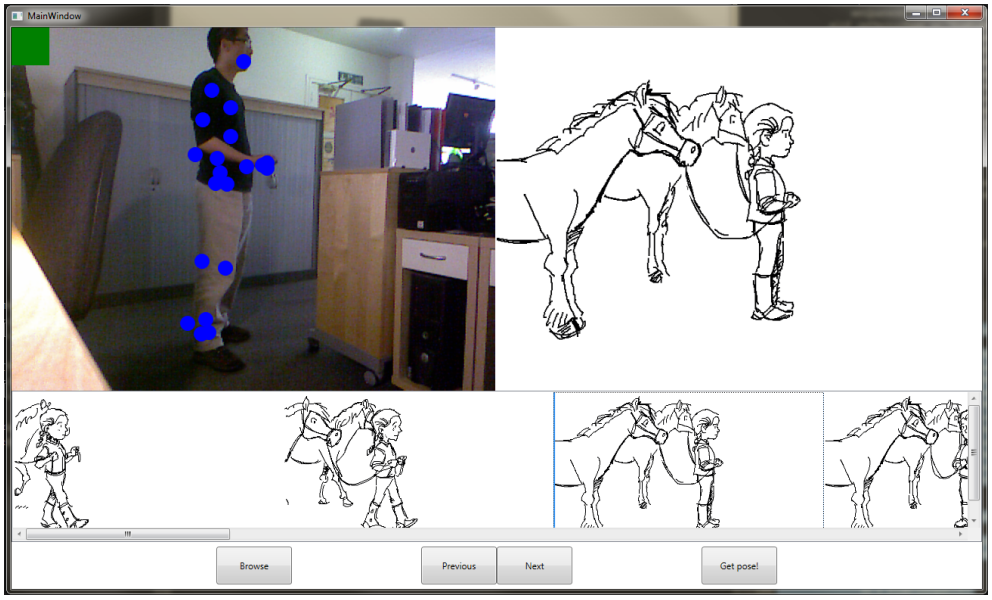



Figure 21: The main UI of the system, as an actor is posing in an office environment, according to the selected storyboard panel.

board panel would be placed in frame 1 of Maya's timeline (Figure 22), while the pose from the second storyboard panel would be placed in frame 25 of Maya's timeline (Figure 23). This information is extracted from the editor's work (e.g. an EDL file), as boards may vary in framerate, may be repeated etc.

### 7.3.2 *Mapping skeleton capture data to 3D rigged assets*

A Kinect pose is represented as 20 joints with their corresponding transformation matrices (Figure 24). From the transformation matrix, the Kinect for Windows SDK makes it possible to extract the absolute position in world space coordinates, the absolute rotation in world space coordinates, as well as the hierarchical rotation in local space coordinates, for each joint through the API. In this case hierarchical rotation is of interest, in order to map it to a 3D rigged asset irrespective of where it is located in a set or of its current root position.

Even though, as mentioned earlier, only the hierarchical rotation of each joint is of interest, the root joint (in this case the pelvis) is an exception. The pelvis' hierarchical rotation is the identity matrix, since it has no parent

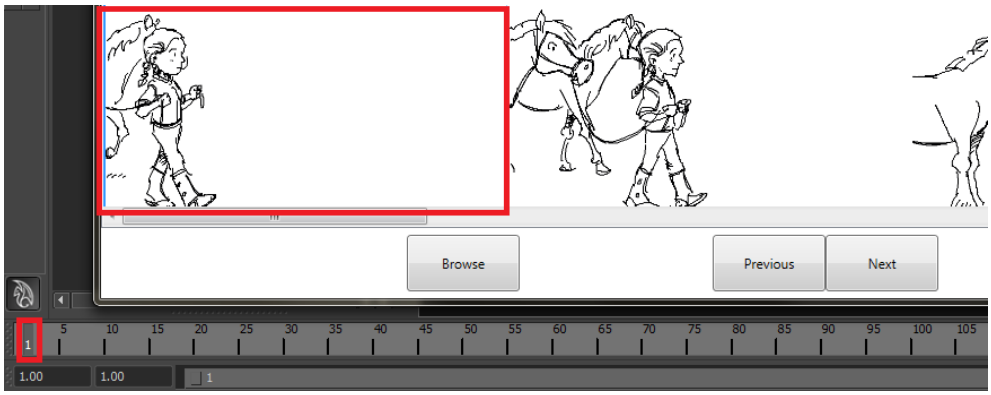



Figure 22: For 25 frames per second, the first panel corresponds to the first frame in Maya's timeline.

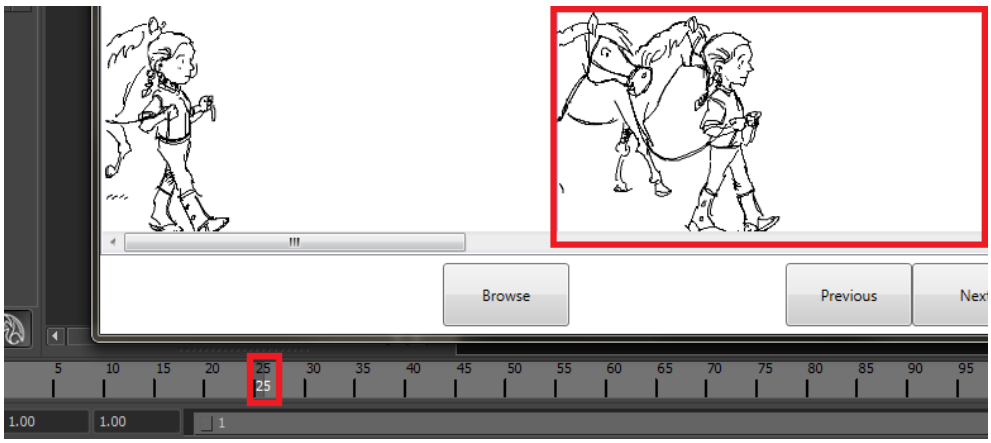



Figure 23: For 25 frames per second, the second panel corresponds to the 25th frame in Maya's timeline.

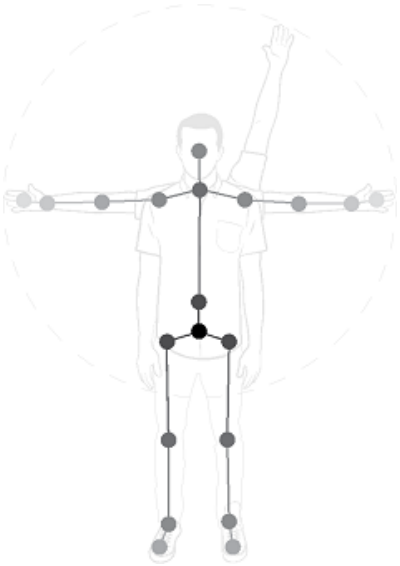


Figure 24: The skeletal structure of the Kinect data. There are 20 joints on the human body tracked from the Kinect sensor (Microsoft 2017).

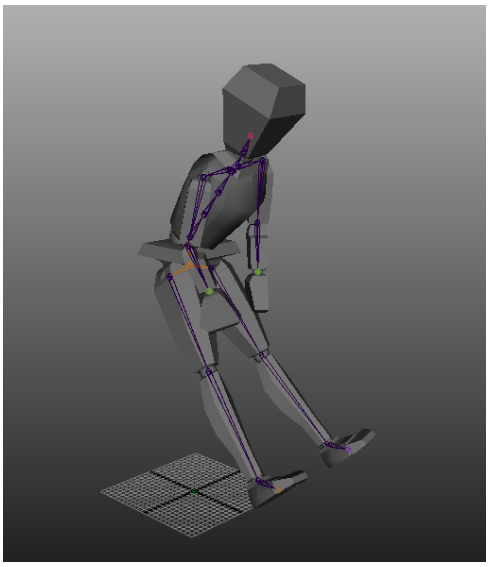
joint. Because the Kinect for Windows SDK skeleton convention rotates the legs to appear as if they are off the ground, rather than the hips to bend (Figure 25a), hierarchical rotation is not enough. As such, the pelvis also requires an additional rotation to fix this (Figure 25b).

To fix this, first the rotation of the root joint (the pelvis) of the 3D humanoid is set using the absolute coordinates from the Kinect data. It is important not to set the absolute coordinates for all the rotation axes of the pelvis, in order to ensure that the storyboard artist's initial translation (set in Redboard) is not affected. As such, only the pelvis is rotated in terms of its local X axis (Code Snippet 3).

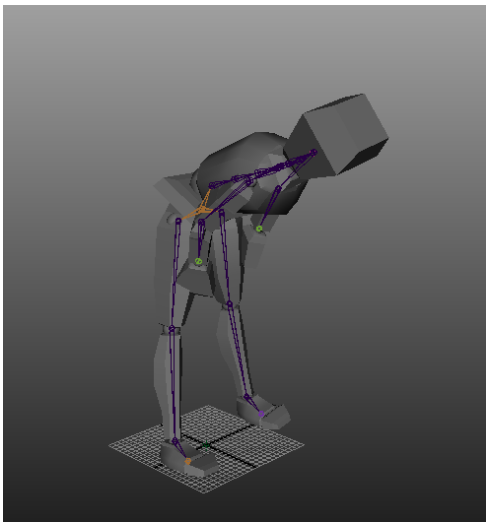
Then for each joint in the hierarchical chain, the hierarchical rotation from the Kinect data is used and concatenated with the transformation matrix of its parent joint.

It is also important to establish a naming and rigging convention for the 3D models to have, in order to easily link them to Kinect data. One issue raised was exactly because of this, as the axis of joints in various rigs had differing orientations, causing wrong mapping from the skeleton capture data to the Maya model.

While it is important for these conventions to exist, a joint-to-joint rig matching is not necessary, as long as the mapping is done correctly. This



(a) Result pose if the root joint is not rotated.



(b) Result pose after the root joint has been rotated using an absolute rotation in terms of the local X axis. Note that now the legs are correctly touching the ground and the overall pose appears as intended.

Figure 25: Kinect data needs to be transformed to be used in Maya.

means that it is not necessary for a rig to have the exact same number of joints that the Kinect data assumes, but for the joints that it does have, the naming and rigging conventions should be followed. In the examples presented in this chapter, there are small differences between the 3D rigged asset and the Kinect data, e.g. the 3D rigged asset has some additional joints in the spine area (Figure 26). This is not an issue as long as the main rig joints are mapped to the corresponding joints on the Kinect skeleton.

There are rigging conventions for a correct mapping function between a 3D model in Maya and the captured pose. The following guidelines for model rigging when it comes to pose capture with Kinect have been established through testing:

1. In order to keep the correct rotation order (x, y, z), the local axis orientation for each joint on the model must follow these rules:
   a) The Y axis is always down along the bone
   b) The Z axis is always out towards the camera
   c) The X axis always follows a right-hand rule
2. The rest pose of the model should not be a T-Pose (Figure 26), but instead should be an A-Pose, which assumes a further 30 degree rotation of the arms along the Z axis (assuming Z axis points to the camera) towards the hips (which is the more modern rest pose convention for video games)
3. Due to the convention of the first version Kinect API, the legs should be pointing upwards (due to the first version Kinect API convention) and this should be achieved by rotating the Z axis by 180 degrees at the hip joints (Code Snippet 3)

## 7.4 RESULTS

The proposed method was evaluated by creating a 2 minute-long sequence. Using the initial screenplay, a storyboard artist was able to efficiently block out the shots and position the characters in the scene. The artist was able to immediately identify issues in shots and reposition characters and objects accordingly. It would not be possible to identify these problems in a traditional

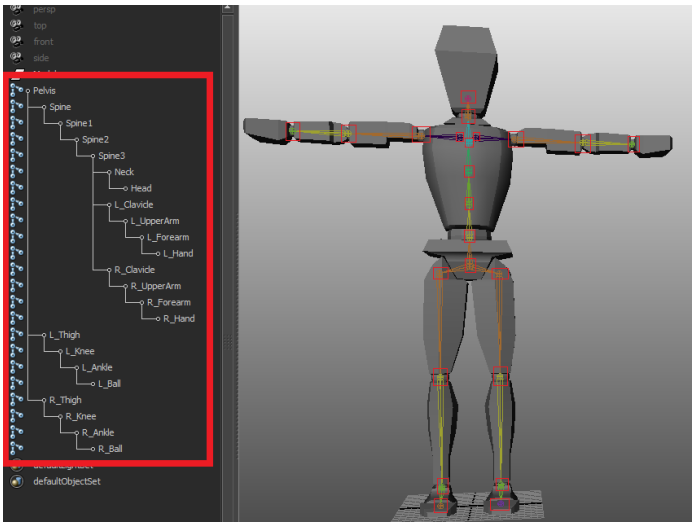



Figure 26: The skeletal structure of an example 3D rigged T-posed asset in Maya. Note that there is not a one-to-one correspondence to the skeletal structure of the Kinect data shown in Figure 24.

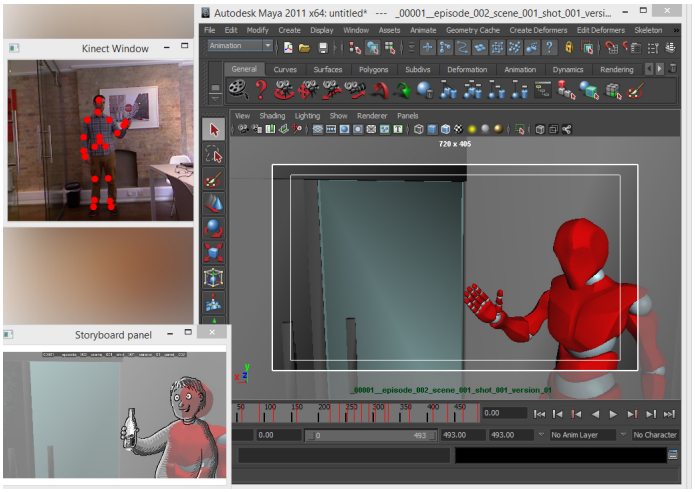



Figure 27: A screenshot during practice for the live demo sequence. The top left window shows the actor acting out the pose in front of the Kinect, with the joints overlaid on top. The bottom left window shows the storyboard panel the actor is currently acting out. The Autodesk Maya window is updated whenever the actor captures a new pose.

storyboarding scenario until much later when the storyboard is transferred to a 3D scene.

The common motion noise issue that may arise from mocap is not an issue in this process, since the result is a static pose rather than a final animation. Because the result is used as a keyframe, noise that may affect the continuity of the poses is not as important and it is easy to retake a pose until the artist is satisfied, as the results are viewed in real-time.

Then one user was able to pose the two human characters in the entire sequence in a few minutes. It would take a professional about an hour using traditional techniques. More importantly, it is worth noting that the user had no previous experience in posing characters and was not a professional 3D artist or animator.

The end result was a ready 3D animatic, with the initial posing stages completed. The entire sequence took half a working day, compared to three or four days using traditional methods. Depending on the quality of the poses at this stage, there can be an additional pass to smooth out some details. For example, there may be more stylised poses that the artist wants for the final look. Once this is done, the production can proceed to the animation stage. Figure 28 shows the system running while an operator is standing in front of the kinect, while Figure 29 shows the system in action posing a human character as part of an overall sequence.

The proposed workflow together with the developed software were presented at the Annecy International Animated Film Festival (Ahoomey et al. 2014) in the form of a live demo, posing two characters in a new environment (the presentation stage) on one machine, for an entire scene, within minutes (Figure 27).

## 7.5 DISCUSSION & FUTURE WORK

One of the advantages of using the Microsoft Kinect sensor with the Kinect for Windows SDK is that it naturally handles self-occlusions and works with various body types (Shotton et al. 2011). Having said that, there could still cases where the captured pose did not match the storyboard 100%, but as discussed earlier, the poses would be close enough for an artist to quickly clean up as a second pass. It is worth noting that this solution does not

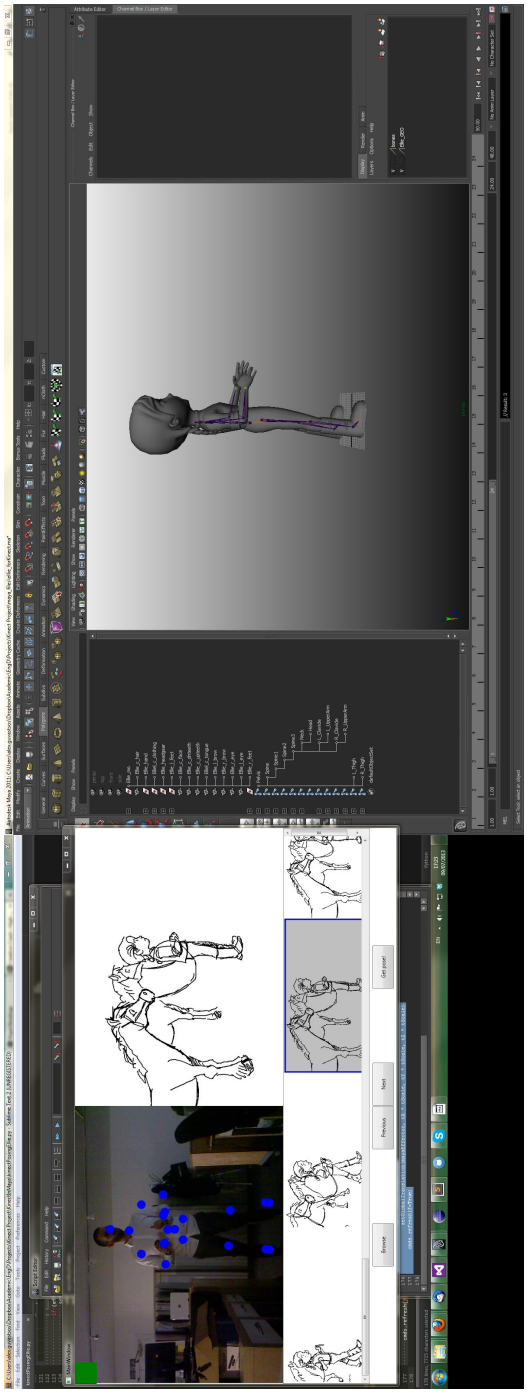


Figure 28: An example of an actor posing a 3D character in Maya, according to the selected storyboard panel.

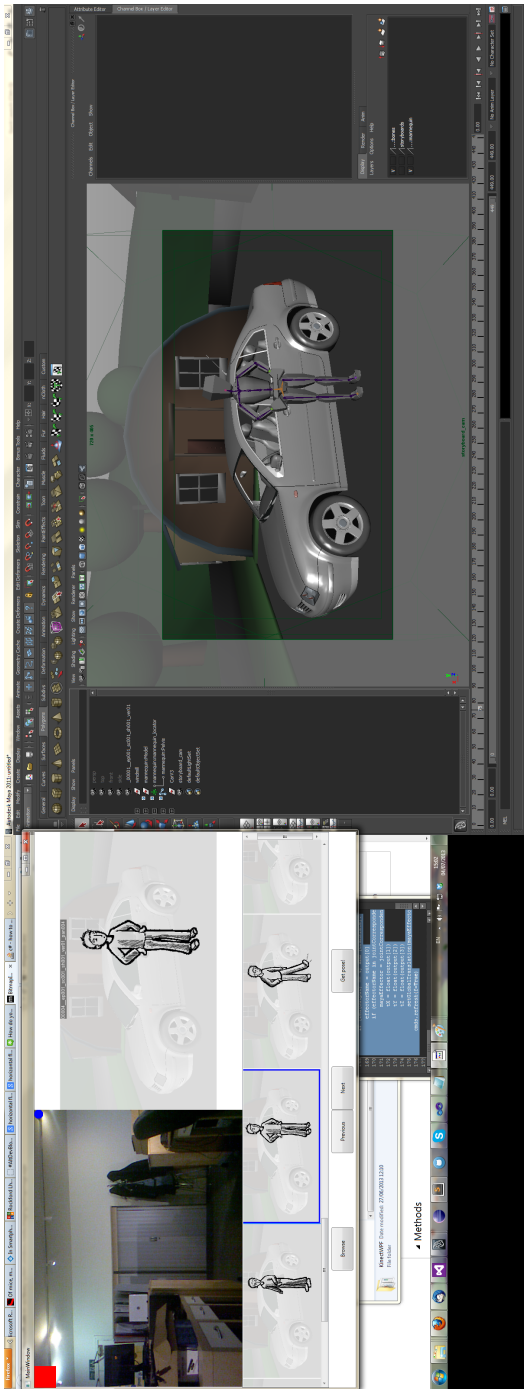



Figure 29: An example of a character in a Maya scene posed using the proposed pose capture system. The actor is in this case controlling the computer and thus does not appear in front of the camera.

handle finger tracking, so it is currently not possible to capture an action like pointing at something.

