Piia Brusi

# MAKING A GAME CHARACTER MOVE

## Animation and motion capture for video games

Bachelor's thesis
Degree programme in Game Design

2021

South-Eastern Finland
University of Applied Sciences

South-Eastern Finland University of Applied Sciences

| Author (authors) | Degree title | Time |
|---|---|---|
| Piia Brusi | Bachelor of Culture and Arts | May 2021 |

**Thesis title**

69 pages

Making a game character move
Animation and motion capture for video games

**Commissioned by**

South Eastern Finland University of Applied Sciences

**Supervisor**

Marko Siitonen


**Abstract**

The purpose of this thesis was to serve as an introduction and overview of video game animation; how the interactive nature of games differentiates game animation from cinematic animation, what the process of producing game animations is like, what goes into making good game animations and what animation methods and tools are available.

The thesis briefly covered other game design principles most relevant to game animators: game design, character design, modelling and rigging and how they relate to game animation. The text mainly focused on animation theory and practices based on commentary and viewpoints provided by industry professionals. Additionally, the thesis described various 3D animation and motion capture systems and software in detail, including how motion capture footage is shot and processed for games. The thesis ended on a step-by-step description of the author's motion capture cleanup project, where a jog loop was created out of raw motion capture data.

As the topic of game animation is vast, the thesis could not cover topics such as facial motion capture and procedural animation in detail. Technologies such as motion matching, machine learning and range imaging were also suggested as topics worth covering in the future.



**Keywords**

3d, animation, game design, motion capture

**CONTENTS**

# 1 INTRODUCTION

The purpose of this thesis is to serve as an introduction and overview of video game animation; how animating for games differs from animating for non-interactive media, what the process of producing animations for games is like, what goes into making good game animations and what animation methods and tools are available.
As animators need to actively cooperate with various other departments, this thesis starts with a brief look into some of the principles the work of which affects the animators the strongest: game design, concept art, modelling and rigging.
In an attempt to narrow down the subject and due to lack of time and personal experience, this thesis will not be covering facial motion capture, rigging nor creature animation, at least not in detail. Instead, the focus will be on game specific animation theory, full body animation and motion capture for humanoid characters. Included at the end of this thesis is a small-scale project demonstrating the step-by-step process of processing mocap footage into a jog loop.

# 2 RESEARCH METHODS

There exist many sources covering the theory behind general animation principles, but very few game industry experts share their game animation experiences in written format. Instead, majority of the sources for this thesis consist of video essays created by industry professionals as well as publicly available recordings of Game Developers Conference talks and lectures.
In 2020 Jonathan Cooper published his book Game Anim: Video Game Animation Explained which served as major source of information for this thesis as well as some of the aforementioned video essays. As the five fundamentals of video game animation are built on the twelve basic principles of animation, those will be covered as well based on Frank Thomas and Ollie Johnston's book The Illusion of Life: Disney Animation (1995).

Parts of the text, especially the project portion are influenced by the author's limited personal experience with animation and motion capture, as well as

teacher lectures and peer discussions at the South-Eastern Finland University of Applied Sciences and Breda University of Applied Sciences.
Material from other academic institutions was also used to fill in information about the various motion capture systems.

# 3 GENERAL VIDEO GAME CHARACTER PIPELINE

## 3.1 Game design and pre-production

Game designers come in many flavors and in some ways, it is hoped that everyone involved in a game is a designer. As an overarching principle, game design is heavily based on back and forth between all the teams or individuals within a project. It is about working around limitations and finding middle ground everyone can work with, all while keeping in mind what the goals and core ideas of the game are.

What a designer needs to be able to do varies greatly. In small-scale development teams a designer might be responsible for figuring out nearly all aspects of the game while in larger companies the designers are split into more specific principles with the different areas of responsibility. As an example of more specialized roles a narrative designer is responsible for the story and world building of the game. If there is dialogue, there needs to be scriptwriters. For the purposes of visual storytelling, storyboarding may also be necessary. Meanwhile a gameplay designer is responsible for the moment-to-moment interactions: what obstacles the player can encounter and what methods they can choose from their given arsenal to overcome said obstacles. In addition to these, there are many more roles to take, from system design and programming to level design, puzzle design, sound design and so on. (Brown 2021.)

Animators are heavily tied to other designers and need to have good awareness of the other principles in order to be successful. According to Cooper (2021), they must actively work together with various designers, programmers, artists, audio technicians and more. After all, game design effectively dictates what the game is and how it functions.

Most essentially to animators gameplay designers dictate what characters and actions are necessary for the game in the first place, artists create the characters and props for the animators to animate, level designers provide the stage for the props and characters to live in and programmers implement everything. Additionally, there is the technical designer, whose role is that of a cross-disciplinary bridge between animators, designers, and engineers, who works to support and build the animated world as well as sustain content requirements (Davis 2017).

The process of making a game usually starts from a core idea, be that a game mechanic, a setting or a story. What the core idea is can change during development, but in order to get everyone on the same page as to what the main goal of the game is and what features should receive priority and emphasis during development, many developers use something called game pillars. Game pillars are essentially a set of core ideas that the game is founded upon and allow a team of developers to highlight and focus on what is important and what is not. Game pillars may relate to narrative focus, gameplay features or the experience the game is expected to provide. For some games, animation might play a significant part in these pillars, while in others less so. (Cooper 2021.)

Despite the fact that design dictates what is required from the animators during the production phase, animators can inspire game mechanics or visual features in the same way concept art can.

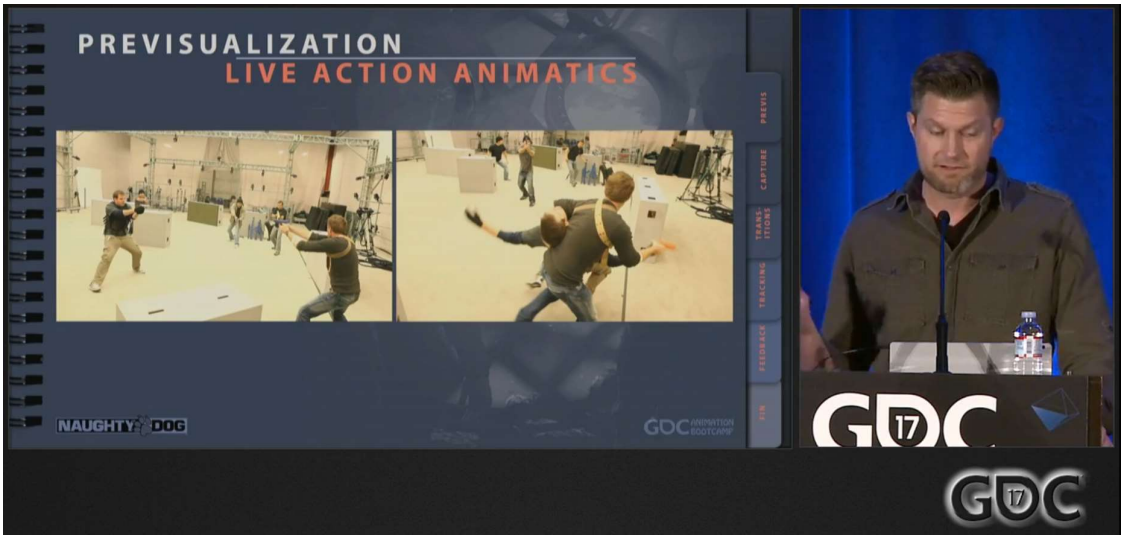



Figure 1: Jeremy Yates talking about the previsualization process for Uncharted 4 (GDC 2017).

One crucial way animators may participate in the concepting and pre-production phases of a game is by previsualizing what the gameplay should look like. This can be done by creating animatics or even by acting out gameplay in live action (figure 1) together with artists and designers, which allows the programmers and other technical experts to continue building the underlaying structures of the game uninterrupted. When a mockup emulates desired real-time gameplay without the game itself yet functioning it is called target footage. Target footage especially can serve as an important guideline for how the game is supposed to play and perform later in development. (Yates 2017.)

Once the core gameplay concept, mechanics and features as well as project scope have been set, the real production phase for all principles and teams can begin.

### 3.2 Character design and modelling

As mentioned above, concept art and animation can affect the mechanical design of a game during the concepting phrase, but the two affect each other greatly during pre-production and production as well. After all, the background knowledge required for both animators and character designers has multiple points of overlap. In terms of visual design and execution, both animators and designers need to have a clear understanding of silhouette, dynamic design

features, body mechanics in terms of weight and volume as well as appeal. These features are expanded further upon with the 12 basic principles of animation, which were originally designed by Disney animators Frank Thomas and Ollie Johnston in 1981 with traditional 2D animation and illustration in mind.

Character design and concept art are rather vast topics on their own, but to condense the process briefly: character creation starts from dozens of concept iterations, going from the overall silhouette to details and colors. The most common ways of concepting are drawing from scratch, photobashing and painting over image bases, but it is also possible to iterate on designs by sculpting directly onto some base model. Photobashing here refers to mixing together elements from multiple images, such as textures or limbs.

Sculpting as a term is quite self-explanatory; it is constructing and shaping a three-dimensional figure, though in this case the process is entirely digital. In a digital format, 3D models consist of large quantities of one-sided surfaces called polygons, which can further be broken down to their edges and points called vertices. A full model of a character or object, whether sculpted or modeled is also called a mesh.
A character sculpt usually consists of thousands of polygons due to their high amount of detail and smoothness, making them very performance heavy. As such, the 3D models used in games are usually not the original sculpts but simplified low-poly reconstructions; the process of reconstructing a model using less polygons is called retopologizing.

There are some design details animators should keep an eye on while a character is being designed and modelled. For example, both animators and riggers need to know which parts of the mesh should be flexible or rigid. If there is armor or other solid features, they should not interfere with points of articulation. Apparel, limbs, and props should not restrict movement or else a certain degree of mesh intersection aka clipping needs to be accepted. Additionally, an animator should be able to tell what areas of a model need extra attention so they can bend and deform comfortably. Areas with fewer polygons

are easier on performance but much stiffer and do not deform well, so the concentration of polygons and poly loops should be higher at points of articulation. (Cooper 2021.)

### 3.3 Rigging

The process of making a character model movable starts with the creation of a skeleton. The closest comparison to a digital skeleton would be an armature (figure 2), as they too consist of only bones and the joints, although in some cases simulated muscles can be added to a digital skeleton for fidelity.

Figure 2 (right): CUZIN family of mechanical armatures (Malvern Armatures no date).

In a digital skeleton each bone tracks their positional and rotational values relative to the joint they are connected to and use as a pivot point. Each joint can be a “parent” to multiple bones with the hip joint acting as the most common core, creating a hierarchy of joints and bones based on how the bones relate to each other (figure 3).

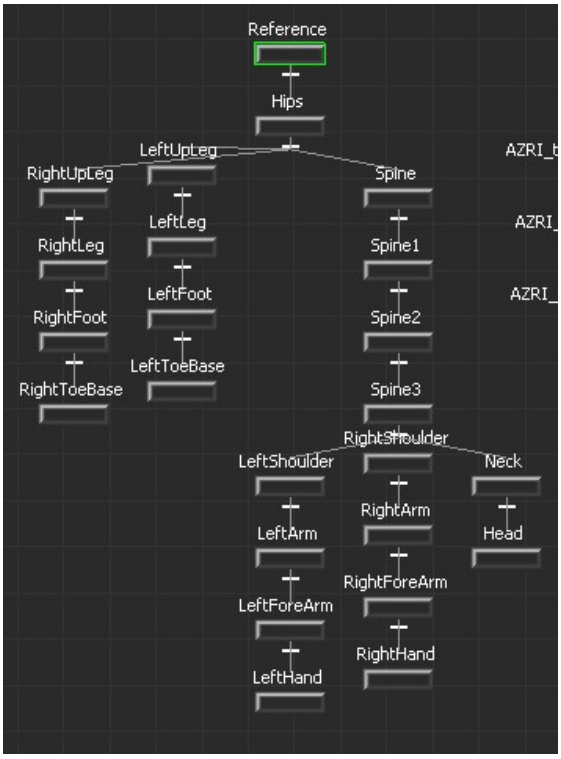


Figure 3 (left): An example of bone hierarchy in a humanoid figure (Brusi 2021).

Skeletons are initially built superimposed and separate from the model and once done, the skeleton is attached to the model in a process called skinning. Skinning involves defining what bones correspond to what surfaces of the model and to what degree; this process is also called weight painting.
According to Cooper (2021) the automated skinning ability of 3D software has improved greatly over the years, but manual work may still be required to a degree.

Though a skeleton can be posed as is, and what is ultimately exported into the game as animation is the positional and rotational values of the bones, animation is usually done with the help of rigs. A rig is an extra controller that drives the skeleton and often features easier to select points of control and various constraints for joint and bone movement. Other features that can be added to an animation rig include collision controls, IK/FK swapping, attach points for props and attribute sliders. In figure 4 of the AZRI rig the spline shapes around the body are controllers for the limbs, while the spheres in front of the legs are pointers that dictate where the knees are pointing. The facial controllers and bones for

fingers, hair and clothes are hidden by default, but become available by turning them on in the control layers.

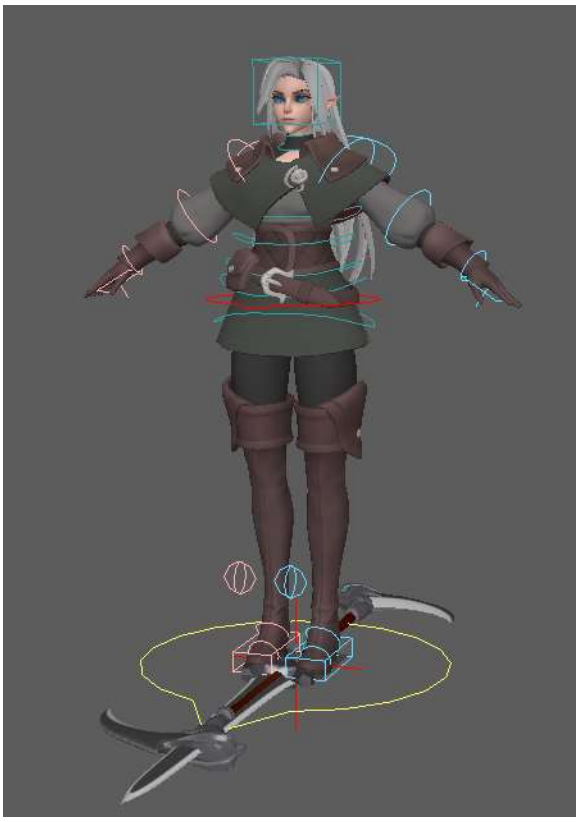


Figure 4: AZRI rig in Autodesk Maya (Brusi 2021).

The hierarchy of the rig is almost always different from the export skeleton's, but it makes manipulating the skeleton much more animator friendly. All in all, rigging takes considerable amounts of time and resources, so one point of consideration for both riggers and animators to address during character creation is whether several characters can utilize the same rig. Though animators at smaller companies may also be responsible for rigging, what rigs can do and how to construct them efficiently is a whole another topic to cover. (Cooper 2021.)

## 4 ANIMATION

Animation is the art of chaining separate images or poses together to create the impression of movement. The 20th century animation mostly followed the standards and visual language of live action film storytelling and many of these

standards are still in use today, but with technological improvements the limits imposed by camera set ups of old have been broken. (Cooper 2021).
While animation is animation regardless of the purpose, there is one major difference separating cinematic animation from video game animation: interactivity.
Unlike in cinematic animation, there can be no strict moment-to-moment vision for gameplay to follow, as player input dictates what actions are happening at any given moment, meaning gameplay animation can be considered non-linear and unscripted.

According to Cooper (2021) cinematic cutscenes are a rare opportunity for the developers to author scenes of a game so they can play exactly according to their vision. However, because games are supposed to be interactive and a more immersive experience, cutscenes are considered a double-edged sword that need to be used sparingly, lest they divorce the player from the story, world, and player character.
While the term cutscene is used to refer to most game scenarios where control is taken away from the player until the scene ends, a cinematic cutscene is a high fidelity pre-rendered video. Because cinematics can be done in a different program and may even have bespoke assets, they may have bespoke teams as well or the work can be outsourced to different companies. As an example, Sega has outsourced cinematic work to Blur studios for some of their Sonic games, particularly Sonic the Hedgehog in 2006 and Shadow the Hedgehog in 2005. (Floyd 2021.) Non-cinematic cutscenes on the other hand are generally done in-engine and using in-game assets including camera systems.

