2016

From Fantasy to Virtual Reality: An Exploration of Modeling, Rigging and Animating Characters for Video Games

Sarah K. Williams
The College of Wooster, swilliams16@wooster.edu

Follow this and additional works at: https://openworks.wooster.edu/independentstudy

Part of the Graphics and Human Computer Interfaces Commons

Recommended Citation
https://openworks.wooster.edu/independentstudy/7171

This Senior Independent Study Thesis Exemplar is brought to you by Open Works, a service of The College of Wooster Libraries. It has been accepted for inclusion in Senior Independent Study Theses by an authorized administrator of Open Works. For more information, please contact openworks@wooster.edu.

© Copyright 2016 Sarah K. Williams
FROM FANTASY TO VIRTUAL REALITY: AN EXPLORATION OF MODELING, RIGGING AND ANIMATING CHARACTERS FOR VIDEO GAMES

INDEPENDENT STUDY THESIS

Presented in Partial Fulfillment of the Requirements for the Degree Bachelor of Arts in the Department of Mathematics and Computer Science at The College of Wooster

by

Sarah K. Williams

The College of Wooster
2016

Advised by:

Dr. Denise Byrnes (Computer Science)
Abstract

In the last few decades video games have quickly become one of the most popular forms of entertainment around the world. This can be linked to the improvement of computer systems and graphics which now allow for authentic and highly detailed computer generated characters. This project examines how these characters are modeled and developed. The examination of game characters entails a brief history of video games and their aesthetics. The foundations of character design are discussed and 3D modeling of a character is explored in detail. Finally, rigging or skeleton placement is investigated in order to animate the characters designed for this study. The result is two animated characters, which can be incorporated into several of the current and popular game engines. By the end of this paper the reader should have a fundamental understanding of how a video game character is designed, modeled, rigged, and animated.
This work is dedicated to my Mother and Father.
Thank you both for your unwavering support and love.
First and foremost I would like to thank my family for always being there and supporting me in everything I do. I don’t know what I did to deserve such a fun and amazing family, but I’m eternally grateful and love you all so much.

Thank you to the many professors who have supported me and instilled in me a life-long desire to learn. Many thanks to Dr. Byrnes, my advisor from day one of FYS, as her mentorship is the reason I discovered my love of computer science.

To all the amazing people at Wooster I have had the opportunity to call “friend,” the deepest of thanks. Late night Mom’s runs, movie marathons and weekend shenanigans with you will always be among my fondest memories.

Special shout out to Team Fax Code ’16 for always being there, whether it was a late night in the lab or an afternoon filled with laughter. I can’t wait to see where we all end up, but I know no matter what, we’ll be friends for life.
# Contents

Abstract v
Dedication vii
Acknowledgments ix
Contents xi
List of Figures xiii

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 History of video games</td>
<td>1</td>
</tr>
<tr>
<td>2 Character Design</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Introduction to character design</td>
<td>5</td>
</tr>
<tr>
<td>3 Computer Graphics and Modeling</td>
<td>9</td>
</tr>
<tr>
<td>3.1 Introduction to Computer Graphics</td>
<td>9</td>
</tr>
<tr>
<td>3.2 3D modeling</td>
<td>10</td>
</tr>
<tr>
<td>3.2.1 Polygonal modeling</td>
<td>10</td>
</tr>
<tr>
<td>3.2.2 NURBS modeling</td>
<td>11</td>
</tr>
<tr>
<td>3.2.3 Tessellation</td>
<td>12</td>
</tr>
<tr>
<td>4 Maya Modeling</td>
<td>13</td>
</tr>
<tr>
<td>4.1 Introduction to Maya Modeling</td>
<td>13</td>
</tr>
<tr>
<td>4.1.1 First look at Maya</td>
<td>13</td>
</tr>
<tr>
<td>4.2 Basic character modeling</td>
<td>16</td>
</tr>
<tr>
<td>4.2.1 Sketching the model</td>
<td>16</td>
</tr>
<tr>
<td>4.2.2 Getting ready to model</td>
<td>18</td>
</tr>
<tr>
<td>4.2.3 Basic Modeling</td>
<td>21</td>
</tr>
<tr>
<td>4.2.3.1 Torso and Head</td>
<td>21</td>
</tr>
<tr>
<td>4.2.3.2 Legs</td>
<td>25</td>
</tr>
<tr>
<td>4.2.3.3 Arms</td>
<td>28</td>
</tr>
<tr>
<td>4.2.3.4 Hands</td>
<td>32</td>
</tr>
<tr>
<td>4.2.3.5 Feet</td>
<td>37</td>
</tr>
<tr>
<td>4.2.3.6 Eyes</td>
<td>40</td>
</tr>
<tr>
<td>4.2.3.7 Face Details</td>
<td>49</td>
</tr>
<tr>
<td>4.2.3.8 Converting NURBS to Polygons</td>
<td>52</td>
</tr>
<tr>
<td>4.2.3.9 Adding color</td>
<td>62</td>
</tr>
<tr>
<td>5 Maya Animation Rigging Toolset and Basic Animation in Maya</td>
<td>65</td>
</tr>
<tr>
<td>5.1 Rigging</td>
<td>66</td>
</tr>
<tr>
<td>5.1.1 Skeleton Creation</td>
<td>66</td>
</tr>
<tr>
<td>5.1.2 Skeleton Placement</td>
<td>68</td>
</tr>
<tr>
<td>5.1.3 Deformation Setup</td>
<td>69</td>
</tr>
<tr>
<td>5.2 Animation Tools</td>
<td>73</td>
</tr>
</tbody>
</table>
5.3 Animation in Maya ................................................................. 76
  5.3.1 Bouncing ball animation .................................................. 77

6 Results 85
  6.1 Female Model ................................................................. 85
    6.1.1 Sketches and Drawings ............................................... 85
    6.1.2 Modeling ................................................................. 87
    6.1.3 Texture Mapping ...................................................... 89
  6.2 Animation ................................................................. 93

7 Conclusion and Future Work 97
  7.1 Future Work ............................................................... 98

References 101

Index 103
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td><em>Dragon's Lair</em> [13]</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td><em>Journey</em>, created by thatgamecompany [14]</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Sackboy from <em>LittleBigPlanet</em> [15]</td>
<td>6</td>
</tr>
<tr>
<td>2.2</td>
<td>Male representations of Ectomorph, Mesomorph and Endomorph [20]</td>
<td>6</td>
</tr>
<tr>
<td>2.3</td>
<td>Female representations of Ectomorph, Mesomorph and Endomorph [20]</td>
<td>7</td>
</tr>
<tr>
<td>3.1</td>
<td>Polygon Sphere</td>
<td>10</td>
</tr>
<tr>
<td>3.2</td>
<td>Polygon torus on left - NURBS torus on right</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>(a) Top view</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>(b) Side View</td>
<td>12</td>
</tr>
<tr>
<td>4.1</td>
<td>The initial Maya screen</td>
<td>14</td>
</tr>
<tr>
<td>4.2</td>
<td>Maya’s four viewports and Hotbox</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>(a) The four viewing options in Maya</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>(b) Maya’s Hotbox</td>
<td>15</td>
</tr>
<tr>
<td>4.3</td>
<td>Appearance of Maya’s three transform tools and their Manipulators</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>(a) Move</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>(b) Rotate</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>(c) Scale</td>
<td>16</td>
</tr>
<tr>
<td>4.4</td>
<td>Simple character sketch on graph paper</td>
<td>17</td>
</tr>
<tr>
<td>4.5</td>
<td>Additional reference images</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>(a) Photoshop made T-pose reference image</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>(b) Top down arm reference image</td>
<td>17</td>
</tr>
<tr>
<td>4.6</td>
<td>Making a NURBS plane for the reference image</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>(a) Make a NURBS plane</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>(b) NURBS plane scaled to 1/100 of reference image size</td>
<td>18</td>
</tr>
<tr>
<td>4.7</td>
<td>Making a new material for the reference image</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>(a) Open Hypershade</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>(b) Hypershade popup box</td>
<td>19</td>
</tr>
<tr>
<td>4.8</td>
<td>Create new lambert material</td>
<td>19</td>
</tr>
<tr>
<td>4.9</td>
<td>Set color to file and choose reference image</td>
<td>20</td>
</tr>
<tr>
<td>4.10</td>
<td>All three reference images set</td>
<td>20</td>
</tr>
<tr>
<td>4.11</td>
<td>EP curve tool</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>(a) EP curve tool</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>(b) The traced EP curve</td>
<td>21</td>
</tr>
<tr>
<td>4.12</td>
<td>NURBS surface</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>(a) The NURBS surface produced by using the revolve tool</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>(b) The NURBS surface is tied to the EP curve</td>
<td>22</td>
</tr>
<tr>
<td>4.13</td>
<td>Creating a lattice</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>(a) Menu to create Lattice</td>
<td>22</td>
</tr>
</tbody>
</table>
4.14 Refined lattices .................................................. 23
   (a) Front ...................................................... 23
   (b) Side ...................................................... 23

4.15 Deleting the history of the NURBS surface makes changes permanent and removes the link to the EP curve .................................................. 24

4.16 Changes to Little Green Man are made as the character is modeled .......... 24

4.17 A cylinder makes up the base of the leg ........................................... 25
   (a) Base cylinder ............................................ 25
   (b) Cylinder with 12 spans .................................. 25

4.18 Moved and scaled cylinder ..................................................... 25
   (a) Side ...................................................... 25
   (b) Front ...................................................... 25

4.19 The lattice is moved to produce the shape of the leg .................................. 26

4.20 Translation and Inversion of the duplicated leg ........................................... 26
   (a) Translation .............................................. 26
   (b) Inversion ................................................. 26

4.21 Little Green Man and his new legs ................................................... 27

4.22 The NURBS cylinder with 10 spans .................................................... 28

4.23 The lattice is moved to produce the shape of the arm .................................. 28
   (a) Resized cylinder ....................................... 28
   (b) Original lattice around cylinder ......................... 28

4.24 Arm lattice deformation .................................................................. 29
   (a) ............................................................... 29
   (b) ............................................................... 29
   (c) ............................................................... 29

4.25 Arm lattice rotation .................................................................. 30
   (a) ............................................................... 30
   (b) ............................................................... 30
   (c) ............................................................... 30

4.26 Arm after rotation .................................................................. 30

4.27 Arm Translation and Inversion ......................................................... 31

4.28 Arm’s translation, rotation and scale values set to zero for the new position .................................................. 31

