EaseMarks: Using Secondary Sketch Marks To Author and Communicate Motion Interpolation

Hadrien Nguyen

EASEMARKS: USING SECONDARY SKETCH MARKS TO AUTHOR AND COMMUNICATE MOTION INTERPOLATION

by

HADRIEN NGUYEN

THESIS

Submitted in partial fulfillment of the requirements

for the degree of Master of Science in Computer Science at

The University of Texas at Arlington

December 2023

Arlington, Texas

I dedicate this thesis to my spouse Tiffany Phan, who has helped me and supported me throughout this whole process and in life, but most importantly kept me in check and gave me motivation to see everything through.

ACKNOWLEDGEMENTS

A special thank you to Jaya Gratts (@ghostlysleuth) for providing me with the assets used as exemplars for the tool. Thank you to Dr. César Torres for guiding me through this long process. Thank you to Joshua Wilson and Dr. Vassilis Athitsos for being on my committee. Thank you to everyone who participated in the studies presented as well as users and reviewers.

12, 13, 2023

ABSTRACT

EaseMarks: Using Secondary Sketch Marks to Author and Communicate Motion Interpolation

Hadrien Nguyen, M.S.

The University of Texas at Arlington, 2023

Supervising Professor: Dr. Cesar A. Torres

Motion interpolation is a process where an animator transforms jerky frame transitions into rich motions that communicate anticipation, urgency, hysteresis, and even calmness. Animators leverage mathematical functions known as easing curves to modify the rate at which *in-betweens* are added to keyframes. While effective, easing curves are tedious to tune since they fundamentally lack the ability to encode spatial information. Inspired by timing charts and other standards from traditional cel animation, we introduce a motion animation technique where secondary marks, which we term *EaseMarks* (e.g., hatches, loops), are used to denote motion interpolation decisions. We synthesize an EaseMark Sketching Language and evaluate it through a crowdsourced study. This sketching language is then put in practice in a tool that allows animators to author motion interpolation by sketching or selecting different EaseMarks that affect an object's visual and spatial expression over time. We discuss how secondary marks can be used to expand the expressiveness and utility of sketching languages in other complex design practices.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Animation as a medium has changed considerably over the past century, but animators consistently struggle with being able to express their design intention [2]. The use of the computer offers great assistance, as it lessens the amount of work needed to be put in overall compared to traditional animation [3]. However, an amount of control can be lost by the animators when relying on tools that attempt to fill out the gaps.

Traditional animation describes the process of drawing each frame of an animation individually, the quick superposition of the frames creating the illusion of motion. The process of drawing the frames can be usually spread out between two people: a principal animator, and an assistant, called inbetweener [4]. The principal animator would draw keys: the most important points of a motion, the extremes. The inbetweener could then come in and fill the frames in between by respecting the instructions given by the principal animator. This is time consuming, especially for modern animation that can require up to 60 frames per second [5].

To alleviate the work needed by animators, modern animation software follows the metaphor of traditional animation by having the computer as the inbetweener. Timelines are widely used in animation software as a way to "key" different properties over time [3]. The computer can then calculate each frame by simply calculating he middle values of these keys. Every possible parameter of a motion can be keyed this way, whether it be the position, rotation, or even color of an object.

On a timeline, every parameter is independent. This allows for more minute control, but creates a lot of overhead during the animation process: if a big change happens at time *t*, every parameter that needs to change needs to be added to the timeline one by one [5]. Since each behavior is only a dot on the timeline, there is also a lack of visualization of the relationship between parameters or visualization of the relationship between parameters to the overall motion. On top of that, extra information can be given to the computer through **Easing Curves**, mathematical functions that describe how the property should grow between two frames [5].

1.1 Problem Definition

The issue with the timeline workflow is that there can be a lack of overall synthesis during the process. Tuning requires having all the prooperties in the timeline already. Information is spread all over. Easing curves are in a different window than the viewport, and as animations get more complex, timelines get full of different parameters. Because each one gets one line with independent keys and easing curves, it is difficult to get an overarching view of the motion.

Applying the principles learned from traditional animation to computer animation requires a way to design layered and expressive motions. Authoring these complex animations involves a streamlined way to manipulate different behaviors while still keeping track of how everything works together. An animator should be able to declare how a motion should behave and the computer should comply to these instructions, in the same way that a principal animator draws keys and timing charts for the inbetweener to use. This requires a redefinition of a language of motion, as well as an exploration of sketch for the purpose of applying a visual language to define visual behavior.

My objective is to come up with a tool that allows users to author complex stylized animations while offering an effective method of communicating said animations. Such tool would be successful if it both allows the authoring of said animations and other users are able to understand and predict the desired motion based on the sketches present within the tool.

1.2 Contributions

Drawing inspiration from traditional animation, I present EaseMarks, a framework that focuses on authoring and communicating motion animation with the use of sketching. EaseMarks represent different behaviors that an object can exhibit. In my tool they represent different affine transformations that the object will follow. An EaseMark can represent the squashing and stretching of the object for example. The user is able to manipulate this EaseMark through sketch and add different EaseMarks together. Our design tool, the EaseMark Sheet brings this concept together in a simple interface that can be presented by itself to describe complex motions.

I explore crowd-sourcing as a method of verifying the correspondence between the EaseMark language and user intuition. I then evaluate the tool by showing examplars created with the tool and comparing it with current animation processes.

1.3 Outline

I first introduce key concepts in Chapter 2 before diving into other research that has inspired this paper and to which I have built upon in Chapter 3. I will then explain my approach including what worked and what did not when it came to understanding the relationship between sketch and motion in Chapter 4. In Chapter 5, I will present my tool and explain how it allows its users to create lively animations in a way that utilizes the visual properties of sketch authoring. Finally, I will evaluate my contribution and discuss future work in Chapter7.

CHAPTER 2

Background

To understand the unique design concerns and opportunities within motion authoring, I present an explanation of principles that are relevant to the animation space. Overall, the following survey of animation principles describes what stylized animation in video games means and how it can improve the visuals of a game.

2.1 Squash and Stretch

Squash and stretch is a well known principle which describes the tendency of an object to deform when moving depending on its material and density [4]. For example, a water balloon will flatten greatly upon reaching the ground whereas a bowling ball will show no deformation. One rule associated with squash and stretch is that while an object can be stretched greatly, that object should keep the same or a similar volume. When added to an animation, squash and stretch can greatly improve the communication of the motion to the audience.

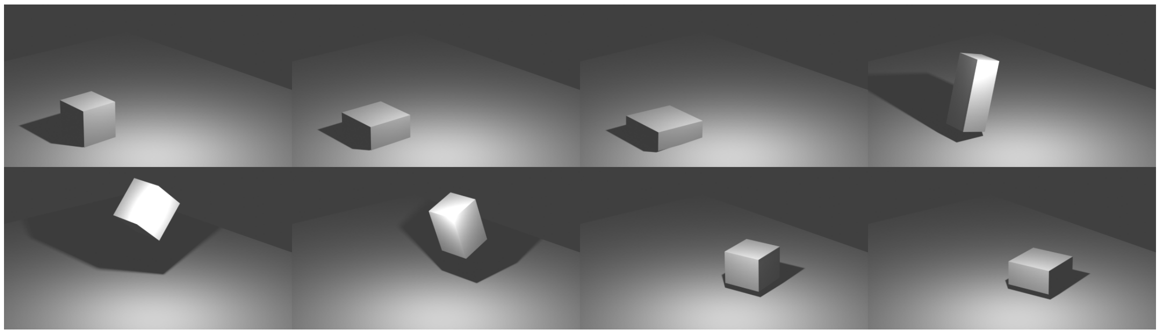


Figure 2.1: Animation of a 3D box jumping, separated into 8 frames, read from left to right and top to bottom.

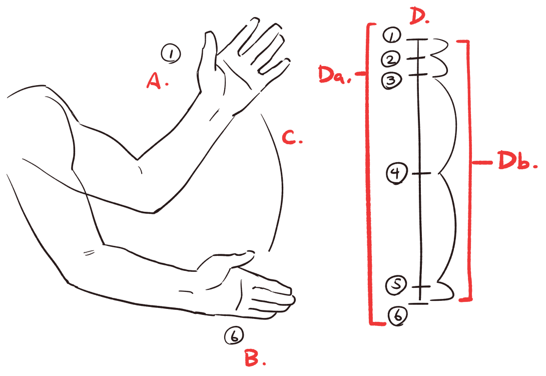


Figure 2.2: Example of a Timing Chart as used in traditional animation. A) First Key. B) Last Key. C) Arc of Motion. D) Timing Chart. a) Using frame numbering notation. b) Using arc notation

Video games with 3D animation can be limited when applying squash and stretch as they rely on models: 3D representations of their objects that can be puppeteered [5]. Models are useful when wanting to maintain consistency and coherence throughout a game. The character will always have the same size and the same proportions in all of their animations. With models, squash and stretch can be applied through temporary deformations like seen in Figure 2.1 which show different frames of an animation depicting a box squashing and stretching. Modern methods of interpolation require the manipulation of independent parameters, which can get even more complex when adding stylization like deformations, when those parameters should be working in tandem.

2.2 Timing

One of the principles presented in a paper by Lasseter of Disney Pixar is Timing [3]. It describes the amount of frames that can be needed for a single motion, whether it is slow (more frames) or sudden (less frames). While animators used to rely on a set number of frames for animations, screens can endure a wide variety of frame rates - 20 to 240 frames per seconds; especially for mediums like video games where irregular frame rate needs to be accounted for at all times. Even so, this idea of timing is still used today through artificial simulation of lower frame rate. In the video game Guilty Gear Xrd, characters are animated at a lower framerate both to emulate the anime style but also to give more appeal to their characters [6]. In a similar manner, the Spider-Verse skin in Marvel's Spider-Man: Miles Morales references the movie by intentionally reducing the smoothness of its animations [7].