Gameplay animation as a whole consists of three states constantly flowing in and out of each other: loops, links and linears, also called cycles, transitions, and linear actions. (Cooper 2021.)
Loops are the standard constant actions the player is likely going to be looking at the most during gameplay: idling and walking as an example. As the name(s) might suggest, these are animation clips of varying lengths that are defined by

the fact that the first and last frame of the animation match, meaning that they can loop over and over again seamlessly.
Links are the transitional animations used to blend two actions together, such as the start of a run, crouching down, standing up, equipping an item and turning to face a direction.
Lastly, linear actions are animation clips that play only once from start to end. In combat for example, each attack is a linear action and once a singular animation is done playing, the action may flow to the next part of a combo or transitions back to the idle combat stance. Brief linear animations called inserts or fidgets can be used to break up the monotony of a constant loop. For idles this can mean the character stretching, looking around or taking a glance at a watch before returning to their usual idle loop. (Root 2020.)

When and what animations play, which animations may override others and how animations blend into each other is determined by state machines and blend trees. State machines are essentially visual diagrams that include all the actions a character can be in the state of doing such as walking, jumping, falling, and attacking. Sometimes various sub-states and animation blends happen within a single state; for example, a jump may be broken down to the state of jumping and falling. Blend trees determine when and how animations blend in and out of each other and are most commonly used for character movement. (Davis 2017).

### 4.1 The twelve basic principles of animation

The twelve principles of animation were originally described by Disney animators Frank Thomas and Ollie Johnston in their 1981 book Disney Animation: The Illusion of Life (later republished as The Illusion of Life: Disney Animation in 1995). They are a set of guiding principles to producing drawings that give a character physicality and appeal, as well as make the animations more effective at selling actions and emotions. These principles were originally designed for traditional pen-on-paper animation but majority of it still applies to the different methods used today.

Squash and stretch

Squashing and stretching of an object is used to emphasize the object's speed, weight, and mass while it is in movement.

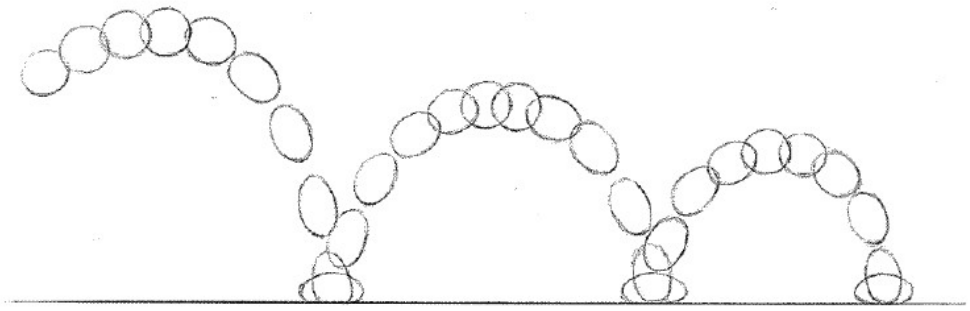


Figure 5: The ball animation test (Williams 2001).

The ball test (figure 5) is a famous tool for demonstrating this and many more principles; soft and bouncy objects tend to stretch out and compress more while moving, while solid and rigid objects remain mostly unchanged. In order to sell the physicality of an object, its overall mass should remain constant even while deforming, which is why the ball also elongates while being squished from the sides. (Johnston & Thomas 1995.)

Because the current AAA gaming landscape tends to favour realism, squash and stretch tends to be applied to lesser extremes in game animation. Additionally, because games are rendered in real time, complex deformable character rigs that save every bone's scale in addition to position and rotation lead into high memory and performance costs, making the technique a harder sell for gameplay animation. (Floyd 2019.) Rigs and models can be and often are pushed further for prerendered cinematics since the whole scene is compressed into a simple 2D video file at that point. Regardless, matching the fluidity of 2D animation with 3D models has only been achieved recently by films such as Hotel Transylvania series.

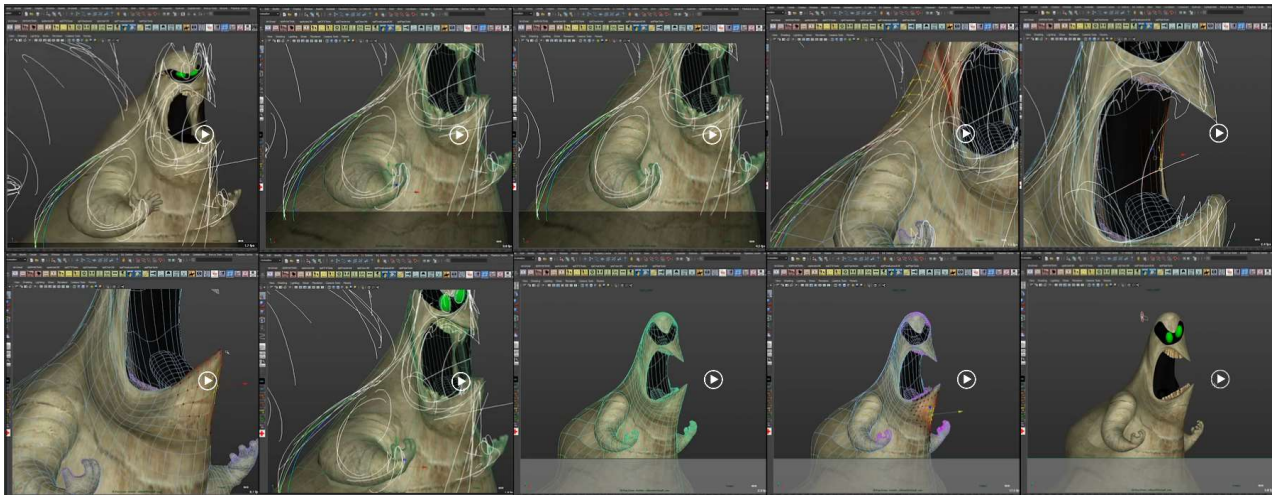


Figure 6: Screenshots of the pose sculpting process for Hotel Transylvania 2 (Sony Pictures Imageworks 2015).

As seen in the figure 6, the models in the movie were basically sculpted repeatedly to create all the key poses, with 2D storyboards and drawings as a reference (Failes 2015). Big budget movies can afford to do full productions like this both in terms of time and workforce investment, but in games it is used in a much smaller scale.

Stretching or sometimes doubling an object or a limb to cover a larger distance for a couple of frames can be used to sell speed, in which case it is called a smear or a blur. For the performance reasons above, smears are not used as much in 3D animation, but they were vital for making 2D animation feel smooth and fluid as well as to sell heavy actions with fewer frames.
Like in the Hotel Transylvania example, to create smear frames in 3D animation, the mesh of the model itself needs to be manipulated instead of bones, since smears are more often than not supposed to break the usual structure of an object. In figure 8, McCree's arm stretches and bends in an unnatural way for a couple of frames to sell the speed at which he whips out his gun. Smearing in real life can be witnessed as a trail of colour by a person waving an object such as their hand in front of them. (Stoeber 2021.)

Figure 8: A McCree smear frame (Blizzard Entertainment 2016).

Lack of mesh deforming for the purposes of gameplay animation is commonly compensated for by exaggerating pose compression and extension for a frame or two, but games like as Jak and Daxter (figure 7) clearly utilize squash and stretch in places where the topology allows for it, such as the torso, arms, and legs. (Floyd 2019.)

Figure 7: Squash and stretch principle in action in Jak and Daxter (Sony Interactive Entertainment 2001).

### Anticipation

Anticipation is the build-up and preparation for an action. The intent is to make the action read as clear as possible and theoretically, with good anticipatory posing, the audience should be able to recognize the upcoming action even if it cut to black before the actual execution. (Becker 2017).

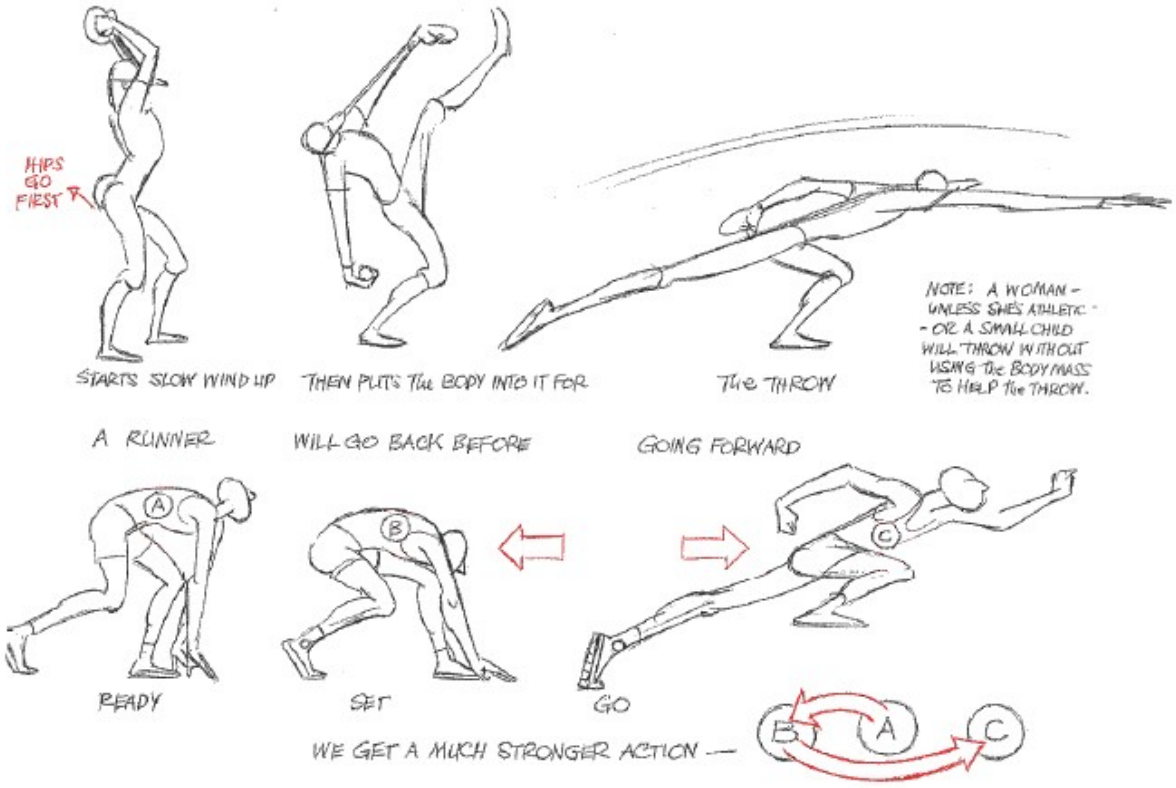



Figure 9: Examples of anticipation (Williams 2001).

An example of real-life anticipation can be seen when a baseball batter twists their body and arms in the opposite direction to their upcoming swing, or how a runner's body leans backwards before springing into a run (figure 9). Sometimes anticipation is used as a red herring to something completely unexpected and the action is purposefully interrupted as a surprise gag. (Johnston & Thomas 1995.)

Anticipation is a very important principle when it comes to game animation, where it is often paired with exaggerating, compressing, and extending a pose. It is common for the length or exaggeration of the anticipation to match and signal the strength of actions. Light and fast attacks are weaker while slow heavy attacks are strong and require more effort. In situations where a gameplay designer or an animator wants to really sell the power of a strike and make it a high-risk-high-reward deal, a longer anticipation period is used. Anticipatory action on the enemy side is also called telegraphing. The purpose of a longer anticipatory phase here is to give the player time to read what is about to happen and to respond accordingly.

In order to make the game respond to player inputs as fast as possible to maintain game feel however, the number of anticipatory frames are often sacrificed and the frames that are there are made as large and clear as possible. As a trade-off, games tend to utilize longer recovery periods instead of windup to sell the effort that was put into the action. (Cooper 2021.)

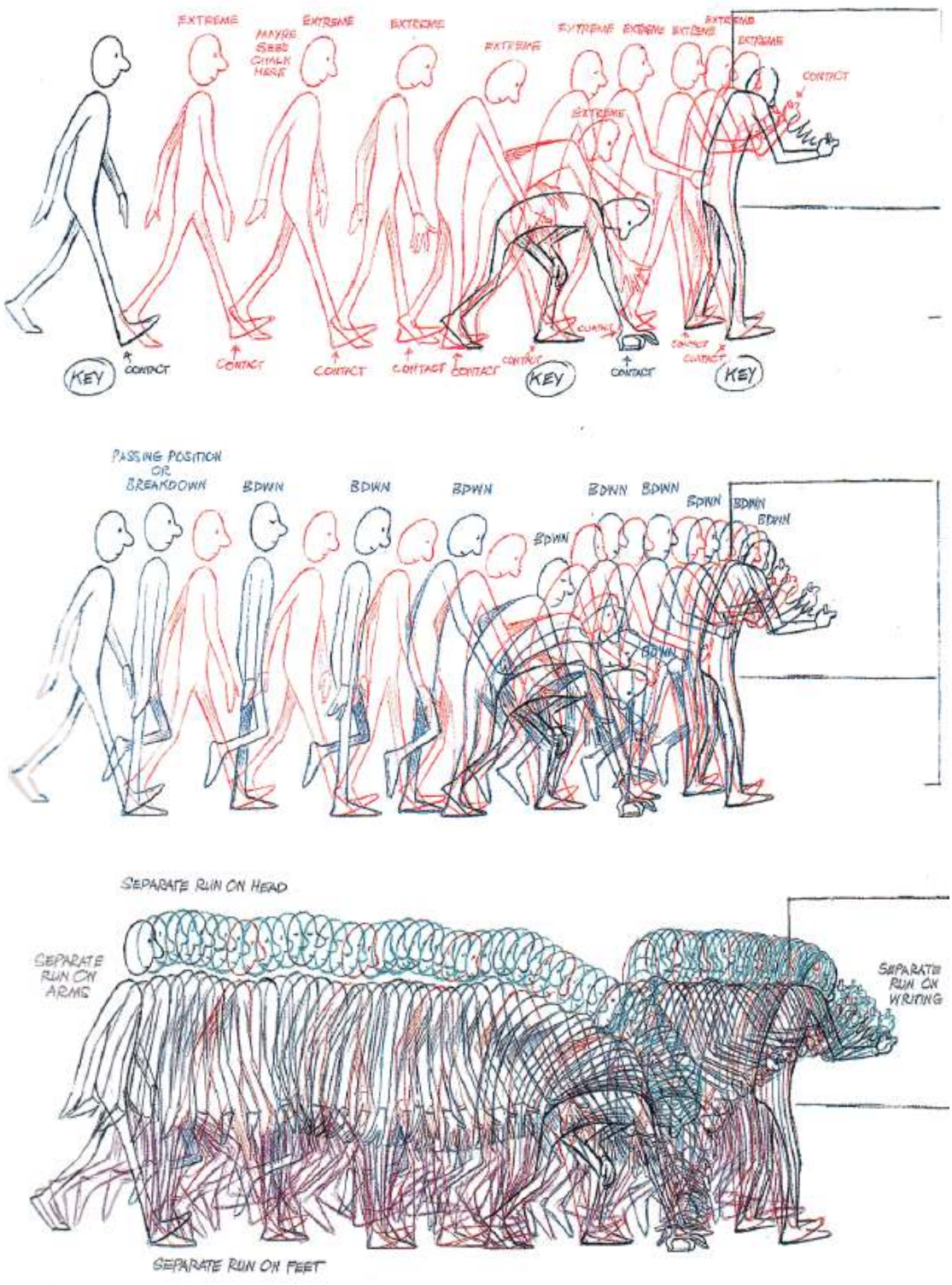
### Staging

Staging, also called framing, is the presentation of any idea so that it is completely and unmistakably clear. This principle can be applied to many aspects of animation, from posing to timing, camera angles and the setting of the scene itself, and other principles may be used to support good staging. Essentially, it is about controlling the audience's eyes to what is important in the scene at any given moment. (Becker 2017.)

Johnston & Thomas (1995) advise showing only one action at a time and emphasize the importance of a clear and readable silhouette. In video games noise is harder to avoid because what is on screen and what is moving from moment-to-moment is mostly out of the hands of the animators. Instead, staging is most relevant to those responsible for the gameplay and level design. There, it can be applied to make the camera focus on specific features when necessary, filtering out unnecessary information, as well as to guide the player within a level with other visual elements such as lighting. (Cooper 2021.)

### Straight ahead action and pose to pose

Simply put, in traditional 2D animation straight ahead action means drawing all the frames in an animation in their chronological order, offering spontaneity. In pose to pose, an animator first draws what are called key poses and fills in the transitions between those poses later, offering clarity and predictability. More precisely as pictured in figures 10-12, the stages or layers of pose to pose go from the key poses to secondary poses called extremes that take the character the farthest they or their limbs will go in each direction, to breakdown poses connecting the extremes and finally all the rest in-between to smooth the transitions out. (Johnston & Thomas 1995.)