4.29 Little Green Man with his new arms .................................................... 32

4.30 Resized and placed NURBS sphere ..................................................... 32

4.31 Increasing the spans to 8 for better deformation during animation .................. 33

4.32 Modeled right hand palm on left arm ................................................... 33

4.33 Scale Z negated to fix backwards hand issue ........................................... 34

4.34 NURBS sphere shape base for fingers .................................................. 34

4.35 Modeled fingers .................................................................. 35

4.36 Adjusting the palm to the fingers ...................................................... 35

4.37 Modeled thumb .................................................................. 36

4.38 Duplicating the palm and fingers ...................................................... 36
   (a) ............................................................... 36
   (b) ............................................................... 36

4.39 Values to make the hand for the right side of the model ......................... 37

4.40 Little Green Man and his new hands .................................................... 37

4.41 Basic foot shape .................................................................. 38
   (a) Front ............................................................... 38
   (b) Side ............................................................... 38

4.42 Foot shape with lattice .................................................................. 38
   (a) Side ............................................................... 38
   (b) Top ............................................................... 38

4.43 Deformed foot .................................................................. 39
4.74 Insert edge loop tool
4.73 Removal and moving of faces and vertices
4.72 Unnecessary faces inside the model
4.71 Maya's mesh combine tool
4.70 Left side of the model is removed
4.69 Created polygon shapes
4.68 NURBS to polygons options
4.67 Converting NURBS to polygons
4.66 Completed face
4.65 Control vertices (seen in purple) used to add detail
4.64 Basic sculpting done
4.63 Sculpt tool with reflection
4.62 Sculpt tool interface seen in red
4.61 Sculpting tool
4.60 Isoparms added to the face
4.59 Little Green Man and his eyes
4.58 Duplicate special options box
4.57 The eye is deformed and manipulated to fit the character
4.56 Eyeball and eyelid grouped together
4.55 Eyelid with detail
4.54 Eyelid with additional isoparms (seen in green)
4.53 Basic eyelid sphere
4.52 Sphere option box for eyelid
4.51 Finished eyeball
4.50 Eye ramp color change
4.49 Original ramp color and application
4.48 Eye material and color changes
4.47 Basic eye shape
4.46 Sphere option box
4.45 NURBS Primitives > Sphere > option box
4.44 Duplicated foot
4.43 Sphere option box
4.42 Sphere option box
4.41 Finished eyeball
4.40 Sphere option box
4.39 Sphere option box
4.38 Sphere option box
4.37 Sphere option box
4.36 Sphere option box
4.35 Sphere option box
4.34 Sphere option box
4.33 Sphere option box
4.32 Sphere option box
4.31 Sphere option box
4.30 Sphere option box
4.29 Sphere option box
4.28 Sphere option box
4.27 Sphere option box
4.26 Sphere option box
4.25 Sphere option box
4.24 Sphere option box
4.23 Sphere option box
4.22 Sphere option box
4.21 Sphere option box
4.20 Sphere option box
4.19 Sphere option box
4.18 Sphere option box
4.17 Sphere option box
4.16 Sphere option box
4.15 Sphere option box
4.14 Sphere option box
4.13 Sphere option box
4.12 Sphere option box
4.11 Sphere option box
4.10 Sphere option box
4.09 Sphere option box
4.08 Sphere option box
4.07 Sphere option box
4.06 Sphere option box
4.05 Sphere option box
4.04 Sphere option box
4.03 Sphere option box
4.02 Sphere option box
4.01 Sphere option box
(a) Top .................................................. 39
(b) Side .................................................. 39
4.44 Duplicated foot .................................................. 39
(a) .................................................. 39
(b) .................................................. 39
4.45 NURBS Primitives > Sphere > option box .................. 40
4.46 Sphere option box .................................................. 40
4.47 Basic eye shape .................................................. 41
4.48 Eye material and color changes .................................................. 41
(a) Phong material .................................................. 41
(b) Ramp color .................................................. 41
4.49 Original ramp color and application .................................................. 42
4.50 Eye ramp color change .................................................. 43
(a) Original .................................................. 43
(b) New colors .................................................. 43
(c) Final Colors .................................................. 43
4.51 Finished eyeball .................................................. 44
4.52 Sphere option box for eyelid .................................................. 44
4.53 Basic eyelid sphere .................................................. 45
4.54 Eyelid with additional isoparms (seen in green) .................................................. 45
4.55 Eyelid with detail .................................................. 46
4.56 Eyeball and eyelid grouped together .................................................. 46
4.57 The eye is deformed and manipulated to fit the character .................................................. 47
(a) .................................................. 47
(b) .................................................. 47
(c) .................................................. 47
(d) .................................................. 47
4.58 Duplicate special options box .................................................. 48
4.59 Little Green Man and his eyes .................................................. 48
4.60 Isoparms added to the face .................................................. 49
(a) .................................................. 49
(b) .................................................. 49
(c) .................................................. 49
(d) .................................................. 49
4.61 Sculpting tool .................................................. 50
(a) Edit NURBS > Sculpt Geometry .................................................. 50
(b) Pull tool .................................................. 50
4.62 Sculpt tool interface seen in red .................................................. 50
4.63 Sculpt tool with reflection .................................................. 51
(a) .................................................. 51
(b) .................................................. 51
4.64 Basic sculpting done .................................................. 51
4.65 Control vertices (seen in purple) used to add detail .................................................. 52
4.66 Completed face .................................................. 52
4.67 Converting NURBS to polygons .................................................. 53
4.68 NURBS to polygons options .................................................. 53
4.69 Created polygon shapes .................................................. 54
4.70 Left side of the model is removed .................................................. 54
4.71 Maya’s mesh combine tool .................................................. 55
(a) Mesh > Combine .................................................. 55
(b) Single polygon surface .................................................. 55
4.72 Unnecessary faces inside the model .................................................. 55
4.73 Removal and moving of faces and vertices .................................................. 56
4.74 Insert edge loop tool .................................................. 56
4.75 Edge loop tool in use ........................................ 57
(a) Before .................................................. 57
(b) After .................................................. 57
4.76 Cleaning up the polygon model ......................... 57
(a) Completed external connection ....................... 57
(b) Internal faces removed ............................... 57
4.77 Leg and body connected seamlessly ................. 58
4.78 Additional connections ................................ 58
(a) Before .................................................. 58
(b) After .................................................. 58
(c) Before .................................................. 58
(d) After .................................................. 58
4.79 Shoulder connection .................................. 59
4.80 Final polygon mesh ................................... 59
4.81 Mirror geometry ...................................... 60
4.82 Gap left by mirroring .................................. 60
4.83 Normals for Little Green Man ....................... 61
(a) Turn on normals ...................................... 61
(b) Initial normals ...................................... 61
(c) Normals facing the wrong way ..................... 61
(d) Fixed normals ...................................... 61
4.84 Before color ........................................... 62
4.85 New material with green color ...................... 62
4.86 Little Green Man is complete ....................... 63
5.1 Screenshot from ART [10] .............................. 65
5.2 Epic Games Menu ...................................... 66
5.3 Skeleton Creation Settings ............................. 67
(a) Body, Arm and Leg options ....................... 67
(b) Changed values seen in orange .................. 67
5.4 Add Rig Modules ....................................... 67
(a) Leaf, Jiggle and Chain joint options ............. 67
(b) Added chain joint with head as parent ........ 67
5.5 Joint mover tool ....................................... 68
5.6 Global, Offset and Mesh mover tools ............... 69
5.7 Physique menu ......................................... 70
5.8 Physique menu changes to mesh .................... 70
5.9 Final prototyped character mesh .................... 71
5.10 Saving the mesh for future use ..................... 71
5.11 Saving the mesh for future use ..................... 72
5.12 Saving the mesh for future use ..................... 72
5.13 ART interface for animation ....................... 73
5.14 FK arm movement .................................... 74
(a) Before .................................................. 74
(b) After .................................................. 74
(c) After .................................................. 74
5.15 Auto Hips ............................................. 74
(a) Off ..................................................... 74
(b) On ..................................................... 74
5.16 ART squash and stretch ............................. 75
5.17 IK arm movement .................................... 76
(a) Before .................................................. 76
(b) After .................................................. 76
5.18 Maya time slider (boxed in red) ..................... 76
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.19</td>
<td>Keyframes denoted in white, interpolated frames in red</td>
</tr>
<tr>
<td>5.20</td>
<td>Ball hierarchy</td>
</tr>
<tr>
<td>5.21</td>
<td>Auto Keyframe</td>
</tr>
<tr>
<td>5.22</td>
<td>Graph Editor</td>
</tr>
</tbody>
</table>
| 5.23    | Graphs of Y-axis change with different tangents  
  (a) Plateau tangents  
  (b) Linear tangents |
| 5.24    | Blocking for squash and stretch  
  (a) Initial keyframe with no scale change  
  (b) Squash on impact  
  (c) Stretch on bounce  
  (d) Reset to round sphere at top of bounce |
| 5.25    | Graph Editor for scale |
| 5.26    | Keyframes for rotation  
  (a) Initial  
  (b) Final |
| 5.27    | Graph Editor for rotation |
| 5.28    | Screenshots from final bouncing ball animation |
| 6.1     | Initial sketch of Polly |
| 6.2     | Polly concept art |
| 6.3     | Polly reference images  
  (a) Front  
  (b) Side  
  (c) Top |
| 6.4     | Polly model  
  (a) Top  
  (b) Front  
  (c) Side  
  (d) Perspective |
| 6.5     | Polly head detail  
  (a) Top  
  (b) Front |
| 6.6     | Polly’s texture map with no color |
| 6.7     | Polly’s texture map with color |
| 6.8     | Final Polly model top view |
| 6.9     | Final Polly model  
  (a) Back  
  (b) Side  
  (c) Front |
| 6.10    | Final Polly model detail |
| 6.11    | Main keyframes of walking animation  
  (a) Frame 0 and 32  
  (b) Frame 8  
  (c) Frame 16  
  (d) Frame 24 |
| 6.12    | Jump keyframes  
  (a) Frame 0 and 32  
  (b) Frame 8 and 24  
  (c) Frame 16 |
| 7.1     | Halo’s Master Chief 2001 - 2012 [2]  
  Sheep character from Disney’s Zootopia [9] |
CHAPTER 1

INTRODUCTION

When compared to mathematics or physics, the field of computing is but a baby still in its cradle. While this may be true, it is one strong baby. Over the past five decades or so, computing has gone from a field of study found only in the wealthiest universities and research laboratories, to a global phenomenon. Nearly every person in America has used a computer at one point in their life, and most carry a portable one in their pocket. This drastic and amazing transformation of computers, their uses, sizes and sheer capabilities is one of the most wondrous things to happen in the late 20th and early 21st centuries.