An advantage of the proposed work flow is that it is suitable for any storyboarding method, as it can be plugged in as a black box between the storyboarding phase and the 3D layout phase. Having said that, it is important to note that it is even more powerful when used in a 3D storyboarding pipeline. Combining the ideas presented in this chapter with a 3D storyboarding pipeline offers some interesting avenues for future work:

1. Explicitly show which character in the storyboard drawing corresponds to which 3D model in the set. This is only possible with 3D storyboarding because the drawing is overlaid on top of the actual 3D model.

2. A more intuitive UI, where rather than having to select each model from a list or similar control, the user can select a model by clicking directly on the drawn character in the storyboard panel.

3. Given the first example in this list, the overall experience of the system can be simplified to only two buttons, by cycling through the characters and presenting only one drawing at a time. This would allow for a very simple and understandable interface for non-technical users.

Code Snippet 3: Hip rotation, legs correction and A-Pose rotations in Autodesk Maya Python/MEL pseudo-code.

```
# Root joints/bones
if (effectorName == "HipCenter HipCenter"):
    setQuaternionRotation(
        mayaEffector,
        float(output[6]), float(output[7]),
        float(output[8]), float(output[9]),
    )
    xRotation = mel.eval(
        'getAttr "' + mayaEffector + '.rotateX"',
    )
    mel.eval('select -r "' + mayaEffector + '"')
    mel.eval('rotate -a -os -yz 0 0')
    if ((xRotation >= 90) and (xRotation <= 270))
    or ((xRotation <= -90) and (xRotation >= -270)):
        mel.eval('rotate -r -os -x 180')
    mel.eval('select -cl')
elif (effectorName == "Spine ShoulderCenter"):
    # The first Kinect version had a standing and seated mode
    if not seated:
        continue
# Legs
elif (effectorName == "HipRight KneeRight")
or (effectorName == "HipLeft KneeLeft"):
    setQuaternionRotation(mayaEffector, x, y, z, w, 0, 0, 180)
# Back
elif (effectorName == "HipCenter Spine"):
    setQuaternionRotation(mayaEffector, x, y, z, w, 40, 0, 0)
# Arms
elif (effectorName == "ShoulderLeft ElbowLeft"):
    setQuaternionRotation(mayaEffector, x, y, z, w, 0, 0, -30)
elif (effectorName == "ShoulderRight ElbowRight"):
    setQuaternionRotation(mayaEffector, x, y, z, w, 0, 0, 30)
# Torso
else:
    setQuaternionRotation(mayaEffector, x, y, z, w)
```

# 8

# AUTOMATIC 3D POSING FROM 2D HAND-DRAWN SKETCHES

## 8.1 MOTIVATION

Sketch-based posing refers to the automatic posing of 3D models using 2D hand-drawn sketches. It also refers to UIs that allow for 3D posing by drawing, instead of operating complex 3D software controls.

Animation encompasses entertainment ranging from traditional hand-drawn pieces like 'Snow White and the Seven Dwarfs' (Walt Disney Productions 1937) all the way to cutting-edge visual effects in blockbuster Hollywood films like 'Avengers: Age of Ultron' (Marvel Studios 2015). As the boundaries of what is possible are pushed and 3D computer animation is widely adopted, this originally artistic field has become highly technical. Today's animators employ cutting-edge techniques, such as 3D skeleton manipulation and motion or pose capture, as seen in Chapter 7.

This technological boost has helped artists give life to characters and worlds that previously existed only in the realm of imagination. Unfortunately, it has also increased the breadth of knowledge a modern animator is required to have. Classically trained animators may find it even more difficult to transfer their skills to 3D computer animation. To address this issue, researchers have attempted to combine technology with traditional methods. An example of these attempts is to use physical, modular components to build puppets which are then used to animate 3D models (Jacobson et al. 2014). Another example is leveraging the pencil-and-paper skills of traditional animators.

This chapter looks at the latter: methods that utilise hand-drawn sketches in order to pose or animate 3D models.

Inferring the 3D pose of a character from a drawing is a complex and under-constrained problem. Solving it may help automate various parts of an animation production pipeline such as pre-visualisation. A novel way of inferring the 3D pose from a monocular 2D sketch is also proposed in this chapter. The proposed method does not make any external assumptions about the model, allowing it to be used on different types of characters.

The inference of the 3D pose is formulated as an optimisation problem and a parallel variation of the Particle Swarm Optimisation algorithm called PARAC-LOAPSO is introduced for searching the minimum. Testing in isolation as well as part of a larger scene, the presented method is evaluated by posing a lamp, a horse and a human character. The results show that this method is robust, highly scalable and is able to be extended to various types of models.

## 8.2 BACKGROUND

Since animation has its roots in hand drawings, being able to show one's ideas on paper is a skill most animators possess. As such, many have argued sketching is an intuitive interface for artists from both traditional and modern backgrounds, be it for posing (Mao et al. 2006), animating (Davis et al. 2003), modelling (Yang and Wünsche 2010), stylising motion (Li et al. 2003) or simulating secondary movement (Jain et al. 2012) and crowds (Mao et al. 2007). By providing an intuitive link between the initial sketching process and the end result, the cognitive load and the necessary training for the artist may be significantly reduced.

Initial sketching is already part of most animation pipelines, usually in the form of storyboarding during pre-production. These drawings are then used by animators or pose artists to set up the initial pose layout before going into the animation stage. Rather than introducing a new workflow or step, the aim of the proposed method is to automate one of the existing steps. Specifically, the focus is on making the initial posing stage of a production less labour intensive.

Posing 3D models from 2D sketches is a complex, open problem, closely related to the computer vision problem of inferring a 3D pose from a 2D photograph but with several differences. Photographs are usually accurate in terms of scale and include colour information. Sketches may have squashing

and stretching, usually lack colour and contain noise in the form of auxiliary strokes, e.g. from shading.

This chapter focuses on using sketches to pose 3D characters automatically, for layout or pre-visualisation. It is general enough to pose any character model, in any stage of the pre-production phases, e.g. storyboard blocks or hand-drawn animation frames.

Either in the field of computer vision (Hua et al. 2005) or in previous attempts at automatic 3D posing for animation (Jain et al. 2009), successful attempts have used data-driven approaches. In this case, a data-driven approach is not suitable because a pose database can be challenging and expensive to build and maintain, given the variety of fantastic characters in animation productions. As such, the method proposed in this paper does not use any database.

A hand drawing is processed and split into two descriptors: an edge map and a silhouette. The user quickly pinpoints the joints over the drawing to form the third descriptor. A variation of the Particle Swarm Optimisation (PSO) algorithm called PARAC-LOAPSO (Section 8.3.2) is used, which works by manipulating the pose of a pre-existing 3D model. The process iterates, comparing the 3D render with the drawing (Section 8.3.3) until a suitable 3D pose is found for the character.

### 8.2.1 *Related work*

This section briefly summarises related work from the fields of computer graphics, vision and animation, when it comes to the problem of both inferring 3D data from 2D data as well as the problem of providing a meaningful and useful interface for the process. Mainstream methods, as well as posing interfaces and automatic methods are summarised.

In traditional 2D animation, the artist is drawing the shape of the character while also drawing the movement of the character, conveying very dynamic movement using the 12 principles of animation (Johnston and Thomas 1981). In 3D animation, the shape of the character already exists in the form of a 3D model, and the artist has to deform it using 3D software like Autodesk Maya. Keyframing is the mainstream method of animating 3D characters, by moving controls on a hierarchical structure, called a rig (e.g. joints on a skeleton). By manipulating controls of a skeletal structure, usually joints (Burtnyk and

Wein 1976), an artist can create separate poses to create a sequence. FK (Burtnyk and Wein 1976) allows the artist to control the pose by rotating joints of the rig. While this is more straight forward, it is less intuitive for artists to manipulate angles on joints, compared to moving an end-effector where they want it. IK (Zhao and Badler 1994) solves that as it allows the artist to translate joints (e.g. the wrist) of a character to a position in 3D space, and the rest of the body follows to fit a pose. Even with the invention of IK, posing and animating remains a time consuming process which requires the user to be skilled and trained in specific suites of software.

A 3D rigged model is still the basis for all the methods that are presented in this chapter. The novel aspects of these methods are not in how a 3D character is modeled or rigged, but in how a user can manipulate the character's rig, replacing the mainstream skeleton manipulation process with a sketch-based one.

The first category of sketch-based approaches for 3D animation consists of supervised methods, where a user guides the system through hand-drawn sketches instead of, say, using a mouse. Examples of this sketch-based interface approach attempt to extract animated sequences from mocap databases (Xiao et al. 2015) or make use of differential blending for expressive poses (Öztireli et al. 2013).

The second category of methods does not use hand-drawn sketches as an interactive interface but as a way to automatically infer the 3D pose. This category can be further broken down into two subcategories, the database approach and the generative approach. The database approach uses a database as a knowledge domain, whether it is to position objects in a scene (Xu et al. 2013) or to classify the shape itself (Eitz et al. 2012). The generative approach attempts to pose a pre-existing 3D model until the pose matches the information from the drawing, whether it is to pose animals (Favreau et al. 2006) or humans (Ivekovic et al. 2008).

A drawing is a representation of an internal model that the artist has about a scene and it contains no depth information. As such, the problem of inferring the depth is under-constrained. Nonetheless, another person is able to recognise the pose of the drawn character if they have a similar internal model to that of the artist. However, even for human observers, this internal model may differ.

Automatic 3D posing In general, inferring a 3D pose from a drawing can be broken down into two sub-problems. Firstly, matching the 3D model projection to the drawing. Secondly, inferring the missing depth. For 3D productions, the camera represents the position from which the characters in the drawing are viewed, according to the artist's internal model.

A comprehensive overview of progress in this computer vision problem (Gavrila 1999) explains more about the aforementioned *database* approach and *explicit shape model* approaches, as well as their subcategory, the *model-based* approach.

It is worth noting that unlike the computer vision problem which has data from photographs or video to use as input, the focus in this case is on finding solutions from monocular drawings only.

#### 8.2.1.1 *Database approach*

The database approach uses a set of data in order to create an internal model with which the contents of the drawing are classified. This is done by directly using the data as a comparison to the sketch, or by training a statistical or probabilistic model which may then be used to infer the 3D pose.

Previous computer vision research has used database approaches not only for tracking and posing but also for positioning in a scene or classifying. For example, solutions like Sketch2Scene (Xu et al. 2013) attempt to position objects in 3D scenes by first finding a match of a sketch in a database. Eitz et al. (2012) attempt to classify sketches by developing a bag-of-features sketch representation from a database of 20,000 sketches. For the problem of posing a 3D model, a database of this size is non-trivial to build, since it would also require an actual 3D pose for each corresponding sketch or render. Jain et al. (2009) follow this approach in order to infer the missing depth and pose humanoid characters.

There are some disadvantages to this approach. Building a database of poses may be expensive, which makes it an ineffective solution for smaller budget productions. While motion capture could provide a relatively cheap way of populating databases with human poses, it becomes more difficult for non-humanoids. Fantastic, non-humanoid creatures are so common in animation, meaning that databases would be less scalable, unless already posed models are added to the database.

#### 8.2.1.2 *Explicit shape model approach*

This approach is very effective when multiple camera views are available, because the problem is no longer under-constrained (no occlusions). It is based on the assumption that a 3D model which matches the image already exists. It consists of comparing the 3D render pose to the 2D image in order to estimate how close it is to a correct pose. One of the limitations of this computer vision approach is that it requires a 3D model that is a good match to the 2D image.

This approach has its roots in computer vision, where assuming that a 3D model already exists to match the 2D image can be a problem, if for example the problem is video analysis. However, in 3D productions this limitation does not apply, since a 3D model exists anyway.

When using only a monocular 2D image, it is important to simulate the point of view that the person creating the 2D image had in mind — the equivalent of the viewer's or the camera's position. For the explicit shape model approach, this is usually done by positioning the camera in a 3D space, relative to the character model (Jain et al. 2012).

When it comes to having data from video Favreau et al. (2006) successfully pose animals using Radial Basis Functions of 3D pose examples, and tackle the forward-backward ambiguity with a PCA-based algorithm. The method proposed in this thesis is also concerned with dealing with non-cooperative subjects such as animals, however it does not have the advantage of using video data.

Ivekovic et al. (2008) infer 3D poses of humans in video sequences using multi-view images. Ivekovic et al. still make human-specific assumptions, limiting its use to a small subset of characters while both Favreau et al. and Ivekovic et al. use video data and exploit temporal relations between frames. Temporal data allows for a previous frame to be used in order to initialise the next frame.

More recent work by Bessmeltsev et al. (2016) proposes a method for posing 3D models using 2D gesture drawings, taking only the drawing and the rigged 3D model as input. Moreover, Bessmeltsev et al. (2016) analyze and formulate the pose cues encoded in gesture drawings and use them as a knowledge base for solving this problem.

#### 8.2.1.3 *Model-based approach*

There is a third approach, which is a subcategory of the two aforementioned ones; a collection of data is used to train a model. This allows for direct processing of the given 2D image through the model, without the need to render or calculate anything from the 3D model in real-time. The trained model removes the need for a predefined error model to compare the probability that a given image belongs to a projection from the training data.

The processing is done during the generation of the model. Furthermore, there is no need to store the data locally after the model is trained. These approaches, be it discriminative (Agarwal and Triggs 2004) or generative (Ramanan and Forsyth 2003; Rosales and Sclaroff 1999) attempt to generalise from the training data.

Of course, while this approach can mitigate the processing and memory scalability issues, it still falls under the same limitations that a database approach does: requiring data to exist in the first place.

The rise of neural networks has increased research interest in data-based and model-based approaches with recent work showing great promise. Han et al. (2017) use convolutional neural networks for inferring 3D face models from 2D sketches, allowing non-experts to create posed 3D face models in less than ten minutes. Convolutional neural networks are also used by Mehta et al. (2017) to infer 3D poses of humans from monocular colour image and video.

#### 8.2.1.4 *Features and correspondences*

All approaches assume some way of comparing the 3D model projection, database entry or training data with the drawing. This is done through the description of the pose for the 3D model and the image. Different features can be extracted from an image which can then be used for comparison (Mikolajczyk and Schmid 2005).

Such features include joint angles (Jain et al. 2009) as well as joint angle limits (Difranco and Cham 2001), silhouette (Ren et al. 2004), various shape descriptors for the silhouette e.g. Hu moments, Fourier descriptors, discrete cosine transform, shape context histogram (Poel 2006; Sedai et al. 2011), colour cluster flow and motion flow (Gavrila 1999).

It is more common to calculate such descriptors directly from the silhouette, although it is also common to perform Principal Component Analysis (PCA) (Favreau et al. 2006). Adding more to the difficulty of the problem is the fact that a sketch does not contain as much information (e.g. colour, accurate shape) as a photograph or video frame. An interesting addition to using the silhouette shape would be to make use of internal edge boundaries to decrease the shortcomings of poor image segmentation or inaccurate readings from the other descriptors.

Recent work has provided methods of reducing noise from hand-drawn sketches that could be leveraged for extracting useful information from the image (Liu et al. 2018; Simo-Serra et al. 2018).

#### 8.2.1.5 *Posing interfaces*

Interfaces to replace the mainstream skeleton manipulation process have been the focus of previous research, e.g. tangible devices (Jacobson et al. 2014). Other approaches at building sketch-based interfaces have looked at using colour to label parts of the drawing, essentially formulating a 'sketching language' (Li et al. 2017). Other examples of sketch-based interfaces make use of differential blending (Öztireli et al. 2013) or the line of action to pose 3D characters (Guay et al. 2013). Seki and Igarashi (2017) use partial contour drawings to pose hair for 3D characters. These sketch-based interfaces may offer improvements, but are still operated by artists. The proposed method aims to reduce the process to an automatic method a user with no artistic background would be able to perform.

#### 8.2.1.6 *Common issues*

Posing 3D models from 2D drawings is a complex, open problem which has many commonalities with the original computer vision problem of inferring a 3D pose from 2D images (Gavrila 1999). The problem is therefore under-constrained, as the 2D drawings do not contain sufficient information to extrapolate accurate 3D pose data. The computer vision problem deals primarily with real photographs, while computer animation is full of fantastic creatures. The anatomical limits of imaginary creatures are often unknown, unlike those of humans. Moreover, a sketch can lack colour or contain noise (e.g. in the form of shading). As exaggeration is often used

Figure 30: An example of the forward-backward ambiguity issue in inferred 3D pose of a lamp when not using the internal lines for additional information.

to add dynamism, sketches can be inaccurate in terms of scale. When dealing with photographic or video data, multiple cameras positioned around the subject can be used to provide multiple viewpoints. For example, it is possible to have a room with cameras on all walls, so that when a person steps in, they are viewed from multiple angles. In contrast, a hand-drawn animation frame would only be drawn with one view angle in mind. Finally, the forward-backward or depth ambiguity is a recurring issue. For specific poses, it is difficult to know if a limb is facing forward (toward the camera) or backward (away from the camera). For example, given only the silhouette of a drawn lamp, more than one interpretation exists: the lamp's head could be facing the camera or away from the camera (Figure 30).

### 8.2.2 *State-of-the-Art work*

In this section, two pieces of recent research are singled out and explained in more detail followed by the approach of the author of this thesis. As an example of the sketch-based interface approach, the line of action is used as a user-friendly interface for posing of 3D characters (Guay et al. 2013). Using drawings and minimal user input, a database approach example leverages a set of mocap animation sequences to use as a knowledge base (Jain et al. 2009). Finally, the work in this thesis is an example of the generative approach, as global optimisation is used to generate new poses on every iteration, until the correct pose is found (Gouvatsos et al. 2014b).

#### 8.2.2.1 *Sketch-based interface approach*

The line of action (LOA) is a tool used by animators that helps capture the principal shape and expression of a character in a single line (Figure 31). These lines are C- or S-shaped 2D curves. In their work, Guay et al. (2013)

propose a mathematical definition of the LOA. By formulating and solving an optimisation problem, they are able to align a 3D character's bones to lines of action that are drawn by the user. The user can view a 3D character from different angles and draw curves on top. For example, a user who wanted to bend a 3D character's knee would look at the character from the side and draw a C-shaped curve, indicating a bent leg (Figure 32).