Figures 10-12: Breaking down the types of frames in an animation (Williams 2001.)

The straight ahead action method is effectively unused in 3D animation. Most game animations require perfect looping so the animator must be absolutely sure that the starting and ending positions match, which is practically impossible

without planning ahead pose to pose. Additionally, in 3D animation the software can estimate transitionary poses between two known ones by the method of interpolation, meaning the animator does not have to manually pose every single frame. Lastly, 3D animation tends to rely on iteration over multiple animation passes, meaning edits to timing and posing anywhere on the animation's timeline are made constantly. (Cooper 2021.)

### Follow through and overlapping action

Follow through, overlapping action and drag are essentially different aspects of the same principle: when the main mass stops moving, lighter body parts, appendages, hair, and garments drag behind.

Follow through means movement that comes after main body has stopped, overlapping action describes the offset between the timing of the main body and its other parts, and drag describes the technique of delaying the movement of body parts in relation to the main body. Like with squash and stretch, the amount of drag can display the object's mass. (Becker 2017.)

Offsetting the main movement and the overlapping action also breaks up the uniformity of an animation and makes it more interesting to look at. Additionally, follow through and drag can help in reading what path, arc, or direction an action took even after it has ended. The point where the main body has completed an action is also where an animator designates the windows for further input, should several actions be chainable together.

In 3D animation, overlap and follow through may be manually animated when the chain of movements is predictable, but features such as clothes or hair can usually be left up to a physics engine that calculates the deformations and movement of objects automatically. (Cooper 2021.)

### Slow in and slow out

Slow in and slow out is about portraying the acceleration and deceleration of an object. Very few objects maintain a constant speed and it is natural for an action to take some effort to start, stop or change direction. How this looks in animation is portrayed by varying how far something changes or moves out from one frame

to the next. As seen in the ball test (figure 5), when the ball is changing directions, going from flying up to falling down, the frames are more closely packed. (Becker 2017.)

As previously mentioned, 3D animation software estimates transitionary poses between two known ones by the method of interpolation. By default, it calculates the most straightforward transition at a consistent pace of change. The algorithm for calculating the change between poses can be manipulated with a curve graph editor (figure 13). When the transitional line is changed into a curve with splines, slow in and slow out to can easily be added to animations and the change in spacing can be adjusted even further. (Cooper 2021.)

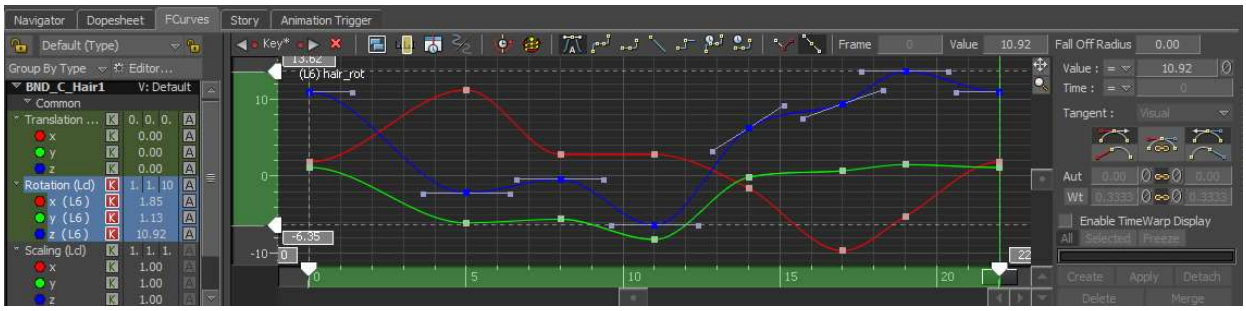



Figure 13: The curve editor in MotionBuilder (Brusi 2021).

### Arc

To expand further on lines and curves; linear in-betweens often lead to very mechanical movement while arcs soften the movement, make it flow better and look more organic. This applies not only when talking about change in velocity, but the path a movement takes as well, which is the main focus of this principle. Coming once again to the ball test (figure 5), when the bouncing of a ball is animated, the movement happens in arcs with soft and gradual changes in direction. If the transitions between points of directional change were linear, the ball would move in a zig-zag pattern reminiscent of a gear shift. Wave-like arcs are particularly clear in how the head and arms move in the previously featured figure 12.

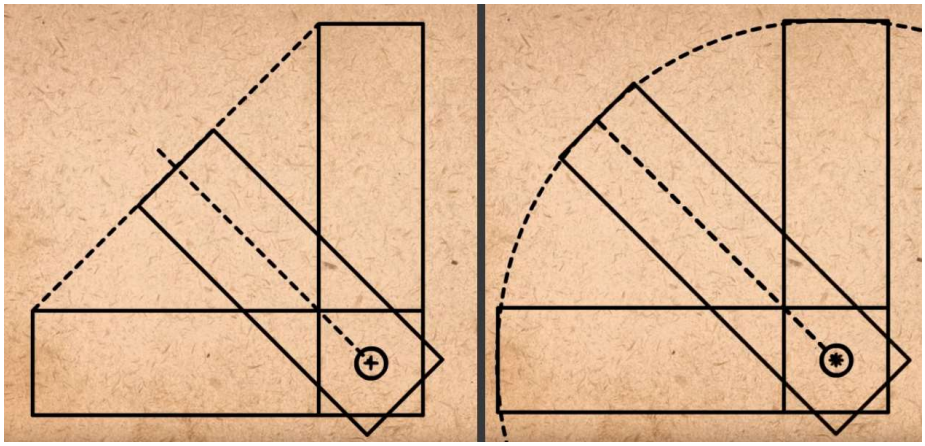


Figure 14: A block turning on a hinge (Becker 2017).

Additionally, not visualizing an arc in movement can easily lead into mistakes, especially in 2D. In figure 14 there is a block and a hinge. If the middle point between the two extreme poses is pictured linearly like on the left, the block clearly shrinks and changes mass. Picturing an arc and actually rotating the block around the hinge in an arc maintains the dimensions of the block correctly. (Becker 2017.)

### Secondary action

Secondary action means gestures that support the main action and add dimension to the character. As an example, in terms of walking, the leg movement is the minimum requirement and the key part to making a walk recognizable as such. Whether a person keeps their arms stiffly by their side or swings them as they walk, however, can tell a lot about the character and their current situation. If a character is sad, they may cry. To add to that crying, they may wipe away their tears, blow their nose, their head may droop, and they may wail. If someone is standing around waiting for another, they may occasionally look around, adjust their posture, look at their wristwatch to check the time. All of these are secondary actions that support the main action. The secondary actions should not overtake the primary action, the core the action should always remain clear. (Becker 2017.)

In game animation an animator can streamline production by making their secondary poses additive to the main ones. This is discussed more in depth in the Elegance section of the five fundamentals of video game animation, but as an

example, if aiming a weapon is the main action, aiming it at a specific direction can be considered a secondary one. (Cooper 2021.)

### Timing

The personality of a character and the meaning of an action is greatly affected by the number of frames inserted between each main action. Thus, timing in terms of animation means change over time as well as speed, or tempo of an action. Johnston & Thomas (1995) describe an example where the reason why and the method how someone turns their head changes greatly with each additional frame between the start and end frames. In this way, timing plays a great role in tandem with many other principles to sell the nature of an object, character, or action.

According to Cooper (2021), timing is finalized in a game animation first, even before the key poses are polished. This is because sound effects, particle effects and the like need to be in sync with the action, and the nature of the action itself needs to be clear as soon as possible.

### Exaggeration

When Walt Disney used to ask his staff to make something more realistic, the animators may have mistakenly tried removing stylization, when Walt actually wanted the action to be more convincing and more obvious (Johnston & Thomas 1995). Exaggeration is about pushing poses further into their extremes to make changes more noticeable. Exaggeration can be applied to practically all the other principles and at times it can mean going as far as to intentionally breaking some of the principle to a degree. As an example exaggerated smears and enlarging a punching fist for a couple frames sell the speed and strength of actions while breaking the squash and stretch mass consistency rule.

### Solid drawing

Solid drawing can be interpreted as a solid understanding of body mechanics, from center of mass to balance to the chains of reaction down a limb or spine as a foot hits the floor (Cooper 2021).

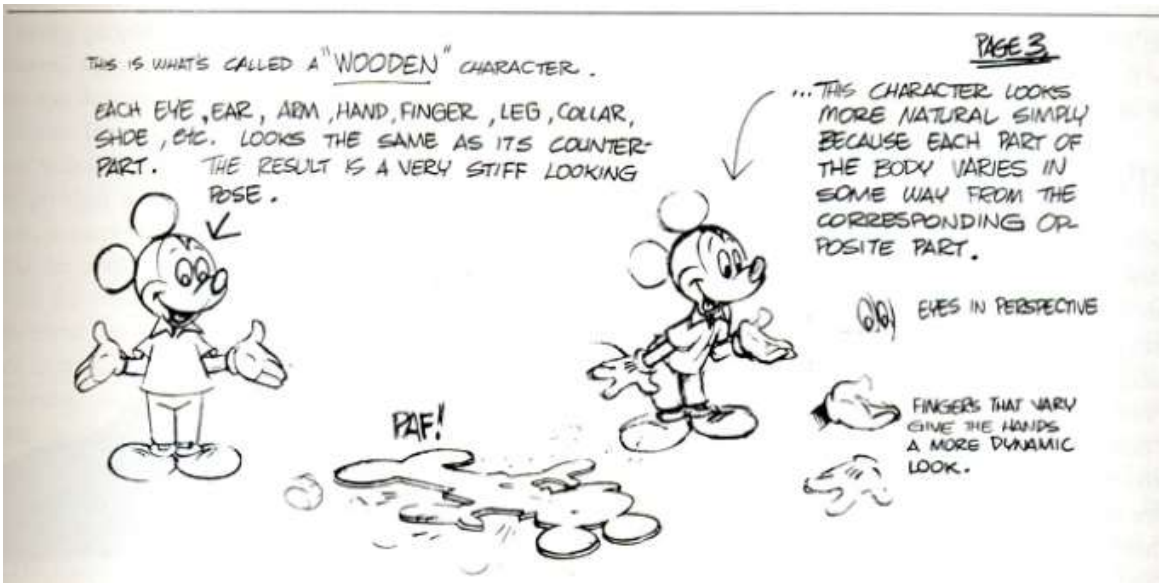



Figure 15: Illustrating the difference between flat and dynamic posing with Mickey Mouse (Johnston & Thomas 1995).

In 2D animation this means looking for nice perspective angles to illustrate contours and posing the character in a naturally asymmetrical way all the while accounting for volume, weight, line of action and balance to make forms feel more corporeal, as seen with the Mickey Mouse drawing on the right in figure 15 (Johnston & Thomas 1995). This may seem to be easy to do in 3d because the shapes actually have volume, but careful attention must be paid to the centre of balance, so the poses have real weight instead of being unbalanced and floaty. Because some 3D software allow for copying, pasting and mirroring of poses or parts of them, the mistake of twinning limbs as a shortcut happens easily, making the limbs pose in the same exact way unnaturally. (Cooper 2021.)

### Appeal

Out of all the principles, appeal is the most difficult to give a one-stop definition to because what appeals to someone changes from person to person. It does not necessarily mean good looking; appealing can mean interesting or just generally likeable in the sense that it is nice and/or interesting to look at.

Appeal may be applied to multiple design principles; it can mean color palettes, the use of shapes and proportions and amount of detail or clarity in simplicity. Even if it was the horrific main monster of the project, it needs to be something a person can at the very least look at and be intrigued by. (Becker 2017.) Thus

character design decisions have an important part to play, but the usage of all these animation principles can have a positive effect on a character's appeal as well.

## 4.2 The five fundamentals of video game animation

The five fundamentals were initially proposed in 2019 by Jonathan Cooper in the first edition of his book Game Anim: Video Game Animation Explained to complement the 12 basic principles of animation. Their purpose is to build on and expand upon game specific aspects of the principles.

### 4.2.1 Feel

As mentioned several times before, interactivity is the main feature separating video games from linear animation. Like appeal, what a good game feel means depends on the emulated properties and the functions within a game.
As an example, if there are two characters with entirely different body builds, one nimble and one heavy, their movement speeds should be adjusted accordingly to match their designs. Actions that happen fast and take little effort are perceived as light if not weak, while putting more force and weight into an action is sold with a longer windup. The catch is that in games this is a delicate balancing act between response time and visual flare. How visually impressive, realistic, weighty, or detailed an animator wants to make an animation look can never take priority over functionality and player control.
Games and characters from different genres and styles have different rules for inertia and momentum, which affects the entire feel and flow of the game. Even more realistic games do not actually stick to realism but exaggerate and take shortcuts if it makes the game feel more satisfying to play, as an example having controls that allow for position adjustment even mid-air. (Cooper 2021.)

Madeleine from 2018's platformer Celeste can be used as an example of a more simplified and cartoony style of movement. She gains speed fast, accelerating to full speed in only 6 frames, but stops even faster in 3 frames. The acceleration

time is long enough that she feels like a human with mass, but short enough that response to player input is almost immediate.

Figure 16: Madeleine elongates as she jumps (Extremely OK Games 2018).

Because these animations are so short, her animations are also rather simple and consist largely of squashing and stretching her character at appropriate times, for example when she jumps in figure 16.
Contrast this to Mario, who is a lot heavier if not a little floaty but who has more defined poses for his actions. As platformers, the feel of both games is built around the movement and jumping controls, yet still different. (Brown 2019b).


Figure 17: Combat in Batman: Arkham Knight (Rocksteady Studios 2015).

Batman in 2015's Arkham Knight (figure 17) moves rather fast for someone who is not supposed to be super-human and is so stacked with gear, crossing several meter gaps between the people he is punching and kicking in a constant flow. Not necessarily realistic, but the ease and speed of the free flow combat feels good to play since something usually happens very soon after a button is pushed.


Figure 18: Combat in Spider-Man (Insomniac Games 2018).

A similar system is implemented in 2018's Marvel's Spider-Man (figure 18), but the feel is different and lighter with the titular Spider-Man constantly utilizing his strings and acrobatic maneuvers in combat, as well as tossing some quips at his enemies. (Root 2020.)

In these ways game feel is very much about the controls and mechanics of game, but feedback can be given in many ways. Animations and special effects, camera effects, brief impact pauses and haptic feedback via controller rumble all stack up to sell the feel of actions. A visual blast can be seen in figure 17, but in addition to that, the game also makes use of slow-down on impact. However, it is important not to go above and beyond for every move, because too much feedback on common actions may become irritating and undermine the more special actions. More challenging controls and animations can also still be implemented, but usually this is done in moderation and for smaller sections. (Root 2020.).

### 4.2.2 Fluidity

In early sprite-based era of game visuals, transitions were not really thought about due to technical limitations and the workload. Sprite animations are called as needed and rarely have transitionary frames, meaning the character either does, or does not do something. Later, when 3D technology had developed far

enough, numerical values in 3D posing allowed for crossfade calculations between poses. Having the transitions between different actions happen as quickly and smoothly as possible is what fluidity is. For a good number of cases, it is enough to give the two main poses weight values that can be decreased or increased as needed to create adequate crossfades. Some other blends like a 180 turn or stopping from a full run still need custom help to look like they make physical sense. Of course, while having more bespoke transitional animations leads to better fluidity, it may also lead to more sluggish action and a more expensive project. (Cooper 2021.)

One recent leap towards more automated realistic movement and transitions is called motion matching. In involves shooting large quantities of mocap data which is then processed and gets sorted into a databank. (Zadziuk 2016). While the characters are moving at runtime, the technology dynamically selects the most appropriate section of motion data from the databank based on data on momentum, velocity, and matching poses instead of relying on just the usual positional and rotational values stored in bones and blend trees. Provided the data fed into the bank has been cleaned and processed properly and no duplicate motions can be found, the transitions between motions are made visually flawless. So far, this technology has mainly been used in sports titles and action games with a slower pace and focus on realistic movement. Motion matching is already facing increasing amounts of competition in form of machine learning, though both use a similar motion databank. (Cooper 2021.)

If cutscenes and gameplay are supposed to blend into each other as well, something called abort frames can be used. An abort frame is a programmed tag from where on the animation can be interrupted, giving the player controls back earlier without having to wait until the interaction animation ends. If no inputs are made, the animation plays all the way to its end as usual. (Yates 2017.) Similar method is applied to combo attacks. As an example, Link in Breath of the Wild has a four-swing combo sword attack. Each swing is cut into its own clip of animation since not every attack is guaranteed to go through the whole combo. Input during one animation’s window of opportunity can lead into the next.

Because these windows of opportunity need to offer some flexibility, what frame each animation ends on is slightly unpredictable. Because of this, the transitions in a combo will actually look a little less fluid, but as a tradeoff this flexibility makes for better game feel. (Root 2019a.)