The majority of Americans own a computer, cell phone and at perhaps a gaming console. It is hard to avoid these three things in today’s internet and entertainment focused society. As a result, when asked, most everyone can name at least one video game. This fact is something that students at MIT in the 1960’s could not begin to appreciate, as at that time, the only "video game" in the world was located in their laboratory on one of the world’s first modern computers, PDP-1 [19]. The game was titled Spacewar! and was a simple program where two players controlled "rocket ships" that launched torpedoes at each other. This game was born out of a desire to show the calculating power of early computers. At one point the game’s creators were asked about monetizing the game to which they replied, "We thought about ways we could make money off it for two or three days, but then concluded it couldn’t be done” [19]. This simple game would go on to inspire some of the most innovative and creative minds and start what is now a billion dollar industry.

1.1 History of video games

As with many other revolutions, gaming was not initially a huge success. The first ever "gaming console" to grace the world was the brain-child of Ralph Baer, titled the Magnavox Odyssey 1TL200.
The simple console, launched in 1972, plugged into a television and had twelve playable games with simple black and white graphics, and no sound [19]. Other games were being developed at the time, but they ran on large university computers that offered no public access. So for a time, gaming was only available to researchers.

Walt Disney did not invent animation, but no one disputes that he is the greatest proponent of the art form. If anyone could claim to be the Walt Disney of electronic games, it is Nolan Bushnell. Few have heard this name, but on July 27th, 1972 the then 27-year-old Bushnell and his two partners filed for the incorporation of their new company, Atari. They set up shop in Silicon Valley and began working. Bushnell told one of his new engineers, Alan Alcorn, that he had landed a contract with General Electric to create an arcade game based on ping-pong. Bushnell said, "The game should be simple, one ball, two paddles" [19]. The reality was, that there was no contract. He merely wanted to motivate Alcorn. This simple act was the first step in the creation of PONG.

The first version of PONG was a simple game that ran through a bargain store TV and had one simple instruction: Avoid missing the ball for a high score. To test out the game, Bushnell attached a simple coin operated device, and placed it in a tavern in Sunnyvale, CA. After two days the game stopped working, but upon inspection it turned out that the coin box was so backed up it had jammed. By the end of the week the bar owner had people lining up to play the game before he even opened for the day. The game pulled in $300 a week, six times the amount of any other pinball games in the bar [19]. PONG is considered one of the great breakthroughs in graphics, as before all computer activity was text-based. Just a few years later Atari was a 2 billion dollar company.

In Japan gaming was growing just as quickly, where in 1978 Space Invaders was released, and in 1980 Pac-Man was released, becoming one of the most successful video games in history. We all know gaming didn’t stop there, and in 1982, the first version of Microsoft Flight Simulator was launched on the new IBM PC. Games that followed in this first "golden age" of video games include Donkey Kong and Mario, both from the now well known company Nintendo.

Up until this point, all games were visually primitive. Most games used sprites-simple graphic shapes, little more than bitmaps, to convey information. 1983 saw the beginning of a shift away from these simple graphics when Disney director Don Bluth partnered with the creators of Space Invaders to develop games such as Dragon's Lair and Space Ace. Instead of geometric shapes darting around the screen, Bluth and his team created Disney-quality cartoon animations. Every possible move and plot within the game was stored on a disc inside the game unit. These games became just as fun to look at and watch as they were to play, and some consoles broke from overuse in the years when Dragon’s Lair was one of America’s most popular games.
1.1. History of video games

By the end of 1982, the video game market had reached a saturation point. There were too many games that all looked and sounded similar. This began the “Great Video Games Collapse” [19]. It was during this lull that Japan became the dominant force in video games. Nintendo introduced eight and sixteen bit home consoles that used raster graphics, as well as launched one of the most popular and lucrative game franchises, *The Legend of Zelda*. The Soviet Union even got in on video game fame when Alexey Pajitnov, a 29 year old student, invented the world’s most purchased cellphone game today, *Tetris* [19].

The 1990s became known to gamers as the years of the console wars, with the release of Nintendo’s NES, Sony’s Playstation and Microsoft’s Xbox. The first handheld game devices arose from this time period as well, with the most prominent being the Nintendo Game Boy. In 1996, the world was introduced to *Pokemon*, and has yet to part with the beloved franchise. The gaming communities’ first major female hero, Laura Croft hit stores in 1993, and is still one of the most recognized characters in gaming [23]. The console wars continued into 1996 when Nintendo released its first 64-bit game system, aptly named the Nintendo 64. It sold half a million units on the first day, and that year Nintendo boasted over 1 billion game cartridges sold [23].

As millennials who were raised on gaming grew to young adulthood, games grew with them. Adult-oriented games like *Grand Theft Auto*, *Mortal Kombat* and *Call of Duty* appeared. Vector graphics yielded to raster graphics and flat staging abdicated to three-point, in-depth perspective. Games began to feature realistic movement, richer story lines and cinematic scenes. Bestselling authors

Figure 1.1: *Dragon’s Lair* [13]
like Tom Clancy and Douglas Adams wrote plots and scripts. Oscar winning directors like Martin Scorcese and Spike Lee directed cut-scenes. Budgets on big games began to overtake that of movies and the salaries of artists and engineers soared [19].

In 2003, the total sales of video games surpassed the total box office of the movie industry for the first time. In 2011 the National Endowment for the Humanities declared interactive games an official art form [19]. Some games like *Journey* embody this fact more than ever imagined in the early days of video games. The graphics in games of today are indeed made possible by the advances in technology of the last few decades, but it is more than that. Characters, environments and creatures that artists could only dream of making 20 years ago can be made in the comfort of one’s own home. Realistic physics in games are now a possibility, with some games altering physics as a gameplay mechanic, such as *Portal*.

![Figure 1.2: Journey, created by thatgamecompany [14]](image)

A recent study by the Entertainment Software Association stated that 155 million Americans play video games, and 80% of households own a gaming system [4]. In America alone the total consumer payout for video games and consoles in 2014 was $22.41 billion [4]. Gone are the days of wondering if the video game industry will collapse again. As a generation raised on the internet and video games comes into its own, it is hard to imagine the field of computing without interactive gaming.
2.1 INTRODUCTION TO CHARACTER DESIGN

Character design in video games is a field nearly as broad and diverse as art itself. No two games are exactly alike, and even among video games where realism is the goal, stylistic decisions and variances keep them from appearing identical. Genre, platform, studio and target audiences change what a game looks like and even how the characters themselves behave and move. Despite these factors, when developing a character it is paramount to understand the human form and how it is perceived.

Human and humanoid characters, regardless of stylistic choices and designs, must look and behave realistically in order to correctly represent the human form. One of the most basic forms of measurement used when drawing the human form is head height. This is a simple way to ensure that a complex character maintains proportions, as the unit of measure is part of the character itself. Character age, rather than sex, is the most important factor in how many heads tall a character should be. A typical adult will be 8 heads high, teenagers 7 heads, older children 6, younger children 5 and infants just 4 heads high [12]. These measurements are by no means rules, but rather guidelines. The character Sackboy from LittleBigPlanet, seen in Figure 2.1, has a head height of 2, making his body the same size as his head [20]. This ratio results in an appearance that is cartoonish and childlike. This exaggeration of head height is not the only way to make a character appear a specific way. The Halo franchise’s main character, Master Chief, is a fantastic example of how a character with a realistic head height can be made to be an imposing figure by simply increasing the size of his head, thus increasing his total height without compromising his body proportions.
In addition to height and correct proportions, character designers emphasize body type when designing characters. There exist three main body types: ectomorph, mesomorph and endomorph. These body types categorize body shapes based on the body’s ability to store fat and build muscle. Ectomorphs are defined as having tall, lean builds with little body fat. Mesomorphs can be considered the naturally athletic body type with clearly defined muscles. Endomorphs have a higher proportion of body fat to muscle and tend to be stocky with strong arms and legs [20]. See Figures 2.2 and 2.3 for examples of these body types.

Although a connection between body type and personality is complete nonsense in the real world, video game characters tend to have traits projected onto them by the player due to observation of body type. Along with particular design choices, ectomorphs may be considered introverts, creative, or shifty. Conversely, a mesomorph body type can suggest courage, dominance or competitiveness.
2.1. Introduction to character design

Figure 2.3: Female representations of Ectomorph, Mesomorph and Endomorph [20]

And finally endomorphs tend to portray funny, social and strong characters. It is interesting to note that in recent big budget video game franchises such as *Halo*, *Uncharted*, and *The Elder Scrolls*, the mesomorph body type has become the go-to body type for both sexes, resulting in an unbalanced amount of idealized body types in video games.

These idealized characteristics are even more pronounced when looking at the differences between male and female characters in video games. While the male and female form vary greatly upon reaching puberty, video games tend to take these differences to extremes in order to emphasize ideas of masculinity and femininity. Historically, video games have been marketed towards young males and thus these over exaggerated ideals were an easy way to attract consumers [20]. Despite this trend, there are examples of video games that deviate from commonly held notions of body shape. Perhaps the best example of this is Samus Aran from the *Metroid* series. The main character is a powerful bounty hunter in the original *Metroid*, Samus’ design throughout the game is blocky and masculine, yet in the final moments of the game it is revealed that underneath the armor is a female character. Her suit suggests the broad shoulders of a male, however her chest to waist ratio in the suit upon closer inspection leans towards the proportions of a female figure. The choice made by Nintendo to not reveal Samus’ identity until the end of the game threw players for a loop, and sparked a movement to create more female game characters.

Body type and gender differences are not the only factors to consider when designing or analyzing video game characters. There are numerous physical cues that players notice that also contribute to how a character is interpreted. Attraction is a major factor in video games, and while beauty is in the eye of the beholder, it is generally accepted that symmetry is more attractive than asymmetry [20]. However, it is important to note that perfect symmetry is often viewed as being alien and unsettling. Male characters with exaggerated masculine traits such as broad shoulders, and strong
limbs are considered more attractive than skinny and weak males. Conversely female characters with soft facial features and narrow waists convey an air of femininity. In addition to attraction, the visible and apparent health of a character contributes to a player’s opinion. The three main categories that determine apparent healthiness are: skin texture, skin color and amount of body fat. Blotches, spots and variations in pigment are often associated with weak immune systems whereas smooth skin with little variation indicates health. Skin that has an underlying tone of red is considered more attractive than skin that is less saturated. This is due to the natural blushing of the skin as blood flows through it, and is perceived as healthier than the skin of someone who has less oxygenated skin [20]. Additionally, the amount of yellow tone in skin is an indicator of health as well. A light tan is typically considered more attractive than skin that is paler. The amount of yellow and red contained in the skin must remain at a moderate level or else the character appears unhealthy. Finally the weight of a character has a tremendous impact on the character’s perceived health. Obviously a grossly overweight or underweight character is perceived as unhealthy and thus impacts a player’s opinion.