There is a relationship between the intentional removal of frames and the process of rotoscoping. Rotoscoping describes the systematic tracing of live-action footage in order to have a basis for animation. An issue that comes with rotoscoping is that because it can easily be a systematic process rather than intentional, information can be lost. In the Animator's Survival Kit, Williams theorizes that "our brain is normally picking up so much information - more than we are conscious of - and the drawings don't pick enough of it up" [4]. In this sense, the animator should act as a filter of information, showing the necessary and only the necessary. Motion clarity can get hurt if the wrong frame gets deleted.

Because timing is difficult to understand, predict, and communicate, traditional animation makes the use of timing charts [3]. Figure 2.2 represents an example of what an animator would provide as a timing chart. A and B represent the keyframes given by the principle animator, superimposed to present them simultaneously. C represents the arc of motion, describing the hand falling in a circular motion as time goes on. D is the timing chart itself, which instructs the inbetweener to draw frames 1, 2, and 3 close to each other, while also isolating frame 4. The chart is meant to be interpreted as the motion *Slowing in* and *Slowing out*: starting out slow, being very fast at 4, and finally slowing down at 5 and 6. In this example, sketching is used as a language to communicate motion intent. Using a visual language to describe a visual

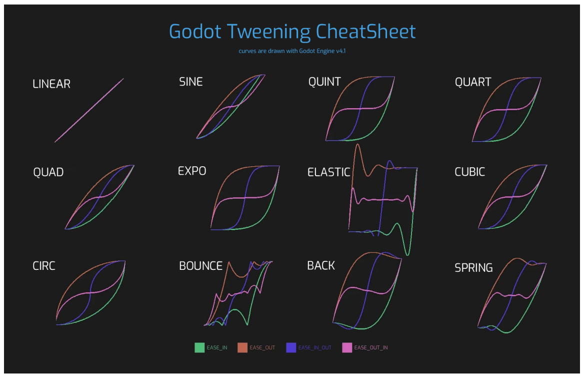


Figure 2.3: A list of built-in easing curves present in the Godot game engine [1]

behavior allows for a more direct approach to authoring, and enables the audience to intuit correspondence more easily.

2.3 Easing

The principle of Slow-in/Slow-out is most commonly known as *Easing* in computer animation [5]. In traditional animation, it represents the spacing of the frames in between two keys. For example, if the motion is one of a drawer opening, the drawer will need to open fast at first and slow down before it is fully open. This would correspond to a motion with slow-out. This would mean that the frames should be closer to each other as they reach the end of the trajectory, but more spaced out when the trajectory begins. In interpolation, since inbetweening can be considered continuous, there is a notion of density rather than amount. If the frames of an object are more dense in an area, that means that the object is moving slower.

Most interpolated animations makes the use of easing curves, mathematical functions that determine the spacing of an animation through time [3]. There are four main behaviors that easing curve can describe:

- Ease-in: the object starts out slow and picks up speed as it goes. Example: A rocket launching.

- Ease-out: the object starts out fast and slows down until it stops. Example: Opening a drawer.
- Ease-in-out: the object starts out slow, picks up speed, then slows down. Example: A car between two stoplights.
- Ease-out-in: the object starts and ends fast, but slows down in the middle. Example: A ball going up and down.

These four curves are then applied in conjunction to a function with a domain of [0, 1] in order to define the specified behavior. Figure 2.3 shows the predefined transition functions built-in the Godot game engine [1]. All of the easing graphs go from the coordinates (0, 0) to (1, 1). If a curve goes above or below these bounds, it means that the interpolation point will overshoot the bounds and will not stay strictly in-between the two keys. This is can be useful when the goal is to show the anticipation - the initial gain of momentum for an object - or the follow-through - the final release of momentum for that object - of a given action.

2.4 Arc of Motion

Finally, the last animation principle is the *Arc of Motion*. Motion can be described as a unit of movement that can be broken down into the intent, the action, and the consequence. A motion can be someone lifting their arm, a tree falling down, or a projectile being thrown. More complex animations, like a character dancing for example, can include several motions. The Arc of Motion describes the trajectory of this motion, the route it will take. It represents the path the audience will follow when reading the motion and the driving force of the overall movement. If too many parts go in too many different ways, the arc will be muddled and the animation more confusing [4]. In video game animation, the arc is not usually taken into account as position is just another parameter. While modern animation tools like Blender allow for visualizing a motion path [8], that path is independent from other parameters like easing or framerate. If the user wishes to modify the easing of an object following a path, the user would have to use an easing curve that can be tuned using information about the timing of the motion but not about its spacing.

My goal is to understand how animators structure their animation and to isolate the important aspects of a motion. These principles are also where other tools drew inspiration from and understanding the principles allows for a better understanding of these tools.

CHAPTER 3

Related Work

3.1 Motion Sketching

The synthesis of sketch-based animation techniques reveals a trend towards using intuitive and simple sketch marks to define and manipulate motion. Popović et al. [9] demonstrated the use of line sketches for specifying object trajectories in rigid-body simulations, highlighting the computation of physically plausible motions that align with the animator's intent. Similarly, Chao et al. [10] developed an interface for human motion retrieval, where continuous lines sketched over a character capture the desired motion, emphasizing simplicity and expressiveness. Guay et al. [11] introduced space-time sketching, enabling the creation of coordinated motion with single-stroke sketches, demonstrating the efficiency of minimalistic sketch marks.

Further, Li et al. [12] explored sketch-based motion retrieval and refinement using basic sketch lines to construct and refine motion for articulated characters, underscoring the utility of simple sketch elements . Xie et al. [13] enriched rotational motion in keyframe animations through curved lines to edit angular velocity trajectories, showing the use of curves for defining rotational dynamics. In the realm of dance choreography, Moghaddam et al. [14] used stick figure sketches and annotations for ballet motion representation in 3D environments, showcasing the effectiveness of rudimentary sketches. Lastly, Shi et al. [15] presented a method using stroke lines for creating reusable animations across different characters, emphasizing the versatility of simple sketch lines.

Overall, these studies collectively highlight the reliance on fundamental sketch marks such as lines, strokes, curves, and stick figures in animation, serving as effective tools for specifying and refining motion, thereby making complex animation tasks more accessible and manageable. I explore the use of complex sketch marks and how the relationship between different type of sketch marks can allow for more complex motions.

3.2 Procedural Inbetweening

In traditional animation, keys are used as anchors for the drawing of inbetweens. Procedural inbetweening streamlines this process by letting the animator set the keys needed and letting the computer automatically filling the missing frames. Easing curves can make interpolation more organic but go hand in hand with an extensive keying process from the animator. Several approaches have been explored to generate a full animation from a very limited number of given keys. It is possible to use a geometrical understanding of the shapes that are to be interpolated in order to interpolate between them. Blender contributors developed shape keys [8], which allow a user to set different poses of an object (a face for example) and the engine will be able to blend the shapes together and to transition between them effectively. Machine learning methods have also been utilized in order to extrapolate frames from a limited input. They are also able to apply styles onto these animations without the need to formally define them [16]. While these approaches allow for a large output of animations, EaseMarks focus on how to fine-tune these interpolations easily and to leave the animator with as much control as possible throughout the animation process.

The method in which keys are set allow for a range of flexibility while still allowing the animator to have control over the final product. Animation of characters is often achieved by rigging a skeleton and puppeteering it over time. Anzovin et al. [17] revised this approach by allowing for temporary rigs and re-rigging on the spot. This unlocks a full range of motion where the behavior of the object depends on the specific motion that is needed. I use this idea of behavior-first approach where the process of animating begins at the desired motion. On top of that, I focus on spatial keys rather than keys based in time, I spotlight the desired regions of interest and fine tune the behavior within these regions.

3.3 Sketch Based Authoring Tools

Sketching is an intrinsic part of the animation process. Keyframes are usually sketched before being rendered. In traditional animation, inbetween intentions are communicated through timing charts. They allow for two different people to work on the same animation by creating a hierarchy betwen key frames and in betweens. They often rely on annotations to sketched animations [4].

Sketching in Computer interaction allows an user to quickly author and display the result of authoring by essentially being a way to render direct manipulation. In animation, this usually comes in the trajectory of the motion of a particular object like in K-Sketch [18], which consisted of drawing the path that it takes. Other tools like Draco [19] took a different approach by focusing on kinetic textures, effects that are meant to loop or that describe a collection of objects. Sketch has also been used to program non-trivial animations like in Motion Doodles [20] where sketch not only described the path of motion, but also the behavior of the character as it followed the path. A limitation of these tools is that they rely on preset notations that the user has to learn. Hashim et al. [21] attempted to "unify procedural authoring" by offering a notation that can be applied to many aspects of an animation like style, position, color. Through crowd-sourcing, I focus on people's intuition of sketch in order to formalize my syntax.

3.4 Rules-based Stylization

Since the rise of 3D animation, there have been interests in translating principles learned from 2D animation over the last century to 3D [3]. Tools like the Shading Rig by Petikam et al. [22] and the linework generation in Spider-Man Across the Spider-Verse [23] focus on nonphotorealistic rendering, while still keeping the artist in mind by giving them a way to tune the resulting render without having to modify post-processing effects under the hood. I use this approach to artist control by letting the user see the end result of the animation as they are tuning it. An approach to stylized motion is the Stretch Engine [24], which uses the concept of squash and stretch automatically over a 3D skeleton. Ma et al. [2] used a layered approach by adding onto one another several stylizations over the course of an animation and parametrizing them. This allows for a process similar to visual shaders in the sense that the user does not need to know the implementation in order to create complex animations. I follow Ma et al.'s guidelines and provide the user with different motion behaviors that can be added onto one another and full customization of the behavior over time rather than restricting that behavior to procedural rules.

The works presented cover three fundamental aspects to motion authoring which are the ease of use, the amount of control given to the artist, and the expressivity of the product. In the following chapters, I will describe how sketching allows for a balance of all three.