Some shortcuts into streamlining the process of creating animations will be discussed in the Elegance section, but one way to help with transitions and more complex animations in general is procedural animation.
The term Forward Kinematics aka FK is used when bones higher in the hierarchy affects the ones below in relatively equal measure. Most animation is done in FK, meaning the process of posing starts from the root (the hips) and moves down to the extremities joint by joint.
The opposite of this is IK aka Inverse Kinematics where the positioning of bones down in the hierarchy affect the ones above. Where in FK all bones down the line of hierarchy react equally to the movement of bones up it, in IK any bone can be the one to lead the movement, but the strength of the pull is incremental.
In procedural animation limbs are controlled by mathematics and calculations rather than anything bespoke. Instead, properties of other objects and input determines what leads the movement. Most commonly, procedural IK animation is used to make the character correctly touch a wall while passing it or to calculate where a foot lands while on changing or uneven terrain without disrupting the rest of the body. As a downside, IK can be expensive in terms of performance. (Cooper 2021, 69.)
Some rare games like Ubisoft's 2015 game Grow Home almost completely forgo bespoke animations and rely entirely on procedural animation, occasionally using some key poses as a weighted guide. In these situations, the animation is very fluid, but very much ragdoll-y and unbalanced. (Root 2020.)

### 4.2.3 Readability

Readability is similar to and derivative of one of the twelve animation principles, staging, though it requires some key elements of solid drawing as well. Once again, the goal is to make actions as clear and easy to read as possible. Where in 2D making a character feel more three-dimensional and appealing may involve

drawing them from a 3/4ths angle or from profile if an easy-to-read silhouette calls for it, in 3D games the camera may be independent from the character's actions. Some camera considerations are discussed more in the next segment, but generally it should be assumed that the animation needs to be identifiable and clear from any angle and even from a distance. Thus, it is recommended that actions and motions that flow through only one particular axis are avoided unless the animator knows for certain the camera will be at a readable angle. (Root 2020.)

The readability of actions can also be improved with the usage of arcs, swipes and secondary movements that use several axes, since these look more appealing and read better in 3D. If the animation is linear and fast, exaggeration and visual effects such as follow through parts or smears help to read the action and its direction better in the aftermath.  Having a clear center of balance and silhouette in general also remains as important as ever. (Cooper 2021.)

For fast animation loops that can play for a long time, it may wise to stabilize and smooth key parts of the movement in order to avoid an uncomfortable strobe effect. As an example, in most Sonic games when the titular hedgehog starts running, his upper body moves up and down as a runner's would. Once he reaches maximum velocity however, the upper body stabilizes so it is less irritating to look at in the long run. (Floyd 2021.)

### 4.2.4 Context

Context here relates to where, how and by whom an animation is used. Since animations can be recycled for multiple purposes or characters, an animator does not always know all these details beforehand. This raises the question of Distinction vs Homogeneity; is the action for a specific character or for multiple playable and non-playable characters? How much character should there be? What character design details need to be accounted for in an animation? (Cooper 2021.)

These questions are not limited to keyframe animation alone, but mocap actors need to consider them as well. According to motion capture actor Andrew Ray in his 2017 GDC talk, an actor is not supposed to just act for the camera nor move

as they would by default. Of course, generic characters can be more generic in their movements, but to portray a specific character the actor would need to physically be that character and carry themselves as that character would. This may include finding neutral body aka idle pose, center of gravity and center line specific to that character. Actors often look for character context and inspiration by looking at concept art and inspecting other references such as character profiles.

As with the fundamental of readability, the camera is one key player when it comes to context. The cinematic camera language still applies to a degree, wide angle shots offer lots of peripheral vision but also less focus, while tight zooms feel more intimate and allow for more subtle movement but obscure a lot. A linear game could benefit from an on-rails camera while a more exploratory game might require a controllable camera. (Brown 2019a.) Different games benefit from different camera views, but as mentioned during the readability section, most 3D animations should at least be made with the assumption that it can be viewed from any angle.
Additionally, if animations are intended to be seen mostly from a distance, say an enemy spotting the player during an encounter, the animation should be exaggerated enough to account for that distance. More insignificant characters or actions should not take as much investment. Objects less relevant or further away do not require detailed rigs either, partially for performance reasons. (Root 2020.)

### 4.2.5 Elegance

Elegance as defined by Cooper (2021) in this context is not about fluidity in motion or visual elegance as might be assumed, but it is more about efficiency and cleanliness in terms of production. It is about optimizing resources so more generic animations can be taken advantage of as much as possible, in order to highlight the fancier bespoke features and animations as well as to have the overall quality of game design be as level as possible. Because of this, the fundamental of elegance is applicable not only to animation but other parts of game design as well.

If a punching animation can be applied to opening a door with the help of minor tweaking, this opportunity should be taken advantage of. If props can be picked up by the player, figuring out standardized measurements for them can take out some extra guessing work for the animators and prop designers. If the changes in climbing terrain for are standardized, in addition to making the animation work easier, the player will have an easier time identifying what terrains are climbable in the first place. (Cooper 2021). Things such as ladders are often portrayed in games as protruding higher than they would in real life, both to make them more visible to players and to make animating ladder climbing easier (figure 19).

Figure 19: Ladder protruding above ground level (Ubisoft 2016).

In Uncharted 4, the team avoided having to code complex climbing AI for NPCs for only a limited number of scenes, by making use of the player character Nathan Drake's pre-established and programmed movement abilities. They played through the necessary scenes as Drake while recording his movements (figure 20), treated following the footage as they would treat mocap data and applied those animations onto the NPCs. (Yates 2017.)

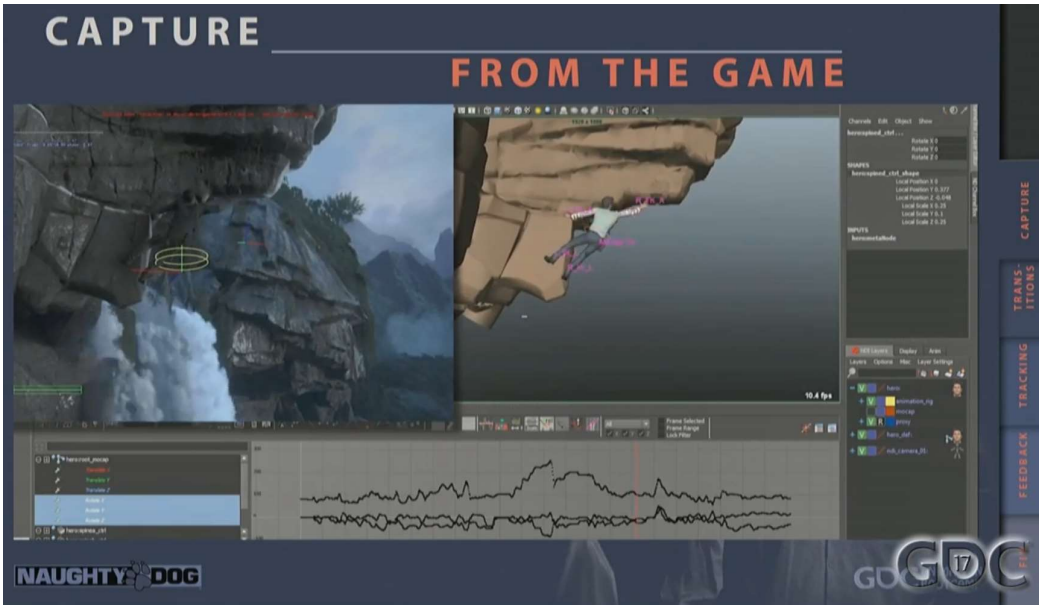



Figure 20: Capturing animation from gameplay to for NPC use (Naughty Dog 2017).

A lot of animation production can be streamlined by using animation layers and additive poses (figure 21). Essentially additive poses are premade separate poses and animations which can be added to other animations either fully or in part and at a designated blend weight. Additive animation requires a pose to be subtracted from the base before it can be overlaid, so only the difference between the additive and selected pose is applied to the underlaying animation. (Cooper 2021.) For Blizzard Entertainment's 2016 game Overwatch, instead of making 9 directional reload animations one by one, all that was needed was one reload animation in a neutral position and one animation for all aims per character. The neutral reload could then be combined with the aims, cut into clips, and saved as the bespoke directional reloads, which could then be additively applied onto the different movement animations in engine. (Davis 2017.)

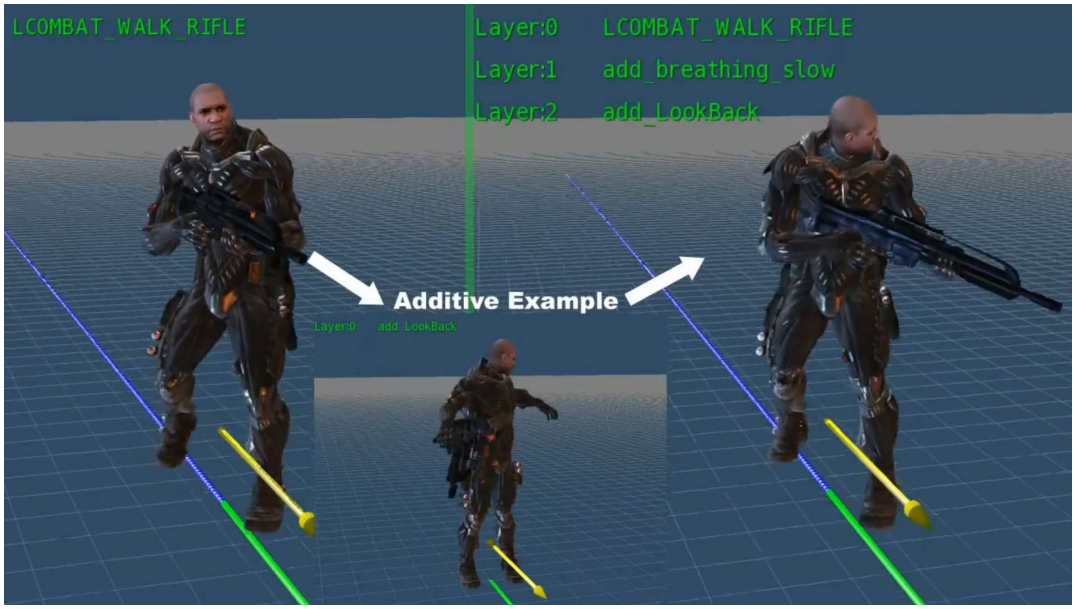



Figure 21: Example of additive animation for Crysis (Crytek 2006).

## 5 KEYFRAMING

Keyframe animation is essentially manually constructing poses for the character throughout their action, whether that is done by drawing or posing 3D models.

To recap what has been mentioned in various sections before: 3D animation utilizes the pose-to-pose method of animation. In this method all the poses in an animation are created non-linearly and in it the initial main poses are called keys, secondary poses are called extremes and further broken-down poses are breakdowns and in-betweens.

3D models are controlled using rigs and skeletons consisting of bones for each limb, sometimes several if the limb needs more flexibility.

3D animation software keeps track of the rotational and positional values of all the bones in a hierarchy that starts from the core, usually hips, and goes down to each of the extremities (figure 3). Motion can affect the hierarchy in two directions: Forward Kinematics or FK means the movements of bones up in the hierarchy affect everything below in equal measure, while in Inverse Kinematics or IK any bone, usually an extremity, is set to lead the movement and this incrementally affects the bones connected to it in the hierarchy.

The animation software calculates transitions between two poses using interpolation, the process of estimating unknown values that fall between known

values. The values in this case being the rotational and positional values in the bones. The algorithm the software uses for these calculations can be manipulated with the usage of curve graphs.

### 5.1 Software

Some of the most famous commercial software include Autodesk Maya, 3Ds Max, MotionBuilder, Cinema 4D and Houdini.
Currently Houdini is used mostly for procedural animation, while Maya is the current industry standard for keyframe animation. MotionBuilder is the industry standard for handling motion capture and will be explored in detail in the project portion of this thesis.

On the freeware and opensource side, some of the most common software include Blender, Art of Illusion, OpenFX, Seamless3d, Toonloop, Clara.io and Daz Studio. Out of these Blender in particular is gathering more and more attention with its continuing development and multitude ways of usage.

Game engines such as Unreal and Unity can handle keyframe animation to some extent, but animation specific software is preferred for their functions and precision. What the game engine animation tools are capable of are constantly under development however, and might see increasing amounts of usage in the future.

### 5.2 Advantages

Keyframe animation is very malleable. Its results can be as swift and exaggerated or as slow and subtle as desired, allowing for both realistic and cartoony movement. Each limb and bone can be adjusted individually since poses are not saved as one big unit, but each bone and joint maintain their own positional and rotational values.
Keyframe animation is highly recommended for cartoony character proportions, since human actors physically cannot move like cartoon characters even if they exaggerate their movements or made use of padding to emulate different

proportions. This phenomenon is sometimes called the mascot problem because the footage often ends up looking more like people acting out a scene in mascot costumes instead of the actual characters being alive and moving. (Floyd 2021.) All in all, keyframe animation offers high amounts of control and versatility in the hands of a capable animator.

### 5.3 Disadvantages

Simply put, keyframe animation is extremely time consuming because everything can be manually adjusted. Especially realistic posing may require a considerable amount of finetuning. How easy this is to do also depends heavily on the character's rig, but rigs are a whole topic on its own.

Starting pose construction from the wrong body parts - working inwards from the extremities, is a common mistake; since majority of keyframe animation is done in FK, where adjusting bones higher in the hierarchy affects everything below, working inwards is very much counterintuitive and leads into endless amounts of readjusting body parts already posed. At worst the animator might just have to count their losses, scrap all the work done so far and start over. To avoid that scenario, the hips and torso should be posed first, then all the limbs joint by joint moving down in the hierarchy.

Another common mistake is setting too many keyframes. What keyframes are shown in a timeline depends on what joints are selected and individual keyframes can rack up fast unmonitored. The amount of keyframes can make a nightmare out of adjusting features such as timing because of how hard to read an overcrowded timeline becomes without constant scrubbing.
Making use out of curve graphs to adjust timing can leave the animation timeline much cleaner and the graphs might be easier to read, assuming there are not too many set keyframes and the limbs do not rotate or turn beyond their limits. Again, too many frames may lead to scenario where one would need to start over to finish the job properly and in a timely fashion. Finding just the right amount of keyframes in tandem with graph usage tends to lead to the best result. (Cooper 2021.)

# 6 MOTION CAPTURE

## 6.1 Optical motion capture systems

When thinking about motion capture, it is quite common to think about optical marker-based motion capture systems. This is not an unfounded thought either since it is the most popular type of motion capture hardware. All the different variants function based on the same principle: a set of 4 to 36+ calibrated cameras are arranged around a stage, creating what is called a performance volume. Within this volume, each camera captures 2D images at 120 frames per second, featuring the locations of markers in its sight. If at least three cameras can see an individual marker, it is possible to triangulate the marker's location in a 3D recreation of the volume based on the sum of all the camera views. These stage setups tend to be very expensive, but their quality is likewise very high. (Cooper 2021, 185.)
Additionally, as long as the stage can contain it, marker-based systems can handle movements and actors of various shapes.

Figure 22: Horse and actor motion capture for Call of Duty: Black Ops II (Williams, D 2012).

Since in most cases the markers are placed manually, the skeletons that can be tracked are not limited to people, but they can be used on props and animals as

well, like in figure 22. (Qualisys no date.) This technology can also be used to map entire surfaces, but this requires many more markers.

### 6.1.1 Passive markers

The functionality of passive optical systems is based on the markers being covered with retroreflective material and every camera having its own light source. This makes it so that the markers reflecting the lights are well illuminated for each individual camera. (Tanveer 2014.)
Software for all the marker-based systems tend to have several premade marker configurations to choose from but easily allow for custom configurations.

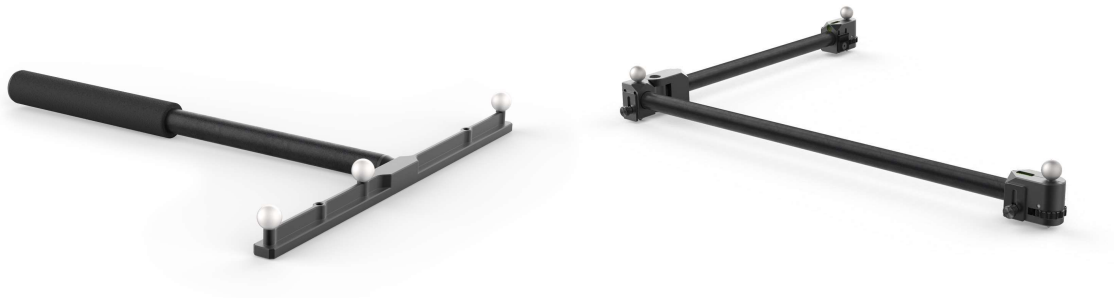


Figure 23 (Left): CWM-250 Calibration Wand (Optitrack no date).
Figure 24 (Right): CS-400 Calibration Square (Optitrack no date).