The LOA is used to determine the screen space positions and tangents of a subset of the 3D character's bones, which Guay et al. (2013) call the body line. The body line is defined as a maximal, connected linear chain in a character's skeleton, for example, from head to a foot or from a hand to a foot. As such, ten body lines exist by this definition. The user can manually select the body line they want to manipulate, or they can draw a LOA to select the body line automatically. Once a body line is selected, the best pose is found by solving an energy minimization problem (Equation 1). The difference between the shape of the character's bones in the body line and the shape of the LOA is minimized, using gradient-based local optimisation.

$$\min_{w} \int_{s} E_X(s, w(s)) + E_K(w(s)) + E_C(w(s)) ds$$

$E_X$: closest point objective function

$E_K$: avoid high curvature constraint

$E_C$: maintain connectivity constraint

$s$: body line coordinate

$w(s)$: warping function for line of action coordinate (1)

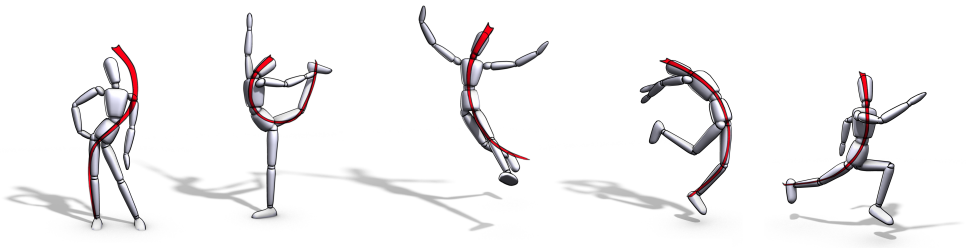


Figure 31: Character poses created by sketching lines of action (Guay et al. 2013).

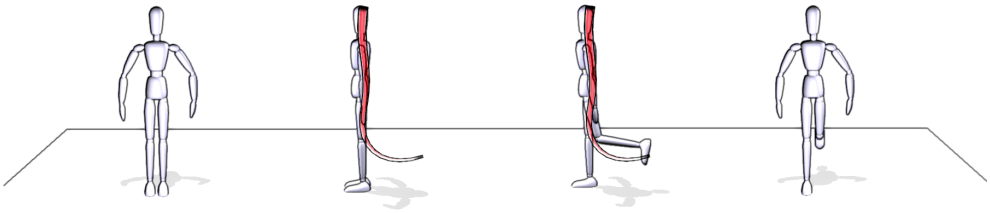


Figure 32: Bone rotations are constrained to the viewing plane. The bones are parameterised as a single axis-angle component. The axis is the camera's viewing direction projected onto the floor plane. From left to right is the initial pose, a side view with the LOA, the result after optimisation, and a frontal view of the final pose (Guay et al. 2013).

The evaluation compares the time it takes to recreate poses with the method by Guay et al. (2013) versus the time it takes to recreate the poses with IK in Autodesk Maya (Table 2). These results suggest that sketching interfaces have the potential to speed up and make more efficient the process of posing and animation. More specifically, the speed up seems to increase as the complexity of task increases.

#### 8.2.2.2 *Database approach for automatic 3D posing and animation*

The work by Jain et al. (2012) introduces an algorithm which takes a set of 2D hand-drawn frames as input. Additionally, a user overlays joints (as dots) on top of the frames and specifies the position of an orthographic camera in 3D space. The algorithm maps the input to a database of mocap information to identify a suitable clip. The clip is projected to 2D and matched to the hand-drawn frames in two ways: timing and exact pose. Finally, the 2D time-aligned and posed result is reconstructed in 3D. Jain et al. (2012) propose a novel pose descriptor to represent 2D poses. This pose descriptor allows for pose comparison while being invariant to translation and scale. Each joint is represented as an angle with respect to a coordinate frame attached to its parent joint. For example, a wrist is represented as the angle between the wrist-elbow bone and a coordinate frame attached to the elbow (Figure 33). Therefore, a given pose P is a vector of polar angles θn:

$$P = [\theta_1,\ \theta_2 \dots \theta_K] \tag{2}$$

where K is the number of joints pinpointed by the user.

| FIGURE | STROKES, ROTATIONS | TIME (LOA) | TIME (MAYA) |
|---|---|---|---|
| Walk | 4 strokes 1 rotation | 20 sec. | 90 sec. |
| S Shape | 3 strokes, 0 rotation | 20 sec. | 120 sec. |
| Hero punched | 6 strokes, 2 rotations | 90 sec. | 150 sec. |
| Hero punch | 7 strokes, 4 rotations | 2 min. | 3 min. 30 sec. |
| Animation | | | |
| Dancers | 6 strokes, 1 rotation | 30 sec. | 2 min. 45 sec. |
| Cartoon swing | 7 strokes, 1 rotation | 45 sec. | 2 min. 30 sec. |
| Muybridge | 34 strokes, 1 rotation | 6 min. | 22 min. |

Table 2: Number of strokes and rotations and time taken using lines of action versus using 3D IK widgets in Maya (Guay et al. 2013).

Jain et al. (2012) break their algorithm down to three steps. Firstly, the mocap sequence is aligned to the timing of the sketches, using the dynamic time warping algorithm (Sakoe and Chiba 1978). Secondly, the time-aligned mocap segment is projected to 2D and made to best match the pose in the sketches. The 3D projection (projecting a 3D space into 2D) is achieved through the user-specified orthographic camera. To match the pose, each joint is rotated in the image plane until its pose descriptor matches the pose descriptor of the sketch. Thirdly, the missing third dimension is inferred. Jain et al. (2012) formulate a linear least squares system to estimate the optimal 3D pose given the time-aligned and fitted 2D pose. This method by Jain et al. (2012) is evaluated by animating a ballet dancer, a happy flower, a stylized sneaky walk, and a sequence of jumping jacks (Figure 34).

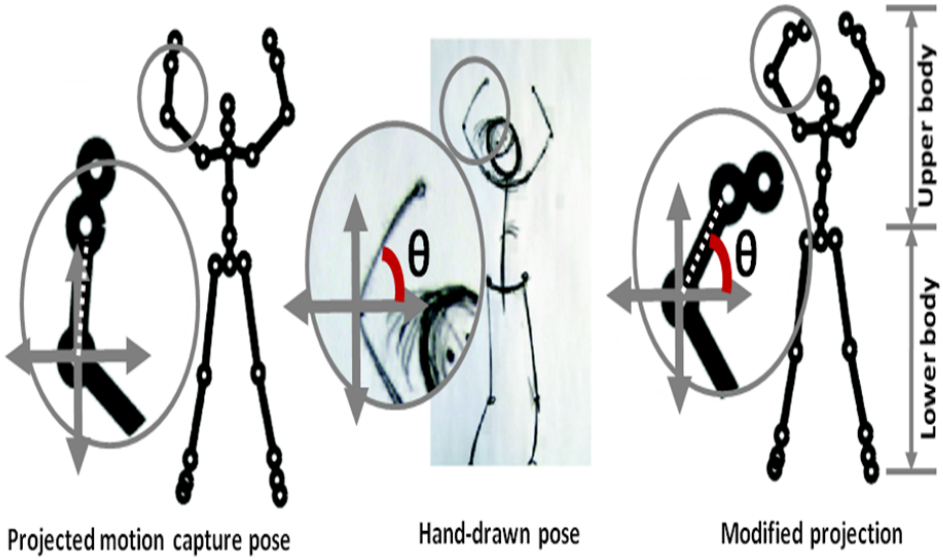



Figure 33: The pose descriptor consists of in-the-image plane angles for every limb segment (Jain et al. 2009).

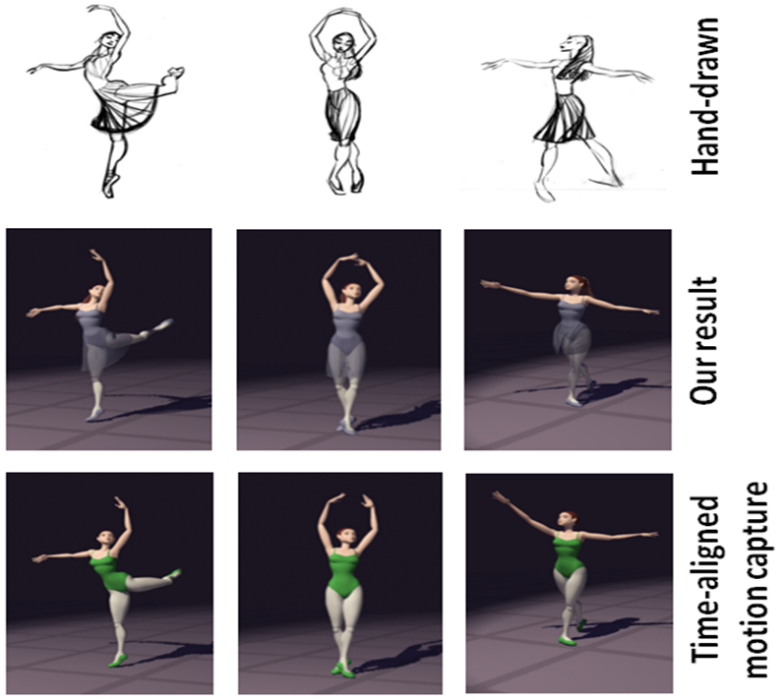



Figure 34: Character poses that match the hand animation much more closely than the mocap animation. 'Our result' refers to the work by Jain et al. (2012).

### 8.2.3 *Proposed Approach*

In order to present a general method for automatically posing 3D models, the proposed method builds upon the works of Jain et al. (2012) and Ivekovic et al. (2008).

The proposed approach is a method for posing 3D models from 2D drawings, aiming at pose layout and pre-visualisation. It makes no assumptions about the 3D model in order to deal with many types of characters. Unlike Jain et al. (2009), it does not rely on a database.

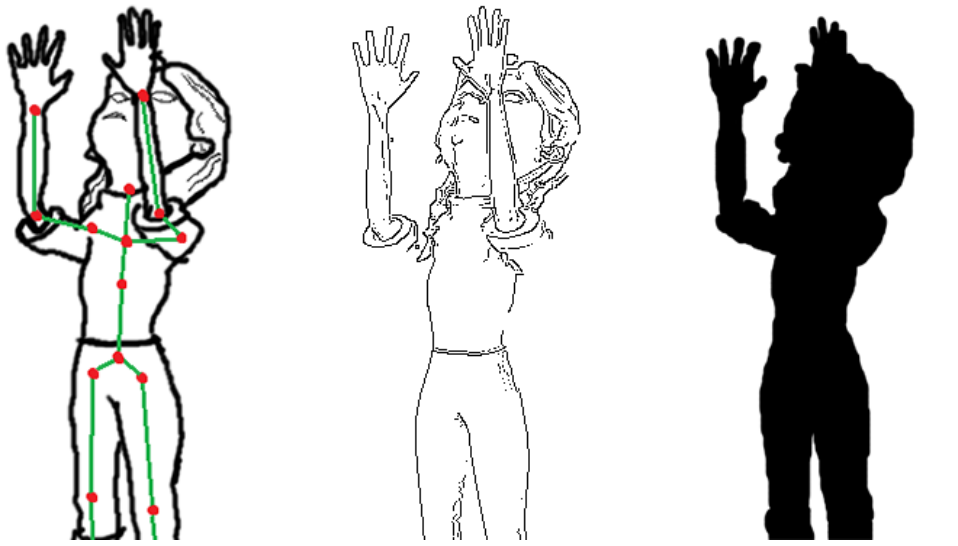
Instead, a PSO variant called Parallel Convergence Local Optima Avoidance PSO (PARAC-LOAPSO) is used. This allows for posing of both cooperative (e.g. humans) and uncooperative (e.g. animals) characters. It does not require the model to have a skeleton as a controller; any type of controller is suitable. Unlike Ivekovic et al. (2008) the input data come from storyboards and are thus monocular, stand-alone drawings with no temporal relations between them.

Minimal user input is required, in the form of pinpointing the joints on top of the drawing. To avoid forward-backward ambiguity, the proposed method uses a combination of various descriptors (Fleischmann et al. 2012). There are currently few other methods (Bessmeltsev et al. 2016) that perform automatic posing for the layout phase of animation that work irrespective of the character model and that can be applied without the need for a pre-existing database.

## 8.3 METHODOLOGY

A camera in 3D space is used to project a 3D pose to 2D. The sketch is compared to a rendered image of the 3D character, using a combination of computer vision descriptors. Different 3D poses are generated and compared until a good match to the sketch is found.

Three descriptors are used to compare drawings to renders. The first comparison descriptor is the position of joints overlaid on top of the sketch (Figure 35a). The second descriptor is an edge map extracted with the Canny algorithm (Canny 1986), which represents the lines, internal and silhouette, of the drawing (Figure 35b). Finally, the third descriptor is a silhouette of the character, which represents the overall shape (Figure 35c).



(a) Joints overlaid by a user on top of a drawing of a human.

(b) Edge map of a drawing of a human, extracted using the Canny algorithm.

(c) Silhouette of a drawing of a human.

Figure 35: Different descriptors for comparing poses in 2D.

The method proposed is to generate new 3D poses and attempt to minimize the difference between them and the sketch. All the possible poses are seen as a search space of solutions. To navigate that search space, a variation of the PSO algorithm is used. The difference between the hand-drawn sketch and the 3D pose is represented by an objective function with three terms, one term for each descriptor.

To improve its global search capabilities, the PSO variation is implemented to run on a graphics processing unit. Moreover, a constriction factor and random variation are added in every iteration, to speed up convergence and avoid local minima, respectively.

This method is evaluated by posing a lamp, a horse and a human.

### 8.3.1 *Overview*

To propose a general posing method which can automate the initial layout pass or pre-visualisation of a production, the solution takes into account several requirements:

- It makes no assumptions about the type of character, in order to be versatile (e.g. humans, horses, lamps, octopuses)
- It provides results that are good enough for the initial posing and not the final animation
- It ensures the necessary user input is quick and easy to learn
- It does not restrict the traditional pipeline structure
- It is feasible for both small and big budget productions

Since the proposed method is focusing on posing and not generating a 3D model from a 2D sketch, it is assumed that there is already a 3D model that corresponds to the character in the drawing. Therefore, the *explicit shape model* approach is utilised for solving the problem.

The 2D sketch is an artistic drawing, e.g. a storyboard panel. It does not need to be of high quality, but it does need to be accurate in terms of the shape and scale of the object that will be posed.

To create a search space containing the desired 3D pose, an interface is used which allows users to overlay drawings on top of a 3D space, moving the camera and placing unposed 3D models. This reduces the search space so that it consists of only local poses of a character, not translation in 3D space (Section 8.3.2). To compare the 2D to the 3D data in the search space the system extracts information from the sketch. That is when the only user input required — the pinpointing of the joints — takes place (Section 8.3.3). To navigate the search space and find the desired pose, the system iteratively poses and renders the 3D model of the character to match the drawing (Section 8.3.2). The rendering, which is a 2D projection of the 3D pose, is used as a comparison to the original 2D drawing.

The process is not disruptive or encumbering on pipelines, as it is no different from a traditional pipeline with a 3D layout pass. More specifically, 3D storyboarding essentially incorporates these steps as part of the process. Additionally, drawings exist early on in pre-production in the form of storyboards. The pinpointing of the joints can be done in a few seconds during the clean-up phase and does not require technical training. More specifically, as someone may go through the storyboard drawings during one of the approval points (Chapter 3) that is when they could go through with a digital

stylus and pin the joints on the drawn character approximately on the correct position. It is important that the correct joint marker is pinned to the correct position on the drawn character, but the positioning itself does not have to be accurate.

### 8.3.2 *PARAC-LOAPSO*

As previously mentioned, the problem of inferring the 3D pose from 2D data is treated as a minimisation problem, by navigating a formulated search space using a variation of PSO called PARAC-LOAPSO.

The original PSO algorithm (Figure 36) was inspired by the social behaviour of animals who work together in order to explore and find desirable positions in a larger area; the classic example is a flock of birds flying over a landscape looking for food, communicating their findings to each other in order to converge on the position with the most food.

PSO as an optimisation algorithm performs global search and a reason for choosing it as the foundation of PARAC-LOAPSO is that it is parallel by nature, which makes it straight-forward to implement on a GPU (Mussi et al. 2010). As a meta-heuristic algorithm, it makes no assumptions about the problem, which means it can be general enough to be applied on different models or model controls (e.g. joints). This is important in order to meet the requirement for a general solution, as explained in Section 8.3.1.

The algorithm consists of a swarm of particles navigating a multidimensional search space looking for solutions that get a lower value when evaluated through a fitness function. In the swarm, each particle represents a possible pose. Therefore, the search space is defined as the set of all possible poses. For more complex models (i.e. models with a greater number of joints) the search space would be larger as the number of possible poses is greater.

Even for less complex models the search space of possible poses can be large, even after reducing it by making assumptions. Complicating things further, the problem of pose estimation from 2D drawings does not necessarily have just one perfect solution when formulated as a minimisation problem, as the drawing may not map perfectly to the 3D asset. As such, the aim is to find a solution that is good enough. Local search methods would identify one of the many local minima as the best, but global search methods

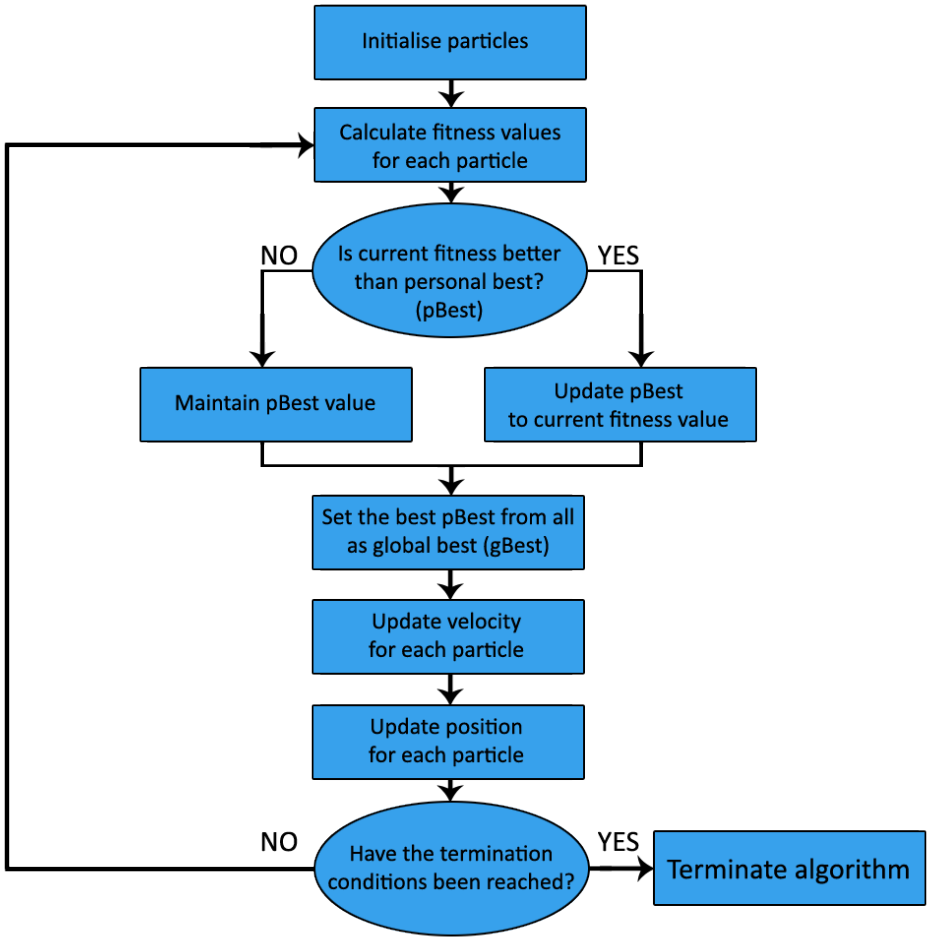



Figure 36: Flow chart of original Particle Swarm Optimisation algorithm.

may be able to perform better given the multimodal nature of the problem, meaning they are able to find poses which are closer to the optimal solution.