CHAPTER 4

Understanding Motion and Sketch Marks

My objective is to be able to come up with a sketching language that can be also interpreted as a motion language. This language needs to reach a certain level of intuitiveness, as a language that is intuitive can be easy to learn, understand, and use. A good mark is one that can represent a motion and/or give motion information in a way that is understandable to a maximum amount of would-be animators. I try to come up with a sketch-to-motion matching, where any instance of motion can be described by a simple drawing. To come up with this matching, I went through an iterative process of refining what sketches and motions are.

4.1 Method

My goal throughout this process was to come up with a mapping of marks to motion, as represented abstractly in Figure 4.1. A space is a broad categories that follow certain rules, and I attempt to map between the space of marks and the space of motions. The space of marks represent any sketch marks or sketch notations that can be added onto a line. The space of motions represents the set of possible movements between two points. Neither of these spaces are exhaustive as we want to focus on motions that are short and limit the complexity of sketch marks. An ideal matching is a matching that is reproducible, understandable, and broad. When shown a motion, the user should be able to represent it with marks, and when shown a mark, the user should be able to predict the motion.

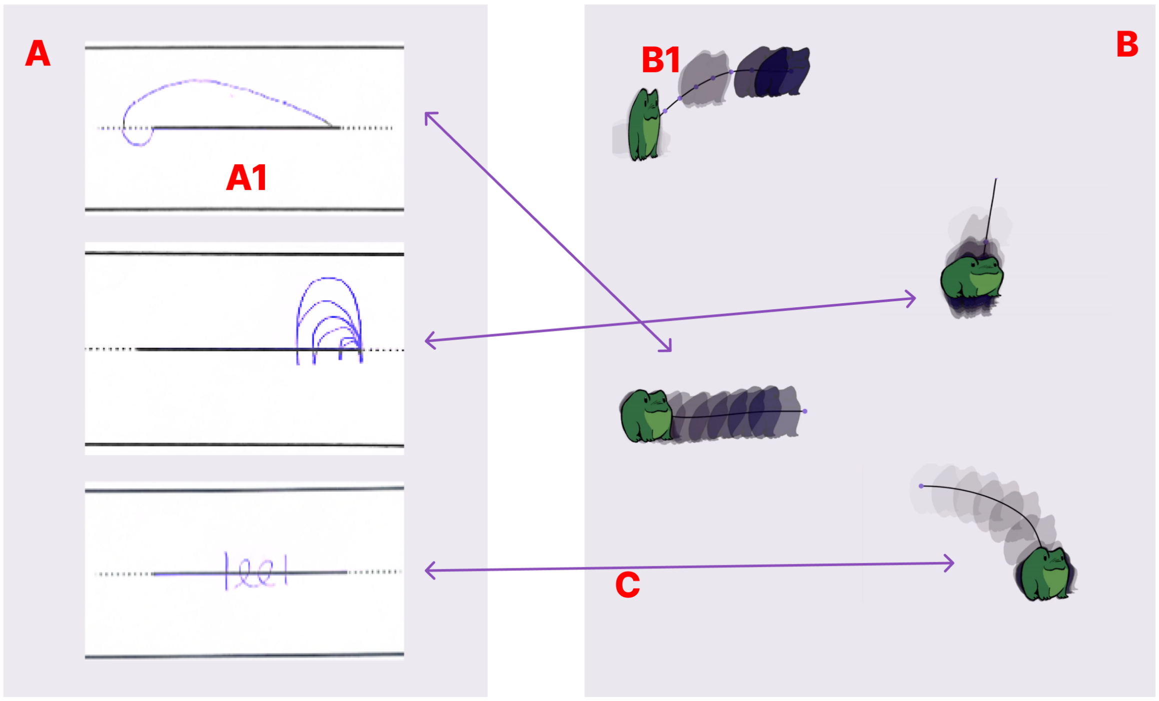
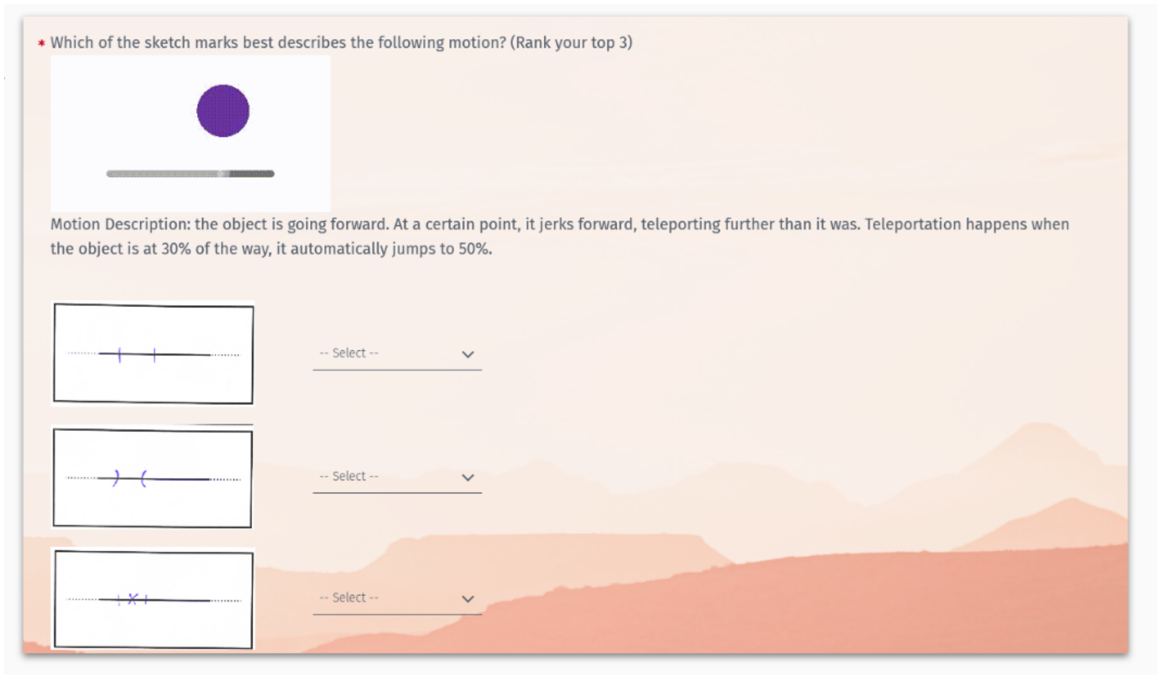


Figure 4.1: Our goal is to be able to produce a mapping between A) the space of marks and B) the space of motion. A valid mapping exists if for a mark A1) there is an existing motion B1) that exists in the space of motion. C) A matching between a mark and a motion.

I structured the two spaces according to typologies. A typology is a list of categories that can divide a space. I came up with these typologies by analyzing different aspects of both marks and motions and coming up with broader and broader categories that could encompass them. If a motion does not match any of the existing categories, I needed to create a new one. If all the motions in one category also fit in another, I needed to merge them. I considered the spaces defined once I could safely categorize any new mark or motion.

I iterated between a total of three different typologies of marks and one typology of motion. The typologies of marks were changed adding more limits to the space, for example, allowing annotations that are independent to the line or not. For each typology of mark and motion, I deployed a Human Intelligence Task (HIT) through


* Which of the sketch marks best describes the following motion? (Rank your top 3)

Motion Description: the object is going forward. At a certain point, it jerks forward, teleporting further than it was. Teleportation happens when the object is at 30% of the way, it automatically jumps to 50%.

Figure 4.2: An example of a survey question. Participants are given a Motion GIF and a description of the motion in words. They are asked to rank the three best marks.

Amazon Mechanical Turk [25], a crowdsourcing platform. Each participant was asked to look at a GIF portraying a motion and rank the marks available to them. I recorded the categories that each motion and mark belonged to and hoped to find a relationship by looking at whether a motion category was more associated with a mark category and vice-versa.

4.1.1 Motion Typology

To create a motion typology, I first surveyed existing motions in both motion pictures and video games. I looked at video clips on YouTube and went through the videos at a slow rate and isolated motions. Once isolated, I was able to go through them frame by frame and note the changes happening on screen, whether it be a change of speed, scale, or color.

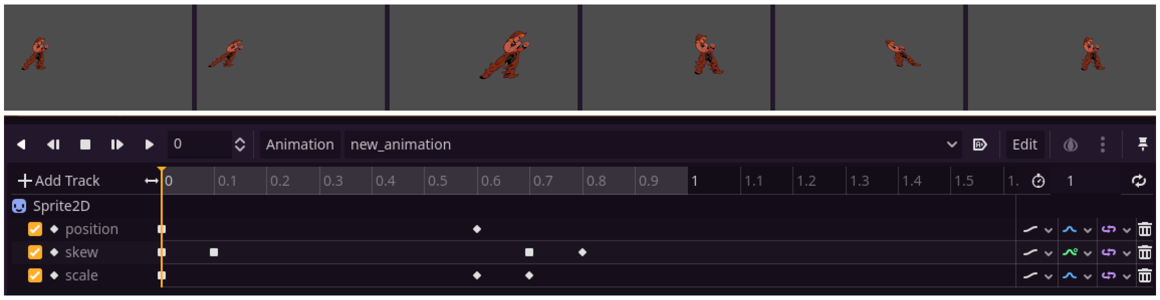


Figure 4.3: Animation of a Character from a Fighting Game. Six frames of the animation are on top of the timeline that helped create the animation

I then authored animations using existing tools. Figure 2.1 was created using Blender [8], and represents a motion that follows an arc while being deformed through squash and stretch. Figure 4.3 represents an animation made in the Godot Game Engine [1]. The animation is simple, going from left to right, but the timing of the different parameters affects how the animation is perceived. For example, upon reaching the end of the animation, the character changes in scale suddenly before reverting to normal. From these motions created, I took important characteristics and came up with a typology.

I define a motion as change in relation to an arc. An arc is a path that an object seems to follow during an animation, relative to an environment. If an object only changes color during an animation for example, I do not consider it a motion. If an object changes position in the screen and its environment moves in the same way, it is not considered to be in motion. Overall, an object is in motion if an arc can be drawn that follows the change on the screen. I then organized motions along 5 traits: Limited, Anticipation, Follow Thru, Easing, and Oscillatory.