Perhaps one of the most important visual characteristics of a virtual character is age. While proportions play a factor in our ability to perceive age, there are a number of other clues character designers consider when defining the age of a character. The younger a character is perceived to be, the more exaggerated facial features are. A large forehead and eyes might be paired with a small mouth, chin and nose [20]. These features create a type of cuteness that humans tend to associate with children. At the other end of the scale, the prominent aspect of age is skin texture. As a person ages, biological and genetic factors cause skin to lose its elasticity and create wrinkles. Yet the amount of wrinkles a character has is not always directly correlated to age, as someone who has lived a hard life exposed to the elements will have aged much more quickly than a character that lived an easy and posh life.

As we have seen, there are numerous details that go into creating a video game character. Game style, genre and story can greatly influence how a character looks, yet at the core of character design we find simple rules and decisions that can determine how a character is perceived by players.
CHAPTER 3

COMPUTER GRAPHICS AND MODELING

One of the most common statements anyone who works with computers hears is something along the lines of, "Well the computer does all the work for you!" Anyone who has heard this phrase can tell you just how untrue it is. Yes, most work done with computers requires immense computational power and resources, but sitting down at a computer and producing something is not always as easy as a few clicks here and there and presto! Perhaps the individual who most often hears that a computer does the work for them is the digital artist. As a relatively new medium, computer generated art is sometimes seen as inferior. Any digital artist will contest this opinion, as who is to say that the undo function, CTRL + Z, is not the same as an eraser. Computer generated art can be anything from paintings to advertisements to 3D animated shorts. This broad category is typically called computer graphics, or CG. Another common name for 3D graphics is CGI or computer graphics imagery. The two are often used interchangeably, and although technically digital painting and other 2D creations fall in these categories it is an industry convention to use CG and CGI to refer to 3D graphics.

3.1 INTRODUCTION TO COMPUTER GRAPHICS

Most 2D software is bitmap based, whereas 3D software is vector based [6]. Bitmap software creates a unified image through the use of pixels similar to a quilt. 3D software relies on mathematical models and instructions that create vectors from one point to another. These mathematical instructions combine to form shapes, lighting, textures and even animations. A computer is used to render the scene, converting the instructions into 2D images that can be linked together in a way similar to film with thousands of still images forming a single movie.
3.2 3D Modeling

Just as with any complex object in the real world, complex 3D models are composed of many smaller, simpler objects. While, there are a number of ways to make these simple objects, here the focus is on polygons and Non-Uniform Rational B-Splines, or NURBS.

3.2.1 Polygonal Modeling

Polygons are made up of faces, vertices and edges. The simplest polygons are triangles: three edges, three vertices and two faces, one for each side of the 2D triangle. While polygons can have any number of edges and vertices, in 3D modeling triangles and quadrilaterals are most commonly used.

Polygonal meshes are created when individual polygons are attached along the edges. These can combine to form more and more complex objects, from simple pyramids to spherical shapes. Figure 3.1 shows how quadrilaterals around the equator and triangles at the poles combine to form a sphere.

![Polygon Sphere](image)

**Figure 3.1: Polygon Sphere**

Polygon models are simple for a computer to render. They are used for gaming applications in which the model needs to be rendered in real time [6]. The higher the number of polygons on a model, the harder a system needs to work to render it. This is why the cut scenes and trailers of games appear to be of a higher quality than any actual gameplay, as these are pre-rendered and saved as video files, and thus not rendered in real time. Despite this, many games today have approached photorealism due to the huge advances made in gaming and modeling in the last decade.

In modelers such as Maya, primitives like cubes, spheres, cones and cylinders are the simplest objects that can be generated. They use basic geometric shapes to create simple 3D objects. Because
the level of detail on a primitive’s surface can be increased nearly infinitely, they offer great versatility through manipulation, and form the base for many great models.

3.2.2 NURBS modeling

NURBS (Non-Uniform Rational B-Splines) are a part of a larger set of non-uniform rational cubic polynomial curve segments. In general, the polynomials in a rational curve can be Bezier, Hermite or any other type. Using B-Splines gives us NURBS. B-Splines consist of curve segments whose polynomial coefficients depend on just a few control points [7]. Cubic B-Splines approximate a series of \( m + 1 \) control points where \( m \geq 3 \), on a curve consisting of \( m - 2 \) polynomial curve segments. When \( m \geq 4 \), there is a join point between each curve segment called a knot [7]. In uniform B-Splines these knots are spaced at equal intervals, while non-uniform B-Splines allow for these knots to be at unequal intervals. This alters the blending functions at each interval as they vary curve segment to curve segment. Rational curves are useful because they are unchanging under rotation, scaling, translation and perspective transformations of the control points or vertices. They also have the ability to precisely define any conic section, as a conic can only be approximated with non-rationals [7].

As NURBS are based on more organic mathematics, they can create smooth curves and surfaces. This is unlike polygons, which can only suggest curved surfaces. The difference between the two can be likened to the difference between raster images and vectors. When zoomed in on, a raster based image eventually blurs into its individual pixels. A vector image, however, can be zoomed in on infinitely and never experiences loss of quality. In Figure 3.2, the difference between polygons and NURBS can be clearly seen, as the NURBS torus on the right has curved lines representing the mesh while the polygon surface is made up of rigid quadrilaterals.

NURBS are typically used for applications where the rendering is done in advance, such as animations for films [6]. NURBS modeling excels at curves, and as such is used to model animals, humans and other highly detailed objects. NURBS surfaces are called patches, and while very powerful and detailed, can be tedious to work with when modeling.

At the base level of NURBS patches, curves are created with a control vertex (CV), or knot, at each end as well as a few other CVs in between to provide curvature [6]. Note that the only CVs on the actual curve are the start and endpoint CVs. The resulting curves are called isoparms, and make up NURBS surfaces. The surface is created between two isoparms to form spans that follow the
3. Computer Graphics and Modeling

Figure 3.2: Polygon torus on left - NURBS torus on right

surface curvature defined by the isoparms. The more spans and isoparms a model has the higher the level of detail, as well as the demand on the computer during rendering.

If the object or character being modeled requires smooth curves or organic shapes then NURBS are the obvious choice. They can be converted to polygons at any time with relative ease, but converting to NURBS from polygons can become a complicated and tricky process [6].

3.2.3 Tessellation

In the end, during rendering, all 3D models and objects are broken down into polygon triangles that form the surfaces that shape any object. This happens regardless of whether polygons or NURBS are used in the modeling process. The computer calculates the position of significant points on the surface of the model and then connects these points to form a polygon mesh, or skin that ultimately represents the model [6].
4.1 INTRODUCTION TO MAYA MODELING

Autodesk Maya, commonly referred to as just Maya, is a 3D computer graphics program used to model, animate and texture a wide variety of objects and scenes. Maya has been used in the creation of video games, animated movies and real world products. More information about the program can be found at its website [16].

When first opening Maya, the number of buttons, options and capabilities of the program are overwhelming. Some professional modelers have been heard to say that even after years of working with the program, they don’t feel like they know 100% of its functionality. Despite this, learning the basics of Maya is easy due to the sheer number of tutorials and books available on the program.

This chapter utilizes a variety of resources to model a simple character, these include [22], [17], [6], [21], [18] and [5]. Each of these books provides different tips, techniques and information about modeling and animating a wide variety of objects. References [6], [21] and [18] provide basic guides on how to use Maya and create simple models. While [22], [17] and [5] provide more detailed tutorials and information about creating human models so that the animation process is more straightforward.

4.1.1 FIRST LOOK AT MAYA

When first opening Maya, the user is presented with a blank scene as depicted in Figure 4.1, showing the major parts of the user interface (UI). This view is a 3D perspective view of a blank scene with a grid view along the x and z axes. The central part of the screen is taken up by the workspace which
is host to the viewports. Hitting the spacebar while hovering the mouse over this area opens all four viewing options as seen in Figure 4.2(a). Clockwise from the top right is the 3D perspective view, right, front and top view. These views all have an important role to play when modeling anything in Maya, as it is easier to think about an object in one plane and work in one dimension at a time, rather than trying to work in three dimensions. Clicking on one of these view volumes and then hitting the spacebar makes that view volume full screen again, allowing for easy transition between the four views. Holding down the spacebar pulls up Maya’s Hotbox, as seen in Figure 4.2(b). The Hotbox is attached to the current mouse location, allowing for easy access to menu items and tools. While all of the menus presented in the Hotbox are accessible through drop down menus, utilizing the Hotbox significantly cuts down on time spent crawling through menu after menu and streamlines the modeling process in Maya. The Hotbox is also customizable, allowing each user to fine tune Maya to their own personal preferences.

Maya requires the use of a three-button mouse, even on Macintosh systems. The left mouse button acts as the primary selection button, the right button activates shortcuts and the middle mouse button zooms. Using the left mouse button to click on an object selects it, this highlights the object and pulls it up in the channel box or attribute editor on the right hand side of the screen.
4.1. Introduction to Maya Modeling

Once an object is selected there are three different transformations that can be applied: move, rotate and scale. When an object and one of these tools is selected, a Manipulator appears at or around the object. The Manipulators are shown in Figure 4.3. Each of these Manipulators has a distinct appearance to avoid confusion and allow for quick recognition of the active tool.

Figure 4.2: Maya’s four viewports and Hotbox
4. Maya Modeling

Figure 4.3: Appearance of Maya’s three transform tools and their Manipulators

The color of each manipulator is also important, as red corresponds to the x-axis, green to the y-axis and blue to the z-axis. This gives the modeler another way to situate themselves in space when working with a model and rotating it. These tools are used in conjunction with many others to create the complex and beautiful models that we see in video games.

4.2 Basic character modeling

The easiest way to understand how to model in Maya is just to start modeling. This section details how to create a simple character in Maya.

4.2.1 Sketching the model

The first step in modeling a character in Maya is to design the character. Character design can be a very intensive process, with numerous factors contributing to the overall look and feel to a character. Most of these decisions are effected by the game that the character is being designed for. The sample character designed here has no indented game and as such is designed based on the need for simplicity. While designing the character, there are a few things that can be done to make the modeling process much easier.

When modeling a character images are imported into the scene to act as references when modeling. These images can be hand drawn sketches, detailed concept art or open source pose references found online. This section demonstrates how simple sketches can be used to model a character.

Most characters need to maintain some kind of proportion. The easiest way to do this is to sketch the basic character on graph paper. As shown in Figure 4.4, a front facing and side facing version of our character, Little Green Man, is drawn. These images are then scanned into a computer and
4.2. Basic character modeling

Figure 4.4: Simple character sketch on graph paper

manipulated in photoshop to produce separate front, side and T-pose images. Figure 4.5(a) shows the composite T-pose reference image. Additionally, a top down view of the arm to be used is seen in Figure 4.5(b). If desired, reference images for the character’s feet, back and more detailed versions of the front and sides could be provided. However these are all surplus and are not necessary to start the modeling process.