The problem is formulated by defining each particle as a possible pose or *solution*. Specifically, each solution is a vector $X = (j_1, j_2 \dots j_n)$, where $n$ is the number of joints to manipulate on the model. It is possible to use an IK skeleton setup instead of rotating the joints, however using translation instead of rotation, which is how IK works, makes it difficult to constrain the search space. As such, each element $j$ represents a joint and is a three dimensional vector $j = (x, y, z)$ where $x$, $y$ and $z$ are its Euler angle rotations in local space. Therefore, the search space contains all the possible poses for that model.

While issues of uncertain convergence, speed and getting stuck at local minima have been explored (Bratton and Kennedy 2007), it is important to re-examine them for this particular problem.

#### 8.3.2.1 *Improved Search*

To deal with slower convergence, a constriction factor K (Equation 3) (Bratton and Kennedy 2007) is preferred over the inertia weight used by Ivekovic et al. (2008) or Salehizadeh et al. (2009). While both methods aim to limit the feasible search area of the swarm, the constriction factor is actually a special case of the inertia weight formula, guaranteed to reach convergence due to its proven stability (Clerc and Kennedy 2002). To speed up convergence and overall runtime, the algorithm is implemented to run in parallel on the GPU. To deal with getting stuck at local minima, the method proposed by Salehizadeh et al. (2009) is used to add variation to the population (Equations 6 and 7). Additionally, the optimisation process is faster and more accurate as a result of navigating a smaller search space. This is achieved by using joint rotation limits from the given 3D model as constraints. Instead of using biological data about human joint limits, only the joint limit data from the 3D model are acquired. Previous work has used real-world anatomical data to recreate physically-plausible animation systems (Kadleček et al. 2016). For the project presented in this thesis, any hierarchical assumptions would decrease the method's versatility, as it would depend on real-world data about humans and animals. Moreover, it would not be easy to match real-world data to fantastical creatures, which are common in animation. The joint limit data is set by the artist who rigs the 3D model for animation. In Autodesk Maya it is defined as a minimum and maximum euler angle rotation value for each axis (X, Y and Z) of each joint or controller.

$$
\begin{aligned}
K &= \frac{2}{|2 - a - \sqrt{a^2 - 4a}|} \\
a &= \phi_1 + \phi_2 \\
\phi_1 &: \text{cognitive influence (by own findings)} \\
\phi_2 &: \text{social influence (by others' findings)}
\end{aligned}
\tag{3}
$$

For a particle $i$, the cognitive influence term $\phi_1$ is defined as the weighting of the particle's personal best position $B_i^t$ in the calculation of its velocity (Equation 6). The social influence term $\phi_2$ is defined as the weighting of a particle's global best position $B_g^t$ from all particles. A value of $a > 4$ guarantees the algorithm's convergence (Bratton and Kennedy 2007).

These terms in turn affect the algorithm's balance between 'exploration', meaning spreading the swarm's particles to cover more of the search space, and 'exploitation', meaning focusing the particles around the same area of the search space. In other words, when the standard deviation of the particle values is greater, the algorithm is exploring more. The balance between exploration and exploitation is based on the idea that for each particle $i$, larger changes in its position $X_i$ are preferred earlier during the optimisation process, in order to examine larger areas of the search space. Towards the end of the optimisation, smaller changes in $X_i$ are preferred in order to refine the solution as much as possible.

When it comes to improving search and ensuring convergence, another important parameter is the communication topology or neighbourhood of the swarm. There are two types of topologies, the global and the local. The global topology means all particles exchange information with all other particles and can update the global best solution found at each iteration step. The local topology means any non-global topology. In this thesis, local topology refers to the ring topology, where each particle has two other particles in its neighbourhood that it influences and can be influenced by (Figure 37).

The global topology converges faster than the local topology, since all particles are affected by the best particle from the beginning and this is often why it is preferred by researchers. It is important to ensure convergence of the swarm, but if the particles are converging towards a local minimum and not the global minimum, then this quick convergence results in a suboptimal solution. For specific problem sets it has been shown that the additional time it takes for the local topology to converge also results in a solution closer to the global minimum (Bratton and Kennedy 2007).

When the search space is simple, like in unimodal problems, Bratton and Kennedy (2007) found that the global topology is better than any local topologies. However, when it comes to multimodal problems, they suggest that a local topology might be better in order to avoid local minima because it spends more time exploring the search space.

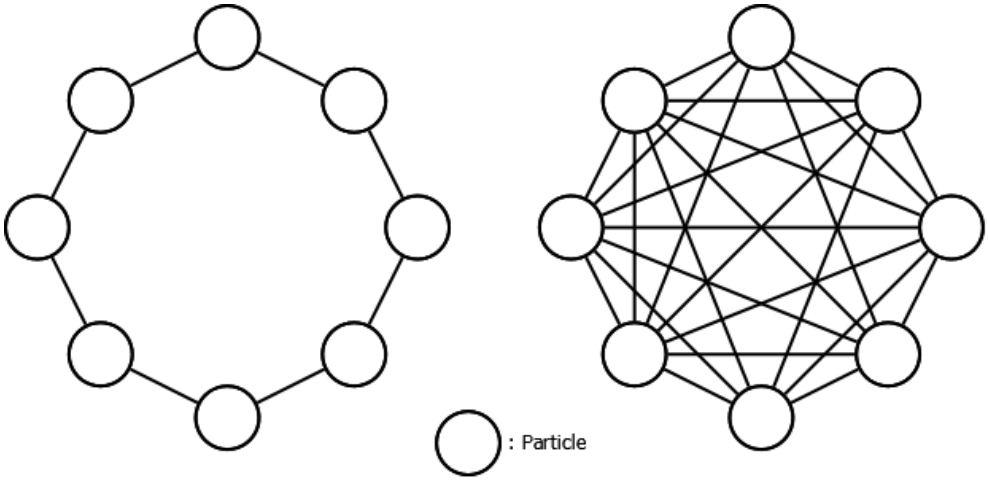



Figure 37: The ring local topology of a particle swarm (on the left) allows each particle to communicate with two other particles, while the global topology (on the right) allows particles to exchange information with all other particles.

Even for problems where the local topology is better, the optimisation needs to run for more epochs in order for the improved results to show, due to its slower rate of convergence. In this case, the word epoch is used to describe an iteration of the optimisation algorithm. Depending on how many operations occur per epoch, this can take its toll on the runtime of the algorithm, depending on the swarm population. Bratton and Kennedy recommend that both topologies are always tested.

#### 8.3.2.2 *Initialisation*

The overall performance of the algorithm is affected by its initialisation. Since no temporal data is assumed, as would be in the case of a video, a previous pose cannot be used as an initial point. Therefore, each element $j_n$ of position $X_i$ of each particle $i$ is initialised from a uniform random distribution: $j_n \sim U(X_{min}, X_{max})$. The upper and lower boundaries $X_{min}, X_{max} \in [-360, 360]$ differ for each element of $X_i$, since the joint rotation limits, as described earlier in this chapter, are used to set them for each particle element.

Given the link between the drawing and the 3D scene, an alternative initialisation approach for the particles would be to use IK handles to match the approximate position of the joints as pinpointed on the sketch. Then the swarm would be initialised based on a random distribution centred on the

IK solution pose. It was decided to not use this approach because it relies on the assumption that the pre-existing 3D model contains IK handles, which in turn would make the proposed method less general in application.

In cases where temporal data exists, like for example in an animation sequence where each frame follows logically from the previous one, the system can easily be configured to initialise each particle randomly in a distribution centred around the previous frame. For example, given a successfully posed model for a frame at the beginning of the sequence, the frame which follows it will have a comparatively similar pose. As such, automatically posing models in an animation sequence with frequent key-framing contains more information about the problem and may improve due to better initialisation. It is also possible to create a second optimisation pass, after the first, where each estimated pose is further corrected based on the previous and the subsequent frames.

#### 8.3.2.3 *Fitness evaluation*

After the initialisation step a fitness function $f$ allows for the evaluation of each particle. Based on this, the personal best position $B_i^t$ of each particle $i$ and the global best position $B_g^t$ from all particles are stored.

$$\begin{aligned} f &= w_1 D_j + w_2 D_e + w_3 D_s \\ &D_j\text{: distance metric} \\ &\text{for joint positions (Equation 9)} \\ &D_e\text{: distance metric} \\ &\text{for edges (both internal and external) (Equation 10)} \\ &D_s\text{: distance metric} \\ &\text{for silhouette (Equation 11)} \\ &w_1, w_2, w_3\text{: weights for each} \\ &\text{distance metric } (\frac{1}{3} \text{ for equal weighting of all parts}). \end{aligned} \tag{4}$$

#### 8.3.2.4 *Particle update*

At the beginning of epoch $t$, the population of $N$ particles is split into two subgroups of size $\gamma N$ and $(1-\gamma)N$, with random variables $\gamma = r$ for random

variable $r_2 \sim U(0,1)$ if $t < T_c t_{max}$, or 1 otherwise. $T_c$ is denoted as the threshold of convergence, which controls how many epochs are dedicated to exploration and how many to exploitation. For example, for $T_c = \frac{3}{4}$ only the last quarter of the epochs will be dedicated to exploitation. $t_{max}$ is the maximum number of epochs for a run.

$$X_i^{t+1} = X_i^t + V_i^{t+1} \tag{5}$$

For every particle $i$, its position $X_i$ is updated (Equation 5). If the particle belongs to the first population subgroup, its velocity component $V_i^{t+1}$ is updated using Equation 6. Otherwise, to attempt and avoid local minima by adding variation, it is updated using Equation 7.

$$V_i^{t+1} = K[V_i^t + c_1(B_i^t - X_i^{t+1}) + c_2(B_g^t - X_i^{t+1})] \tag{6}$$

$$\begin{aligned} V_i^{t+1} &= K\{V_i^t - L^t[r_1(B_i^t - X_i^{t+1}) + r_2(B_g^t - X_i^{t+1})]\} \\ L^t &= 2 - \frac{2t}{t_{max}} \end{aligned} \tag{7}$$

$$\begin{aligned} t &= [t_1, t_2 \dots t_{max}] \text{ for } t_{max} \text{ the maximum number of epochs} \\ B_i^t &\text{: personal best solution} \\ B_g^t &\text{: global best solution} \\ r_1, r_2 &\sim U(0,1) \text{ for uniform distribution } U(0,1) \\ c_1 &= \phi_1 r_1 \\ c_2 &= \phi_2 r_2 \end{aligned} \tag{8}$$

#### 8.3.2.5 *Overall algorithm*

The above components are combined in the following steps to form the overall algorithm:

1. Initialise the swarm population, with global best $B_g^t$ and/or neighbourhood bests (depending on topology)

2. For each particle $i$, with personal best $B_i^t$ (in parallel):
   a) Update particle position (Equation 5)
   b) Update descriptors of 3D pose (Section 8.3.3)
   c) Evaluate fitness of particle (Equation 4)
   d) If current fitness < $B_i^t$, set $B_i^t$ to current fitness
   e) If current fitness < $B_g^t$, set $B_g^t$ to current fitness

### 8.3.3 *Comparing Drawings to Renders*

The main issue when trying to infer a 3D pose from a 2D drawing or image, is that the problem is under-constrained: there is not enough information in the 2D drawing to get a perfect estimation of the pose. The results show that a combination of various descriptors is able to provide enough information to infer the 3D poses.

Three ways to compare the drawing with the 3D render were chosen given their previous reported advantages and disadvantages. Joints can be used in order to find close poses from a database (Jain et al. 2009). In order to make use of the internal edges, edge maps are simple, fast and as effective (Tresadern and Reid 2007) as more expensive methods like Shape Context Histograms (Agarwal and Triggs 2006). The Shape Context Histogram method was initially tested in this project, however it was not used for further experiments because each run was taking over 10 hours. Finally, silhouettes have been used to find poses generatively (Ivekovic et al. 2008) or to train models (Agarwal and Triggs 2004).

#### 8.3.3.1 *Joints*

The user overlays the positions of the joints on top of the drawing (Figure 35a). The position of the overlaid joints on the drawing is compared to that of the 3D positions of the joints, transformed into screen space coordinates. This process differs from the work from Jain et al. (2009) in that it is not contextual. This is because the orientation of the pose has to match the drawing in order for the other two comparison methods to work.

The metric for comparing joint positions between a drawing and a projection, defined as $D_j$, is the sum of the Euclidean distance $d(p, q)$ between

all joints in a drawing p and their respective joints in the render q (e.g. the wrist in the drawing and the wrist of the rendered 3D model) (Equation 9). The Euclidean distance is normalised by dividing the Euclidean distance by the image diagonal $diag(W, H)$. The sum for all joints is also normalised by dividing by the total number of joints, in order to yield a value between 0 and 1, where 0 means the joints are matching perfectly.

$$
\begin{aligned}
D_j &= \frac{1}{n} \sum_{i}^{n} \frac{d(\zeta_{j_i}, \psi_{j_i})}{\mu(W, H)} \\
j_i &= [j_1,\ j_2 \ldots j_n] \text{ for } n \text{ joints} \\
\\
\zeta_{j_i} &: \text{position of } j_i \text{ in drawing as 2D point} \\
\psi_{j_i} &: \text{position of } j_i \text{ in render as 2D point} \\
\\
d(\zeta_{j_i}, \psi_{j_i}) &= \sqrt{(\psi_{j_{i_x}} - \zeta_{j_{i_x}})^2 + (\psi_{j_{i_y}} - \zeta_{j_{i_y}})^2} \\
\\
\mu(W, H) &= \sqrt{W^2 + H^2} \\
W &: \text{image pixel width} \\
H &: \text{image pixel height}
\end{aligned}
\tag{9}
$$

#### 8.3.3.2 *Edge map*

To use internal lines in the drawing, the Canny algorithm (Canny 1986) implementation of OpenCV 2.4.8 is used to extract a binary map of the external and internal edges (Figure 35b). For the results presented in this thesis, values equal to 50.0 and 60.0 were used for the first and the second hysteresis threshold respectively and a kernel equal to 3 for the Sobel operator. Then the extracted binary map of the drawing is compared with that of the rendered image.

For a rendered potential pose, a number of points are sampled from its edge map. Then for each sampled point it is checked whether the same point exists in the edge map of the drawing. For the results presented in

this thesis, 256 points were sampled. Therefore for a list of sampled points $S$ where $|S| = 256$, the edge map difference or distance $D_e$ is calculated:

$$D_e = 1 - \left(\frac{1}{|S|}\sum_{i}^{|S|} p_i\right) \tag{10}$$

$p_i$: edge map value at $i_{th}$ pixel

By dividing by $S$, the value of this calculation is normalised to be between 0 and 1, where 0 is a perfect match. It is worth noting that this metric favours poses where the 3D character makes itself as small as possible or to lean on the side with most edge information. The reason for this is that if the 3D character makes itself as small as possible, its edges will come closer and therefore there will be a higher chance of points in $S$ to overlay with other points. Therefore this metric can give inappropriate results if used on its own.

#### 8.3.3.3 *Silhouette*

Finally, for the overall shape, the silhouette of the drawing is used as a binary map (Figure 35c). The *silhouette distance* between the rendered image and the drawing can be calculated (Fleischmann et al. 2012) (Figure 38).

$$D_s = 1 - \left(\frac{1}{2}\frac{U}{U+D} + \frac{1}{2}\frac{U}{U+R}\right)$$
$$U = D \cup R \tag{11}$$

D: silhouette from drawn image

R: silhouette from rendered image

Similarly to the edge map comparison, it favours poses where the 3D model makes itself as big as possible, in order to fit into the silhouette. The reason for this is that by making itself as big as possible it maximises the term $U$ (Equation 11).

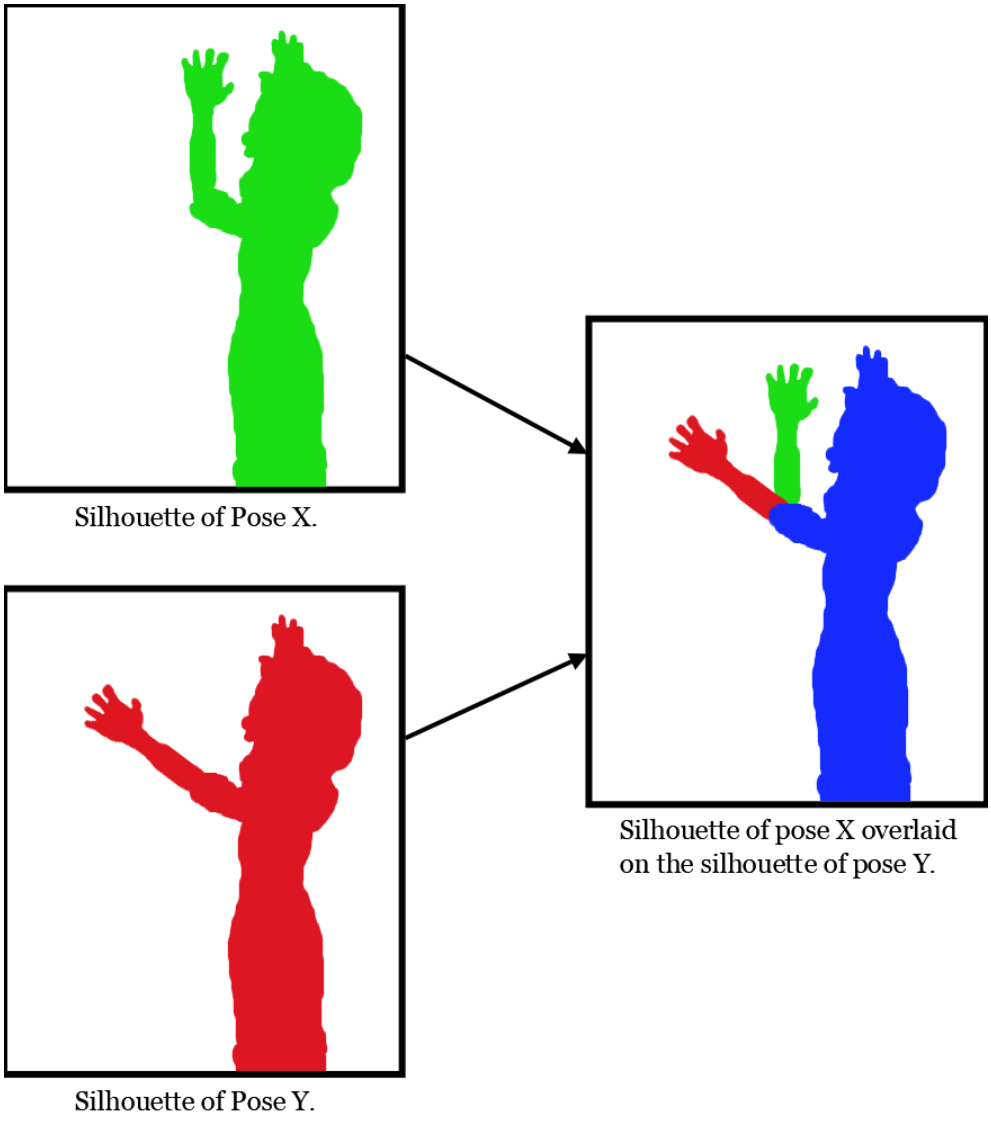



Figure 38: Example of silhouette comparison to calculate similarity between two poses X and Y. Green colour signifies the silhouette of pose X, red colour the silhouette of pose Y and blue colour the overlap between the two poses.