- **Limited:** A motion has the Limited trait if at any point the object does not change in a continuous manner. In traditional animation, it could perhaps represent all motions where an animator draws frames one by one. Interpolation has the specificity that it is able to easily calculate almost-exact values for any continuous function. In fact, it is its default setting. Including the limited trait allows for differentiating any motion where an abrupt change is intentional. Abrupt changes can represent speed or even force of impact [4]. In video games, because animations need to be short, abrupt changes can help sell the clarity of a motion in a few frames.

- **Anticipation:** A motion has the Anticipation trait if it has any special behavior at the beginning of the arc, that is, if there is a defined difference between its behavior at the beginning of the arc and while it is following it. One of the more famous animation principles, anticipation represents the preparation needed and the storing of potential energy for a motion. In video games, this is a double-edged sword as anticipation can increase the impact of an action but feel less reactive as a result. It is often associated with enemies telegraphing their actions to make it easier for the player to predict, or with the player "charging" a strong attack [26].
- **Follow Thru:** In a similar manner, a motion has the follow-thru trait if there is a defined difference between its behavior at the end of the arc and while it is following it. Follow-Thru is similar to anticipation, but instead relates to the consequences of the motion. Strong movements acquire momentum and that momentum is shown in follow-thru [4]. Video games use follow-thru when selling the impact of a motion, but another aspect of it is un-cancellable animations, meaning animations that need to finish playing before the game can accept any other user input. This creates consequences for the player and a sense of danger after using a strong attack, as they are left vulnerable while the animation plays [26].

- **Easing**: A motion has the Easing trait if any change happening during the arc is nonlinear, that is, that the change is nonuniform over the interval of time in which the object travels the arc. Called Slow-in/Slow-out in traditional animation, it is the easiest way to bring life to an animation, as linear motions - motions without easing - can be often seen as "wrong" to an audience without them even knowing why [3]. Going back the the "charging" example, easing would be useful to show an attack that gains speed as it goes, therefore gaining strength.
- **Oscillatory:** A motion has the Oscillatory trait if it travels any point on the arc more than once. Oscillatory represents any object that goes back and forth within the same arc. There is some overlap between Oscillatory motions and motions that have Anticipation or Follow-Thru, as they often go off the main arc. Oscillatory motions usually involve quick movements like the flapping of a wing or shaking a bottle.

Using this terminology, we can say that a bouncing ball would have the Easing trait as it would lose speed as it rises and would gain speed as it drops. The motion would also have the Follow Thru trait as it would Squash once it reaches the ground. Finally, it would have the Oscillatory trait since it would bounce back in the same trajectory that it followed previously.

4.1.2 Secondary Sketch Mark Typology

Coming up with a Sketch Mark typology was different than for the motions. Before we started, we set the scope of what would be considered a mark. The most important one was that they would be **secondary sketch marks**, meaning that they are annotations onto a main line. From there, we attempted different ways to mark this line and determined the possibilities that those marks could be interpreted and how.

The first characteristic of secondary marks was whether they are **Inked** or **Inkless** (**Materiality**). The former means that the translation process takes into account the lengths of the curves constituting the secondary mark. An example could be a secondary mark made up of two curves, one twice as large as the other, intersecting the arc at different intervals, indicating that the object should hold longer at the larger mark. On the other hand, an Inkless secondary mark does not use the length of curves in the translation process. The information gathered is instead based on the intersections or the type of mark.

The second characteristic of secondary mark is their **Complexity**. An **Atomic** secondary mark is one that defines the existence of several independent points of interest, where each point is a behavior in the resulting motion. They can take the form of crosses, ticks, loops, etc. where each one can map to one event in the motion. A **Dual** EaseMark is one that describes a behavior within ranges. It can be a single curve intersecting the arc at two or more points indicating a specific behavior between those points. A **Complex** secondary mark describes any combination of several marks to define a change in one or more behaviors over a range.

The third characteristic is the **Position** of the mark. A mark can be on the line itself, presenting information by intersecting it. It can also be an annotation away from the mark used to define overall parameters like a heart shape or a loop. The inclusion of the position was debated as I am looking for the use of sketch to denote spatial cues; accepting annotative marks would be counter-productive to that effect and become an computer vision exercise of recognizing drawings.

ID	Name	Limited	Anticipation	Follow Through	Easing	Oscillatory
M1	Teleportation	✓	-	-	-	-
M2	Limited Frame Rate	✓	-	-	-	-
M3	Anticipation	-	✓	-	✓	✓
M4	Pull Release	✓	✓	-	-	✓
M5	Elastic	-	-	✓	✓	✓
M6	Bounce	-	-	✓	✓	✓
M7	Ease In	-	-	-	✓	-
M8	Ease Out In	-	-	-	✓	-
M9	Wiggle	-	-	-	-	✓
M10	Back Forth	-	-	-	-	✓

Table 4.1: A table representing the 10 motions used in the surveys as well as the traits associated with each of them.

4.2 Prototypes

The first step in coming up with an EaseMark Language was to use crowd-sourcing in order to gather insights over what a large population would see as an "intuitive" language. Since I needed results that could be analyzed and a large sample size, I conducted an elicitation study which asked participants to rank the marks that corresponded best to a motion.

4.2.0.1 Pilot 1: Full Typology Study

The first pilot study consisted of using the motion and secondary mark typologies defined above. I chose 10 motions sampled from the motion typology, all of which were linked to an easing curve whose only parameter is an object's position along a trajectory over time. For each motion, I randomly selected 6 combinations of the three sketch mark categories. In order to reduce bias, the combinations were randomized for each sketch mark choice; however each mark had to be drawn by me, which ended up introducing some sketch bias.

I synthesized the results by first weighting the ranks. A rank of 3 meant a score of 1, a rank of 2 meant a score of 2, a rank of 1 meant a score of 3, and not being

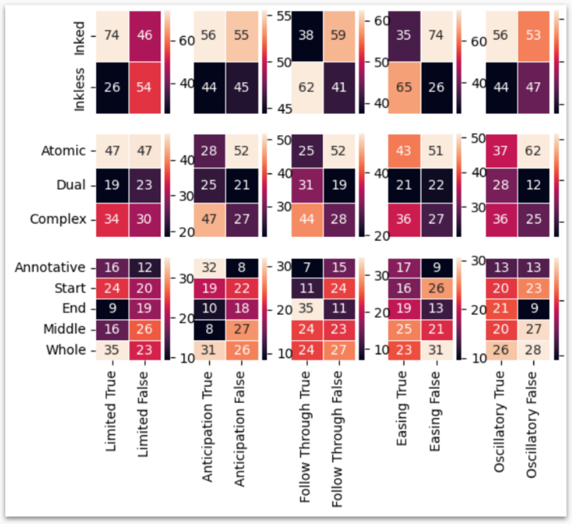


Figure 4.4: Participant preferred matching of Mark attributes (vertical axis) to Motion attributes (horizontal axis)

ranked meant a score of 0. The next step was to group each motion by their traits, If a motion had the Limited trait, it would go in the *Limited True* category, and if it didn't have that trait it would go in the *Limited False*. Finally, each mark and motion were divided by trait categories into heatmaps.

The first results shown in Figure 4.4 were inconclusive. The goal was to match sketch mark characteristics to motion characteristics but none of the connections were statistically significant. One of the issues was in the pool of Ease Marks presented to the participants. Since they were produced by the same person and randomized, a lot of matches were not present for the participant to choose. For example, 4 out of 10 motions did not have an option for an annotative EaseMark. Another issue is that some of the options that were presented had no real connection to the motion and were only there because our randomized sampling did not fully cover all characteristics of the typology. This resulted in participants choosing "least worse" marks rather than the "best" ones. The diversity of marks also had the issue of being visually complex.