Mapping of the volume is done by waving around a T-shaped calibration wand or a calibration square (figures 23 and 24) in the view of the cameras, one marker on one prong and two on the other. (Sigal 2012.)
Once a marker configuration is selected, markers will be placed accordingly on a spandex or lycra suit with an outer lining made of Velcro or directly on the subject with glue or tape as seen in figure 22. Because the markers have so few requirements in terms of how they attach to the actors or interact with the cameras, they are very non-intrusive and allow for unrestricted movement. Markers on matching limbs will be at a slightly different positions relative to each other. These placement differences and temporal continuity allow for the tracking of each individual marker. However manual cleanup is often relied on because of tracking errors. (Sigal 2012.)

### 6.1.2 Active markers

The main difference between active and passive markers is that the active systems the markers themselves light up with infrared LED lights. This makes a massive difference in marker disambiguation for two reasons. Firstly, the cameras used with these systems can filter out visible light and thus detect the infrared emitting markers much easier. Secondly, since the LEDs are made to activate one at a time in a very quick succession, it is much easier for software to tell which marker is which.

Figure 25: Motion capture for Dawn of the Planet of the Apes (20th Century Fox 2014).

Active markers can also be used outdoors, and the intensity of the LED lights can be adjusted to stand out better at the location. The Planet of the Apes movie made good use out of this kind of system (figure 25). (Sigal 2012.)

### 6.1.3 Time modulated active markers

Active marker systems can further be refined by modulating the emission frequency and amplitude of individual markers, allowing for several markers to be strobed simultaneous instead of the markers being cycled through one at a time. This grants them even more distinct individual IDs and eliminates tracking errors

further. LEDs with onboard processing and radio synchronization allow motion capture outdoors even in direct sunlight and under clothes, however these improvements require much more processing power. (Tanveer 2014.)

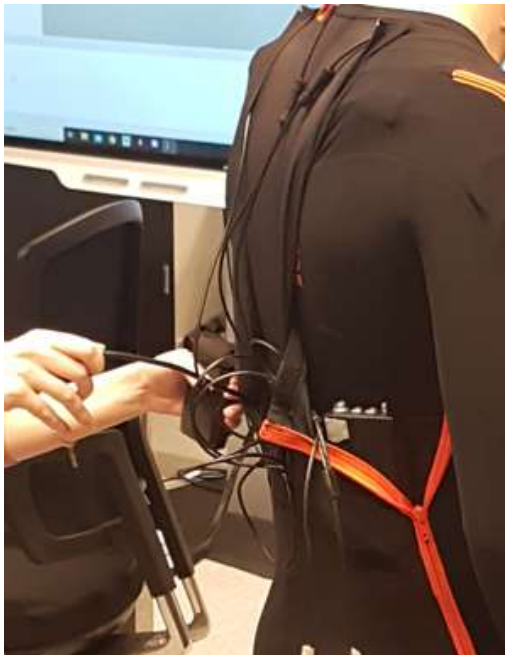
### 6.1.4 Semi-passive imperceptible markers

Semi-passive imperceptible markers utilize another advanced kind of unique identification and setup. It is based on multi-LED infrared projectors that emit spatially varying patterns, and photosensitive marker tags that decode the signals and estimate their position. Each tag can uniquely identify itself as well as calculate features such as position, orientation, incident, illumination, and reflectance. The last two from that list make it relatively easy to apply matching scene lighting when the scene is recreated. This system is ideal for on-set motion capture or real-time broadcasting of virtual sets. (Tanveer 2014.)
Since the system eliminates the need for cameras and the corresponding high-speed image stream, it is fast and requires less processing power. As a tradeoff however, this system cannot track global positions (Sigal 2012).

## 6.2 Non-optical systems

### 6.2.1 Inertial systems

Also called accelerometer suits by Cooper (2021) or inertial motion capture technology by Volpe (2016), inertial systems are based on miniature inertial sensors that measure acceleration and gyroscopes that measure orientation, as well as biomechanical models, and sensor fusion algorithms. A simpler example of this technology would be video game controllers such as the Nintendo Wii or Nintendo Switch remotes, that can keep track of their orientation and how fast they are swung, except the technology used for inertial suits are more sensitive and have higher resolution and update rates.




Figures 26 and 27: Preparing an Xsens suit for a shoot (Brusi 2020).

The sensors are worn by the subject in a lycra suit designed to house all the sensors, cables, batteries and transmitters such as the suit seen in figure 26 and 27. The trackers are placed on the performer's hips, chest, head, shoulders, arms, hands, legs and feet and the connecting wires run through gaps and channels in the suit. All of this is then secured in place by closing the zippers on the outer layer of the suit. Because all the sensitive technology used in the suits, caution should be used in both setting up and wearing an inertial suit. Mixing parts from two individual system sets or breaking a part of a set makes it so the system might not work at all.

Unlike the optical systems, inertial system suits are not necessarily restricted when it comes to the capture location. These suits can store data until a receiver transfers the data into the accompanying software, allowing for mocap outside of the receiver range and making them very portable. According to Kay (2021) the data that can be captured with an inertial system is as good but not necessarily as accurate as an optical system.

Inertial suits are somewhat more affordable than optical systems and hypothetically well suited for smaller companies that do not have funds for or

access to a bespoke mocap stage. Some companies producing inertial mocap systems include Xsens, Qualisys, Perception Neuron and Rokoko.

### 6.2.2 Mechanical motion

Figure 28: Gypsy Torso™ - Upper Body Mocap system (Meta Motion 2004).

Mechanical systems, also called electromechanical suites consist of an exoskeleton seen in figure 28 or armature of rods connected by potentiometers worn on the body. Potentiometers are like nobs on the radio, recording analog voltage changes and convert them to digital values. (Sigal 2012.) As the performer moves, the exoskeleton is forced to move with it and the potentiometers in each joint measure the rotations. Calibration is critical to this system and needs to be done often because the potentiometers can only record changes relative to their calibrated positions. While the results will be very accurate, the exoskeleton is highly restrictive needs to be hooked up to a computer. (Tanveer 2014.) Additionally, the system cannot calculate which way the performer's body is pointing and the positions are actually not known but are calculated from the rotations (Volpe 2016).

Figure 29: Craig Hayes with the Dinosaur Input Device (Knep, B 1993).

Jurassic Park used this kind of system with armatures (figure 29). The small dinosaur armature was manually posed, after which the pose was captured and made into a keyframe to drive the big 3D models. (Sigal 2012).

### 6.2.3 Magnetic systems

According to Volpe (2016) magnetic systems utilize a low-frequency magnetic field generated by an external transmitter and sensors placed on the body to measure position and orientation. These values are based on the relative magnetic flux and intensity of voltage or current of three orthogonal coils on both the transmitter and the sensors (Tanveer 2014).

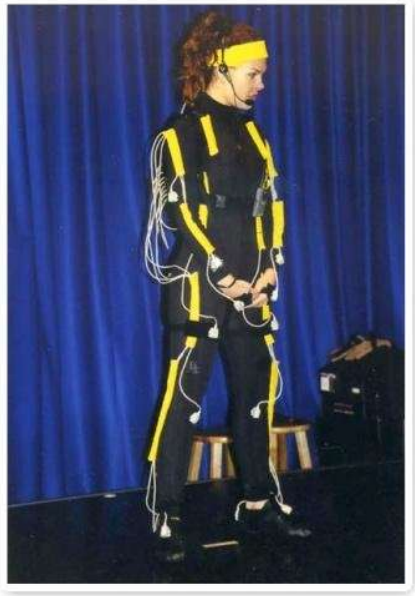


Figure 30: A performer wearing a suit for magnetic motion capture (AMM, 2010).

One suit used in tandem with these systems is features in figure 30. The small white boxes are the sensors while the yellow stripes hold the cables in place. The data from the sensors is transmitted to an electronic control unit that is networked with a host computer either wirelessly or across a wire attached to the suit. As with optical and mechanical systems, the range and maneuverability are limited, and this system is prone to interference from metallic objects and other magnetic fields as well as noisy data. (Volpe 2016.)

### 6.2.4 Stretch sensors

Optical fibers are stretch sensors that are typically used for data gloves and the like. Fibers of the glove and sensors sewn into the glove bend with the fingers, attenuating light and electrical currents, which is then converted to measurements. (Sigal 2012.) This technology seems to be used almost exclusively for hands in motion capture and measuring body posture or movement for medical purposes.

### 6.2.5 Range imaging

Range imaging techniques produce 2D images showing the distance points in a scene by using corresponding relative pixel values. The most common example of a range imaging device is the Kinect. Using structured light aka how a projection of a known pattern deforms based on the surfaces it hits, it computes a depth map by analyzing a speckle pattern of infrared light in the scene. It then applies machine learning to infer body position in the form of a skeleton. (Volpe 2016.)

With phones and webcams achieving ever better image resolutions, it has become possible to use range imaging technology on them too. A more casual form of usage for this technology would be Augmented Reality features like Snapchat filters, but increasing amounts of proper mocap software is being developed as well. Majority of such software are made for the more recent and advanced iPhone systems, such as the apps Motion LIVE, MocapX and Rush. Reallusion’s Motion LIVE is a plugin for their iClone 7 software, that can work not

only off other real time phone or webcam footage, but also other mocap systems and even series of static 2D images. The software has different templates to choose from for facial tracking and after the footage has been recorded and edited, it can be exported directly into other 3D software or game engines. (Reallusion no date.)

According to Cooper (2021), depth sensing cameras are good for reference gathering and previsualization, but the data needs heavy modifications and is not recommended for actual production work.

## 6.3 Shooting motion capture

**Motion Capture Gameplay Shootlist**

| Project Name: | Shoot list by: | List approved by: | Date + time shoot: | Timeframe shoot: |
|---|---|---|---|---|
| Feywild | Rozalie Craens | [Ids Boonstra] | [26-02-19 13:00] | [3 hrs] |

| Client Contact: | Animation Director: | Shootlist manager: | Mocap Member 1: | Mocap Member 2: |
|---|---|---|---|---|
| rozalie_craens@live.nl | Rozalie Craens | Rozalie Craens | [Ids] | [Robin] |

| | Rig: | Suit: | Actor Height (cm) | Actor Shoesize (cm) |
|---|---|---|---|---|
| Actor 1 | Robin van den Eerenbeemd | [S] | [160] | [37] |
| Actor 2 | Robin van den Eerenbeemd | [M] | [170] | [39] |

| Move # | Category | Move Name | Description | Hands | Props |
|---|---|---|---|---|---|
| 1 | Idle | player_idleRestStand | Standing pos3, breathing normally, weapon ready in hand | Right weapon ready | |
| 2 | Idle | player_idleRestInsert01 | Stretch arms | Both stretching | |
| 3 | Idle | player_idleRestInsert02 | Shift weight | Right weapon ready | |
| 4 | Idle | player_idleRestInsert03 | Looking around | Right weapon ready | |
| 5 | Idle | player_idleRestInsert04 | Stretch legs | Right hand moving | |
| 6 | Idle | player_idleRestInsert05 | Plie (down) | Both moving | |
| 7 | Idle | player_idleResttoCombat | Idle rest to idle combat | Right weapon ready | |
| 8 | Idle | player_idleCombatStand | Standing, breathing heavily, weapon ready in hand | Right weapon ready | |
| 9 | Idle | player_idleCombatInsert01 | Stretch arms | Both stretching | |
| 10 | Idle | player_idleCombatInsert02 | Shift weight | Right weapon ready | |
| 11 | Idle | player_idleCombatInsert03 | Looking around | Right weapon ready | |
| 12 | Idle | player_idleCombatInsert04 | Move weapon hand around | Right hand moving | |
| 13 | Idle | player_idleCombatInsert05 | Reposition arms | Both moving | |
| 14 | Idle | player_idleCombattoRest | Idle combat to idle rest | Right weapon ready | |
| 15 | Move | player_walkForward_LF | Walk forward, start with left foot | Right weapon ready | From idle rest |
| 16 | Move | player_walkForward_RF | Walk forward, start with right foot | Right weapon ready | |

Figure 31: Example of a shootlist (Craens 2019).

Before mocap can be shot, preparations need to be made. First comes the shootlist (figure 31), which includes all the necessary information regarding the shoot: date and time, location, director, actors, type of the footage, required props, the moves, and their levels of priority. Planning beforehand and reserving plenty of time for shoots saves a lot of money and trouble in the long and short run, since mocap shoots require not only the presence of the actors and the mocap technicians, but also a director and possibly animators and other staff. If time is short, moves with low priority may be skipped, while if there is extra time,

extra moves can be added to the shoot list during a shoot. As long as the moves are well listed, the shootlists can come in handy while organizing and labeling the data from the shoot.

Sometimes the animators tend to double as the actors in case of more generic movements, but more common performers, stunt actors and performance actors - people who also act as the face and voice of a character, may be used as well. When performance actors are used for a shoot, their body, face, and voice can be recorded all at once with the use of head mounted cameras and microphones. Because of the expenses, mocap stage and actor locations however, performance actors tend to be reserved only for cinematics and voice over. (Cooper 2021.)
Regardless of who is acting and what system is used, new actors always need to be calibrated into the system software. This involves creating a digital puppet of the actor that matches their height and proportions, including measures such as shoe size, shoulder width, arm and leg length and distance from foot to knee.

Both Cooper (2021) and Ray (2017) discuss how the director- actor relationship on set should work. During shoots, there should be only one designated director giving instructions to the actors to avoid confusion and mixed messages. A director should be forthcoming about what exactly they want, but not too critical of the performance.
Cleanup allows for some tweaking and mirroring of poses when necessary, but the amount of reworking a shot would require should be kept to a minimum when possible. A couple ways to help with this is to have reference for what the key poses of the character are and trying slightly different actor approaches for different takes. These so-called coverage takes offer more options when the direction for a scene is not entirely set in stone, and parts of different takes can be mixed together in post. (Cooper 2021.)


Figure 32 and 33: Motion capture for Uncharted 4 (Naughty Dog 2017).

As for the stage itself, the capture volume needs to be able to contain the intended scene and reflect it, since mocap data uses the real-life measures and scale. In addition to simple tape markers used to block out the scene, various building blocks may be used to emulate key features, terrain, and props. According to Yates (2017) Naughty Dog have a digital version of their set building blocks in order to make shoot preparations smoother, since setting up the stage takes considerable amounts of time. First, the scene is previsualized in 3D software, followed by the capture volume and set blocks being imported and placed into the scene as required. Once the digital version of the set is built, it is sent to the mocap stage to provide a blueprint for the shoot. As an example, in figures 32 and 33, various pillars are used to represent the gateway, while a bar acts the bottom of the gate. The data can be applied back onto the digital previz scene and character models in real-time to monitor the data and performance.

Having physical props for the actors to interact with, such as the bar acting as a substitute for the gate can be very helpful for performance, provided the props are made to proper scale and weighed accordingly. The movement of the prop can also be recorded with markers, but this essentially adds another actor into the scene and complicates cleanup. If the grip on the object is supposed to stay the same throughout, it is easier to not record the prop but pose the grip and position the prop manually during cleanup instead. (Cooper 2021.)

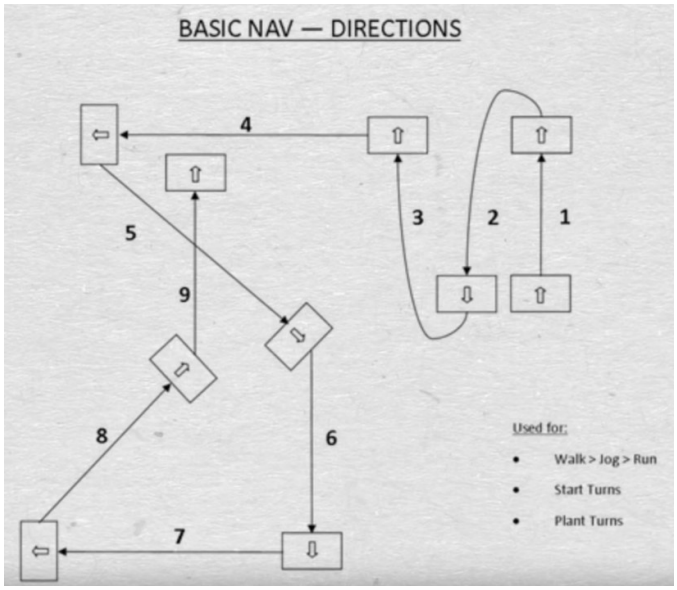



Figure 34: A mocap dance card (Zadziuk 2016).