## 8.4 RESULTS

The system was tested on three problems of different dimensionality and difficulty: a horse (Figure 41), a human (Figure 42) and a lamp (Figure 43). The subjects were chosen in order to evaluate whether the proposed method is suitable irrespective of the problem. The criteria for choosing them were vari-

ation in terms of joint count and degrees of freedom, type (innanimate object, biped, quadruped) and cooperation. The lamp and the horse models would be difficult to be posed via methods such as motion capture, although methods that explore this exist (Chapter 7), especially the horse. Most motion capture solutions focus on humanoids, but since they are often not affordable for smaller studios, posing a human is an important test. The resulting poses are visually compared to the drawings, with the criteria being the pose as seen from the specific camera view.

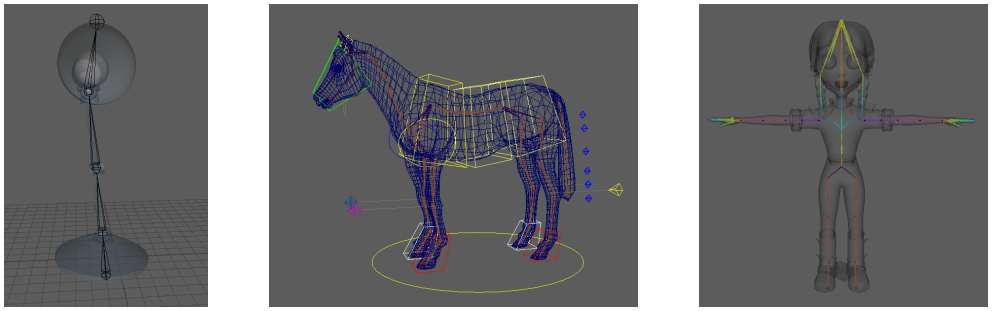


Figure 39: The 3D assets of the lamp, horse and human shown with their joint hierarchies and controllers in Autodesk Maya.

All three 3D models are setup as kinematic hierarchical chains of joints manipulated through rotation. The lamp has 6 controllers (5 joints and a root controller), the human 20 joints and the horse over 25 joints. The horse was however manipulated through 10 controllers set in Autodesk Maya (Figure 39). All drawn images (and as such, the corresponding renders) for the lamp results were 720 pixels by 405 pixels. All drawn images for the human results were 512 pixels by 512 pixels. All drawn images for the horse results were 640 pixels by 480 pixels. These sizes were arbitrary depending on the shape of the drawing.

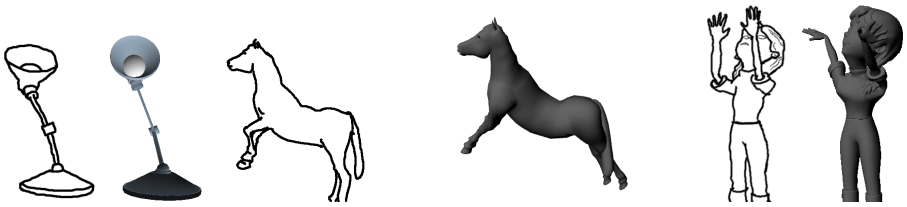


Figure 40: Side by side comparison of the drawings and less successfully estimated 3D poses (bottom).

Apart from posing models in isolation, the proposed method is evaluated by posing a lamp within a scene using an existing storyboard drawing (Figure 44), in order to test the feasibility of the proposed method for posing models within a complex scene. Moreover, the method is also tested in an animation sequence of a lamp, where temporal coherence between the frames exists (Figure 45). The test was designed to analyse possible discontinuities created in the sequence if the posing is inaccurate.

In an actual drawing, such as a storyboard, multiple characters may appear and may even overlap one another. In a complex drawing where multiple characters may be overlapping, separating their individual lines constitutes an additional problem. The system needs to calculate metrics as described in Section 8.3.3 for each character separately, thus making character segmentation necessary.

In order to do this, two approaches were considered. The first approach was to perform the drawing process with layers, where the lines of each character are in a separate layer. The second approach was to add an additional 'highlighting' stage which is performed together with the pinpointing of joints. This allows the user to quickly trace the outline of the character to be posed.

With the first approach of drawing in different layers, the artist who performs the creative task of drawing is encumbered by a different workflow. This is not the case with the second approach, as both the pinpointing of joints and tracing of character outline is performed by a non-skilled user in a separate pass. Since the aim is to reduce the time skilled artists spend on non-creative tasks, the second approach was preferred and thus employed for these tests.

For all the tests, the algorithm was run for $t_{max} = 1000$ epochs. The values $\phi_1$ and $\phi_2$ as defined in Equation 3, were set to a value of 2.05, which is the default value in the canonical PSO (Bratton and Kennedy 2007) and ensures convergence of the swarm. The minimum and maximum velocity of each particle were set to $-360.0$ and 360.0 respectively, since this is the space of possible rotations. This means that in one epoch, a joint can rotate at most by one complete rotation either clockwise or counter-clockwise. The minimum and maximum position of each particle was set dynamically based on the angle rotation limits of each joint of the model, reducing the search space of possible poses. The angle rotation limits are set as part of the model creation

process by the 3D artist. The weights $w1$, $w2$ and $w3$ were set to $\frac{1}{3}$ where each metric contributed equally. The exploration phase (Section 8.3.2), was set to $T_c = \frac{3}{4}$, so the first $t_{max}T_c = 750$ epochs were dedicated to exploration and the last quarter of epochs was focused on exploitation. These parameters were chosen in order to have most of the run exploring and only the last run exploiting, in order to converge.

A runtime comparison of the algorithm on different hardware can be seen in Table 3. All tests except the ones under the 'Discrete GPU' column were performed on a system with 8GB of RAM, an Intel HD Graphics 3000 GPU and an Intel Core i5 2520M 2.50GHz CPU, unless stated otherwise. A run of 1000 epochs with a swarm size of 50 particles took approximately 81 seconds on average. The runtime tests used all three methods of comparing the drawing to the 3D render but with varying population sizes to examine the scalability of the algorithm. The discrete GPU tests were performed on a system with 8GB of RAM, an Intel Core i5 2.5GHz CPU and an NVIDIA GeForce GTX 260 GPU.

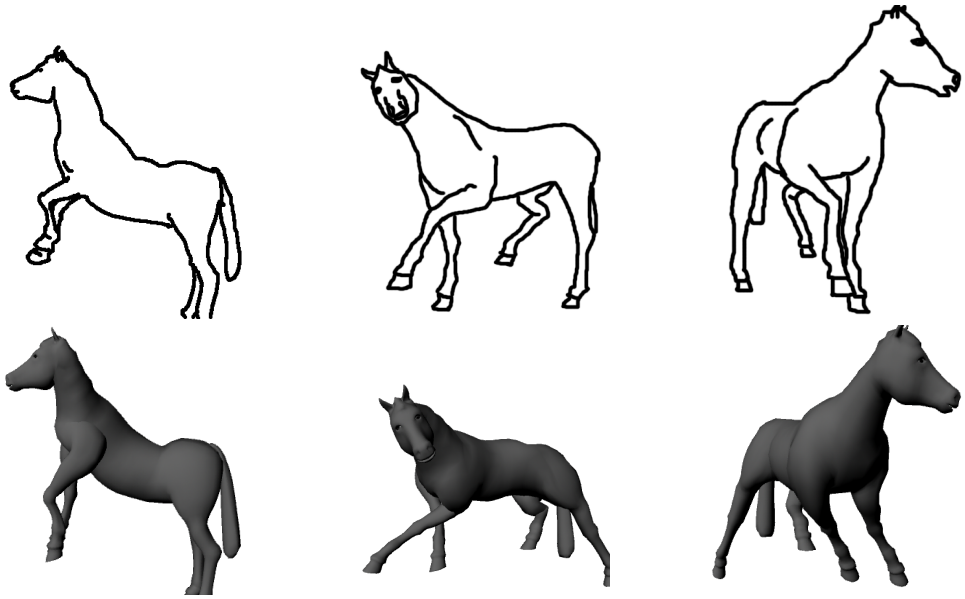


Figure 41: Side by side comparison of the drawings (top) and the estimated 3D pose (bottom) for the horse model.

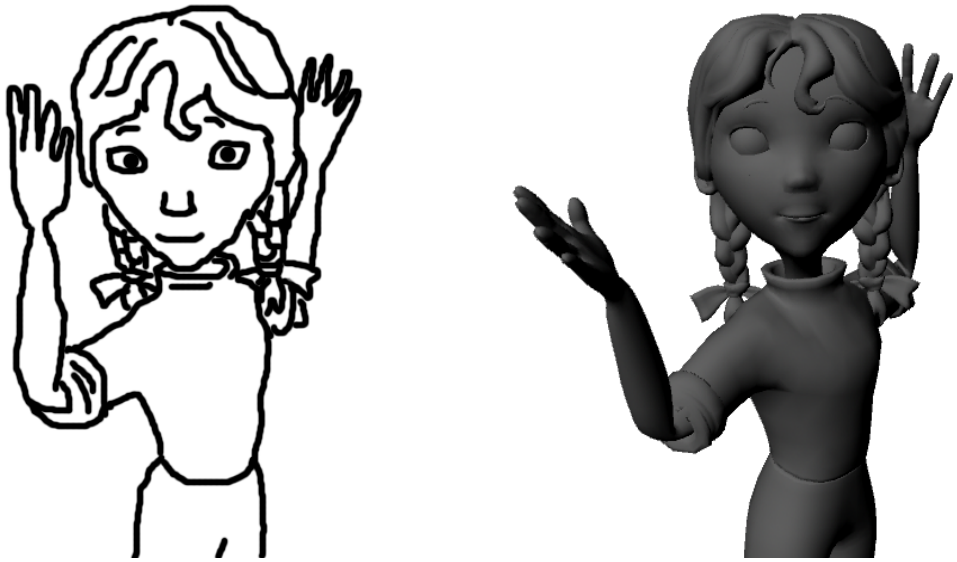


Figure 42: Side by side comparison of a drawing (left) and the estimated 3D pose (right) for the human model. The estimated 3D pose is close to the drawing.

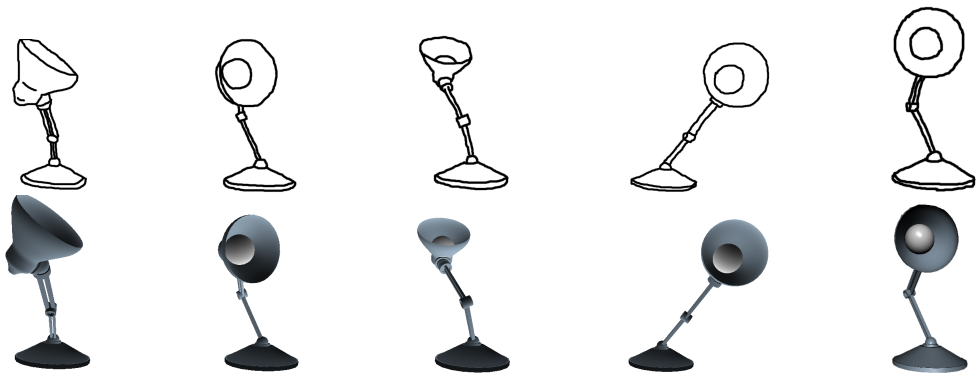


Figure 43: Side by side comparison of the drawings (top) and the estimated 3D pose (bottom) for the lamp model.

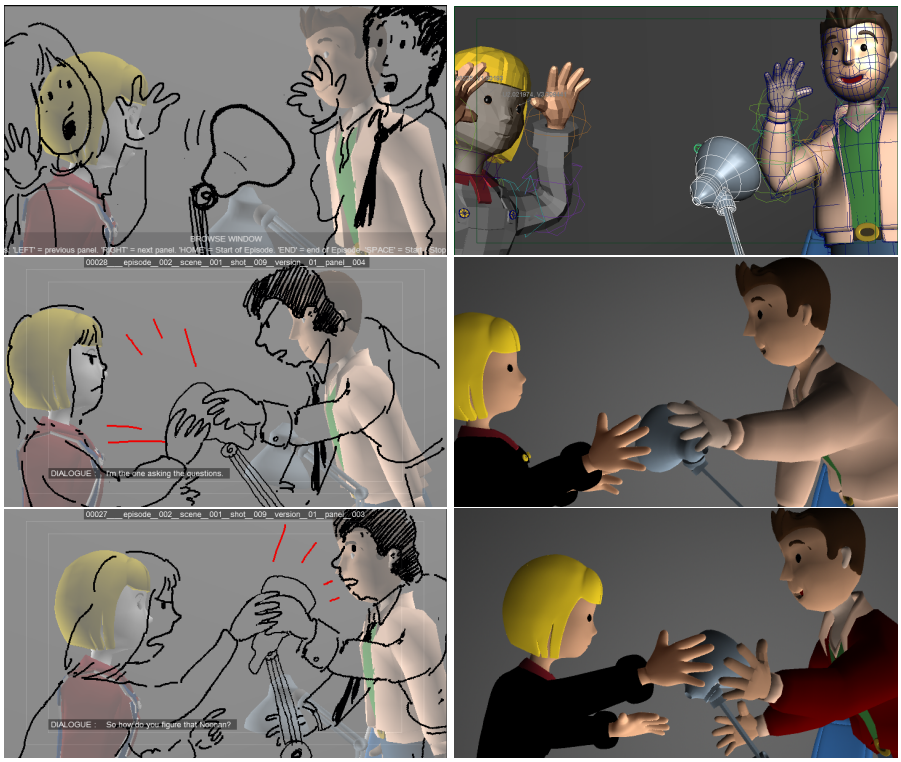



Figure 44: From initial storyboard to pre-visualisation. The lamp in these three shots is automatically posed using the storyboard drawing. On the left column, drawings are overlaid on top of the 3D scene with the assets unposed. On the right column, the lamp has been posed automatically using the proposed method.

Figure 45: Side by side comparison of a sequence of drawings (top) with temporal relationships and a sequence from the estimated 3D poses (bottom) for the lamp model. Lighter opacity and colours signify frames earlier in the sequence while darker opacity and colours signify frames later in the sequence.

## 8.5 DISCUSSION & FUTURE WORK

Guay et al. (2013) present an interesting interface that allows beginners to quickly and intuitively pose 3D characters. However, more detailed, complicated poses are not possible to reconstruct, as the LOA is limited to C- and S-shaped curves. Jain et al. (2012) present an automatic posing method that produces impressive results, but the dependence on a mocap database means it is not always an option. Building a database can be expensive, especially for non-humanoid animation.

An automatic posing method is proposed that does not require a database and can be applied to a range of different characters. However, it takes longer to run than both other methods and may have less accurate results. While this could be an issue for a production environment, the trade-off is that it can run in the background, as 3D layout generation from storyboards is not something that needs to be interactive. Moreover, it is more general than other methods.

The proposed method has been evaluated by posing a lamp, a horse and a human from both simple and complex drawings as part of an actual production scene. It has also been tested on a sequence of drawings. The results show that the proposed method is a general approach that is able to perform accurately on different problems and is not constrained to one type of character.

However, it is this flexibility which leads to the main disadvantage of this method. Since the approach is generative, it requires a rendering step in every iteration, which may mean the time taken to return a result is longer. It is important to note that this method does not require the ongoing feedback of an artist, like for example other sketching interface methods, meaning that this method is still able to contribute to reducing costs even if it takes longer to return a result. Furthermore, implementing the method on a GPU reduces the effects of this disadvantage significantly.

The runtime tests (Table 3) show that PARAC-LOAPSO scales very well on better hardware. It is worth noting that a GPU implementation is much faster compared to a threaded CPU implementation. Moreover, while a swarm with more than 100 particles is very impractical to run on the CPU, a GPU can handle it and return results in under 15 minutes, even on onboard GPU hardware. When using a discrete GPU, the runtimes are approximately

halved. These measurements show a trend which suggests that the algorithm can scale very gracefully and return results faster as hardware with parallel capabilities improves. As such, while it may be unsuitable to run on individual artist machines as part of a workflow with multiple iterations, it would scale greatly as a hand-off step in the pipeline (e.g. from storyboarding to 3D layout), as it could be run to a render farm.

As the 3D model complexity increases, the search space increases too. This makes each particle, which is formulated as a vector, contain more values and as such increases the memory used. As the complexity of the drawing increases, especially in terms of resolution, the metric calculation times per particle per iteration would increase too. It is due to the fact that the metric and comparison calculations are done per particle per iteration, that parallelising the algorithm yields improved performance results.

Since the PARAC-LOAPSO algorithm is stochastic, result accuracy may vary between runs (Figure 40). This inconsistency is due to the stochastic element which may lead to the algorithm being stuck on a local minimum on one run, while overcoming it on another. However, by running for longer and using a larger population of particles, the search space would be explored more consistently. A system with a powerful graphics card can normalise the results through the use of a large population of particles, which helps both in searching as well as in having a more varied initialisation.

The canonical PSO recommends a population of 50 particles (Bratton and Kennedy 2007). Swarms of 5, 15, 25 and 50 particles were used early on during testing and the best results as presented in this thesis were output by the swarm of 50 particles. This hints that larger populations may perform better when it comes to the problem of inferring 3D poses from 2D drawings.

Having tested the runtimes of PARAC-LOAPSO on different hardware, the results are encouraging in terms of running tests for thousands or more epochs with swarm populations in the thousands and local topologies, while modern GPUs handle the computational downsides that would normally arise from these parameters. In effect, the scaling nature of the algorithm means that better hardware will allow for better results.

A swarm with global topology was preferred for these runtime tests to ensure timely convergence of the swarm, since all tests were limited to 1000 epochs. However, when the optimisation ran for longer than 1000 epochs, the local topology produced better results. This difference in performance

was more pronounced for complex models like the human (Figure 42). The reason for this is that the local topology explores the search space more, which is larger for complex models, while given time to reach an optimum despite its slower rate of convergence.

It is important to note that this method does not aim to produce final animated output. It aims at automating the initial pose layout phase or previsualisation, which is currently performed manually by skilled operators. Depending on the complexity of the pose, a skilled operator may be faster than the proposed method, but can still take minutes for each pose.

The simple lamp model was successfully posed in most tests, as was the horse. However, the estimated pose for the more complex human model was not always as accurate (Figure 40). It is likely that for more complex models, better results may be obtained by running the optimisation for more epochs, with a larger swarm and using a local topology. This was not tried and would be an interesting experiment to explore as future work. For both simple and more complex models, a main reason for sub-optimal results was the stochastic nature of the algorithm, as discussed earlier in this section. Moreover, poses are always estimated based on the given viewing angle. Observing the newly posed 3D character from a different angle may in some cases show that the estimated pose is improbable.

The choice of method for comparing drawings to renders may have a notable effect on the end result. For example, by not using the internal lines of a drawing, the forward-backward ambiguity problem can become an issue as seen in Figure 30. On the other hand, by only using joint locations or the silhouette, runtimes are shorter. This is because the method for calculating the internal lines is computationally more expensive, at least in the current implementation. Future work can look into introducing additional comparison methods, like the internal lines, during the exploitation part of the runtime. In this way, overall runtimes may be improved as less expensive methods can be used for most iterations.

An advantage of this approach is that it does not require changes in the pipeline to accommodate it. Minimal user input is still required to pinpoint the joints on the drawing, which takes a few seconds for each pose and can be performed by an unskilled operator.