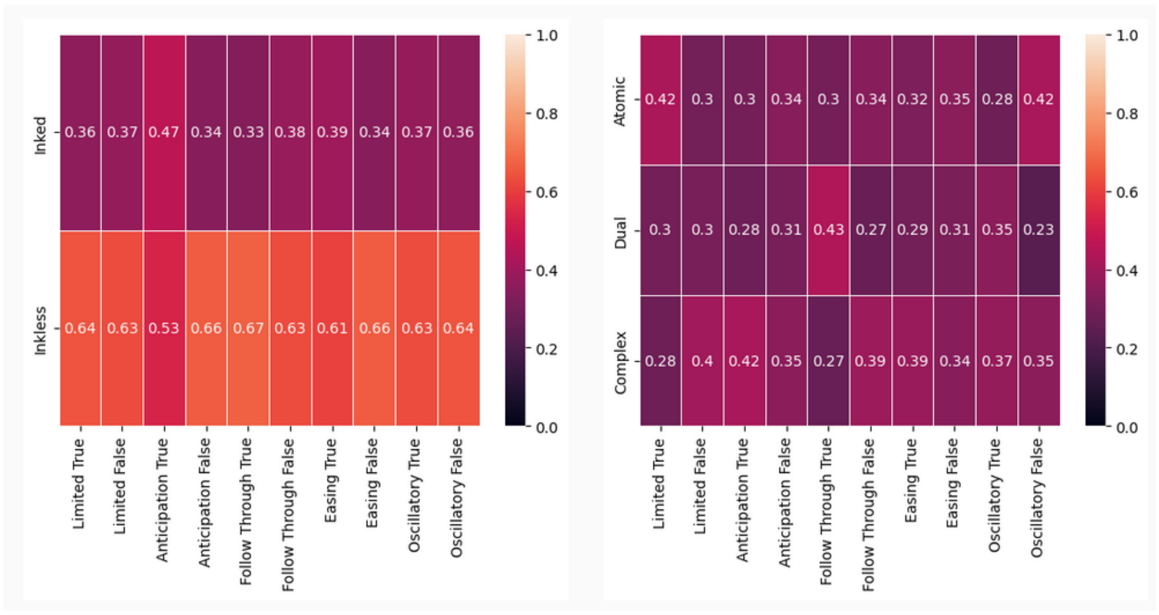


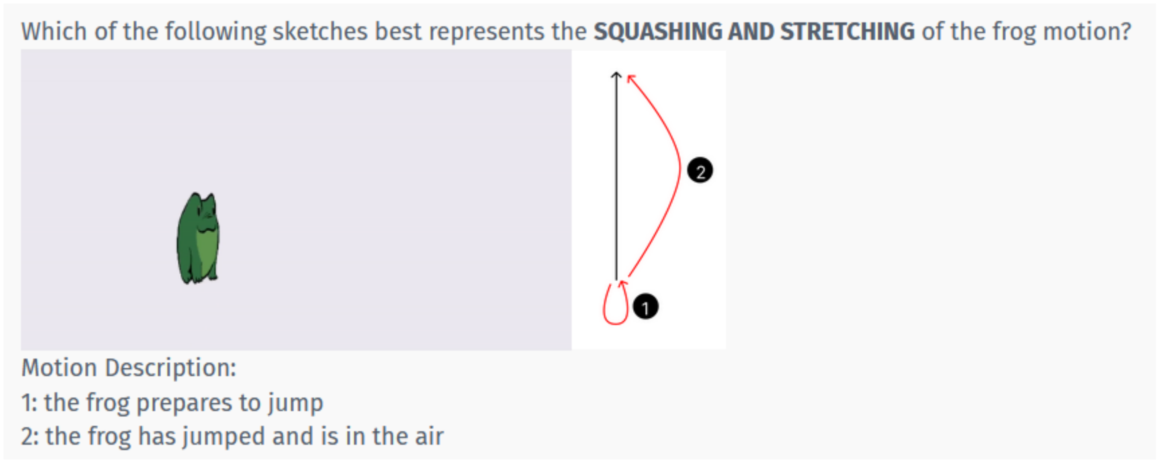
Figure 4.5: Participant preferred matching of Mark attributes (vertical axis) to Motion attributes (horizontal axis)

Participants had trouble determining if one mark was better than another if they were completely different.

4.2.0.2 Pilot 2: Motion Marks to Motions

The second approach was to reduce the amount of sketch mark characteristics presented to the participants. I first eliminated the "position" characteristics – all marks were positioned in the place that makes the most sense for the motion presented. For example a motion with the *anticipation* trait will have a corresponding mark at the beginning of the trajectory. This reduced the realm of possibilities to 6 combinations with one of two levels of *Materiality* and one of three levels of *Complexity*. This also allowed me to disregard the need to randomize since I could exhaustively sample every possible combination of sketch marks.

The results in Figure 4.5 were easier to analyze but were once again inconclusive. The results i was looking for were strong preference for a specific mark characteristic


Which of the following sketches best represents the **SQUASHING AND STRETCHING** of the frog motion?

Motion Description:
1: the frog prepares to jump
2: the frog has jumped and is in the air

Figure 4.6: Example Question presented to participants. Each participant is presented with a Motion GIF, a description of how the trajectory is segmented, and a caption of the motion.

for a type of motion. For example, if motions with the *Limited* trait were intuitively linked to the *Inkless* mark trait, I should see a high frequency of *Inkless - Limited True* and a high frequency of *Inked - Limited False*. This is not what I see, preferences seem to end up being uniformly distributed.

Both pilots 1 and 2 were associated with motions that were only defined by a change in position over time. The results indicated that there needed to be a change in both how I defined the marks and how I presented the question to the participants.

4.2.0.3 Pilot 3: Marks to Stylized Motions

The first break from the previous studies was separating the trajectory into segments. Only having a straight line meant that I did not gain more insight about the use of spatial cues, so a new assumption that was made is that every new motion presented will be separated by me into segments. The participants, like last time will determine the best secondary mark, but only for a chosen behavior like Squash and Stretch or Rotation. This allowed me to also structure the language around a

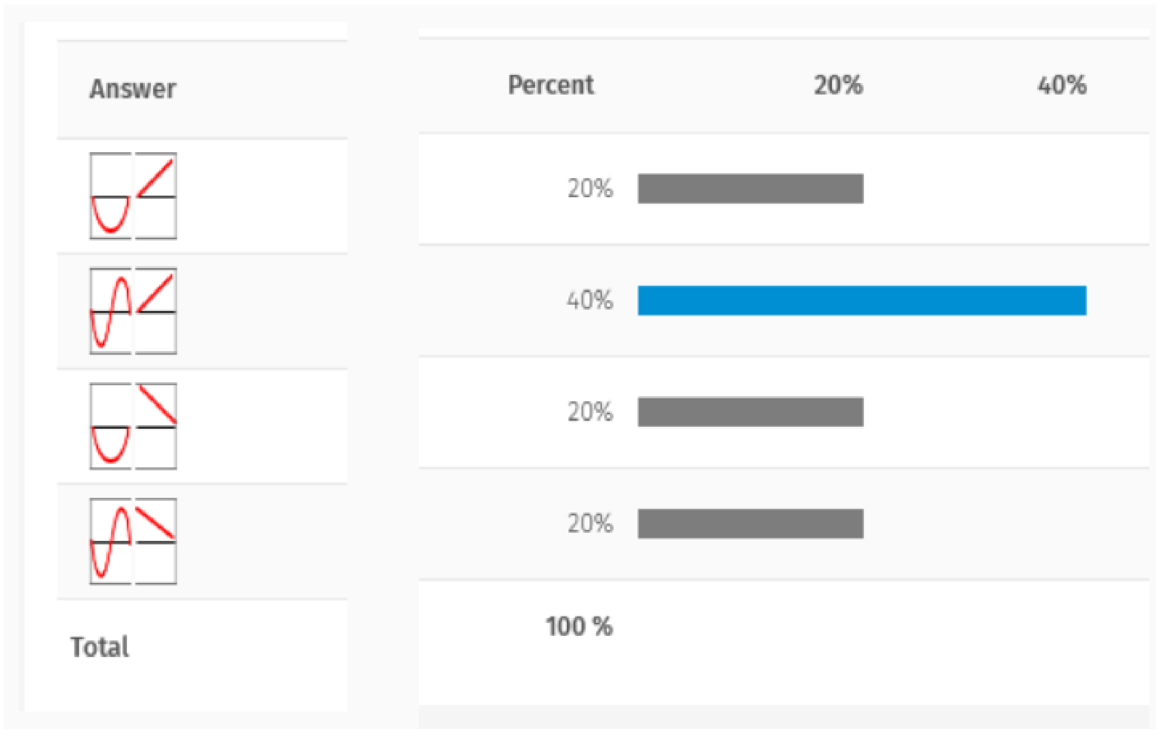


Figure 4.7: Participant preferred mark for a jumping frog

segmentation, which reduces the visual clutter and the overall complexity presented to both participants and the resulting design tool.

The study was also changed in giving a broader context to the animations. For example, the survey in 4.6 represents a jumping frog. The description given was: "1: the frog prepares to jump, 2: the frog has jumped and is in the air", which is less abstract than the previous studies. The focus was on giving as much context as possible for the motion so that the participant would not feel like they are choosing randomly.

Participants were given four choices for each motion, Figure 4.7 shows the results for the question in Figure 4.6. There is a disconnect between the mark chosen the most by people (mark 2) and the mark that was used to author the animation (mark 3) with easing curves. This disconnect can be explained either by participants focusing on the overall position of the frog rather than the squash and stretch behavior, or by the participants not having a frame of reference.

Overall, elicitating sketch through crowd-sourcing has drawbacks. The best way to elicitate sketch is by letting participants sketch, but if analysis is needed, there needs to be some categorization of the sketches drawn, either by the investigator or by a computer. On top of that, the thought process behind the choices made is missed, this could possibly be resolved by asking participants why they chose the mark they chose after each question.

The third pilot, which focuses on segmentation of trajectories and one mark per behavior helped me focus on what needed to be done for an animation tool that authors lively and organic animations. The iterative process of the elicitation helped me focus down and redefine spaces by first reducing the number of categories, and then rethinking how the motion itself is presented.

CHAPTER 5

EaseMark Design Tool

The design tool provides a proof of concept for using spatial cues in the animation process. I focus on an interface where the relationship between parts is clearly defined and provides a structured approach to animating.

5.1 User Interface

I present *EaseMarks*, and a tool built around *EaseMarks* that helps users generate animations in a way that is fast, iterative, and intuitive. The focus is on animating a singular 2D polygon that can be affected by any affine 2D transformations such as translation, rotation, or scaling. Every animation made with the tool corresponds to a singular motion, meaning that the object goes through one defined trajectory. That trajectory can then be separated into one or more segments. A single *EaseMark* describes the change of a single behavior throughout a single segment. EaseMarks can be drawn on to modify the behavior they represent. They will automatically correct what the user draws to be interpretable by the computer. There are also presets available that can be cycled through, which offer a good basis that can be modified through sketch. The tool consists of three main regions:

- **Cel Sheet** The *Cel Sheet* contains the whole animation by drawing the overall *trajectory* (or Arc) and how that *trajectory* is separated in *Segments.* The User can add "*Keys*" directly on the *trajectory* in order to tune the animation to their liking. The *Cel Sheet* also includes the *Object* that is to be animated and the *Onion Skinning* of the object (and how it is modified over time).