For the purposes of gameplay mocap, the goal is to capture minimal amount of moves possible needed to create the maximum amount of coverage for a basic movement set. One way to make the movement capture process easier is the usage of dance cards (figure 34), which are essentially one-shot routines that help act out, capture, and cut data more efficiently. Dance cards are shot in smaller sections and in several variants, such as walk, jog, run, and crouch. If strafing and diagonal movement are implemented, those also need their own footage. Dance cards are essential for shooting motion matching data efficiently with as little duplicate movements as possible, although freeform movement can cover unpredictable transitions better. (Zadziuk 2016.) Both general mocap data and mocap data for motion matching or machine learning purposes may be stored in their raw forms for later reuse (Cooper 2021).

### 6.4 Cleanup

Once a shoot is done, it is time for organizing and cleanup. As mentioned in the shooting section, shootlists and dance cards may help in naming and organizing the shot files and the raw data can be saved for later use.

Raw mocap data can be applied straight onto a character in engine, but this is not recommended, as adjustments are harder to do this way. According to Cooper (2021) cleaning up mocap data is like reverse keyframing; raw mocap

data could be considered all in-between animation, which then needs to be timed and to which key poses need to be added. Re-timing an animation can mean speeding up or slowing down parts of it or even removing some keyframes completely, but this is not recommended. Adjustments to the mocap data are usually done on a separate animation layer to make the edits non-destructive and reversable.

Mocap cleanup process also uses rigs for easier control (figure 35) over the skeleton, though these control rigs are not the same as the rigs used in manual animation and depend on the software used for cleanup.

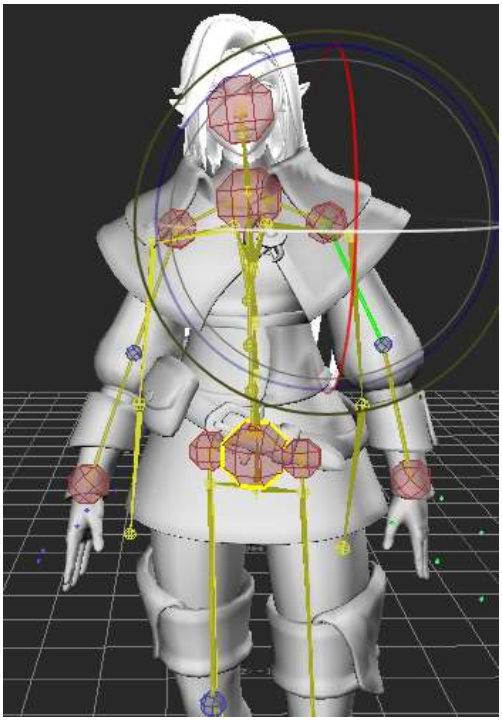


Figure 35: AZRI on a control rig (Brusi 2021).

Mocap data is essentially all positional and rotational point data, even though it is usually visualized as a figure or a skeleton of its own. In order to play the animation data on a character, first both the data figure and the character skeleton need to have at least the following bones defined: ground reference, head, one neck bone, one spine bone, hips as well as left and right upper and lower arms, wrist, thigs, shins and ankles.
The two are then characterized and connected by a control rig that drives the target skeleton based on the mocap footage baked into it from the data figure. Baking in this context refers to the software essentially locking and keyframing

the entire animation in its current form. It saves all the changes to the animation while resetting the default values used to calculate positions and rotations, thus removing dynamic variance from future calculations aside from changes made after the baking.

Raw mocap data takes its scale from real life, so the data needs be scaled down and retargeted onto a game-sized character before other adjustments get made. Doing this incorrectly can lead into problems such as reaching arms and sliding or extended steps as the character moves in a scale not meant for it.
Since mocap data does not include the thickness of limbs, one of the most common parts of the cleanup process is adjusting limb positions so they do not clip inside other body parts. Other usual targets include removing jittering and sliding from feet, making contact points between limbs and objects consistent, finetuning character posture and center of gravity and fixing possible gaps or miscalculations in the captured footage. (Cooper 2021.)

In order to create loops out of raw mocap footage, a clip is taken from the middle of the take where the action is at a stable speed and not in the acceleration or deceleration phrase. This means the mocap footage often gets clipped in various places and sometimes footage from different takes may be spliced together.
Once an animation has been cleaned, it should be realigned to face towards the positive Z-axis and the character's point of origin should be set to zero-coordinates, so the animation plays from the correct position and in the correct direction in the game engine. The following project chapter of this thesis goes through this process step-by-step in Autodesk MotionBuilder.
Once cleanup on a take is done, the data from the control rig is baked onto the character's skeleton and can be exported. If the take includes several actions, the material is cut into clips and saved as individual animations. These animations can then be imported to game engines and be played on any character with a compatible skeleton. (Cooper 2021.)

An animator's job does not end once all the animating is done. As previously mentioned, animations are polished over several passes, and there are points of

adjustment that are only noticeable once the animation is in full context inside the game engine, such as whether an action is obvious enough when there are multiple actors and effects in a scene. Gameplay animators are expected to add tags in engine on certain frames to trigger events and effects such as animation abortions and input windows, facial poses, prop interactions, hit frames, special effects and sound effects. Tags can be added via script, software tools or state machines.
Not all animators necessarily need to work directly with the animation blend trees and state machines if templates are available, but this is more likely to be limited to larger projects where templates can be shared by multiple characters with similar movesets and conditions. (Davis 2017).

# 7 ANIMATION PROJECT

## 7.1 Resources

The target character used for this project was the free non-commercial AZRI rig made to accompany the Game Anim: Video Game Animation Explained book by Jonathan Cooper. The character belongs to Matthew Bachnick and was rigged by Sol Brennan. (Game Anim, no date.)

The author of this thesis spent September 2020 to February 2021 in student exchange at Breda University of Applied Sciences and was a member of the Animation and Mocap guild. The guild is a learning community run by the BUAS head teacher of animation Ralph Palmer and student supervisors Xander Brondijk and Iris Teparić, with student actors Michelle Lipman and Lizzy Wilmer. They conduct motion capture shoots for the BUAS Creative Media and Game Technologies students as well as other clients. After the exchange period was over, permission to use their existing mocap materials for the purpose of this thesis and personal practice was received. The material used for this project was shot in March 2021 for an unrelated student project.

During the same exchange period the author studied the usage of Autodesk Maya as well as MotionBuilder by following Simon Kay’s Mocappys tutorials

(Mocappys, no date). The latter software was chosen for this project as it is better suited for processing motion capture data. Due to time constraints, the goal was to make a single jog loop out of raw data.

## 7.2 Setup

AZRI's initial file format was the MotionBuilder incompatible Maya ASCII so it needed to be exported as an FBX file first. This was done by opening the file in Maya, selecting the export_mesh and export_skeleton from the Scene Outliner and navigating to File options from the toolbar and selecting Export Selection with FBX chosen as the new file type (figure 36).

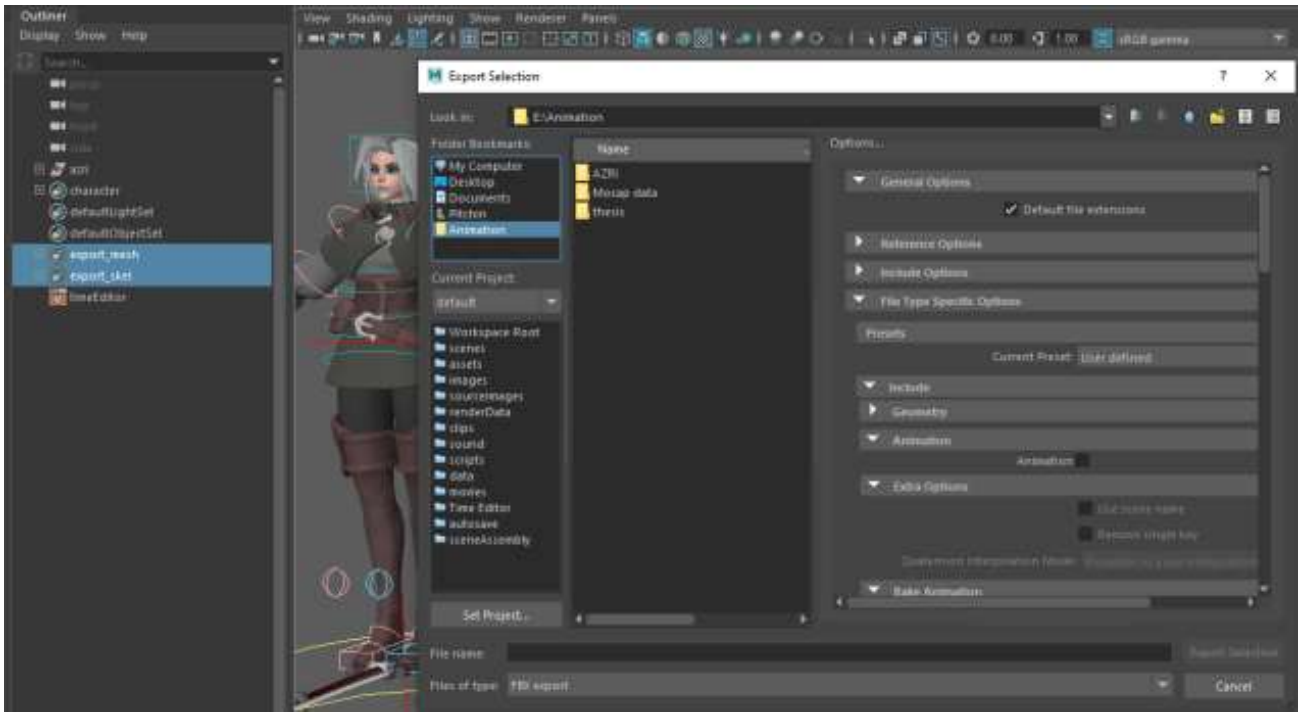



Figure 36: Exporting AZRI from Maya (Brusi 2021).

Under File Type Specific Options, the Autodesk MotionBuilder preset was selected and the file was exported. Maya gave a brief list of warnings (figure 37) as the FBX format does not support rig constraints and only includes the skeleton data, but this was acceptable as the constraints were not as relevant a concern for this project. If there had been any errors however, those would have required some attention. As the weapons included with AZRI were also unnecessary for

creating a jog cycle, their geometry and bones were removed from the outliner list before exporting.

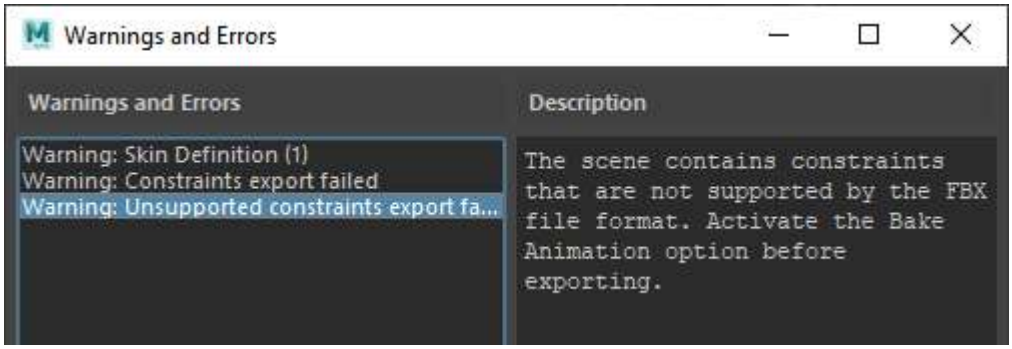



Figure 37: Maya export warnings (Brusi 2021).

Moving on to MotionBuilder, the exported FBX was opened. MB has a list of options regarding what is included every time a file is opened, and the default options for including all the features did not need changing. The mocap data was already in FBX format, so it could be opened in its own file without any changes.

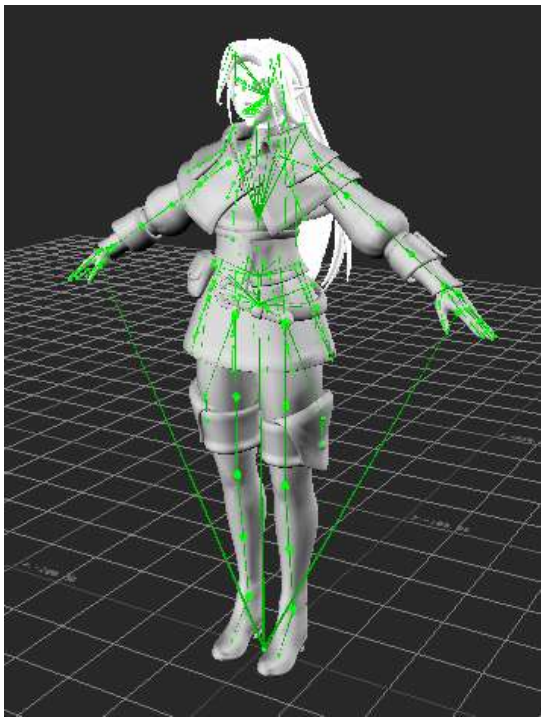


Figure 38: Freshly opened AZRI in MotionBuilder (Brusi 2021).

If the user is already familiar with how other Autodesk 3D software Maya and 3ds Max are controlled and interacted with, they can change the interaction mode under Settings in the toolbar menu.

While rigs and constraints do not get exported to MB by default, the skeleton, mesh and textures are. For unknown reasons, while all of AZRI's textures worked as expected in Maya, something broke the file containing the base colors for AZRI's body and clothes but they were not necessary for animation, so all textures were turned off.
Another feature that did not survive FBX exporting was the rig control layers, so by default everything was turned on simultaneously. To fix this, objects were split into different groups visible under the Groups tab in Resources (figure 39). Objects can be selected via the Navigator tab or by clicking them in the viewport or the Schematic view. The Schematic view, which can be opened and closed by pressing Ctrl+W, shows the hierarchies of all objects in a scene, such as the mocap data figure's skeleton tree featured in figure 3. AZRI's hierarchy tree was more complex, including geometry for different parts of the mesh, bones for fingers, hair and clothes and twist controllers. The main parts of the skeleton were made into one group, geometry into another and lastly clothes, fingers, hair and miscellaneous controllers were put into a third group and the latter two were hidden for time being (figure 39).

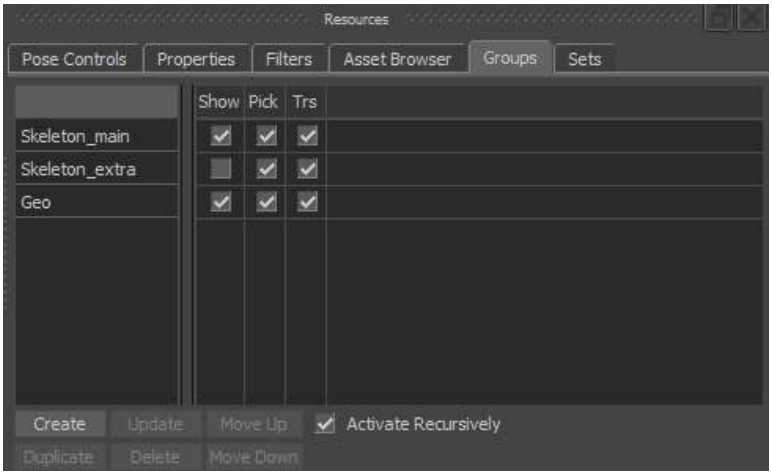



Figure 39: Initial object groupings (Brusi 2021).