Figure 4.5: Additional reference images
4.2.2 Getting ready to model

The first step is to open Maya and set up a project. Maya automatically opens a new scene, so it is important to set the scene before doing anything else and save it for easy access later. The next step is to select ‘Set Project’ and set the destination folder where all the work is saved. In order to start modeling the Little Green Man we first set up our reference images. This entails creating a NURBS plane for each reference image, Figure 4.6(a). To set up the front facing reference, create a NURBS plane on the Z axis and rename it front_ref_plane. In the channels box on the right of the screen, rescale the plane to be 1/100 of the dimensions of our reference image (saving reference images with the names in the form “sidename-ref-heightxwidth” makes this process simple). Scaling the image makes the model have a more reasonable and workable size within Maya. This scaled plane can be seen in Figure 4.6(b).

![Make a NURBS plane](image1)

![NURBS plane scaled to 1/100 of reference image size](image2)

**Figure 4.6:** Making a NURBS plane for the reference image

Next the rendering editors and hypershade are used to add the reference image to the plane. Inside the hypershade window, seen in Figure 4.7 and Figure 4.8, create a new material called a lambert. This material isn’t shiny so it is easy to distinguish the object when working. We can’t use the lambert already in the hypershade menu as the lambert material is a default and changing it means changing the material of everything created already and for future objects. Double clicking on the new lambert, activates the attribute editor. Renaming the lambert to front_ref_lambert allows for easy access to the material later should it need to be changed. The reference image is attached by
4.2 Basic character modeling

(a) Open Hypershade

(b) Hypershade popup box

Figure 4.7: Making a new material for the reference image

clicking on the black and white checkered box next to "Color", as seen in Figure 4.9. Here we select
file and then locate our front facing reference image. Once that is done click on the hypershade
window again and with the middle mouse button click and drag the lambert onto the plane in the
main view volume. As long as textured views is enabled, the image should be visible on the plane.
This process is repeated for the side and top view reference images.

Figure 4.8: Create new lambert material

This process is repeated with each reference image until three reference planes and images are
created, as seen in Figure 4.10. The three planes are grouped together into a layer titled "ref_layer"
which is then set to "R" so that it cannot be interacted with, allowing for modeling without the worry of moving the reference images.

Figure 4.9: Set color to file and choose reference image

Figure 4.10: All three reference images set
4.2. Basic character modeling

4.2.3 Basic Modeling

With our reference images all set we can finally begin modeling a character. There are many different ways to do this, and the techniques used in the following example are just one of many ways to produce a character.

4.2.3.1 Torso and Head

To begin modeling a character, it is best to start with the main portion of the character’s body, typically the character’s torso and head. Since this part of a character can vary drastically and consist of some irregular geometries, a simple way to tackle it is by tracing a side of the character with an End Point curve, or EP curve, and then revolving this curve into a three dimensional object. The EP curve tool can be seen in Figure 4.11. When tracing the character it is important to trace just half of the character, as the bright green curve in Figure 4.11(b) demonstrates.

![EP curve tool](a) EP curve tool  
![The traced EP curve](b) The traced EP curve

**Figure 4.11:** EP curve tool

After tracing the side of the character, refinements to the curve can be made. Once satisfied with the curve, the revolve tool is used to rotate the curve around the y-axis creating a 3D object. The final trace and revolve tool can be seen in Figure 4.12(b). The generated NURBS surface in the shape of the profile curve is shown in Figure 4.12(a).
4. Maya Modeling

At this point the character looks the same from the front and the side, a fact that is rarely true to the final design of the character. Refinements need to be made to the generated surface to amend this problem. Creating a lattice around the surface allows for controlled changes. When generating the lattice it is important to take into account the number of divisions a torso has so that the character appears believable. For an accurate torso division, 10-14 horizontal or "T" divisions are recommended. The example character, Little Green Man, has 12 T divisions. The lattice tool can be seen in Figure 4.13.
4.2. Basic character modeling

Clicking and dragging is necessary for selecting the lattice points that need to be moved, as the 2D front and side views do not show the 3D nature of the lattice. Failure to select both points results in only half of the surface being modeled correctly, a mistake that is not easily amended. By clicking and dragging it is guaranteed that the modeler selects the lattice points in the positive and negative Z positions (if using the front view plane) and can avoid this issue all together. The final body shape and lattice changes for Little Green Man can be seen in Figure 4.14.

Once satisfied with the look of the surface and the structure of the lattice, the modeler needs to remove the lattice as well as unlink the EP curve and the surface. This is done by deleting the object history, as seen in Figure 4.15. It is important not to delete the history if there are any changes that need to be made to the surface, as doing so makes the changes to the surface permanent.

As seen in Figure 4.16, the reference images are just a guide to follow. Making artistic changes to the model is acceptable, and often happens once the character is viewed in 3D. The side profile of Little Green Man is modeled differently than the reference image, as we wished to change the way he carries his weight, a tactic often used to imply personality and character in video games.
Figure 4.15: Deleting the history of the NURBS surface makes changes permanent and removes the link to the EP curve

Figure 4.16: Changes to Little Green Man are made as the character is modeled
4.2. Basic character modeling

4.2.3.2 Legs

The next step is to give the character legs. The basic geometry of a leg is a long cylinder, once created we increase the number of spans to 12, seen in Figure 4.17. This is the correct number of divisions for a leg, and allows for realistic movement of the character once animated.

![Figure 4.17: A cylinder makes up the base of the leg](image)

Once the cylinder has the correct spans it is moved and scaled in relation to the reference images as depicted in Figure 4.18.

![Figure 4.18: Moved and scaled cylinder](image)

Just as with the body of Little Green Man, the next step is to make a lattice and begin refining the
shape of the leg. By starting at the hip and working down, switching between the front and side views, the leg begins to take shape as seen in Figure 4.19.

Figure 4.19: The lattice is moved to produce the shape of the leg

Once satisfied with the look of the leg, the history of the leg is deleted and the changes are made permanent. Rather than modeling the right leg on its own, it is much more time effective to duplicate the left leg. In order to make this new duplicate leg visually correct, the x translation and x scale of
the leg are made negative. This inverts and translates the leg to the correct position and saves the modeler the time it would take to model a separate leg. This translation and inversion is shown in Figure 4.20. For simple characters this tactic is acceptable, but characters that have defining features or clothing require that each individual limb is modeled separately so the character is not too symmetrical.

Once these changes are made to the duplicated leg, the character’s legs are complete. Note that the feet of the character are not modeled at this time. This is due to the fact that modeling them separately allows for better movement when animated.

Figure 4.21: Little Green Man and his new legs
4. Maya Modeling

4.2.3.3 Arms

Modeling the character’s arms is very similar to modeling the legs. We model one arm and then
duplicate it to represent the other arm. There are a few things that should be done to the arm during
modeling to allow for easy and realistic animation later on. To start, a NURBS cylinder is created
with 10 spans, see Figure 4.22. The cylinder is then resized and a lattice is placed around it, as seen
in Figure 4.23. Starting at the shoulder, the lattice points are moved to shape the arm. Figure 4.24
shows the beginning and end of this lattice deformation.

![Figure 4.22: The NURBS cylinder with 10 spans](image)

Note that in Figure 4.24(b) and Figure 4.24(c) the reference images are not followed exactly,
further emphasizing how reference images are by no means the defining point of a character’s
structure.

![Figure 4.23: The lattice is moved to produce the shape of the arm](image)
Here is where modeling the arms changes from modeling the legs. In an effort to model our arms with some semblance to real-life, it is necessary to rotate the last three lattice points of the arm. This provides better deformation of the arm during animation. To see why this is the case, examine an arm held out at shoulder height with the palm forward. Now rotate the hand palm down. The skin on the arm moves during this rotation. A realistic character model should exhibit this movement as well. In order to replicate this natural deformation, the last three lattice point rows are rotated at varying degrees until the last row is rotated 45 degrees forward. These transformations can be seen in Figure 4.25. The final product of these transformations is depicted in Figure 4.26.

**Figure 4.24:** Arm lattice deformation
Figure 4.25: Arm lattice rotation

Figure 4.26: Arm after rotation
After the arm is complete, the lattice is deleted and the changes made to the arm become permanent. The opposite arm is formed by duplicating the existing arm and negating the Translate X and Scale Y values as seen in Figure 4.27. Once satisfied with the arms, the transformations are frozen resulting in all values being set to 0, as in Figure 4.28. This concludes the modeling of the arms; Little Green Man and his arms can be seen in Figure 4.29.
4.2.3.4 Hands

Modeling hands is no easy task. As the primary way that a character interacts with the world, hands play a vital role in animation and the final character seen on screen. As hands need to be nimble and moveable in many different ways, they are not modeled as a solid object. Rather they are broken up into smaller sections, typically the palm and individual fingers.

Figure 4.29: Little Green Man with his new arms

Figure 4.30: Resized and placed NURBS sphere
4.2. Basic character modeling

The modeling of a hand begins with NURBS spheres and then the hand is manipulated using lattices into its final form, just as with the legs and arms. Figure 4.30 shows the original sphere, while in Figure 4.31 the number of spans in the sphere is increased to 8. Doing this allows for more deformation when animating and produces more realistic movement.

Figure 4.31: Increasing the spans to 8 for better deformation during animation

After modeling the palm, seen in Figure 4.32, it is observed that the reference image is set up backwards, meaning the right hand is being modeled on the left arm. This blunder can be fixed by using Maya’s amazing software without having to redo the modeling already completed. Figure 4.33 shows the values of our hand after making the Scale Z value negative. This inverts the shape in the Z direction and makes our right hand model into a left hand. This is just one example of how Maya can easily remedy modeler errors.

Figure 4.32: Modeled right hand palm on left arm
Once the palm is completed the fingers can be modeled. These start out as NURBS spheres, just as with the palm. They are pulled into an elongated shape, and a lattice is placed around them as in Figure 4.34. The deformed finger shapes can be seen in Figure 4.35. When modeling a finger, the 3D perspective view becomes the preferred working view as the other fingers interfere with modeling the side and top views.
While working in the 3D view it is observed that the palm model looks a little odd. Figure 4.36 shows how the palm is rotated and edited slightly to match the fingers. Finally the thumb is modeled as in Figure 4.37.
Just as in the previous parts of the character that are modeled, the hand shapes are duplicated and then translated and inverted to make the right hand as in Figure 4.38. The fingers are rotated so the right side of the shape is connected to the palm. These values can be seen in Figure 4.39.
Finally, it is time to give our character feet. Like many other parts of the character, feet start out as spheres, which are then stretched into a basic foot shape. This basic shape can be seen in Figure 4.41.
The spans of this NURBS sphere is changed to 6 from 8; this allows for the foot to deform in a more natural way when animated.