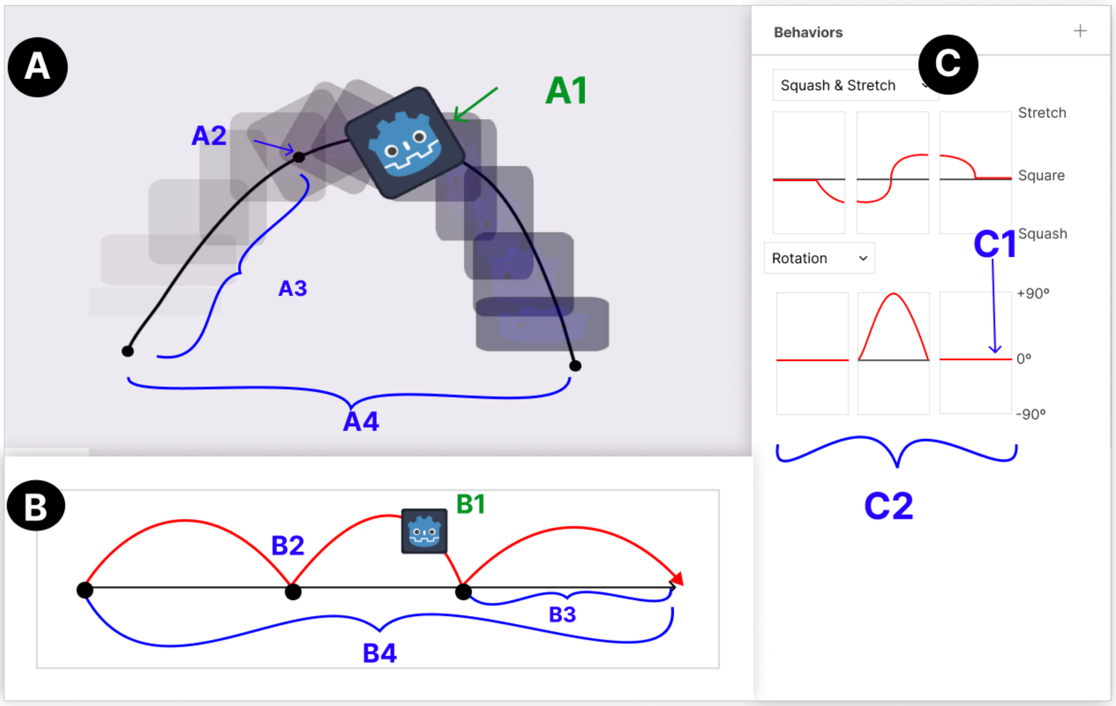


Figure 5.1: Description of the EaseMarks design tool. A) Cel Sheet A1) Object that is being animated A2) Spatial key A3) Segment A4) Full trajectory. B) Timeline B1) object representing the current position along the segments B2) Spatial key positioned proportionally along the trajectory B3) Segment B4) Full trajectory flattened out. C) Behavior Panel C1) EaseMark C2) EaseSheet composed of one EaseMark per segment

- **Timeline** The *Timeline* allows the user to determine the order in which the keys will appear in the animation. This works by letting the user select *segments* in order. This panel automatically shows the flow of the animation with a red arrow so that the user can keep track of it. The user can select one of the *segments* in order to work on it by itself.
- **Behavior Panel** The *Behavior Panel* represents all the *behaviors* that the user will add on to the object. The user can choose which *behavior* to add amongst a list (all affine transforms) and is able to modify the strength of that behavior by sketching an *EaseMark* or choosing one of the default *EaseMarks* available. Each behavior is represented by an *EaseSheet* which contains as many

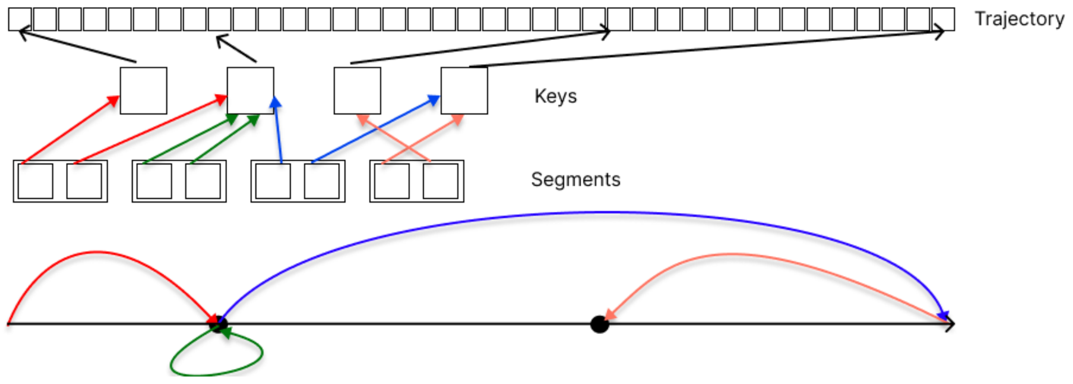


Figure 5.2: Mid-level view of the implementation of segmentation as it relates to arcs. The trajectory is an array of point, Keys is an array of indexes of the trajectory, and Segments is an array of tuples of keys.

EaseMarks as there are segments on the trajectory. Separating *EaseMarks* by segments allows the user to focus on one segment at a time and for each *EaseMark* to be anchored spatially.

5.1.1 Implementation

We implemented the tool in the Godot game engine as it offered the framework necessary for 2D manipulation as well as a streamlined UI builder. All sketch implementation is based on paths made from collection of 2D points passed through Philip Schneider's recursive curve-fitting algorithm [27] which transforms the path into its Bezier form.

The process of creating motions starts at a trajectory T represented as a sequence of n 2D points, with T_1 being the first point of the trajectory and T_n being the last. A segment S_i is a tuple composed of two keys T_j which can exist at any point on the trajectory - for example, a motion with only one segment spanning the whole trajectory would be defined as $\{T_1, T_n\}$. A segment can also start and end at the same point $\{T_j, T_j\}$. The sequence of segments S represents the path that the *object* will take during the motion.

A user can add a behavior B_i to the set of behavior B. Each behavior correspond to an EaseSheet. For each behavior B_i, there will be as many EaseMark as there are segments. Overall, the set of EaseMarks can be represented as a matrix E of size $n \times m$, where n is the size of B and m is the size of S. As such, $E_{i,j}$ represents the EaseMark of behavior B_i at segment S_j.

Each Behavior represents a type of affine transformation applied to the object, with its own predefined interpretation of an EaseMark. For example, the Rotation Behavior has a maximum of 100 and a minimum of -100, 100 meaning a rotation of 360° and -100 meaning a rotation of $-360°$.

By default the top most Behavior is the Speed Behavior. For each segment, this EaseMark describes how fast the object will go. This will be the first EaseMark to be called every frame as it determines which segment the animation is currently playing. Once the interval is determined, every EaseMark will be called sequentially in order of their initialization and each EaseMark will modify the object according to the behavior associated.

Since each EaseMark represents an affine transform, the order matters. Applying the rotation behavior before the squash and stretch behavior will result in different animations. This is something that users will need to keep in mind while using the tool.

5.2 Walkthrough

I showcase the capabilities of the tool by going through a short example. The goal is to demonstrate the ability to create animations that are already possible in a

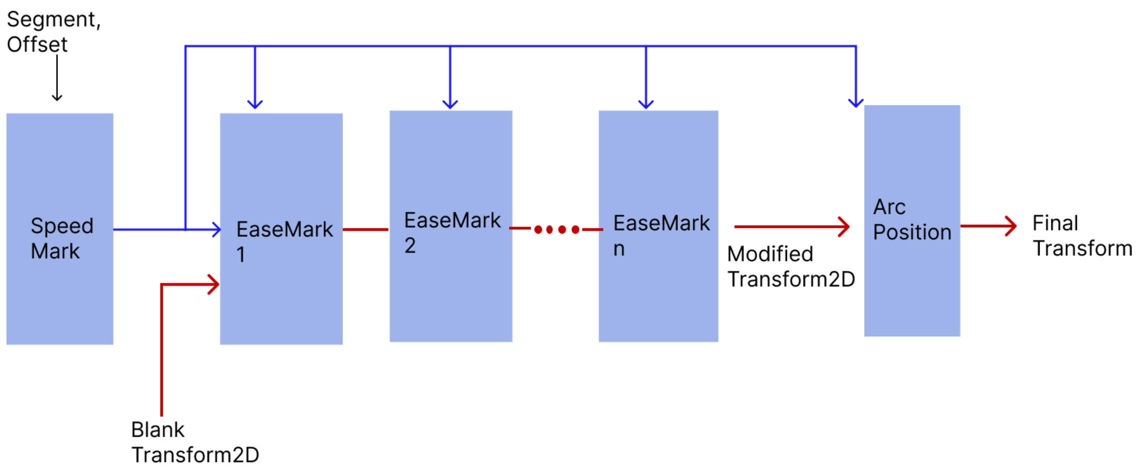


Figure 5.3: Mid-level representation of the flow of EaseMarks, an object's transform is generated by going through each one by one.

way that is faster and more intuitive. In this walkthrough, I present a user wanting to animate a falling and bouncing tire.

The first step to creating an animation is to draw the trajectory on the Cel Sheet. The user can draw as many lines as they wish, and each will be interpreted as a segment going from the beginning of the line to the end of it. Several lines might be needed in the case of more complex motion where the direction and trajectory changes midway. Once the line is drawn, the Onion Skinning of the object will appear and follow the trajectory. Onion Skinning allows for a quick way to see how the object will behave through time without having to play the animation every time. An added benefit is that every EaseMark change in the future will affect the Onion Skinning in real time, making the process even faster. In our case, the user will draw a simple line going from top to bottom.