Analyzing existing approaches side by side with the approach proposed in this thesis provides insight into patterns that can be used for successful

| PARTICLES | CPU | GPU | NON-PARALLEL CPU | DISCRETE GPU |
|---|---|---|---|---|
| 1 | 2m 13s | 20s | 2m 8s | 10s |
| 50 | 2h 30m | 1m 21s | > 4h | 1m 3s |
| 100 | > 4h | 2m 15s | > 4h | 1m 15s |
| 500 | > 4h | 6m 44s | > 4h | 3m 31s |
| 1000 | > 4h | 12m 37s | > 4h | 6m 18s |

Table 3: Runtime comparison for various PARAC-LOAPSO swarm sizes, on different hardware as well as a non-parallel implementation on the CPU. CPU: Intel i5–2520M 2.50Ghz, GPU: Intel HD Graphics 3000, Discrete GPU: NVIDIA GeForce GTX 260.

sketch-based 3D posing. Assuming one is able to formulate a representation for the poses, it is still necessary to match a 3D posed model to the 2D sketch and infer the missing depth. To compare a 3D pose to the 2D sketch, both need to be in the same space. As such, the 3D pose can be reduced to 2D through a camera (e.g. by rendering or using the depth buffer). This camera represents the imaginary camera an artist has in their head when drawing a scene. The missing depth can be inferred by using what information is not missing to formulate an optimisation problem, and most related works, including the one presented in this thesis, use an objective function to compare poses.

These common building blocks or patterns for successful sketch-based 3D posing are deduced by generalizing from the findings of recent works combined with findings presented in this thesis. While these patterns are addressed one by one, they can also be combined depending on the implementation aim. By presenting the problem in three parts with common patterns for each, the method becomes more modular so individual parts can be interchanged without affecting the rest of the algorithm.

The first part of the problem is about comparing poses. Jain et al. (2012) as well as other recent works (Xiao et al. 2015) propose pose descriptors that are translation and scale invariant. This is useful, as two characters with the same pose can have different sizes or be in different positions on the sketch. Descriptors that are not invariant to translation and scale can still be used, assuming that the characters in both the sketch and the 3D projection match (e.g. drawn storyboards that match the 3D scene).

To use various descriptors (e.g. edge maps and silhouettes like in the work proposed in this chapter), it is good practice to normalize all values to be between 0 and 1. This enforces values from different methods to be compared on the same scale. For example, using joints to describe a pose can be effective, but using only a Euclidean distance to compare two poses is not invariant to translation or scale.

It is worth noting that translation and root rotation is irrelevant if this method is used in combination with 3D storyboarding, as through the process the 3D asset should be placed correctly in the scene. Moreover, scale invariance is less important too as the drawing would more accurately match the scale of the 3D asset.

The second part of the problem is about supplying a knowledge base, in order to help solve the problem by reducing the search space. The underconstrained nature of the problem can be resolved by viewing the same pose from different angles (Guay et al. 2013; Ivekovic et al. 2008). However, when only one sketch is available, a knowledge base is necessary to reduce the possible poses. Humans understand a drawn character's pose due to prior knowledge, e.g. anatomically plausible poses or poses that make sense in context. Enforcing rotation limits on character's joints, assumptions about the type of character (e.g. humanoid), and databases are all suitable knowledge bases.

Restrictions on the 3D character's joints can be set by an artist with good intuition or by using real data (e.g. degrees of freedom of humans). However, for more stylized animation with exaggerations, the restrictions can be distracting for animators. Making assumptions about the type of character can help reduce the number of possible poses greatly at the cost of reducing the problem cases a method can be applied to. For example, human-specific assumptions would mean that a method can only be applied on humanoid characters.

Databases come with similar trade-offs. For humanoids, mocap databases can be populated cheaply, using devices such as Microsoft Kinect. The data can consist of poses or whole 3D models (Kholgade et al. 2014). However, building comprehensive databases for noncooperative characters (e.g. elephants) is more complicated. Additionally, storing a large database and extracting information from it requires infrastructure and budget that smaller studios may not have.

The third part of the problem is using an optimisation method to search for the pose. Choosing an optimisation method depends on how the problem is formulated. Jain et al. (2009) formulate the problem as a linear least squares system. Their algorithm is fast but only works with an orthographic projection. The work presented in this chapter uses the particle swarm optimisation algorithm, a global optimisation method which makes no assumptions about the problem. Although the PSO scales well when parallelised on more powerful hardware, it is slower.

In both cases, there is one common parameter: an objective or energy function. This function takes into account the chosen descriptors and transforms them into values that can be minimized. The proposed method uses a weighted combination of three pose descriptors. Choosing and constructing this objective function is crucial and has to be done together with the design of the pose descriptors.

The proposed approach may serve as a direct link between the storyboarding and pose layout phases of the pipeline. Linked together with the work presented in the previous chapters, it concludes a series of projects aiming at making 3D storyboarding more accessible, more efficient, more integrated and leveraging some of its unique advantages to explore new avenues.

# Part III

# CONCLUDING REMARKS

# 9

# DISCUSSION & FUTURE WORK

This thesis highlights problematic areas of transition in 3D animation pre-production as well as between the pre-production phase and the production phase. Animation is a medium in itself, but it is also a part of other media forms. Additionally, other forms of media are part of the creative process of animation. This thesis defines those as *media in animation*. By recognising the ubiquitous use of media within a production and acknowledging producers as consumers, this thesis proposes the term *resumer* to describe this emergent trend. A resumer is a professional consumer-producer who consumes media as part of their job, taking their interpretation of it back to their own production. An example is a storyboard artist reading a screenplay and then creating a visual representation of the story as they interpreted it. Novel methods can offer ways to remediate screenplays (Chapter 5), storyboards (Chapter 6) and 3D layout (Chapters 7 and 8). Such methods have the potential to make pre-vis a default part of the pipeline for all, by lowering costs (Chapter 3). This remediation to a more interactive form can bridge the pre-production and production phases. Moreover, it can pave the way for modern workflows and technologies to be more easily integrated. For example, it is already possible during pre-vis to navigate the 3D space for the best shot using a virtual camera (Anderton 2017), but actors still act in front of green screens. In the future, with pre-vis as default, actors could navigate the 3D space in VR to get a better understanding of the final scene.

VR is an emerging sector in entertainment, using game engines, whether it is 3D VR or a '360' video experience. Using a modern pipeline like the one proposed in this thesis is useful for these new types of production. A storyboard artist would have to setup shots in all directions of the camera (since VR allows the user to move the camera interactively), which would be more

difficult to do on a 2D medium like hand-drawn sketches. There is currently a shortage of VR products on the market and since this research project focuses on making 3D animation production more effective and efficient, the VR sector would benefit from the proposed research and development.

If this transition is done in a mediated way that possesses some of the aspects which Boyd (2007) recognises as characteristic of mediated publics (in this context people that gather and communicate mainly through technological processes), namely persistence and searchability, then locating the fault becomes easier.

An example of this idea is the version control system Git. The function 'git blame' allows any contributor to see what revision and author last modified each line of a file (Git 2017). Although the name of this function is perhaps aggressive, being able to see exactly when, by whom and why a change was made, is valuable. While blaming and shaming can be unhealthy, it is also important to mention that if the number of contributors to a project is too high, the characteristic of 'invisibility' that Boyd (2007) lists can lead to other community issues,such as toxic behaviour between peers (Salter and Blodgett 2012). The worth of a new process should not only be evaluated in financial terms, but also in social.

More specifically, given that the director has to step in to resolve issues during the transition from pre-production to production, mistakes are expensive in terms of team chemistry too: who is at fault that the 3D layout does not match the hand-drawn storyboards? Did the storyboard artist draw something that does not fit? Did the 3D modeller create assets in the wrong scale? Did the 3D layout operator position the camera in the wrong spot?

The technology for this, in many cases, is not pending or new. For example, the Microsoft Kinect device used successfully (Chapter 7) was the first version released in 2012. These changes, although requiring development and technical restructuring, mostly require organisational restructuring. The animation industry is not criticised for the process but for the product. The metrics for films, TV series or commercials revolve around whether they have been developed according to the vision and within the budget constraints, as well as whether the product achieved its goal (e.g. box-office sales for Hollywood films). This means that bad or outdated methods and processes may be repeated until proven faulty on an economic level. Discus-

sion with HRA and other companies reveals that introducing restructuring of that scale is seen as unnecessary risk.

That is not to say there are no attempts to improve the shortcomings of the current production structures. HRA had clients coming in exactly because they were looking for something better. As another example, researchers at Dreamworks recognise the importance of synchronising everyone involved in the creative process through the use of digital media (Sanders et al. 2013), but only focus on the production phase and not the pre-production. Movidiam (Movidiam 2017) is an online platform that works as a way for producers to connect with each other. Moreover, it also takes it a step further providing the tools for time-management, task delegation, collaborative feedback and even payments. Platforms like this may in fact make the already strong bonds in creative communities around the world (e.g. Soho in London, UK) even stronger, as 'geographic space can become more, not less, important' (Matei 2004, p. 27). Finally, it is interesting to note that Blue Zoo, after using HRA's 3D storyboarding tool on their own productions, began working on their own tool named PanelForge (Panel Forge Ltd. 2017).

Stepping away from a professional, budget-constrained environment, a lot more experimentation can occur, revealing how viable new media and methods can really be. An example is the viral video 'Bears on Stairs' (Gilbert and Box 2014) made by Tom Box, co-founder of Blue Zoo, and Grant Gilbert, founder of DBLG, in their spare time. The project was an attempt to reimagine traditional stop motion animation, by creating it as 3D animation and then using 3D printing to build a physical model for each frame. The 'Bears on Stairs' short film has 50 frames, resulting in 50 3D print-outs. Grant Gilbert explained in the Annecy Festival 2014: '[...] some people first thought that these were 3D models, but given the unusual skinning, with all of the grooves from the 3D printing, it would have been virtually impossible without countless hours of work to reach the result in that manner. We took our time to experiment with 3D printing and stop-motion, and it worked.' (para. 20). In time-critical and budget-constrained productions, such experimentation would not be possible. Therefore, stepping away from their role as professionals is what allowed them to test this new method.

This case can provide insight as to how the future of animation may evolve from innovations that only low-risk producers (e.g. prosumers) have the courage to experiment with. This kind of transformation would primarily

be a social one, which as demonstrated depends on the nature and characteristics of the interaction among shared interpretations, social actions and technological artifacts within organisational and professional contexts (Ashuri and Frenkel 2017). In an industry where many technical innovations are adopted, organisational changes prove to be much slower. Ultimately, adopting new media over old media is a matter of the new media proving its worth.

Building around 3D storyboarding is an attempt to make its worth more evident, but there is always room for improvement. Future work can focus on the overall pipeline as a whole, or on each individual research project presented in the previous chapters.

Looking at the overall pipeline first, it is important to evaluate how *media in animation* is used, how useful each medium is and how changing it may improve the pipeline; for example, the time it takes to move to 3D layout using traditional storyboarding compared to 3D storyboarding.

Although certain companies such as HRA and Blue Zoo use 3D storyboarding to reduce costs, it is still not the mainstream way of storyboard creation because it is new and intimidates producers and traditional artists. Moreover, 3D layout teams are threatened by this change in technology. To propose 3D storyboarding as a centre-piece of a pipeline, showing more ways that it can help to increase effectiveness and reduce costs is necessary. While 3D storyboarding has been used to animate thousands of minutes for shows like 'Fireman Sam' (HIT Entertainment 2012), 'Bob the Builder' (HIT Entertainment 2015) and 'Ritter Rost' (Caligari Film und Fernsehproduktions 2013), more hard data could help technical directors convince producers who are reluctant to make the jump. Ultimately, pipeline processes can be examined both quantitatively and qualitatively through experiments. The overall aim is to understand what is truly valuable in a pipeline, as well as how and why.

Apart from 3D storyboarding, which happens in pre-production, it would be interesting to research in more depth one of the earliest steps of production: 3D layout. Based on conversations with stakeholders from companies like Pixar as well as HRA, it becomes apparent they break down 3D layout into two stages: unposed and posed. The posed 3D layout is further refined to include the poses of the characters for the keyframes. Examining how useful posed layout is for the animator can determine whether this stage should

actually be part of a pipeline. It can also determine whether methods that automatically pose 3D models from drawings can help speed up or automate the generation of 3D posed layout. Moreover, it may help answer the question of how accurate a pose needs to be to help the animator without suppressing their creative freedom.

Future work can also focus on any of the individual projects presented in the portfolio of this thesis.

Automatic asset creation (Chapter 4) can be improved by better understanding of what is useful as a reference to the storyboard artist. More extensive feedback from the artists would help make sure that the generated 3D pegs are a suitable character representation, striking a balance between being too abstract and restrictively detailed. Therefore, future work should evaluate whether position, orientation and scale alone are necessary for this task. Following from this, if it is shown that the current results are too abstract for storyboard artists, future work can include research in how to incorporate a simple 3D model from the outline of the 2D character drawing. Taking the character outline and triangulating it to generate a plane is a naive approach to achieve this. The reason for this is that often the shapes of the characters are not concave. Triangulation algorithms such as the Delaunay triangulation (Cignoni et al. 1998) are based on finding the convex hull of a set of points. Moreover, unless dealing with simple or monotone polygons, there might be complications even with techniques such as the Ear-clipping method (Berg et al. 2008) if the outline is not perfectly extracted. The current segmentation method returns a set of points for each character. Therefore, an approach with potential is processing the available set of points using alpha-shapes (Akkiraju et al. 1995) to correctly extract the outline and then generate a rough 3D model.

The screenplay analysis system (Chapter 5) also has room for further development. After the storyboarding process is finished and there are specific shots, a clean-up stage follows. During this stage, all the characters that do not appear in the shot (in front of the camera), have to be removed in order for the export process to be as generalized as possible (e.g. when exporting to Autodesk Maya). This clean-up stage is not considered to be creative and doing it manually is always prone to user error. Therefore, future work can focus on ways to remove completely from a shot any characters which are not visible through the camera. Potential ways to approach this is through

ray-tracing (Shirley et al. 2005), where imaginary rays parallel to the ground and the camera normal vector can be traced to see with which character models they intersect. If there is no intersection with a 3D model, then it should be safely removed from the shot, before the exporting process takes place. As artists' feedback from evaluating the screenplay analysis suggested that not everything should be automated, since they considered breaking shots down to actions a part of their creative contribution, this project further motivates the argument of analysing pipelines in more depth.

Looking past pre-production, to the bridge with the production phase, cheap mocap devices and specifically the Microsoft Kinect, were examined for allowing non-experts to experiment with poses. However, another use of the proposed method is creating pose databases. While databases can be useful in inferring 3D poses from 2D drawings, it is expensive and time-consuming to populate them with data. The proposed system presents an intuitive and accessible way of generating humanoid poses for a database actively or in the background as the 3D layout phase progresses.

Such a database can also be used as an additional constraint or helper to the system proposed for 3D pose estimation from 2D drawings. In fact, a hybrid between the current optimisation approach and a data-driven approach (Jain et al. 2009) is an area where future work may expand to, to get results faster while remaining more general than a pure database method. This would be particularly effective when it comes to humanoids. Moreover, poses that have already been found in previous attempts can be stored in a database and be used as an initialisation point for the PARAC-LOAPSO run. The PARAC-LOAPSO results can then feed back to the database, storing successfully inferred poses for future use.

Despite limitations in terms of possible poses and accuracy of results, production pipelines can benefit from sketch-based methods. Final animation may still need the hand of an artist, but time and effort can be saved. The method proposed in this thesis requires little user input and allocates heavy work to the machine. Even if the accuracy of the results is not production ready, this method can serve as an automatic first pass for artists to build on. Examining how accurate an initial 3D pose needs to be in order to be considered helpful by a professional animator may lead to a new benchmark for evaluating automatic 3D posing, not only in terms of accuracy, but also in

terms of usefulness. The user input itself can take place during a clean up pass of the drawings.

Completely removing the need for user input may be possible by using medial axis methods (Abeysinghe et al. 2008) to extract the curve skeleton automatically from the drawing and then fit the character skeleton to the curve skeleton. It is also worth examining more general or accurate descriptors and methods to extract information from drawings, like the perceptual cues Bessmeltsev et al. (2016) successfully use to recover artist-intended poses. Having broken down the problem into three 'modules' (Chapter 8) means that new research such as this can be plugged in to the existing proposed pipeline. Future work can also look at formally evaluating the method using more examples. A framework of examples that can be compared to other state-of-the-art methods in the field could be built.

Furthermore, for complex scenes with many characters, it would be beneficial to segment each character automatically. This can be very challenging, especially for drawings where characters overlap one another. As described in Chapter 8, restricting the search space is a way to improve the results of the optimisation process. One way to do this is by optimising different character parts such as the legs, the arms etc. individually. The task of quickly and effortlessly segmenting the drawings into different character parts and matching them to the rendered image for comparison is not trivial, but if possible it could improve the optimisation results by breaking the problem down into a set of smaller search spaces.. Ivekovic et al. (2008) perform the optimisation in parts with success, however their method makes human-specific assumptions, removing the benefit of generality which is important to the animation industry.

The proposed method for inferring the 3D pose from drawings also fits into traditional animation pipelines. Unlike Jain et al. (2009), it does not require pre-existing data and can use perspective and orthographic camera models. The system is flexible and applicable to a broad range of models, from objects like lamps, to quadrupeds like horses. Additionally, while information such as joint limits help improve the results, it is by no means a necessity for the algorithm to run. This information usually comes together with the actual 3D model. It is not even necessary to work with skeletons and joints or any type of hierarchy; any type of control may be used instead, according to the preferences of the artist who created the model.

Moreover, similar works state the need for the 3D model to be positioned correctly and for an operator to place a camera that fits the view from the drawing (Jain et al. 2012), or solving the problem of character positioning in addition to character posing. When viewed in isolation, this is additional work, but when combined with 3D storyboarding, these are steps that are already part of the workflow — solving a complex problem through the modern 3D storyboarding pipeline proposed.

Finally, dealing with traditional cartoon exaggeration techniques such as squash and stretch is another interesting venture for the future. Assuming that a model is rigged to allow for squash and stretch deformations, then these rig controllers could be directly used to match the drawing. However, this would potentialy require the algorithm to translate the controllers, rather than rotate them, in which case translation boundaries would be required too.

# 10

# CONCLUSION

The pre-production phase of a pipeline can be modified in order to help transition to the production phase. The focus is in pre-production due to the fact that it is the least evolved phase, in contrast to the production phase.

By first analysing animation as a medium, the research presented in this thesis identifies all the available pre-existing information in mainstream pipelines. This information is then combined and used to motivate a new pipeline as well as individual technical contributions, with a focus on 3D storyboarding as the transition point between the pre-production and the production phase. Building around this core concept, intelligent solutions are proposed to bridge the two sides of the pipeline.

A method for automatically generating simple, rectangular representations of 3D characters using drawings as input is introduced. The impact of such a solution is saving time from modelling when high resolution is not necessary, especially since this can be used before the modelling team has even been assembled yet. In relation to the new proposed pipeline, this mitigates the requirement of 3D storyboarding to have 3D models early on in the pipeline.

A method for automatically analysing screenplay texts is also introduced in this thesis, in order to break them down into scenes, shots and actions. Information such as in which set each scene is taking place as well as which characters appear, is extracted and combined with action information like dialogue. The impact of such a solution is saving time breaking down the text for use further down the pipeline, e.g. by storyboard artists. Moreover, producers can use it to track the appearance of certain characters in the shots, which can be useful in the case of certain contracts (e.g. toy companies). In relation to the new proposed pipeline, this mitigates the problem that 3D

storyboarding has, which requires the storyboard artist to start by creating a structure of scenes and shots and importing sets, characters and textual information.

Both of the above systems are combined so that the screenplay analysis system can extract information such as character names and feed them into the asset creation system. The asset creation system segments characters from the drawing and names the output 3D models with the names appearing in the text. The generated place-holder assets are then fed back to the screenplay analysis system, for the creation of the storyboarding project file where they are automatically imported.