![Figure 4.41: Basic foot shape](image)

Once the basic shape is set, a lattice is placed around the foot to allow for further deformation. Before any deformation takes place, it is important to change the T divisions to 2 and the U divisions to 5. This can be seen in Figure 4.42. We do this so that the foot maintains a more or less flat bottom, while still changing the top of the foot. If there were more than 2 T divisions, the foot could have odd bulges in it, where in reality a foot’s mass spreads out from top to bottom.

![Figure 4.42: Foot shape with lattice](image)
In Figure 4.43 the deformation of the foot can be seen, and as in many other parts of the character, the reference images provide a basic guide rather than a final form. Finally in Figure 4.44 the right foot is duplicated, flipped and inverted to form the left foot. Thus the character finally has feet to stand on.

![Figure 4.43: Deformed foot](image)

![Figure 4.44: Duplicated foot](image)
4. Maya Modeling

4.2.3.6 Eyes

Little Green Man’s eyes are made using spheres. Unlike spheres that were used to make the other parts of our character’s body, when creating the eyes we need to open the option box and change some values of the sphere. This is done by clicking on the box next to the Sphere option in our NURBS Primitives as seen in Figure 4.45.

![Figure 4.45: NURBS Primitives > Sphere > option box](image)

This brings up the option box seen in Figure 4.46. The axis of the sphere is changed to Z so that the pole of the eye is pointing in the forward direction relative to the character and the spans are changed to 6. This creates the sphere seen in Figure 4.47.

![Figure 4.46: Sphere option box](image)

![Figure 4.47: Sphere created](image)
4.2. Basic character modeling

Figure 4.47: Basic eye shape

(a) Phong material

(b) Ramp color

Figure 4.48: Eye material and color changes

Up until this point, every object used the lambert material made at the start of the character’s modeling. Rather than try to add a skin to a blank sphere once modeling is complete, it is easier to add color and reflectiveness to the character’s eye while still modeling. This is done by changing the material of the eye sphere to Phong material, as in Figure 4.48(a). This is a reflective material that has similar reflective properties to natural eyes. To add color to the eye, the checkered box next to color is selected, which brings up the Create Render Node menu, seen in Figure 4.48(b). The Ramp
option is selected and is applied to the sphere as in Figure 4.49. Note that the color on the sphere is wrapped in a way that does not suggest the natural color of an eye.

![Figure 4.49: Original ramp color and application](image)

This is fixed by changing the Ramp from a "V Ramp" to a "U Ramp." Now that the color wraps the correct way, the colors of the Ramp can be edited. This change can be seen in Figure 4.50. When choosing colors for the eye it is extremely important to not use solid black or white. These colors create a visual death on the screen, which means that any part of the character where these colors occur, appears to be incorrectly rendered. Using a very light orange for the whites of the eye avoids this problem. Similarly, the pupil can be a very dark blue or brown, depending on the eye color of the character. Using ramps also allows for the iris to be multiple colors, and can be used to make some very interesting and unique eyes. After deciding on iris colors and positions for the transitions of these colors, a beautiful eyeball is created as seen in Figure 4.51.
4.2. Basic character modeling

With the eyeball completed, the next step is to make eyelids for the eye. The NURBS sphere option box is opened again and the axis is changed to X. Additionally the start sweep angle and end
sweep angle are changed to 55 and 355 respectively, seen in Figure 4.52. This creates an open portion of the sphere for the eye to look out of.

Figure 4.52: Sphere option box for eyelid
4.2. Basic character modeling

In the channel box for the new sphere, the value of Rotate X is changed to 206, and the sphere is scaled to 1.02 in all values. The resulting sphere is seen in Figure 4.53. Scaling the eyelid sphere to be slightly bigger than the eye allows for there to be no collision between the two spheres when animated.

![Figure 4.53: Basic eyelid sphere](image)

In order to add some dimension to the eye, additional isoparms, or horizontal breaks in the sphere, are inserted. Figure 4.54 shows the eyelid with these additional isoparms.

![Figure 4.54: Eyelid with additional isoparms (seen in green)](image)
In order to add even more detail to the eyelid, a bump is created at the edges of the eyelid by pulling some of the control vertices of the additional isoparms away from the eye, shown in Figure 4.55.

![Eyelid with detail](image)

**Figure 4.55:** Eyelid with detail

With the eyelid complete, the eye and eyelid are grouped together to avoid any chance of collision during animation later on. This is shown in Figure 4.56.

![Eyeball and eyelid grouped together](image)

**Figure 4.56:** Eyeball and eyelid grouped together

With the eye now completed it can be added to the character model. Just as with any other part of the character, a lattice is used to manipulate and deform the shape into it’s final form as seen in Figure 4.57.
4.2. Basic character modeling

Figure 4.57: The eye is deformed and manipulated to fit the character

Because the eye is a group of objects and not a single shape, duplicating the eye for the other side of the body is a little more complicated than other duplications. The duplicate special options box is opened and duplicate input graph is selected, seen in Figure 4.58. From there the duplicated eye is translated and inverted just as with the previously modeled body segments. Finally, Little Green Man has eyes as seen in Figure 4.59.
Figure 4.58: Duplicate special options box

Figure 4.59: Little Green Man and his eyes
4.2.3.7 Face Details

It is possible to add detail to the face of a character by adding additional spheres shaped by lattices, just as with the previously modeled parts of the character. Another technique is to sculpt the existing geometry; this is the technique that is used here.

To begin, additional isoparms are added to the head, as the number of isoparms on the head define its level of detail. Figure 4.60 shows the addition of these horizontal and vertical isoparms. Once all the isoparms are added, the character should look similar to Figure 4.60(d).

![Isoparms added to the face](image)

**Figure 4.60:** Isoparms added to the face

In order to add detail to the face, the sculpt tool is used, as shown in Figure 4.61(a). To sculpt the
eyebrows and nose, the pull tool is used as seen in Figure 4.61(b). The tool itself is a simple click and
drag tool, shown in Figure 4.62, that pulls the points where the isoparms intersect.

![Figure 4.61: Sculpting tool](image1)

![Figure 4.62: Sculpt tool interface seen in red](image2)

When modeling the eyebrows, the reflection on the X axis is turned on in the tool settings for
the sculpt tool, seen in Figure 4.63(a). This ensures that the tool reflects as in Figure 4.63(b) and the
eyebrows are sculpted symmetrical to each other.
4.2. Basic character modeling

The basic sculpting is done and the result can be seen in Figure 4.64. The mouth is modeled by using control vertices along the isoparms to indent a row and pull two additional rows together to suggest a mouth, as in Figure 4.65. Control vertices are also used to add additional detail to the nose and eyebrows. Once done adding detail, Little Green Man has a basic face, seen in Figure 4.66.

Figure 4.63: Sculpt tool with reflection

Figure 4.64: Basic sculpting done
4. Maya Modeling

4.2.3.8 Converting NURBS to Polygons

While not necessary for some modeling and animation, sometimes it is convenient to have a polygon version of a model made with NURBS. To begin this conversion, all the NURBS are converted to...
4.2. Basic character modeling

polygons using Maya’s “Convert” utility, seen in Figure 4.67. In the pop up box that appears, the
tessellation method used is control points, as seen in Figure 4.68.

Figure 4.67: Converting NURBS to polygons

Figure 4.68: NURBS to polygons options
With the conversion done, the Outliner box shows all the new polygon shapes made by Maya, Figure 4.69. In an act that at first appears counter intuitive, the left half of the character is now deleted by removing any faces to the right of the origin shown in Figure 4.70. Finally the separate polygon objects are combined into one shape as seen in Figure 4.71.

**Figure 4.69:** Created polygon shapes

**Figure 4.70:** Left side of the model is removed
4.2. Basic character modeling

On inspection it is obvious that there are a fair number of unnecessary faces and unconnected vertices in the polygon object as seen in Figure 4.72. Removing these extra internal faces reduces the number of polygons and thus reduces the complexity of the model. The faces are removed and the unconnected vertices moved around to connect to one another, as seen in Figure 4.73.

In the areas where a vertex is missing in a face that should connect to another face, an edge loop can be inserted, as shown in Figure 4.74. Figure 4.75 shows the leg before an edge loop is inserted.
and after the loop is added. While the addition of these loops increases the polygon count, they also allow the model to retain believable connection points between body parts.
4.2. Basic character modeling

Once the leg is connected externally as seen in Figure 4.76(a), the internal faces can be deleted as seen in Figure 4.76(b). The final connection of the leg and body can be seen in Figure 4.77. This process is repeated for each overlapping body part, and is shown in Figure 4.78 - Figure 4.79. The final polygon shape is shown in Figure 4.80.
Figure 4.77: Leg and body connected seamlessly

(a) Before  
(b) After  

(c) Before  
(d) After  

Figure 4.78: Additional connections
The model is now a polygon mesh and the right side can be mirrored to create the left side as shown in Figure 4.81. The mirroring is not exactly correct and often leaves a gap between the two halves as seen in Figure 4.82. This gap is easily closed by moving the vertices, and the result is a unified polygon model.

When dealing with polygons, it is important to check the surface normals, being calculated, as converting from NURBS to polygons sometimes incorrectly computes these values. To see the normals in Maya they must be turned on under Display > Polygons as seen in Figure 4.83(a). The initial normals for Little Green Man are seen in Figure 4.83(b). Note that there do not seem to be any normals on his belly or head. Closer inspection shows that the normals are inverted and facing inward as in Figure 4.83(c). Using Maya’s built in normals software the normals are recalculated, inverted and then reversed so that they all end up facing outward as seen in Figure 4.83(d).
With the normals fixed, Little Green Man’s conversion from a NURBS model to a polygon model is complete.
4.2. Basic character modeling

(a) Turn on normals

(b) Initial normals
(c) Normals facing the wrong way
(d) Fixed normals

Figure 4.83: Normals for Little Green Man
4.2.3.9 Adding color

As Little Green Man (LGM) is indeed a green man, he deserves to have a little color. The simplest way to do this is to set his body to a new material with a green hue. Figure 4.84 shows LGM before his new color, as well as the new material menu. Figure 4.85 shows the color change given to the new material and finally Figure 4.86 shows LGM with his new green body.