In the Timeline, the user will be presented with an arrow representing the trajectory in space, similar to a timing chart. The tire will first come down, stop when it reaches the ground, then bounce back up. This is represented by 3 segments:

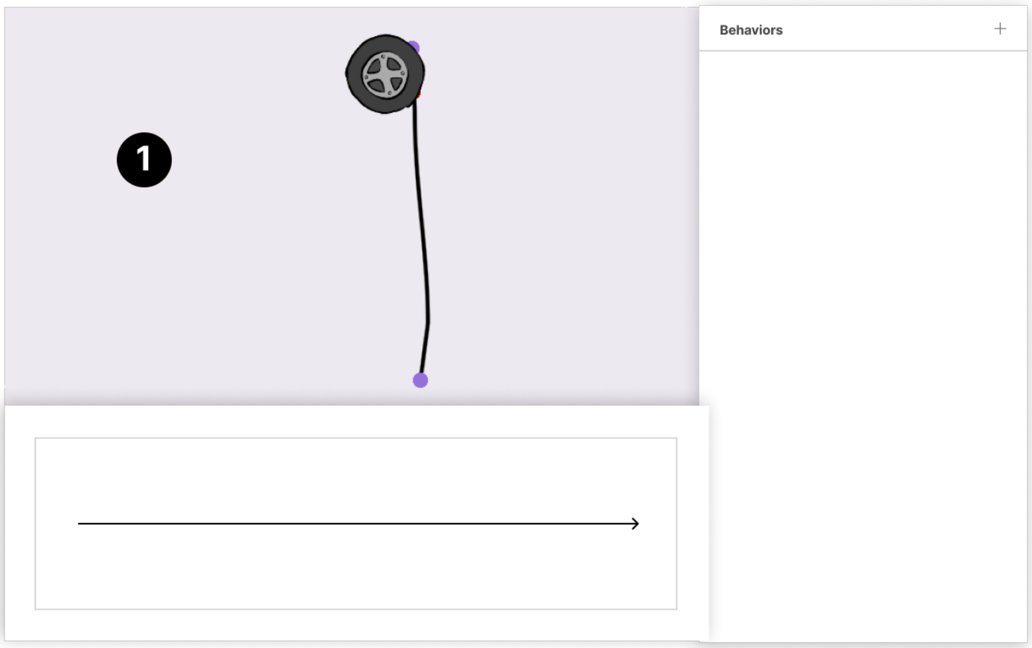


Figure 5.4: A trajectory is drawn on the main canvas.

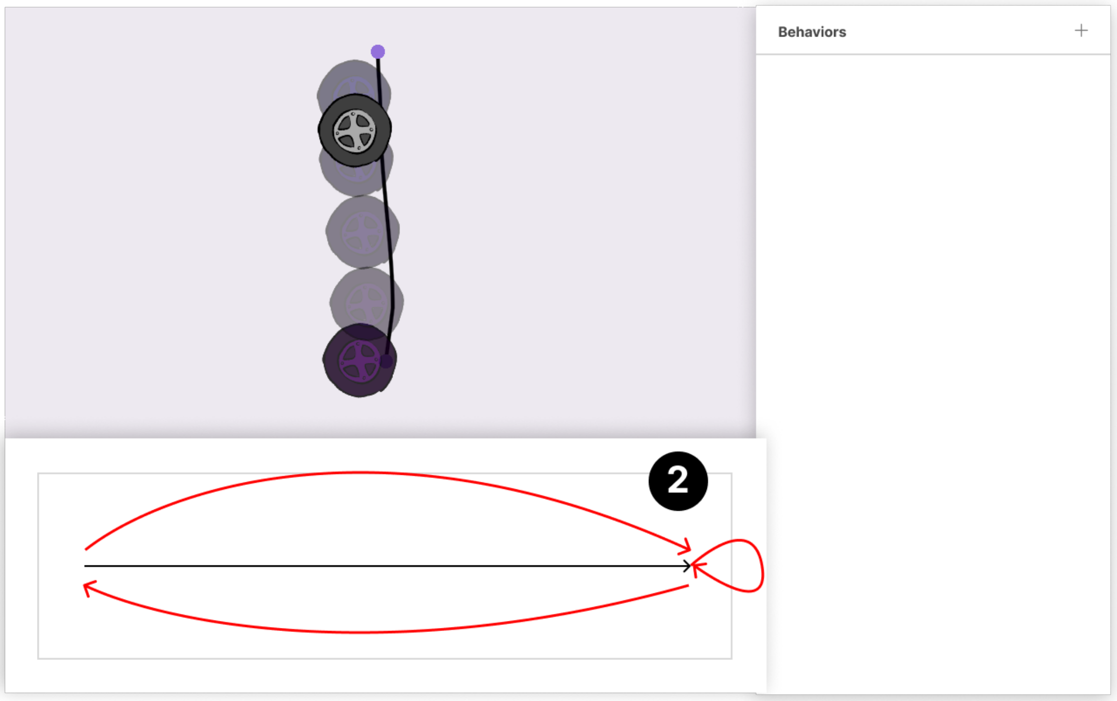


Figure 5.5: The timeline is configured and separated into three segments

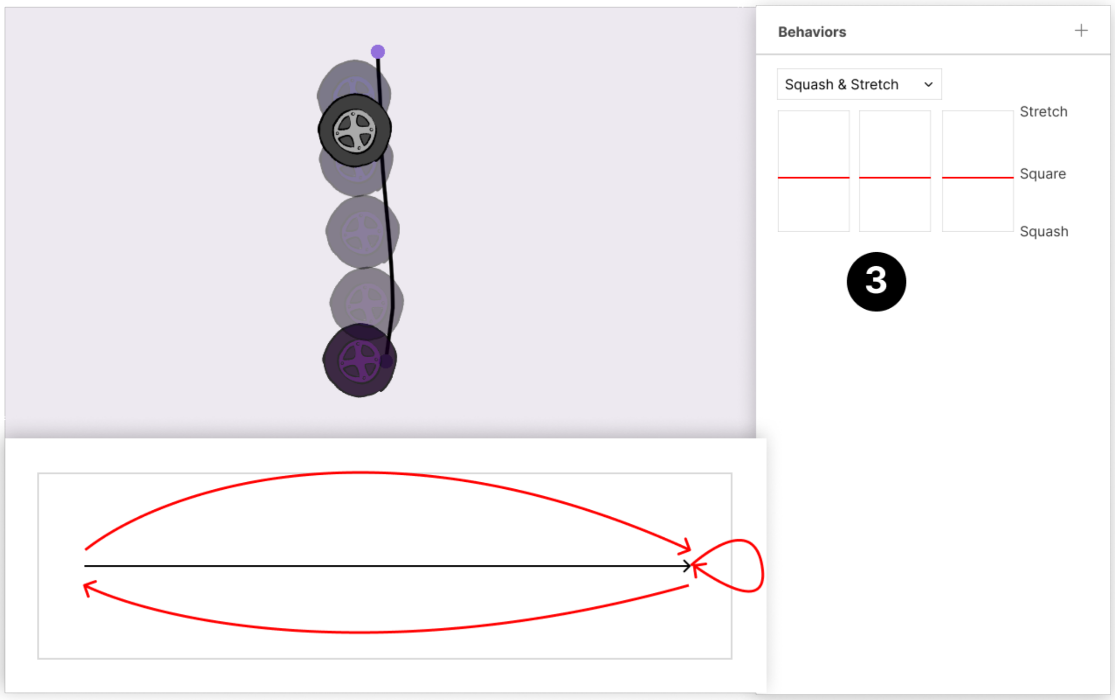


Figure 5.6: A Squash and Stretch behavior is added

$\{T_1, T_n\}, \{T_n, T_n\}, \{T_n, T_1\}$. When the user plays the animation, the tire will go down at a constant velocity, stop, then go back up at the same velocity.

The user can then choose a Behavior from a list of available behaviors. Each new behavior will open a new EaseSheet on the Behavior Panel. In our case, the user chooses the Squash and Stretch behavior, which represents a non-uniform scaling either horizontal (squash) or vertical (stretch). Automatically, three panels appear each representing a segment.

The user can tune the animation by drawing on one of the EaseMark or by cycling through the default EaseMarks provided to them. As the EaseMarks get modified, the onion skin reflects the changes. The user can also play the animation and tweak accordingly.

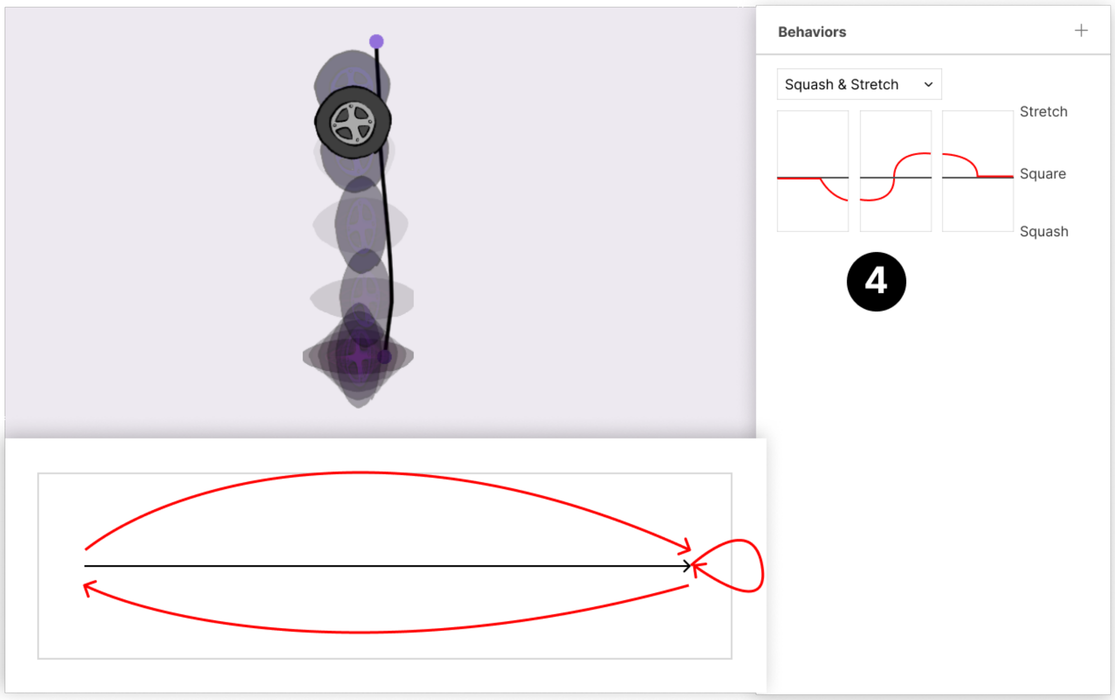


Figure 5.7: The Squash and Stretch EaseMarks are modified and tweaked through sketch.

The ability to see every part of the UI at once means that the user can get an overall view of how the motion works and how parts fit together. It is possible to have a screenshot of the UI as reference to recreate the same motion because every part is uncovered.