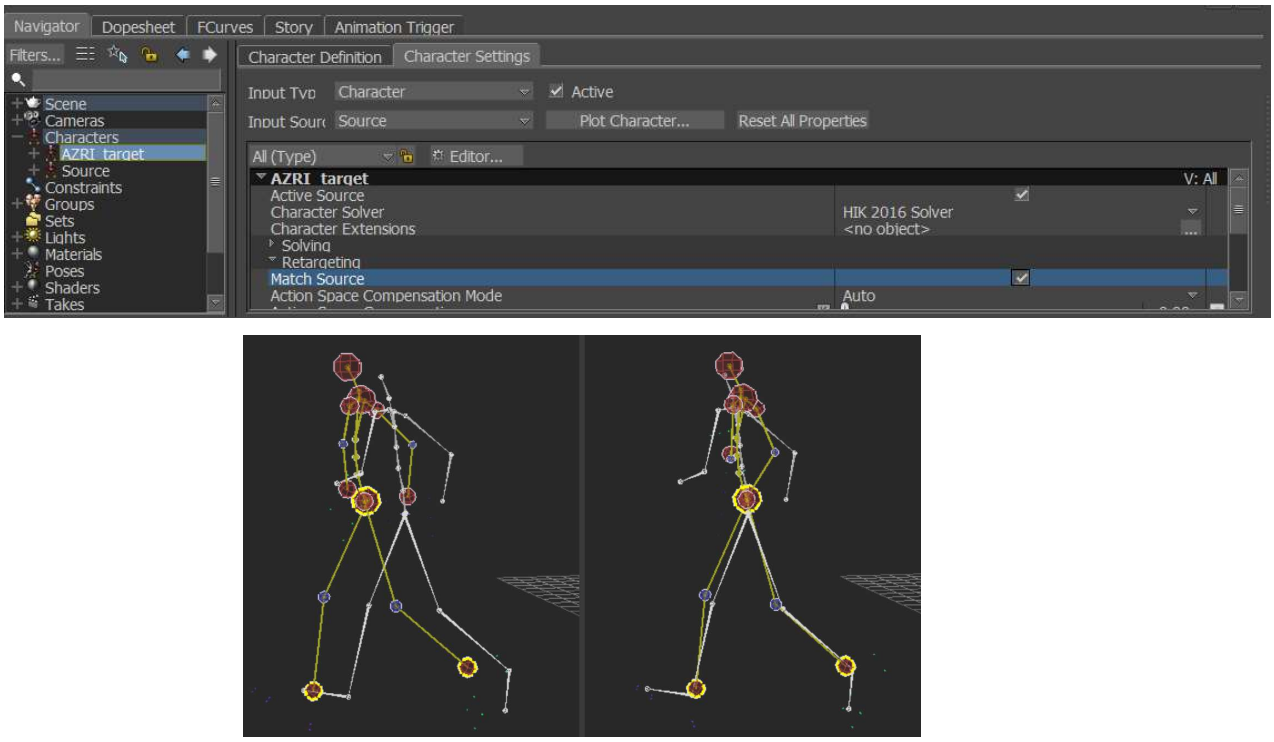
Next, AZRI's and the mocap data figure's skeletons were defined in their respective files. This meant assigning bones to correspond to body parts in the definition graph (figure 40). In addition to the reference point, head, lowest neck and spine bones, hips, upper and lower arms, wrist, thigs, shins and ankles, the clavicles and other spine and neck bones were defined as well. Because the naming schemes for opposing limbs matched, one side were defined simultaneously with the other. If parts were defined wrongly or left undefined, they would not move with the rest of the body and leave geometry behind.
MB had some warnings about the positions of the hands and upper arms and in the data figure's case about the reference point and hip bone overlapping but ignoring them did not cause any problems.

Figure 40: Defining bones (Brusi 2021).

Once the skeleton and figure were defined, they needed to be characterized. This was done by double clicking the character in the Navigator to access the Character Definition and Character Settings tabs. When the Characterize option under Character Definition was checked, MB asked for the type of the character and Biped was selected. This added the figures to the Characters navigator

category. For AZRI, the Active box next to Input Type was checked in the Character Settings tab.

At this point the two files could be combined by selecting Merge from the File options in the toolbar. The default merge settings were fine. With both figures in in the Character category, the AZRI character was named AZRI_Target and the mocap data Source for distinction. The upper part of figure 40 includes AZRI's dropdown slot for the Input Source, which was set to Source the mocap figure. Next, under the Retargeting section of AZRI's Character Settings, the Match Source option from was turned on to make the target character match the source data better (figures 41 and 42). Not doing this would have led to sliding steps.




Figures 41 and 42: Retargeting settings and the difference between untargeted (left) and source matched (right) data (Brusi 2021).

Finally, with the AZRI character selected, the blue button next to the character and source slots in figure 40 was pressed to select the Bake (Plot) To Control Rig option to create an FK/IK control rig. The source figure and the control rig were put into their own groups and their visibility was toggled as necessary. The mocap data was shot in 240 frames per second, but for this project the framerate was changed to 30fps in the Transport Controls, as the framerate signifies how

often data is sampled and how many keyframes were displayed, and because games mostly run 30-60fps.

### 7.3 Cleanup

The raw mocap data involved a short jog (figure 43), but as it included the acceleration and deceleration, the clip was cropped down to a stable speed two-step stable portion in the middle (figure 44). This was done by searching for two similar feet passing poses and by setting the start and end frames of the clip to match in the Transport Controls timeline.

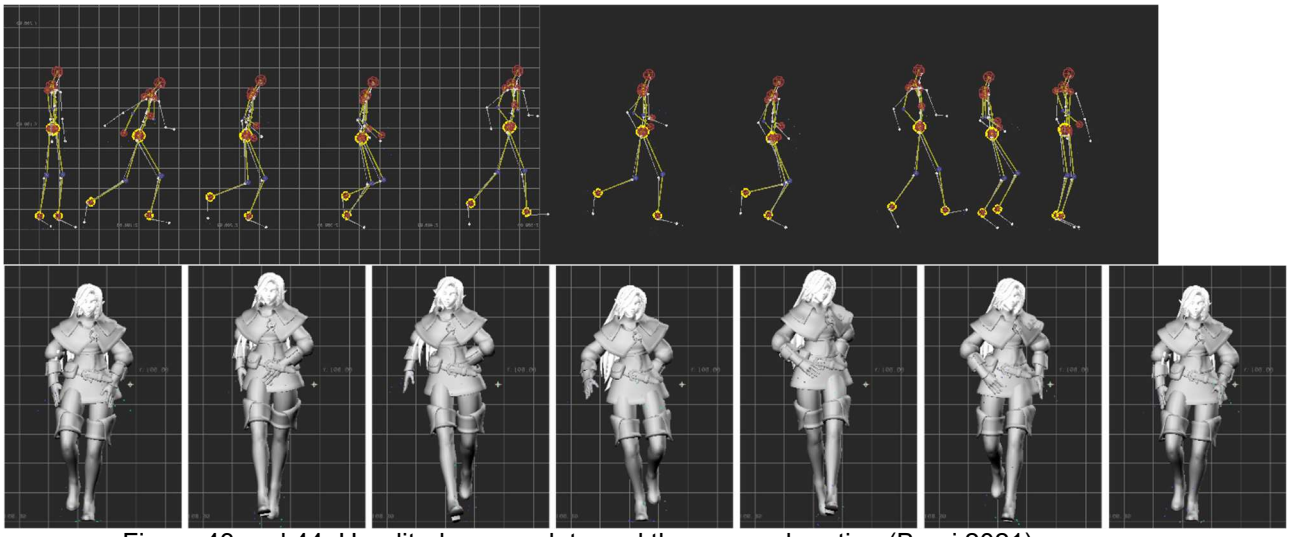


Figure 43 and 44: Unedited mocap data and the cropped portion (Brusi 2021).

Next the character needed to be reoriented as the figure ran slightly diagonally. This was done in MB's story editor, which was found in the navigator's Story tab and by making sure the Story button was turned on and blue (figure 45). An Animation Character Track was inserted to the track section by right clicking the track bar and AZRI was set as the character. By right clicking the story timeline, Insert Current Take was chosen to insert the cropped clip into the timeline editor, which was then moved to start at frame 0. Clicking the small eye icon in the character track turns on ghost mode and displays a straight path between the first and last frames of the clip, which helped with reorientation.

Cropping a take does not automatically update the pivot point of the clip, which can be done manually by double clicking the take clip in the story timeline to open

its Asset Settings and by turning the Auto-Calculate Loop option off and back on. The animation was then realigned by setting all the Clip Offsets Translations to zero to place the character to the correct start position and Clip Offset Rotations were set so the ghost path pointed straight towards the positive Z axis.

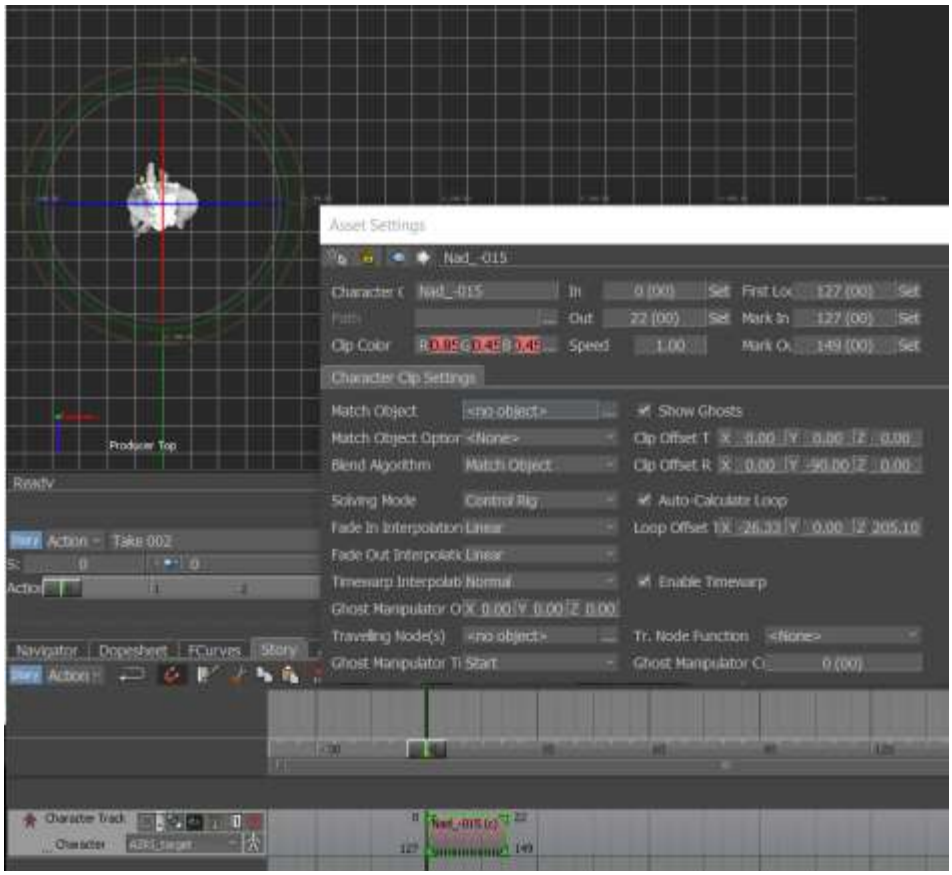



Figure 45: Reorienting in Story mode (Brusi 2021).

After realigning, a new take was created from the Transport Controls and the no data copying option was selected. The realignment edits were then baked by right clicking the the story timeline and choosing Plot Whole Scene To Current Take. Default settings were fine but if the framerate were not changed earlier, it would have been done now. After this, Story mode could be turned off and the changes made in it remained.

Following the mocap data the character still moved forward on the Z axis by default but using the In Place modifier available under the Character Settings tab allowed for movement on certain axes to be locked in place. Using this modifier, it

was possible to rotate the camera around the character to make sure everything looped correctly without the character constantly changing position.

Adjusting limb position and movement was done by editing the curves in the navigator FCurves tab on a desired animation layer. Adjustments could be done directly on the default BaseAnimation layer that contains all the raw positional and rotational data but make the editing process reversable and easier to track, new animation layers were made for different sections.
Some adjustments were easier to do on the base layer however, such as rotating the arms more outwards by adjusting the curves and making sure the end and start frames were near identical. This was done in the Pose Controls tab under Resources. The Character Controls in figure 40 include options for what parts of the skeleton react to manipulation and Full Body was chosen. Then, the start frame pose was saved into the pose library by clicking the plus icon and pasted over the last frame while the hips were selected to anchor the transformation point.

Editing started from AZRI's posture, as she leaned left in the raw clip. On a new layer for general adjustments (AnimLayer1 in figure 47) keyframes were set at the start and end of the of the lean and the rotation was adjusted so the character stayed straight throughout. Other edits done on the general layer included lessening the head turning too much to the right. The clip did not seem to have any foot sliding issues so no adjustments to the feet were necessary aside from a very minor case of knee popping. Knee popping is when a leg is extended straight enough to make it seem like the knee is bending inwards and usually happens if the hips are not low enough or the step is reaching. This was fixed by slightly lifting the foot higher. This was the only time when positional adjustments were required and all other edits were made rotational only.
How the animation was looking after the main parts of the body were cleaned is featured in figure 46.

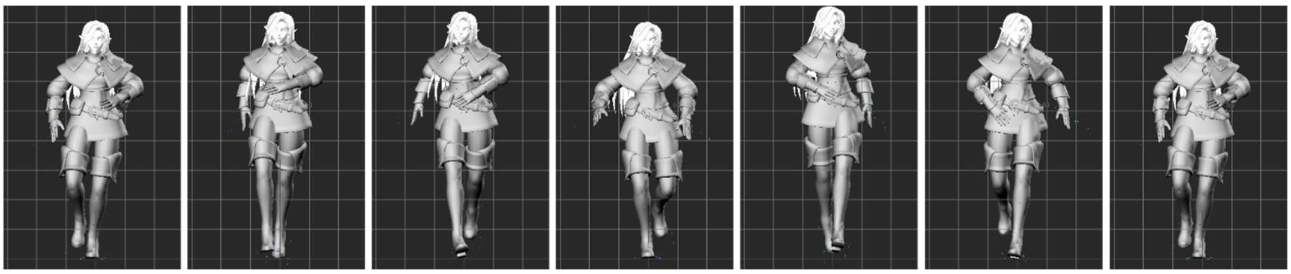


Figure 46: The animation after the core parts of the cleanup were finished (Brusi 2021).

After the main parts of the body came the details, namely hands, clothes, and hair. The face was left completely unchanged due to time constraints. Since none of these features could match the mocap data automatically nor did they have any IK controls, this was all manual FK bone animation from roots to the tips of the bone chains.

The hair, skirt, poncho, and hand bones were separated into their own groups from the skeleton_extra group in order to hide features that were not being worked on. Additionally, new layers were made to focus on specific aspects without disturbing features that had been already edited. Both of these additions can be seen in figure 47.

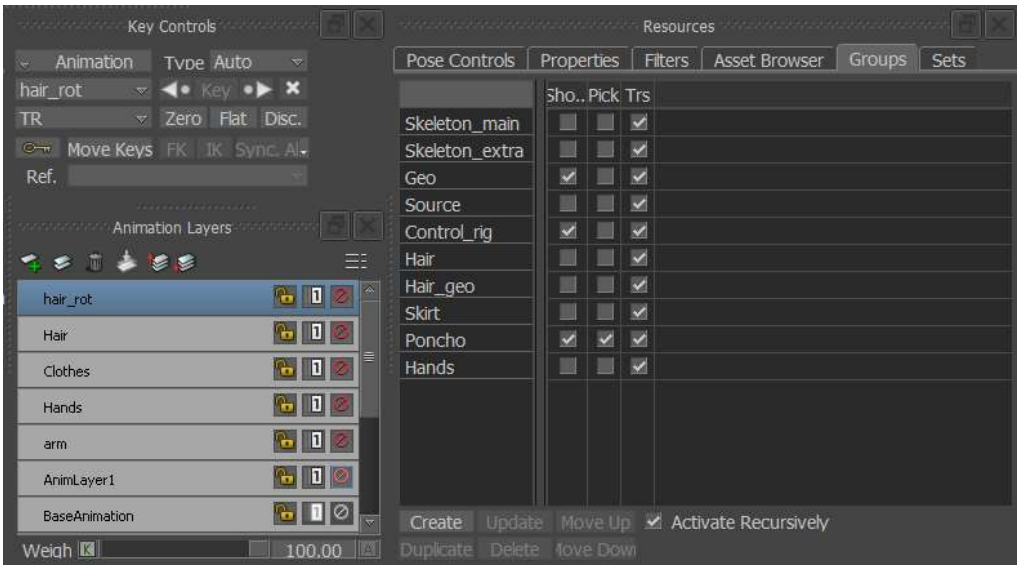



Figure 47: More groups and animation layers (Brusi 2021).

This portion of the cleanup started with some adjustments to the timing of the arm swings, relaxing the hand poses and making them open and close.

After this came the skirt and the poncho. The visibility of the hair mesh was turned off, so it did not obscure the back of the character. The fabric was

animated to move with the main body while accounting for some follow through action with the fabric being trailing slightly behind the body's movements. Lastly the hair was animated. The main mass of AZRI's hair is controlled by a chain of 6 bones and getting the poses and timing right proved very difficult. Rotating each bone one by one was extremely time consuming but the graph editor was difficult to read if all the bones were selected simultaneously (figure 48). In the end one layer was reserved for the hair's movement up and down the Z axis while another was used for the hair twisting and swinging from side to side in the X and Y axes.