![Figure 4.84: Before color](image1)

![Figure 4.85: New material with green color](image2)
4.2. Basic character modeling

Figure 4.86: Little Green Man is complete
CHAPTER 5

MAYA ANIMATION RIGGING TOOLSET AND BASIC ANIMATION IN MAYA

The Maya Animation Rigging Toolset (ART), is a plug-in for Autodesk Maya developed by Epic, the company that developed Unreal Engine. It is a comprehensive toolset that takes a character through skeleton creation, skeleton placement, and rig creation, and also includes many animation tools [10]. Intended for simple creation of bipedal humanoid characters from the ground up, the tool was developed to be used in-house, but was released to the public and continues to be updated and modified. The original plug-in was written for the PC but various employees at Epic reworked the toolset for the Mac, allowing it to be used for this project.

Figure 5.1: Screenshot from ART [10]

The ART was created to allow individuals who wished to use Unreal Engine to create their own
characters. Epic boasts, "An entire rig can be published and sent to the animation team in under a minute" [10]. This is assuming that there is no preexisting model or mesh of a character to be used.

If rigging and animating is done in Maya, the model created in the previous chapters, Little Green Man, could be used. However rigging and animating in Maya is a very complicated and intensive process. Having discovered the ART, it made sense to dabble in rigging and animation using this simple and apparently intuitive system. Upon actually using the program it became clear that using a preexisting model in ART is not simple. Despite that fact, it is indeed possible to use the ART on a modeled character, as evidenced by a tutorial found online [11]. However, before adding in a character modeled in Maya let us first explore ART for its intended use.

5.1 Rigging

5.1.1 Skeleton Creation

To begin working in ART the "Epic Games" menu is opened and the "Character Rig Creator" is selected, shown in Figure 5.2. (The Epic Games menu option shows up once the ART plug-in is installed in Maya.) This brings up the Skeleton Creation Settings menu seen in Figure 5.3(a). The menu has Body, Arm and Leg options for skeleton creation. Within this menu the number of joints, twists and bones for the character can be changed. Note in Figure 5.3(b) how a changed value is highlighted in orange so that any modifications are very obvious to the user. Figure 5.3(b) also shows the "Same as left Leg" check box under the options for the right leg. When selected this ensures that the character has symmetrical leg bones and joints. There is a similar option for the arms and hands.

In addition to the normal skeleton creation options there is an "Add Rig Modules" option in the menu seen in the bottom of Figure 5.3(a) and in Figure 5.4(a). This subsection allows for the addition
of three different types of joints: Leaf, Jiggle and Chain. These joints are useful for adding additional movement and details to a character. Leaf joints can be used for things like jaws where another bone and joint is needed. Jiggle joints are exactly the same as leaf joints, however they have built in dynamics. They can easily be used as a "fat" bone in an overweight character or to add dynamics to loose clothing or pouches. Chain joints are exactly as they sound, a chain of joints. These have built
in physics and can easily represent ponytails or any other long and dynamic shapes needed. Any added joints are given a "Joint Parent" as seen in Figure 5.4(b). This is simply the connection point for the new joint.

Once satisfied with the changes made in the Skeleton Creation Settings, the blue button labeled "Skeleton Placement" at the top of the menu in Figure 5.3(a) is clicked, and skeleton creation is completed.

5.1.2 Skeleton Placement

Once the "Skeleton Placement" button has been clicked ART asks if these are the parameters to be used to build a skeleton. While the creators of ART say that it is possible to go back and change these values once set, it has not been an easy task. Thus double checking values and names of added joints is recommended at this point.

![Joint mover tool](image)

**Figure 5.5:** Joint mover tool

Once confident in the skeleton set up of the character, skeleton placement can continue. ART generates a rig and mannequin mesh shown in Figure 5.5. This is a joint mover which allows for the quick placement of a skeleton in a mesh. The mannequin does not have to be used, but in the
case that there is not a pre-made mesh, or the programer simply wants to prototype a character, the mannequin becomes an invaluable tool.

![Joint Mover Tool](image)

**Figure 5.6:** Global, Offset and Mesh mover tools

The joint mover tool has three modes, G, O, and M, seen on the righthand side of Figure 5.6. The first mode is the yellow Global mover. This tool moves the selected joint and everything down the chain as seen in the selection and movement of the shoulder joint in Figure 5.6. The second mode is the blue Offset mode that allows for movement of individual joints with no impact on joints up or down the chain. The final mode is the pink Mesh mode which only moves the mannequin’s geometry and has no effect on any joints.

Underneath the last mode button “M” there is a button of a silhouetted muscled figure. This is the Physique mode. When selected it pulls up the menu seen in Figure 5.7. This menu effects the mannequin’s body and can be used to prototype characters. Figure 5.8 shows just how varied a character the editor can create. Using this menu the mannequin example in Figure 5.7 is changed into the model seen in Figure 5.9. When editing the physique of the mannequin there is no need to match each side perfectly as the ART has a symmetry button which can replicate the structure of one side onto the other. Once happy with the prototyped character’s mesh, it can be saved to use on other characters made using ART as seen in Figure 5.10.

5.1.3 **Deformation Setup**

Once satisfied with the mesh and placement of joints the last step of rigging can begin. This step in ART is called deformation set up. The first step in ART is to create a Rig pose, otherwise known as
a T-Pose. This means that the pose of the model is changed so that the character’s feet and arms are world axis aligned. This means that the joints are straightened and the character assumes the T-Pose. By clicking create rig pose in the pop up that appears after selecting deformation setup, ART attempts to put the model into a T-Pose. This automated system is not completely accurate and thus once completed the user is presented with the generated T-Pose and a pop up as seen in Figure 5.11. From here the user can edit the pose and rig to ensure the best T-Pose.

Once done editing the T-Pose, Save Rig Pose can be selected. This enables the pop up seen in
5.1. Rigging

Figure 5.9: Final prototyped character mesh

Figure 5.10: Saving the mesh for future use

Figure 5.12. If the character being rigged is a prototype or does not have a preexisting model then selecting Yes is the best option. If there is a modeled character for this rig then No should be selected.

In the case of this model there is indeed no model that is used and Yes is selected.

This leads us into the skin weight menu. If Yes is selected in the above dialog then the mannequin
was automatically skin weighted. If not the skin weights need to be added to the mesh. When using the ART on LGM this tool is used, as he is an imported mesh and the mannequin is used.

After this step is complete the character rigging is complete and it is ready to animate. If at any time however the rig needs to be changed for the character there is an option to do so. This option is seen in Figure 5.2 and labeled "Edit Existing Character." In the pop up that follows, clicking "Edit Export Rig" opens the file and allows for editing of the character’s skeleton, deformation and skin weight.
5.2 Animation Tools

To begin animating a character in the Epic Games menu select "Add Character For Animation." This menu is shown in Figure 5.2. This brings up a pop up from which a character previously rigged using the ART can be loaded.

Once opened, the animation interface for ART is pulled up on the right side of Maya with the rigged character on the left as seen in Figure 5.13. The representation of the humanoid figure on the right is the joint picker and allows for selection of a specific part or parts of the character without having to select them in the model view. The joint picker greatly improves the speed of the model's placement and animation. The ART has a large number of features, not all of which can be covered here, but some that are invaluable to animating the human form are discussed in the following.

Some interesting features of ART are addressed next. The first is called "Auto Hips" and is used to align and rotate the hips between the two legs. When a leg is moved then the hips readjust, as when a human walks, the hips do not remain facing forwards, rather they rotate and sway naturally. Figure 5.15(a) shows the hips and legs positioned without Auto Hips on and Figure 5.15(b) shows them with Auto Hips enabled.
In addition to auto hips, ART has built in squash and stretch abilities for hands, feet and spine. The two are independent so the user can decide how much of each they want in their animations,
5.3. Animation in Maya

shown in Figure 5.16. The values can range from 0 to 1, allowing for a wide range of possible animations using these tools. Squash and stretch are two of the most useful characteristics that can be implemented in a character to suggest weight, as giving visual weight to an animated character can be extremely difficult. For example a cartoon character squishes when getting ready to jump and then stretches once in the air suggesting gravity is acting on the character.

![Figure 5.16: ART squash and stretch](image)

Another feature of the ART is called "Auto Spine." This feature is similar to Auto Hips in that it can be turned on and off, but Auto Spine keeps the middle section of the spine averaged between the hips and chest.

By default the limbs and the spine are in inverse kinematics (IK) mode, as opposed to forward kinematics (FK). The two are different options used for animating. The simplest way to differentiate the two is the effect they have on the movement of a limb. IK allows for the wrist to be selected and translated and the elbow and shoulder move with the wrist. Moving the shoulder moves the hand slightly, but it attempts to stay in its original position as much as possible. FK does not allow for the translation of child objects. Rather it only allows the rotation of joints and child objects move with it. See Figure 5.17 for IK movement of an arm and Figure 5.14 for FK movement. Notice how the IK movement only takes one translation of the wrist, while the FK takes rotations of parent and then children joints to fully move the arm.

These are just a few of the numerous features of the ART that make animating the human figure easier.
5.3 Animation in Maya

Animating in Maya is hardly different from how old hand-drawn cartoons were made. A series of still images are strung together in quick succession to produce what our brains interpret as motion. Each of these individual drawings are called frames, their equivalent in Maya can be seen at the bottom of Figure 5.18. Keyframes are frames in which the animator creates a pose for a character or whatever is being modeled. In CG a keyframe can be set for nearly any aspect of an object, whether it is color, size, position or other value that can be changed. Unlike classical animation, Maya animators do not need to set or draw each and every frame in a scene. Rather they can set as few or as many as they want. In the frames where the pose of a model is not specifically set, Maya interpolates the in-between frames. The keyframes from a bouncing ball can be seen Figure 5.19
denoted by white frame numbers. The red frames represent just two of the many frames that Maya interpolates.

![Figure 5.19: Keyframes denoted in white, interpolated frames in red](image)

This a very basic and over exaggerated example of a ball bouncing up and down. The next section details a more accurate animation of a bouncing ball.

### 5.3.1 Bouncing ball animation

A classic example of animation in Maya is that of a bouncing ball. This example uses translation, rotation, and squash and stretch to emulate a bouncing ball.

First the animator creates a ground plane and a ball to bounce. The ground plane is at \((0, 0, 0)\) and gives the ball something to bounce against. The ball is a unit sphere initially located at \((0, 1, 0)\), where the positive Y-axis is up. Before any animating can be done it is important to set up a hierarchy of the animation effects on the ball. Figure 5.20 shows the hierarchy for the ball. This ensures that translation is applied to the ball, then scaling and finally rotation.

The first step in any animation is called blocking. This is a basic rough edit of the animation’s base. In the case of the ball, it is the location of impacts with the ground, and the high point of its trajectory. In frame 1 of the time slider, and in the outliner seen in Figure 5.20 "translate" is selected so the ball can be moved to it’s position at the beginning of the animation, \((-8, 10, 0)\). Then Auto Keyframe is turned on as seen in Figure 5.21. This feature sets a keyframe in the current frame if any translation changes are made.
Thus the first keyframe is set. Moving to frame 10 the ball can be translated to (-4.75, 0, 0). Moving to frame 20 the ball is translated back up the Y-axis and along the X-axis. This pattern continues with the ball traveling less in X between each keyframe and not bouncing as high in Y, thus suggesting the ball comes to a rest and is no longer bouncing. After about 110 frames the ball is again resting on the ground plane and not moving.