Next, a mocap or rather pose capture system using Microsoft Kinect is introduced. The system interfaces directly with Maya and allows a single user on a single machine to go through storyboard panels and pose all the characters individually. The impact of this is allowing non-expert operators to experiment with poses directly in Maya, for completing the first layout pass. In relation to the new proposed pipeline, this links the unposed layout created by 3D storyboarding with an intuitive and easy way to pose humanoid characters quickly.

Finally, a novel and general method for automatically inferring 3D poses from 2D drawings is introduced. The impact of this is contributing to the overall field of sketch-based posing with an automatic method that works irrespective of the type of character. The nature-inspired algorithm proposed in this thesis offers a scalable solution. In relation to the new proposed pipeline, this can be used as a step between unposed and posed layout. Since storyboards are overlaid on top of a 3D scene, the camera and the character models are already placed correctly in relation to the drawing, meaning that only the poses need to be inferred to achieve a complete second layout pass.

Overall, by introducing new individual methods and combining them into a single pipeline, a novel 3D animation pipeline is proposed with a value greater than the sum of its elements.

For some studios collaborating with HRA, training their artists in a new pipeline and software had additional overhead, as it was common that a majority of them were working remotely. Moreover, when starting a new show, they had no existing 3D assets. On the other hand, in cases where they had already delivered a first season of a show, changing their workflow for the following seasons seemed risky.

Following from the analysis of advantages and disadvantes of 3D storyboarding (Chapters 2 and 3), HRA was able to help make clear what the advantages would be despite this additional initial overhead. Moreover, by understanding the disadvantages, HRA was able to help setup and consult on a pipeline around 3D storyboarding. Being able to identify how information flows within the pre-production pipeline helped studios successfully incorporate 3D storyboarding in real productions such as 'Chugginton' (Ludorum 2008), 'Fireman Sam' (HIT Entertainment 2012) and 'Bob the Builder' (HIT Entertainment 2015).

Moreover, during the pre-production of 'Chugginton', the screenplay analysis system (Chapter 5) was used to build Redboard project files faster, by parsing screenplays provided by the client, speeding up the setup process. Since 3D assets already existed from previous seasons of the series, a 3D artist simplified them to work in a game engine environment.

The overall pipeline, including the projects presented in this thesis, was used to create a short sequence for the Technology Strategy Board, as well as a separate short sequence which was presented at the Annecy Animated Film Festival (Ahoomey et al. 2014). The first sequence started off as a screenplay, which was parsed to generate a Redboard project file. Moreover, the sequence contained a lamp character, which was posed automatically using the system presented in Chapter 8. The second sequence contained only humanoid characters, which were posed live on stage at the Annecy International Animated Film Festival using the system presented in Chapter 7.

# REFERENCES


3D Clic (2014). *SPI Pre-Production*. URL: http://www.3dclic.com/preproduction.html (visited on 06/05/2018).

Abeysinghe, S. S., M. Baker, W. Chiu, and T. Ju (2008). "Segmentation-free skeletonization of grayscale volumes for shape understanding." In: *Shape Modeling and Applications, 2008. SMI 2008. IEEE International Conference on*, pp. 63–71. DOI: 10.1109/SMI.2008.4547951.

Adobe Systems (2017). *Get started with Creative Cloud in just minutes*. URL: https://creative.adobe.com/plans (visited on 06/17/2018).

Agarwal, A. and B. Triggs (2004). "Learning to track 3D human motion from silhouettes." In: *In International Conference on Machine Learning*, pp. 9–16. DOI: 10.1145/1015330.1015343.

Agarwal, A. and B. Triggs (2006). "Recovering 3D human pose from monocular images." In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 28.1, pp. 44–58. DOI: 10.1109/TPAMI.2006.21.

Agarwala, A., A. Hertzmann, D. H. Salesin, and S. M. Seitz (2004). "Keyframe-based Tracking for Rotoscoping and Animation." In: *ACM SIGGRAPH 2004 Papers*. SIGGRAPH '04. Los Angeles, California: ACM, pp. 584–591. DOI: 10.1145/1186562.1015764.

Ahoomey, A., A. T. A. Gouvatsos, N. Marsden, T. Abdel-Gawad, T. Box, A. Charrier, and G. Gilbert (2014). "Emerging Tools." In: *Annecy Festival*.

Akkiraju, N., H. Edelsbrunner, M. Facello, P. Fu, E. Mücke, and C. Varela (1995). "Alpha shapes: definition and software." In: *Proceedings of the 1st International Computational Geometry Software Workshop*. Vol. 63, p. 66.

Anderton, E. (2017). *This 'Rogue One' Shot Was Created, Animated, Rendered & Released in Under a Week*. URL: http://www.slashfilm.com/rogue-one-virtual-camera-system/ (visited on 04/04/2017).

Andrews, S., I. Huerta, T. Komura, L. Sigal, and K. Mitchell (2016). "Real-time Physics-based Motion Capture with Sparse Sensors." In: *Proceedings of the 13th European Conference on Visual Media Production (CVMP 2016)*.

CVMP 2016. London, United Kingdom: ACM, 5:1–5:10. DOI: 10.1145/2998559.2998564.

Andriluka, M., L. Pishchulin, P. Gehler, and B. Schiele (2014). "2D Human Pose Estimation: New Benchmark and State of the Art Analysis." In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3686–3693. DOI: 10.1109/CVPR.2014.471.

Ashuri, T. and A. Frenkel (2017). "Online/Offscreen: On Changing Technology and Practices in Television Journalism." In: *Convergence* 23.2, pp. 148–165. DOI: 10.1177/1354856515577776.

Autodesk (2016). *Autodesk IPR*. URL: https://knowledge.autodesk.com/support/maya/learn-explore/caas/CloudHelp/cloudhelp/2016/ENU/Maya/files/GUID-9887F11F-E4DF-4F35-815A-9CB7417DD98D-htm.html (visited on 06/05/2018).

Autodesk (2018). *Subscribe to Autodesk Maya*. URL: https://www.autodesk.co.uk/products/maya/subscribe?plc=MAYA&term=1-YEAR&support=ADVANCED&quantity=1 (visited on 06/09/2018).

Baak, A., M. Müller, G. Bharaj, H. P. Seidel, and C. Theobalt (2011). "A data-driven approach for real-time full body pose reconstruction from a depth camera." In: *2011 International Conference on Computer Vision*, pp. 1092–1099. DOI: 10.1109/ICCV.2011.6126356.

Berg, M. d., O. Cheong, M. v. Kreveld, and M. Overmars (2008). *Computational Geometry: Algorithms and Applications*. 3rd ed. Santa Clara, CA, USA: Springer-Verlag TELOS.

Bessmeltsev, M., N. Vining, and A. Sheffer (2016). "Gesture3D: Posing 3D Characters via Gesture Drawings." In: *ACM Trans. Graph.* 35.6, 165:1–165:13. DOI: 10.1145/2980179.2980240.

Bijker, W. E. (1997). *Of Bicycles, Bakelites, and Bulbs: Toward a Theory of Sociotechnical Change*. English. Reprint edition. Cambridge, Mass.: The MIT Press.

Blizzard Entertainment (2017). *Software Engineer, Gameplay, World of Warcraft*. URL: https://careers.blizzard.com/en-gb/openings/oMnp4fwm (visited on 06/05/2018).

Blue-Zoo Productions (2013). *Q Pootle 5*. URL: https://www.imdb.com/title/tt2779884/ (visited on 07/08/2018).

Bolter, D. J. and R. Grusin (2000). *Remediation: Understanding New Media*. MIT Press.

Boyd, D. (2007). "Why Youth (Heart) Social Network Sites: The Role of Networked Publics in Teenage Social Life." In: *Youth, Identity, and Digital Media Volume*. The John D. and Catherine T. MacArthur Foundation Series on Digital Learning. Ed. by D. Buckingham.

Bratton, D. and J. Kennedy (2007). "Defining a Standard for Particle Swarm Optimization." In: *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, pp. 120–127. DOI: 10.1109/SIS.2007.368035.

Bruns, A. (2007). "Produsage." In: *Proceedings of the 6th ACM SIGCHI Conference on Creativity &; Cognition*. C&C '07. Washington, DC, USA: ACM, pp. 99–106. DOI: 10.1145/1254960.1254975.

Burtnyk, N. and M. Wein (1976). "Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation." In: *Commun. ACM* 19.10, pp. 564–569. DOI: 10.1145/360349.360357.

Caligari Film und Fernsehproduktions (2013). *Ritter Rost - Eisenhart und voll verbeult*. URL: https://www.imdb.com/title/tt2408734/ (visited on 07/08/2018).

Canny, J. (1986). "A Computational Approach to Edge Detection." In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* PAMI-8.6, pp. 679–698. DOI: 10.1109/TPAMI.1986.4767851.

Catmull, E. and A. Wallace (2014). *Creativity, Inc.: Overcoming the Unseen Forces That Stand in the Way of True Inspiration*. Random House Publishing Group.

Chen, J., S. Izadi, and A. Fitzgibbon (2012). "KinÊTre: Animating the World with the Human Body." In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST '12. Cambridge, Massachusetts, USA: ACM, pp. 435–444. DOI: 10.1145/2380116.2380171.

Cignoni, P., C. Montani, and R. Scopigno (1998). "DeWall: A fast divide and conquer Delaunay triangulation algorithm in $E^d$." In: *Computer-Aided Design* 30.5, pp. 333–341. DOI: 10.1016/S0010-4485(97)00082-1.

Clerc, M. and J. Kennedy (2002). "The particle swarm - explosion, stability, and convergence in a multidimensional complex space." In: *IEEE Transactions on Evolutionary Computation* 6.1, pp. 58–73. DOI: 10.1109/4235.985692.

Cowley, D. (2016). *Unreal Engine 4 Powers Real-Time Cinematography at SIGGRAPH*. URL: https://www.unrealengine.com/blog/unreal-engine-

4 - powers - real - time - cinematography - at - siggraph (visited on 06/05/2018).

Davis, J., M. Agrawala, E. Chuang, Z. Popović, and D. Salesin (2003). "A Sketching Interface for Articulated Figure Animation." In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '03. San Diego, California: Eurographics Association, pp. 320–328.

De Goussencourt, T., J. Dellac, and P. Bertolino (2015). "A Game Engine as a Generic Platform for Real-Time Previz-on-Set in Cinema Visual Effects." In: *Advanced Concepts for Intelligent Vision Systems (ACIVS)*. Catania, Italy. DOI: 10.1007/978-3-319-25903-1_76.

Deuze, M. (2007). *Media Work*. English. 1 edition. Cambridge: Polity.

Deuze, M. (2011). "Media life." In: *Media, Culture and Society* 33.1, pp. 137–148. DOI: 10.1177/0163443710386518.

Difranco, D. E. and T.-j. Cham (2001). "Reconstruction of 3-d figure motion from 2-d correspondences." In: *In Computer Vision and Pattern Recognition*, pp. 307–314.

Douglas, T. (2016a). "Tales of the tribes: animation as a tool for indigenous representation." PhD thesis. Bournemouth University. eprint: http://eprints.bournemouth.ac.uk/25018/.

Douglas, T. (2016b). *Tales of the tribes: animation as a tool for indigenous representation*. URL: https://www.epsrc.ac.uk/newsevents/events/connectednation/blog/talesofthetribes/ (visited on 06/05/2018).

Eitz, M., J. Hays, and M. Alexa (2012). "How Do Humans Sketch Objects?" In: *ACM Trans. Graph. (Proc. SIGGRAPH)* 31.4, 44:1–44:10.

Failes, I. (2017). *From Performance Capture To Creature: How The Apes Were Created In 'War for the Planet of the Apes'*. URL: https://www.cartoonbrew.com/vfx/performance-capture-creature-apes-created-war-planet-apes-152357.html (visited on 07/08/2018).

Favreau, L., L. Reveret, C. Depraz, and M.-P. Cani (2006). "Animal Gaits from Video: Comparative Studies." In: *Graph. Models* 68.2, pp. 212–234. DOI: 10.1016/j.gmod.2005.04.002.

Fernandez-Baena, A., A. Susín, and X. Lligadas (2012). "Biomechanical Validation of Upper-Body and Lower-Body Joint Movements of Kinect Motion Capture Data for Rehabilitation Treatments." In: *2012 Fourth In-*

*ternational Conference on Intelligent Networking and Collaborative Systems*, pp. 656–661. DOI: 10.1109/iNCoS.2012.66.

Fleischmann, P., I. Austvoll, and B. Kwolek (2012). "Particle Swarm Optimization with Soft Search Space Partitioning for Video-Based Markerless Pose Tracking." In: *Advanced Concepts for Intelligent Vision Systems*. Ed. by J. Blanc-Talon, W. Philips, D. Popescu, P. Scheunders, and P. Zemcik. Vol. 7517. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 479–490. DOI: 10.1007/978-3-642-33140-4_42.

Follows, S. (2014). *How many people work on a Hollywood film?* URL: https://stephenfollows.com/how-many-people-work-on-a-hollywood-film/ (visited on 06/05/2018).

FremantleMedia, CBeebies, Blue-Zoo Productions (2012). *Tree Fu Tom*. URL: https://www.imdb.com/title/tt2299207/ (visited on 07/08/2018).

Galloway, A. (2013). "Emergent Media Technologies, Speculation, Expectation, and Human/Nonhuman Relations." In: *Journal of Broadcasting & Electronic Media* 57.1, pp. 53–65. DOI: 10.1080/08838151.2012.761705.

Gavrila, D. (1999). "The Visual Analysis of Human Movement: A Survey." In: *Computer Vision and Image Understanding* 73 (1), pp. 82–98. DOI: 10.1006/cviu.1998.0716.

Gilbert, G. and T. Box (2014). *Bears on Stairs*. URL: http://dblg.co.uk/work/stairs (visited on 06/05/2018).

Git (2017). *git-blame*. URL: https://git-scm.com/docs/git-blame (visited on 06/05/2018).

Glebas, F. (2009). *Directing the Story: Professional Storytelling and Storyboarding Techniques for Live Action and Animation*. Animation masters. Elsevier/Focal Press.

Gouvatsos, A., N. Marsden, and J. Hibbert (2014a). "Optimising TV series pre-production: Redboard." In: *Annecy International Animated Film Festival 2014 Conference Summaries*. Annecy, France.

Gouvatsos, A. and Z. Xiao (2015). "Sketch-Based Posing for 3D Animation." In: *Encyclopedia of Computer Graphics and Games*. Ed. by N. Lee. Cham: Springer International Publishing, pp. 1–10. DOI: 10.1007/978-3-319-08234-9_47-1.

Gouvatsos, A., Z. Xiao, N. Marsden, and J. J. Zhang (2014b). "Automatic 3D Posing from 2D Hand-Drawn Sketches." In: *Pacific Graphics Posters*. The Eurographics Association, 1:1–1:1.

Gouvatsos, A., Z. Xiao, N. Marsden, and J. J. Zhang (2014c). "Automatic 3D Posing from 2D Hand-Drawn Sketches." In: *Pacific Graphics Short Papers*. Ed. by J. Keyser, Y. J. Kim, and P. Wonka. The Eurographics Association. DOI: 10.2312/pgs.20141264.

Gouvatsos, A., Z. Xiao, N. Marsden, and J. J. Zhang (2017). "Posing 3D Models from Drawings." In: *Comput. Entertain.* 15.2, 2:1–2:14. DOI: 10.1145/2729984.

Gouvatsos, A., Z. Xiao, K. Pang, N. Marsden, D. Van der Ark, J. Hibbert, and J. J. Zhang (2016). "Efficient Storyboarding in 3D Game Engines." In: *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation: Posters*. SCA '16. Zurich, Switzerland: Eurographics Association, 1:1–1:1.

Gregor, S., J. Hutson, and C. Oresky (2002). "Storyboard Process to Assist in Requirements Verification and Adaptation to Capabilities Inherent in COTS." English. In: *COTS-Based Software Systems*. Ed. by J. Dean and A. Gravel. Vol. 2255. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 132–141. DOI: 10.1007/3-540-45588-4_13.

Guay, M., M.-P. Cani, and R. Ronfard (2013). "The Line of Action: An Intuitive Interface for Expressive Character Posing." In: *ACM Trans. Graph.* 32.6, 205:1–205:8. DOI: 10.1145/2508363.2508397.

Haberer, A. (2007). "Intertextuality in Theory and Practice." In: *Literatura* 49.5, pp. 54–67.

Han, X., C. Gao, and Y. Yu (2017). "DeepSketch2Face: A Deep Learning Based Sketching System for 3D Face and Caricature Modeling." In: *ACM Trans. Graph.* 36.4, 126:1–126:12. DOI: 10.1145/3072959.3073629.

Held, R., A. Gupta, B. Curless, and M. Agrawala (2012). "3D Puppetry: A Kinect-based Interface for 3D Animation." In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST '12. Cambridge, Massachusetts, USA: ACM, pp. 423–434. DOI: 10.1145/2380116.2380170.

Hibbert Ralph Animation (2014). *Redboard*. Hibbert Ralph Animation. URL: http://www.redboard.tv/ (visited on 06/05/2018).

Hillin, J., D. Rand, S. Squires, S. Ross, S. Kaplan, and D. Yocis (2013). "The State of the Visual Effects Industry." In: *SIGGRAPH13 Talks*.

HIT Entertainment (2012). *Fireman Sam*. URL: https://www.imdb.com/title/tt0329829/ (visited on 07/08/2018).

HIT Entertainment (2015). *Bob the Builder*. URL: https://www.imdb.com/title/tt0262151/ (visited on 07/08/2018).

Hoshino, J. and Y. Hoshino (2001). "Intelligent Storyboard for Prototyping Animation." In: *2012 IEEE International Conference on Multimedia and Expo*, p. 96. DOI: 10.1109/ICME.2001.1237735.

Hua, G., M.-h. Yang, and Y. Wu (2005). "Learning to estimate human pose with data driven belief propagation." In: *CVPR*, pp. 747–754.

Ibrus, I. and C. Scolari (2012). *Crossmedia Innovations: Texts, Markets, Institutions*. Peter Lang GmbH, Europaischer Verlag der Wissenschaften.

Ichikari, R., K. Kawano, A. Kimura, F. Shibata, and H. Tamura (2006). "Mixed reality pre-visualization and camera-work authoring in filmmaking." In: *2006 IEEE/ACM International Symposium on Mixed and Augmented Reality*, pp. 239–240. DOI: 10.1109/ISMAR.2006.297823.

Igarashi, T., S. Matsuoka, and H. Tanaka (1999). "Teddy: a sketching interface for 3D freeform design." In: *Annual Conference on Computer Graphics*, pp. 409–416. DOI: 10.1145/311535.311602.

Ivekovic, S., E. Trucco, and R. Y. Petillot (2008). "Human Body Pose Estimation with Particle Swarm Optimisation." In: *Evolutionary Computation* 16 (4), pp. 509–528. DOI: 10.1162/evco.2008.16.4.509.

Izadi, S., D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon (2011). "KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera." In: ACM, pp. 559–568.

Jacobson, A., D. Panozzo, O. Glauser, C. Pradalier, O. Hilleges, and O. Sorkine-Horning (2014). "Tangible and Modular Input Device for Character Articulation." In: *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)* 33.4.

Jain, E., Y. Sheikh, and J. K. Hodgins (2009). "Leveraging the talent of hand animators to create three-dimensional animation." In: *Symposium on Computer Animation*, pp. 93–102. DOI: 10.1145/1599470.1599483.