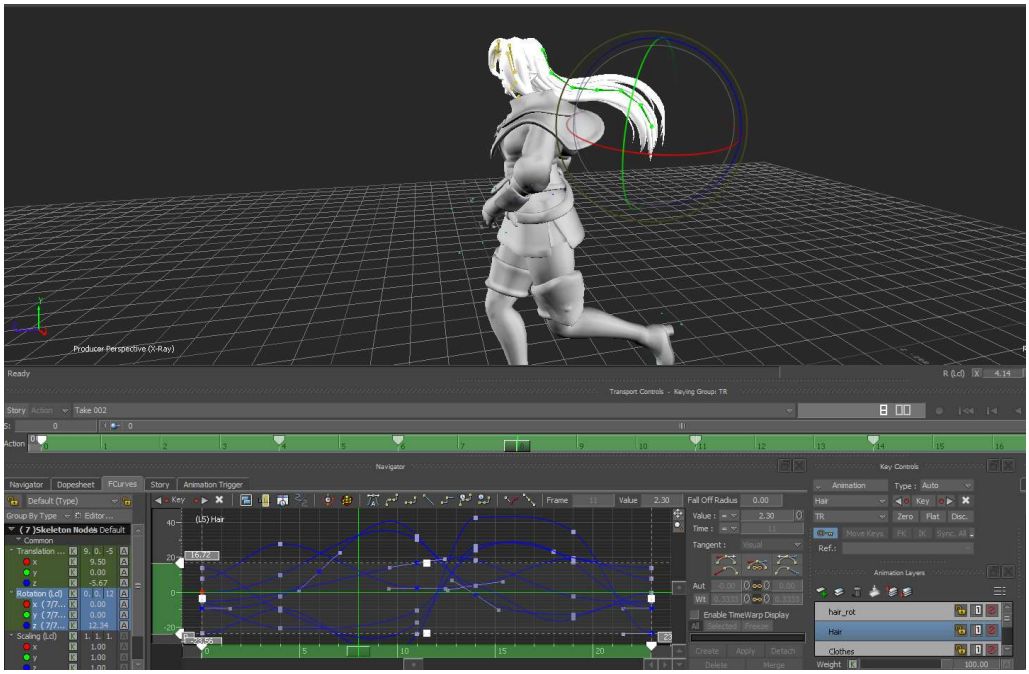



Figure 48: The Z axis rotation curve graph for the hair (Brusi 2021).

Once all the edits had been made, the animation was baked back onto the target skeleton from the Character Controls, making the animation ready for exportation. The end results are presented in figures 49 and 50.

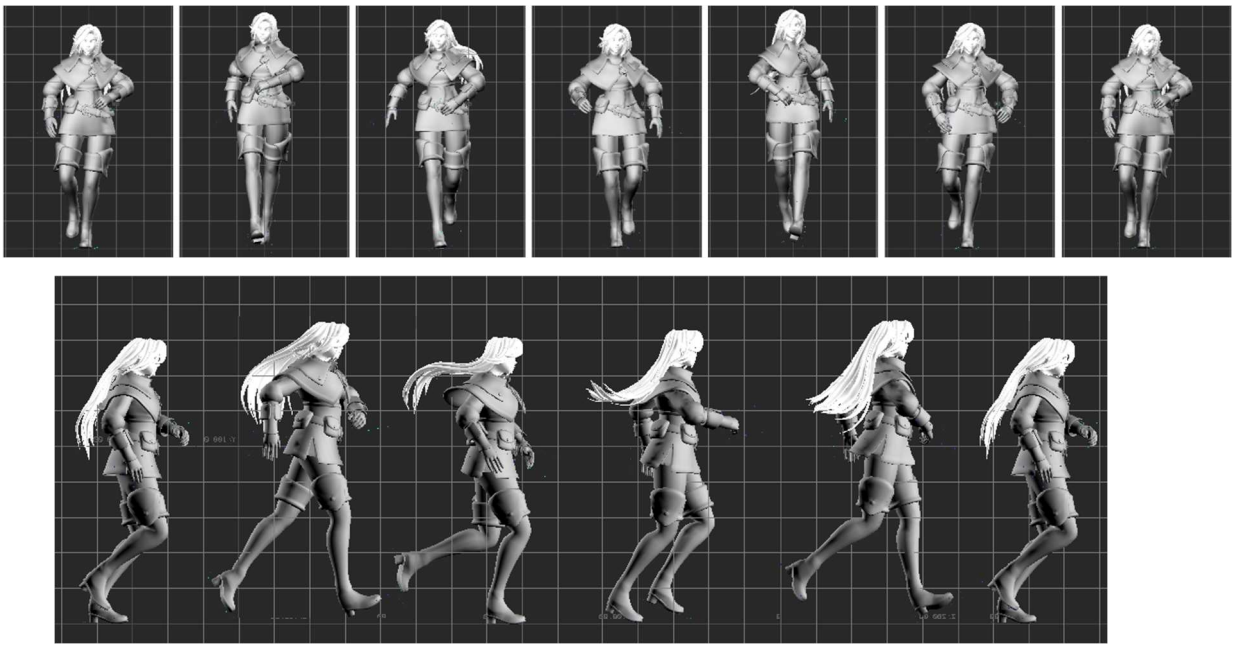


Figure 49 and 50: The end result of cleanup and manual animation (Brusi 2021).

## 8 CONCLUSION

The interactive nature of games differentiates game animation from standard cinematic animation in many different ways and with how many other principles are connected to it, it is a vast topic.

As this thesis covers mainly the full body 3D animation process and game animation theory, topics of that could use further coverage include but are not limited to 2D sprite animation, facial motion capture and animation, rigging, skinning and all the engine features such as state machines and blend trees. Motion capture technologies are constantly evolving, so procedural animation, motion matching and machine learning technologies also have further research potential, as well as range imaging technologies for cheaper and more accessible forms of motion capture.

## REFERENCES

Becker, A. 2017. 12 Principles of Animation. Video series. Available at: https://youtube.com/playlist?list=PL-bOh8btec4CXd2ya1NmSKpi92U_I6ZJd [Accessed 9 May 2021].

Brown, M. 2019a. The Challenge of Cameras. Video essay. Available at: https://youtu.be/bHdi5Ar8GXw [Accessed 17 March 2021].

Brown, M. 2019b. Why Does Celeste Feel So Good to Play? Video essay. Available at: https://youtu.be/yorTG9at90g [Accessed 16 April 2021].

Brown, M. 2021. How to Become a Game Designer. Video essay. Available at: https://youtu.be/PMXf0e8n2Oc [Accessed 21 April 2021].

Cooper, J. 2021. Game Anim: Video Game Animation Explained. 2nd edition. Boca Raton: CRC Press.

Davis, J. 2017. The Animation Pipeline of 'Overwatch'. GDC 2017 lecture. Available at: https://www.gdcvault.com/play/1024267/The-Animation-Pipeline-of-Overwatch [Accessed 19 March 2021].

Failes, I. 2015. Going off-model: Hotel Transylvania 2. FXguide. WWW document. Available at: https://www.fxguide.com/fxfeatured/going-off-model-hotel-transylvania-2/ [Accessed 1 April 2021].

Floyd, D. 2019. SQUASH & STRETCH - The 12 Principles of Animation in Games. Video essay. Available at: https://youtu.be/1kFRU_xBZnE [Accessed 10 March 2021].

Floyd, D. 2021. The Animation of Sonic Games. Video essay. Available at: https://youtu.be/GxCcHOvSVJQ [Accessed 10 March 2021].

Game Anim. No Date. AZRI rig. WWW document. Available at: https://www.gameanim.com/product/azri-rig/ [Accessed 30 April 2021.]

Johnston, O. & Thomas, F. 1995. The Illusion of Life: Disney Animation. Revised edition. Glendale: Disney Publishing Worldwide.

Kay, S. 2021. How to Use an Xsens MVN Mocap System. WWW document. Available at: https://mocappys.com/how-to-use-an-xsens-mvn-mocap-system/ [Accessed 23 March 2021].

Mocappys. No date. MotionBuilder tutorials. Website. Available at: https://mocappys.com/category/motionbuilder-tutorials/ [Accessed 30 April 2021].

Qualisys. No date. Animal motion capture. WWW document. Available at: https://www.qualisys.com/applications/equine-animal/animal-bio/ [Accessed 5 April 2021]

Ray, A. 2017. Animation Bootcamp: Motion Capture Performance: An Actors Approach. GDC 2017 talk. Available at: https://www.gdcvault.com/play/1024318/Animation-Bootcamp-Motion-Capture-Performance [Accessed 19 March 2021].

Reallusion. No date. Motion LIVE - Unified Motion Capture for Face, Body and Hand. WWW document. https://mocap.reallusion.com/iclone-motion-live-mocap/default.html [Accessed 18 April 2021].

Root, D. 2019a. How Link Swings a Sword // Video Game Animation Study. Video essay. Available at: https://youtu.be/Hh-CkW_P1e0 [Accessed 17 March 2021].

Root, D. 2019b. How Trico Was Animated / Video Game Animation Study. Video essay. Available at: https://youtu.be/TfZtIKMhULw [Accessed 17 March 2021].

Root, D. 2020. The Five Fundamentals of Video Game Animation. Video series. Available at: https://youtube.com/playlist?list=PLy445mOFNkd_zljA7uH2Fqndr4-jqE9D8 [Accessed 17 March 2021].

Sigal, L. 2012. Human Motion Modeling and Analysis; Lecture 3: (Marker-based) Motion Capture. Carnegie Mellon University School of Computer Science. PDF document. Available at: http://www.cs.cmu.edu/~yaser/Lecture-3-MarkerBasedMocap.pdf [Accessed 1 April 2021].

Stoeber, J. 2021. How devs break bones to make animation feel right. Video essay. Available at: https://youtu.be/vIdeGmN__Pw [Accessed 20 April 2021].

Tanveer, S. 2014. Motion capture technology. WWW document. Available at: https://www.slideshare.net/shaiktanveer14/seminar-report-33462669 [Accessed 1 April 2021].

Volpe, G. 2016. Multimodal Interfaces@ EyesWeb Week 20; 4. Capturing physical movement signals. Casa Paganini InfoMus Lab. PDF document. Available at: ftp://ftp.infomus.org/pub/Events/EyesWebWeeks/EYWweek2016/Slides/02_MultimodalInterfaces_2015-2016.pdf [Accessed 1 April 2021].

Yates, J. 2017. Animation Bootcamp:" 'Uncharted 4': Naughty Dog's Animation Workflow" GDC 2017 lecture. Available at: https://www.gdcvault.com/play/1024309/Animation-Bootcamp-Uncharted-4-Naughty [Accessed 2 April 2021].

Zadziuk, K. 2016. Animation Bootcamp: Motion Matching: The Future of Games Animation...Today. GDC 2016 lecture. Available at: https://www.gdcvault.com/play/1023478/Animation-Bootcamp-Motion-Matching-The [Accessed 19 March 2021].

LIST OF FIGURES

Figure 5: The ball animation test. Williams, R. 2001. The Animator's Survival Kit: A Manual of Methods, Principles, and Formulas for Classical, Computer, Games, Stop Motion, and Internet Animators, p. 39. London: Faber and Faber.

Figure 6: Screenshots of the pose sculpting process for Hotel Transylvania 2. Sony Pictures Imageworks. 2015. Original video available at: https://www.fxguide.com/fxfeatured/going-off-model-hotel-transylvania-2/ [Accessed 26 April 2021].

Figure 7: Squash and stretch principle in action. Jak and Daxter: The Precursor Legacy. Sony Interactive Entertainment. 2001. Original image available at: https://www.gameanim.com/2019/05/15/the-12-principles-of-animation-in-video-games/ [Accessed 26 April 2021].

Figure 8: A McCree smear frame. Overwatch. Blizzard Entertainment. 2016. Screenshot from: https://youtu.be/vIdeGmN__Pw [Accessed 20 April 2021].

Figures 9: Examples of anticipation. Williams, R. The Animator's Survival Kit: A Manual of Methods, Principles, and Formulas for Classical, Computer, Games, Stop Motion, and Internet Animators, p. 275. London: Faber and Faber. 2001.

Figures 10-12: Breaking down the types of frames in an animation. Williams, R. The Animator's Survival Kit: A Manual of Methods, Principles, and Formulas for Classical, Computer, Games, Stop Motion, and Internet Animators, p. 65 &66. London: Faber and Faber. 2001.

Figure 13: The curve editor in MotionBuilder. Brusi, P. 2021.

Figure 14: A block turning on a hinge. Becker, A. 2017. Screenshots from: https://youtube.com/playlist?list=PL-bOh8btec4CXd2ya1NmSKpi92U_I6ZJd [Accessed 9 May 2021].

Figure 15: Illustrating the difference between flat and dynamic posing with Mickey Mouse. Johnston, O. & Thomas, F. The Illusion of Life: Disney Animation p. 67. Glendale: Disney Publishing Worldwide. 1995.

Figure 16: Madeleine stretches as she jumps. Celeste. Extremely OK Games. 2018. Screenshot from: https://youtu.be/yorTG9at90g [Accessed 26 April 2021].

Figure 17: Combat in Batman: Arkham Knight. Rocksteady Studios. 2015. Screenshot from: https://youtu.be/QydZoW8Uwnc [Accessed 26 April 2021].

Figure 18: Combat in Spider-Man. Insomniac Games. 2018. Screenshot from: https://youtu.be/SeGveiYnB0I [Accessed 26 April 2021].

Figure 19: Ladder protruding above ground level. Tom Clancy's The Division. Ubisoft. 2016. Original GIF available at: https://gfycat.com/apprehensivedopeybonobo [Accessed 9 May 2021].

Figure 20: Capturing animation from gameplay to for NPC use. Uncharted 4. Naughty Dog. 2017. Screenshot from: https://www.gdcvault.com/play/1024309/Animation-Bootcamp-Uncharted-4-Naughty [Accessed 26 April 2021].

Figure 21:. Example of additive animation for Crysis. Crytek. 2006. Screenshot from: https://youtu.be/0JFYt8kGYhM [Accessed 26 April 2021].

Figure 22: Horse and actor motion capture for Call of Duty: Black Ops II Williams, D. 2012. Original image available at: https://tumblr.iamdanw.com/post/36240428321 [Accessed 26 April 2021].

Figure 23: CWM-250 Calibration Wand. Optitrack. No date. Original image available at: https://www.optitrack.com/accessories/calibration-tools/ [Accessed 26 April 2021].

Figure 24: CS-400 Calibration Square. Optitrack. No date. Original image available at: https://www.optitrack.com/accessories/calibration-tools/ [Accessed 26 April 2021].

Figure 25: Motion capture for Dawn of the Planet of the Apes. 20th Century Fox. 2014. Original image available at: http://www.cinema.com.my/Articles/features_details.aspx?search=2014.f_amazingmotion_20346&title=Amazing-Motion-Capture-in-Movies [Accessed 26 April 2021].

Figures 26 and 27: Preparing an Xsens suit for a shoot. Brusi, P. September 2020.

Figure 28: Gypsy Torso™ - Upper Body Mocap system. Meta Motion. 2004. Original image available at: https://metamotion.com/gypsy/old/gypsy-torso.html [Accessed 26 April 2021].

Figure 29: Craig Hayes with the Dinosaur Input Device. Knep, B. 1993. Original image available at: http://www.blep.com/rd/special-effects/dinosaur-input-device/ [Accessed 26 April 2021].

Figure 30: A performer wearing a suit for magnetic motion capture. AMM. 2010. Original image available at: https://www.researchgate.net/figure/3-A-performer-wearing-a-suit-for-magnetic-motion-capture-AMM-2010_fig4_255990108 [Accessed 26 April 2021].

Figure 31: Example of a shootlist. Craens, R. 2019.

Figure 32 and 33: Motion capture for Uncharted 4. Naughty Dog. 2017. Screenshot from: https://www.gdcvault.com/play/1024309/Animation-Bootcamp-Uncharted-4-Naughty [Accessed 26 April 2021].

Figure 34: A mocap dance card. Zadziuk, K. 2016. Screenshot from: https://youtu.be/_Bd2T7uP9VA [Accessed 9 May 2021].

Figure 35: AZRI on a control rig. Brusi, P. 2021.

Figure 36: Exporting AZRI from Maya. Brusi, P. 2021.

Figure 37: Maya export warnings. Brusi, P. 2021.

Figure 38: Freshly opened AZRI in MotionBuilder. Brusi, P. 2021.

Figure 39: Initial object groupings. Brusi, P. 2021.

Figure 40: Defining bones. Brusi, P. 2021.

Figures 41 and 42: Retargeting settings and the difference between untargeted (left) and source matched (right) data. Brusi, P. 2021.

Figure 43 and 44: Unedited mocap data and the cropped. Brusi, P. 2021.

Figure 45: Reorienting in Story mode. Brusi, P. 2021.

Figure 46: The animation after the core parts of the cleanup were finished. Brusi, P. 2021.

Figure 47: More groups and animation layers. Brusi, P. 2021.

Figure 48: The Z axis rotation curve graph for the hair. Brusi, P. 2021.

Figure 49 and 50: The end result of cleanup and manual animation. Brusi, P. 2021.