Once all the translations are made the animation Graph Editor can be opened. This is one of the most powerful tools that an animator has in Maya. Every moment that is set in Maya generates a graph of value versus time. The Graph Editor grants direct access to the curves generated by an animation and allows for fine tuning. Keyframes are represented on the curves as points that can be freely moved to adjust this timing or value. This is a very mathematical approach to animating, and can be useful if you have a mathematical background. Figure 5.21 shows the graph editor for the translation of the bouncing ball example. The green line is the change in the Y value over time and the red line is the change in the X value. From this it can be seen that over time the ball is rising and falling while moving along the X at a slowing rate.
5.3. Animation in Maya

Each keyframe point on the curve has tangency handles that control the curvature of the curve. The nature of the curve itself has an impact on the way the animation behaves. Take for example the two graphs seen in Figure 5.23. The two graphs represent the change in the Y value of the ball over time. They both have the same values for Y at the keyframes yet they look very different. This is because they have different tangents at the keyframes, that is the curve is being manipulated in different ways on each graph. Figure 5.23(a) has plateau tangents on every keyframe, that pull the curve to a flat point. Figure 5.23(b) has linear tangents on the low points of the graph.

What does this mean for the ball? In Figure 5.23(a) the ball’s speed slows down as it reaches each keyframe and speed up gradually towards the next keyframe. This is not representative of a bouncing ball as it is known when it is observed, that a ball does not gradually speed up to its next highest bounce point. Rather it impacts the ground and then bounces back up with a nearly identical speed, allowing for some loss of energy due to gravity and friction. Thus the animation graph is changed to that seen in Figure 5.23(b). This graph more accurately represents the vertical motion of a bouncing ball as it collides with the ground and bounces off quickly.
At this point the ball correctly translates as a bouncing ball should. However there is no “bounce” to the ball. That is to say that the ball does not squash when hitting the ground and stretch when lifting off again. To animate this the ball’s scale is blocked and edited just as seen with the ball’s translation. Figure 5.24 shows the basic blocking for the squash and stretch of a bouncing ball. Just as with translation the amount of squash and stretch seen in the ball will decrease as the height of the bounce decreases and the ball comes to a stop. The Graph Editor for the scale changes of the ball can be seen in Figure 5.25. Very similarly to the translation graph in Figure 5.23, the graph of the squash and stretch decays over time. That is the amount of movement or change seen in the ball decreases, this is an intentional pattern and is used to mimic the loss of momentum and effects of gravity that a real ball would encounter.
At this point in its animation, the ball appears to correctly bounce as well as deform. However, the ball always bounces perfectly on its bottom pole and never rotates. The odds of a ball exhibiting this type of bounce in the real world are slim to none, so the final step in animating a bouncing ball is to add rotation. This is the simplest part of animating a ball as there are only two key frames necessary: the initial point with zero rotation and the final point with -480 degrees of Z axis rotation. The two key frames can be seen in Figure 5.26. The Graph Editor for the rotation can be seen in
Figure 5.27. Notice that this graph is linear unlike the graphs for translation and scale. This is because the ball’s rotation never speeds up or slows down, rather it maintains a constant speed of rotation. If the animator wanted to make a hyper realistic animation of a bouncing ball this graph would be different as the ball’s rotation would in actuality slow down as the ball came to a stop, but for the purposes of this example a linear graph works perfectly well.

![Figure 5.26: Keyframes for rotation](image)
After rotation is implemented the bouncing ball animation is complete. Screenshots from the final animation can be seen in Figure 5.28.

This simple example shows all the basics necessary to animate more complex models. The workflow remains the same and while there is more time involved in setting keyframes for complex models.
Figure 5.28: Screenshots from final bouncing ball animation
CHAPTER 6

RESULTS

This chapter covers the software created for this project not previously discussed elsewhere in this paper.

6.1 FEMALE MODEL

In addition to Little Green Man, this project also explores the creation of a detailed female character. Given the name Polly, after the polygons that she is composed of, the following subsections cover her development.

6.1.1 SKETCHES AND DRAWINGS

Figure 6.1: Initial sketch of Polly
Just as with many creative processes the first step is to begin sketching out ideas. Polly was sketched many different ways, however, as her companion is an alien figure, her first iteration had her depicted as an astronaut of sorts. This evolved into an explorer in a modern jacket. The final sketch of Polly can be seen in Figure 6.1. As Little Green Man and Polly are not intended for a specific video game or even genre, their design is kept rather neutral.

After sketching out the basic idea of Polly, a more detailed image of her is drawn and edited in Adobe Photoshop [3]. This image can be seen in Figure 6.2.

Figure 6.2: Polly concept art
6.1.2 Modeling

In the interest of modeling Polly to be more anatomically correct than LGM, reference images for Polly are not hand drawn. Rather, images of a model made using an interactive character creator called Fuse are used as references for Polly [1]. These references can be seen in Figure 6.3.

As with LGM, reference images are not closely followed at times and thus using this basic female as the foundation of Polly is an understandable decision. Similarly to LGM, Polly’s sketches and concept art did not wind up representing her final form. In a professional game development company this would not be the case, as a modeler would receive detailed concept art and references from the company’s art department. Polly also varies from her concept art as in her final model she is not wearing a jacket and pants but rather a bodysuit. This is partially due to the reference images not containing the clothing, as modeling detailed clothing with no reference is a challenge to even experienced modelers.

![Figure 6.3: Polly reference images](image-url)
Polly’s final base model can be seen in Figure 6.4 and Figure 6.5.

Figure 6.4: Polly model
6.1. Female Model

6.1.3 Texture Mapping

Unlike LGM, Polly’s model does not lend itself to being a single color. Her detailed model requires a more complex surface. The solution to this is to use texture mapping. This process takes all the
polygons that make up Polly and places them on a single plane. This flat representation, or mapping, can then be saved as an image and colored. Maya has the ability to link with Adobe Photoshop and create a PSD (Photoshop Data File) version of this mapping. This allows for quick edits to be made on the texture map and then viewed directly on the model in Maya.

Figure 6.6: Polly’s texture map with no color
6.1. Female Model

Figure 6.6 shows the PSD mapping of Polly before editing. At this point Polly would show up as entirely white as none of the mapping has any color. Figure 6.7 shows the PSD file after being edited. With these changes Polly’s model is complete and can be seen in Figures 6.8 - 6.10.

Figure 6.7: Polly’s texture map with color

Figure 6.8: Final Polly model top view
Figure 6.9: Final Polly model
6.2 Animation

Figure 6.10: Final Polly model detail

6.2 Animation

Animating detailed characters is very similar to the previously shown simple bouncing ball example. The only difference is these animations have more moving parts and can take significantly more time to perfect. When animating a simple walk there are in theory only four main keyframes needed. These frames can be seen in Figure 6.11. Note that Figure 6.11(a) is both the first and last frame of the animation. This is to ensure that the animation loops correctly and delivers a smooth animation. However, when the animation is exported, the frames used would be 0 to 31, as leaving in frame 32 would cause a stutter in the animation.

Polly has 34 different joints, and animating a walk can change from a simple task to quite an undertaking in no time at all. Due to this fact her walk animation consists only of movement below the waist at the time of this writing.

Little Green Man was also rigged for animating and he was used to explore a jump animation. The main keyframes for this animation can be seen in Figure 6.12.
6. Results

Figure 6.11: Main keyframes of walking animation
6.2. Animation

Figure 6.12: Jump keyframes
CHAPTER 7

CONCLUSION AND FUTURE WORK

Video games are a form of entertainment sure to remain at the forefront of technological advancement and innovation for the foreseeable future. The last few decades alone have seen a drastic and remarkable evolution in game detail and realism. Franchises such as Microsoft’s Halo paint a clear and remarkable picture of just how much games have changed in a few short years. Figure 7.1 shows how the level of detail in one character has improved in just over a decade.

![Figure 7.1: Halo’s Master Chief 2001 - 2012][2]

Animated films are another facet of computer graphics in which new and amazing advancements are being made. Disney’s newest movie, Zootopia, features brand new technology with the ability to create numerous different type of realistic animal fur [9]. The coat of a rabbit is vastly different from that of a sheep, and Disney’s new technology allows for an extremely accurate depiction of these varying textures. Figure 7.2 shows the base model and final render of a sheep character from the movie.
These new and powerful technologies allow for the creation of characters and worlds only dreamed of in past years. This project intended to examine and explore how characters in video games are brought to life. Two characters were created for this project using a tool that professional character designers use. A toolkit made by a game engine company was used to rig and animate these characters. This project is by no means comprehensive examination of character design; however the reader should have an understanding of the basic steps necessary for character creation.

7.1 Future Work

Character design for video games in a professional setting is nearly never a one man job. Typically, a team of artists, modelers, riggers and animators work together to produce a single character. A significant amount of time and effort went into the characters produced for this project. However, there is always room for further exploration and development.

For example, the level of detail on Little Green Man is exceptionally low. Polly’s model is indeed more detailed, yet her original design had to be diverged from in order to save enough time for texture mapping and animating. Even given a decent amount of time to accomplish these two last tasks, the quality desired from each was not fully achieved. Polly’s texture mapping lacks detail on her face and hair, and these two areas could be given unique texture mappings to achieve greater detail.

In regards to animation, this topic could become an entire project of its own. The brief exploration of animation in this project made it clear that a detailed understanding of anatomy and human movement are needed to produce fluid and realistic animations.
Finally, although this project is drawing to a close, there is still work to be done with Polly and Little Green Man. They were both rigged and animated using Unreal Engine’s toolkit for Maya. This will allow for an easy import process into Unreal Engine itself. From there the characters can be used to populate a video game or another project.

The vast worlds and detailed adventures that exist in video games have become a major part of many peoples’ lives. The characters in these stories remain with players long after the games are completed. It is a combination of a character’s design, history and personality that brings these games to life. Hopefully this project sheds light the immense amount of work that goes into each and every character in a video game. The long hours spent staring at a screen are well worth it when a character that previously existed only in concept comes to life. From fantasy to reality, this paper has provided the reader with a small peek into the complex process of creating realistic and dynamic characters for the enjoyment of gamers worldwide. I hope you have enjoyed the view!
REFERENCES


11. NA. Unreal engine 4 rigging a custom asymmetric model with art. URL https://www.youtube.com/watch?v=KH76mXiRV10.