Jain, E., Y. Sheikh, M. Mahler, and J. Hodgins (2012). "Three-dimensional proxies for hand-drawn characters." In: *ACM Trans. Graph.* 31.1, 8:1–8:16.

Jenkins, H. (2006). *Convergence Culture: Where Old and New Media Collide*. NYU Press.

Jhala, A., C. Rawls, S. Munilla, and R. Young (2008). "Longboard: A sketch based intelligent storyboarding tool for creating machinima." In: *Pro-*

*ceedings of the Florida Artificial Intelligence Research Society Conference (FLAIRS)*.

Johnston, O. and F. Thomas (1981). *The Illusion of Life: Disney Animation*. Abbeville Press.

Jung, Y., S. Wagner, C. Jung, J. Behr, and D. Fellner (2010). "Storyboarding and pre-visualization with x3d." In: *In Proc. Web3D 2010: 15th Intl. Conference on 3D Web Technology, ACM*. Press.

Kadleček, P., A.-E. Ichim, T. Liu, J. Křivánek, and L. Kavan (2016). "Reconstructing Personalized Anatomical Models for Physics-based Body Animation." In: *ACM Trans. Graph.* 35.6, 213:1–213:13. DOI: 10.1145/2980179.2982438.

Kholgade, N., T. Simon, A. Efros, and Y. Sheikh (2014). "3D Object Manipulation in a Single Photograph using Stock 3D Models." In: *ACM Transactions on Computer Graphics* 33.4.

Kirillov, A. (2011). *AForge.NET Framework*. URL: http://www.aforgenet.com/ (visited on 06/05/2018).

Kubo, T. (2013). *My Neighbor Totoro: A Novel*. VIZ Media LLC, p. 192.

Lange, B., S. Koenig, E. McConnell, C. Y. Chang, R. Juang, E. Suma, M. Bolas, and A. Rizzo (2012). "Interactive game-based rehabilitation using the Microsoft Kinect." In: *2012 IEEE Virtual Reality Workshops (VRW)*, pp. 171–172. DOI: 10.1109/VR.2012.6180935.

Li, C., H. Pan, Y. Liu, X. Tong, A. Sheffer, and W. Wang (2017). "BendSketch: Modeling Freeform Surfaces Through 2D Sketching." In: *ACM Trans. Graph.* 36.4, 125:1–125:14. DOI: 10.1145/3072959.3073632.

Li, Y., M. Gleicher, Y.-Q. Xu, and H.-Y. Shum (2003). "Stylizing Motion with Drawings." In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '03. San Diego, California: Eurographics Association, pp. 309–319.

Lin, A., G. S. Lee, J. Longson, J. Steele, E. Goldberg, and R. Stefanovic (2015). "Achieving Real-time Playback with Production Rigs." In: *ACM SIGGRAPH 2015 Talks*. SIGGRAPH '15. Los Angeles, California: ACM, 11:1–11:1. DOI: 10.1145/2775280.2792519.

Lintmeijer, L. L., G. S. Faber, H. R. Kruk, A. J. " van Soest, and M. J. Hofmijster (2018). "An accurate estimation of the horizontal acceleration of a rower's centre of mass using inertial sensors: a validation." In: *European Journal of Sport Science*, pp. 1–7. DOI: 10.1080/17461391.2018.1465126.

Liu, C., E. Rosales, and A. Sheffer (2018). "StrokeAggregator: Consolidating Raw Sketches into Artist-Intended Curve Drawings." In: *ACM Transaction on Graphics* 37.4. DOI: 10.1145/3197517.3201314.

Ludorum (2008). *Chuggington*. URL: https://www.imdb.com/title/tt1307510/ (visited on 07/08/2018).

Luebke, D., M. Reddy, J. D. Cohen, A. Varshney, B. Watson, and R. Huebner (2002). *Level of Detail for 3D Graphics*. Elsevier.

Malmelin, N. and M. Villi (2017). "Co-Creation of What? Modes of Audience Community Collaboration in Media Work." en. In: *Convergence* 23.2, pp. 182–196. DOI: 10.1177/1354856515592511.

Mao, C., S. F. Qin, and D. K. Wright (2006). "Sketching-out virtual humans: From 2d storyboarding to immediate 3d character animation." In: *Proceedings of ACM SIGCHI International Conference On Advances In Computer Entertainment Technology*. ACM.

Mao, C., S. F. Qin, and D. Wright (2007). "Sketch-Based Virtual Human Modelling and Animation." In: *Proceedings of the 8th International Symposium on Smart Graphics*. SG '07. Kyoto, Japan: Springer-Verlag, pp. 220–223. DOI: 10.1007/978-3-540-73214-3_24.

Marvel Studios (2015). *Avengers: Age of Ultron*. URL: https://www.imdb.com/title/tt2395427/ (visited on 06/05/2018).

Matei, S. (2004). "The Impact of State-Level Social Capital on the Emergence of Virtual Communities." In: *Journal of Broadcasting & Electronic Media* 48.1, pp. 23–40. DOI: 10.1207/s15506878jobem4801_2.

McDermott, S., J. Li, and W. H. Bares (2002). "Storyboard frame editing for cinematic composition." In: *Intelligent User Interfaces*, pp. 206–207. DOI: 10.1145/502716.502759.

Mehta, D., S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt (2017). "VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera." In: vol. 36. 4. DOI: 10.1145/3072959.3073596.

Microsoft (2017). *MSDN - Kinect for Windows - Joint Orientation*. URL: https://msdn.microsoft.com/en-us/library/hh973073.aspx (visited on 08/27/2017).

Mikolajczyk, K. and C. Schmid (2005). *A Performance Evaluation Of Local Descriptors*.

*Minimalist Lego Cartoon Characters* (2012). http://mymodernmet.com/jung-von-matt-imagine-lego-ad/.

Moran, G. (2015). "Pushing Photorealism in "a Boy and His Kite"." In: *ACM SIGGRAPH 2015 Computer Animation Festival*. SIGGRAPH '15. Los Angeles, California: ACM, pp. 190–190. DOI: 10.1145/2790329.2790332.

Mori, S., R. Ichikari, F. Shibata, A. Kimura, and H. Tamura (2011). "Enabling On-set Stereoscopic MR-based Previsualization for 3D Filmmaking." In: *SIGGRAPH Asia 2011 Sketches*. SA '11. Hong Kong, China: ACM, 14:1–14:2. DOI: 10.1145/2077378.2077396.

Movidiam (2017). *Movidiam*. URL: https://www.movidiam.com/ (visited on 06/05/2018).

Muller, M. (2017). *Mr Carton – The world's first cartoon series MadeWithUnity*. URL: https://blogs.unity3d.com/2017/02/23/mr-carton-the-worlds-first-cartoon-series-madewithunity/ (visited on 07/08/2018).

Mussi, L., S. Ivekovic, and S. Cagnoni (2010). "Markerless Articulated Human Body Tracking from Multi-view Video with GPU-PSO." In: *Proceedings of the 9th International Conference on Evolvable Systems: From Biology to Hardware*. ICES'10. York, UK: Springer-Verlag, pp. 97–108.

New Line Cinema (2014). *The Hobbit: The Battle of the Five Armies*. URL: https://www.imdb.com/title/tt2310332/ (visited on 07/08/2018).

Nitsche, M. (2008). "Experiments in the Use of Game Technology for Previsualization." In: *Proceedings of the 2008 Conference on Future Play: Research, Play, Share*. Future Play '08. Toronto, Ontario, Canada: ACM, pp. 160–165. DOI: 10.1145/1496984.1497011.

Northam, L., J. Istead, and C. S. Kaplan (2011). "RTFX: On-Set Previs with UnrealEngine3." In: *Entertainment Computing – ICEC 2011*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 432–435.

O'Brien, J. F., R. Bodenheimer, G. Brostow, and J. K. Hodgins (2000). "Automatic Joint Parameter Estimation from Magnetic Motion Capture Data." In: *Graphics Interface*, pp. 53–60.

Öztireli, A. C., I. Baran, T. Popa, B. Dalstein, R. W. Sumner, and M. Gross (2013). "Differential Blending for Expressive Sketch-Based Posing." In: *Proceedings of the 2013 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '13. Anaheim, LA, USA: ACM.

Panel Forge Ltd. (2017). *panelforge: storyboard with simplicity, efficiency and style*. URL: http://panel-forge.com/index.html (visited on 06/05/2018).

Patel, M. (2009). *The Digital Visual Effects Studio: The Artists and Their Work Revealed*. Pap/Psc edition. CreateSpace Independent Publishing Platform.

Pixar Animation Studios (2015). *Pixar RIS*. URL: https://renderman.pixar.com/resources/RenderMan_20/risOverview.html (visited on 06/05/2018).

Poel, M. (2006). "Comparison of Silhouette Shape Descriptors for Example-based Human Pose Recovery." In: *IEEE FGR* 2006, pp. 541–546.

Pouet (2000). *Pouet.net*. URL: https://www.pouet.net/ (visited on 07/08/2018).

Prahalad, C. K. and V. Ramaswamy (2000). *Co-Opting Customer Competence*. https://hbr.org/2000/01/co-opting-customer-competence.

Ramanan, D. and D. A. Forsyth (2003). *Finding and Tracking People from the Bottom Up*.

Ren, L., G. Shakhnarovich, J. Hodgins, H. Pfister, and P. Viola (2004). *Learning Silhouette Features for Control of Human Motion*.

Ritzer, G., P. Dean, and N. Jurgenson (2012). "The Coming of Age of the Prosumer." In: *American Behavioral Scientist* 56.4, pp. 379–398. DOI: 10.1177/0002764211429368.

Rosales, R. and S. Sclaroff (1999). "Inferring Body Pose without Tracking Body Parts." In: *IN CVPR*, pp. 721–727.

Sakoe, H. and S. Chiba (1978). "Dynamic programming algorithm optimization for spoken word recognition." In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26.1, pp. 43–49. DOI: 10.1109/TASSP.1978.1163055.

Salehizadeh, S. M. A., P. Yadmellat, and M.-B. Menhaj (2009). "Local Optima Avoidable Particle Swarm Optimization." In: *Swarm Intelligence Symposium, 2009. SIS '09. IEEE*, pp. 16–21. DOI: 10.1109/SIS.2009.4937839.

Salter, A. and B. Blodgett (2012). "Hypermasculinity & Dickwolves: The Contentious Role of Women in the New Gaming Public." In: *Journal of Broadcasting & Electronic Media* 56.3, pp. 401–416. DOI: 10.1080/08838151.2012.705199.

Samet, H. and M. Tamminen (1988). "Efficient component labeling of images of arbitrary dimension represented by linear bintrees." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10.4, pp. 579–586. DOI: 10.1109/34.3918.

Sanders, C., M. Baer, M. Edwards, and D. Crowe (2013). "DreamWorks Animation Presents: A Journey to the Croodaceous." In: *SIGGRAPH13 Talks*.

Sedai, S., M. Bennamoun, and D. Huynh (2011). "Evaluating Shape and Appearance Descriptors for 3D Human Pose Estimation." In: *Proceedings of the 2011 6th IEEE Conference on Industrial Electronics & Applications*. Vol. Single, pp. 287–292.

Seki, S. and T. Igarashi (2017). "Sketch-based 3D Hair Posing by Contour Drawings." In: *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. SCA '17. Los Angeles, California: ACM, 29:1–29:2. DOI: 10.1145/3099564.3106638.

Selby, M., A. Rao, D. Fagnou, C. Cameron, A. Valdez, and H. Ricklefs (2013). "Put That in Your Pipe!" In: *SIGGRAPH13 Talks*.

Seol, Y., C. O'Sullivan, and J. Lee (2013). "Creature Features: Online Motion Puppetry for Non-human Characters." In: *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '13. Anaheim, California: ACM, pp. 213–221. DOI: 10.1145/2485895.2485903.

Shirley, P., M. Ashikhmin, and S. Marschner (2005). *Fundamentals of Computer Graphics*. 2nd. Natick, MA, USA: A. K. Peters, Ltd.

Shotgun Software Inc. (2018). *Shotgun Software*. URL: https://www.shotgunsoftware.com (visited on 06/05/2018).

Shotton, J., A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake (2011). "Real-time human pose recognition in parts from single depth images." In: *CVPR*. Vol. 3.

Simo-Serra, E., S. Iizuka, and H. Ishikawa (2018). "Mastering Sketching: Adversarial Augmentation for Structured Prediction." In: *ACM Trans. Graph.* 37.1, 11:1–11:13. DOI: 10.1145/3132703.

*Snow White and the Seven Dwarfs (1934) [DVD] - Behind the Scenes* (2009).

Sony Interactive Entertainment (2017). *Sony Playstation Jobs Engineer*. URL: https://www.playstationjobs.co.uk/vacancy/1741-UE4-Engineer/page/3 (visited on 06/05/2018).

*Star Wars Trilogy (Episodes IV-VI) (1977) [DVD] - Behind the Scenes* (2013).

Studio Ghibli (1986). *Castle in the Sky*. URL: https://www.imdb.com/title/tt0092067/ (visited on 06/05/2018).

Studio Ghibli (1988). *My Neighbour Totoro*. URL: https://www.imdb.com/title/tt0096283/ (visited on 06/05/2018).

Sucontphunt, T., Z. Mo, U. Neumann, and Z. Deng (2008). "Interactive 3D facial expression posing through 2D portrait manipulation." In: *Graphics Interface*, pp. 177–184. DOI: 10.1145/1375714.1375745.

Sullivan, K., G. Schumer, and K. Alexander (2008). *Ideas for the Animated Short: Finding and Building Stories*. Focal Press.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. 1st. New York, NY, USA: Springer-Verlag New York, Inc.

The Creative Assembly (2017). *CAMPAIGN GAMEPLAY PROGRAMMER*. URL: http://www.creative-assembly.com/job/170307/campaign-gameplay-programmer (visited on 06/05/2018).

The Foundry Visionmongers Ltd. (2014). *Nuke*. URL: http://www.thefoundry.co.uk/products/nuke/ (visited on 06/05/2018).

The Foundry Visionmongers Ltd. (2017). *flix - collaborative story development - foundry*. URL: https://www.foundry.com/products/flix (visited on 06/05/2018).

Toffler, A. (1980). *The Third Wave*. Morrow.

Tresadern, P. and I. Reid (2007). "An Evaluation of Shape Descriptors for Image Retrieval in Human Pose Estimation." In: *Proc. 18th British Machine Vision Conf., Warwick*. Vol. 2, pp. 800–809.

Unreal Engine (2015). *A Boy & His Kite*. URL: https://www.youtube.com/watch?v=JNgsbNvkNjE (visited on 07/08/2018).

Usher, N. (2010). "Goodbye to the news: how out-of-work journalists assess enduring news values and the new media landscape." In: *New Media and Society* 12.6, pp. 911–928. DOI: 10.1177/1461444809350899.

Vaillancourt, O. and R. Egli (2011). "Placeholders beyond Static Art Replacement." In: *Game Engine Gems, Volume 2*. CRC Press.

Vlasic, D., I. Baran, W. Matusik, and J. Popović (2008). "Articulated Mesh Animation from Multi-view Silhouettes." In: *ACM Trans. Graph.* 27.3, 97:1–97:9. DOI: 10.1145/1360612.1360696.

Walt Disney Productions (1937). *Snow White and the Seven Dwarfs*. URL: https://www.imdb.com/title/tt0029583/ (visited on 06/05/2018).

Wang, C. C., T. K. Chang, and M. M. Yuen (2003). "From laser-scanned data to feature human model: a system based on fuzzy logic concept." In: *Computer-Aided Design* 35.3, pp. 241–253. DOI: https://doi.org/10.1016/S0010-4485(01)00209-3.

Warner Bros. Pictures (2017). *Bladerunner 2049*. URL: https://www.imdb.com/title/tt1856101/ (visited on 06/05/2018).

Wei, S., V. Ramakrishna, T. Kanade, and Y. Sheikh (2016). "Convolutional Pose Machines." In: *CoRR* abs/1602.00134. arXiv: 1602.00134.

Wei, X. and J. Chai (2010). "VideoMocap: Modeling Physically Realistic Human Motion from Monocular Video Sequences." In: *ACM Trans. Graph.* 29.4, 42:1–42:10. DOI: 10.1145/1778765.1778779.

Weta (2015). *FX Guide TV - Weta on the Hobbit*. URL: http://media.fxguide.com/fxguidetv/fxguidetv-ep199.mp4 (visited on 07/08/2018).

Winder, C. and Z. Dowlatabadi (2001). *Producing Animation*. Focal Press visual effects and animation series. Focal Press.

Winter, T. and J. Belfort (2013). *Wolf of Wall Street, The Script*. URL: http://www.imsdb.com/scripts/Wolf-of-Wall-Street,-The.html (visited on 06/05/2018).

Xia, S., L. Gao, Y.-K. Lai, M.-Z. Yuan, and J. Chai (2017). "A Survey on Human Performance Capture and Animation." In: *Journal of Computer Science and Technology* 32.3, pp. 536–554. DOI: 10.1007/s11390-017-1742-y.

Xiao, J., Z. Tang, Y. Feng, and Z. Xiao (2015). "Sketch-based human motion retrieval via selected 2D geometric posture descriptor." In: *Signal Processing* 113, pp. 1–8. DOI: 10.1016/j.sigpro.2015.01.004.

Xu, K., K. Chen, H. Fu, W.-L. Sun, and S.-M. Hu (2013). "Sketch2Scene: Sketch-based Co-retrieval and Co-placement of 3D Models." In: *ACM Transactions on Graphics* 32.4, 123:1–123:12.

Yamane, K., Y. Ariki, and J. Hodgins (2010). "Animating Non-humanoid Characters with Human Motion Data." In: *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '10. Madrid, Spain: Eurographics Association, pp. 169–178.

Yang, R. and B. C. Wünsche (2010). "LifeSketch - A Framework for Sketch-Based Modelling and Animation of 3D Objects." In: *Proceedings of the Australasian User Interface Conference (AUIC 2010*, p. 2010.

Ye, G., Y. Liu, Y. Deng, N. Hasler, X. Ji, Q. Dai, and C. Theobalt (2013). "Free-Viewpoint Video of Human Actors Using Multiple Handheld Kinects." In: *IEEE Transactions on Cybernetics* 43.5, pp. 1370–1382. DOI: 10.1109/TCYB.2013.2272321.

Zagal, J. P. and R. Altizer (2015). "Placeholder Content in Game Development: Benefits and Challenges." In: *Proceedings of the 2015 Annual Sym-*

*posium on Computer-Human Interaction in Play*. CHI PLAY '15. London, United Kingdom: ACM, pp. 745–750. DOI: 10.1145/2793107.2810319.

Zhao, J. and N. Badler (1994). "Inverse kinematics positioning using nonlinear programming for highly articulated figures." In: *ACM Transactions on Graphics* 13, pp. 313–336.

Zhu, H., Y. Liu, J. Fan, Q. Dai, and X. Cao (2017). "Video-Based Outdoor Human Reconstruction." In: *IEEE Transactions on Circuits and Systems for Video Technology* 27.4, pp. 760–770. DOI: 10.1109/TCSVT.2016.2596118.

# GLOSSARY

HRA Hibbert Ralph Animation

DRY Don't Repeat Yourself

API Application Programming Interface

VFX Visual Effects

CGI Computer Generated Imagery

2D Two Dimensional

3D Three Dimensional

UI User Interface

WPF Windows Presentation Foundation

MVVM Model-View-ViewModel

PSO Particle Swarm Optimisation

SDK Software Development Kit

PCA Principal Component Analysis

FK Forward Kinematics

IK Inverse Kinematics

VR Virtual Reality

mocap Motion Capture

app Software Application