CHAPTER 6

EVALUATION

Easemarks and its accompanying tool act as a proof of concept for the use of spatial cues and sketching for authoring. I evaluate the tool by its expressiveness both in animation but also in how it presents motion statically. Following Ledo et al.'s strategies for evaluation of toolkits, I use evaluation by demonstration to showcase the abilities and limitations of my tool [28]. Each of the exemplars are connected to one of the motion types described earlier in Chapter 4, in order to show both the ability of the tool to replicate existing motions, but also how having knowledge of these motion types increases the liveliness of these animations. In the descriptions, EaseMarks are denoted in *italics*.

6.1 Anticipation

Anticipation is the preparation and the build-up of momentum that an object goes through before a motion. Anticipation is crucial in understanding a motion as it sets the audience's expectations as to what is coming next. It can be a good way to distinguish between a motion that is ongoing or one that is now starting. The trajectory of anticipation usually follows the opposite trajectory of the arc, announcing where the point of interest will go next. In traditional animation, the anticipation frames and the main action frames are independent, which can be difficult when wanting to create one cohesive motion. The animator can draw the motion first and mirror the direction of change of the beginning frames, or draw the anticipation before drawing anything else [4].

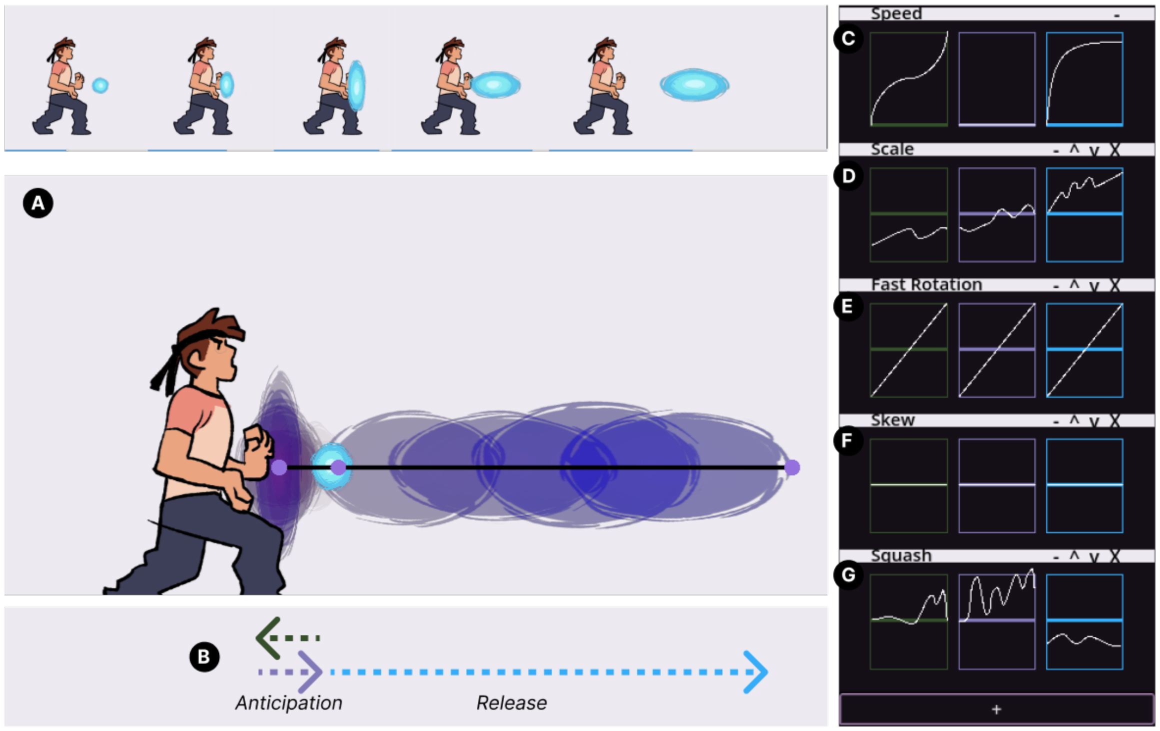


Figure 6.1: A) An animation of a orb being charged and released; B) the trajectory is split into three; the first frame acts as the anticipation phase; C)*Speed* winds up during anticipation and accelerates when released; D) *Scale* grows in pulses, reaching its largest size at the end, E) the orb *Rotates* three times across each segment of the trajectory; F) *Skew is unaffected*, G) the *Squash* key controls vertical elongation in the back-and-forth of the anticipation phase, then flattens during the release stage.

To show anticipation with EaseMarks, I present a character preparing a charged attack in the form of an "orb" of energy. The character is one that can be found in a fighting game and the orb is reminiscent of the special attacks that characters like Ryu or Ken would perform in Street Fighter [29]. Figure 6.1 represents the frame of the motion. The orb squashes and stretches while slowly growing in the same position. It is then pro-pulsed forward at a fast pace once it reaches its maximum size. Figure 6.1shows how the motion is presented in the tool. The behaviors used are *Squash and Stretch*, *Scale*, and *Rotation*.

An important element of the behaviors used in this animation in particular is the use of both *Squash and Stretch* and *Scale*. The latter describes uniform scaling accross both x and y axis while the former represents inverse scaling: as the x goes up, y goes down. Because we focus on behaviors rather than parameter, we can have an object that has a squash and stretch behavior while continuously growing. In a timeline, this combination would be represented in a singular line representing the scale factor and would not be distinguishable. We also show here that the ability to sketch the behavior is utilized to create a fluid almost random motion for the orb as it gains speed. The increase in size is neither linear, nor its rate always positive, giving it the impression that it is pulsing.

My method here is shown as more convenient with parameters that go back and forth, like the scale in this case. Modern Interpolation would require different keys at different extremes of the scale. On top of that, I am able to have different behaviors that share the same parameters because each one is independent, making it possible to focus on one at a time.

6.2 Bouncing Follow Through

Follow-Thru represent the consequence of a motion, where the momentum goes once the object has reached its destination. A sudden change of state between the motion and the follow-thru could mean a huge impact, whereas a smooth one could mean a soft landing. In traditional animation, follow-thru is drawn in a similar manner as anticipation but instead focuses on the end of the arc of motion. It is also often synonymous with collision, as such, the direction of the arc might drastically change, sometimes reflected by the plane of collision [4].

I show follow-thru by presenting a frog landing from a long jump. The animation starts in mid-air, with half of it taking space on the ground. As it lands, it bounces

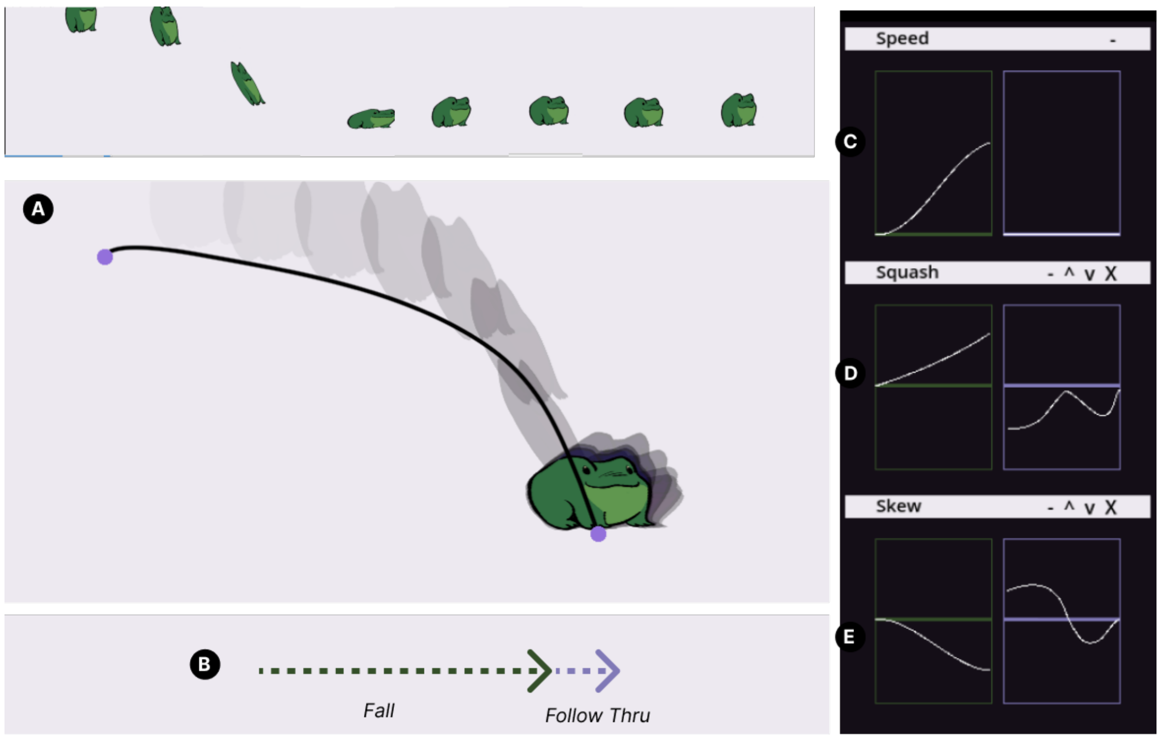


Figure 6.2: A) An animation of a frong landing down and bouncing in place; B) the trajectory is split into two; the last segment acting as the follow-thru phase; C) *Speed* accelerates as it is going down; D) the frog *Stretches* as it reaches down, before *Squashing* back and forth as it hits the ground E) the frog *Skews* forward and backwards with the momentum of the motion

up and down as well as wiggling back and forth, absorbing the momentum from the fall as seen in 6.2. We see in the next figure 6.2 that the behaviors used are *Squash and Stretch* as well as *Skew*. *Squash and Stretch* are often linked to collisions as they represent common deformations with sudden changes of speed. Having an overall view of the animation allowed us to quickly determine the direction of skewing. Since the frog jumps forward, it first skews forward.

6.3 Limited Animation

In traditional animation, creating a scene where a tire bounces off a wall involves a meticulous, frame-by-frame drawing process. Animators would first sketch the

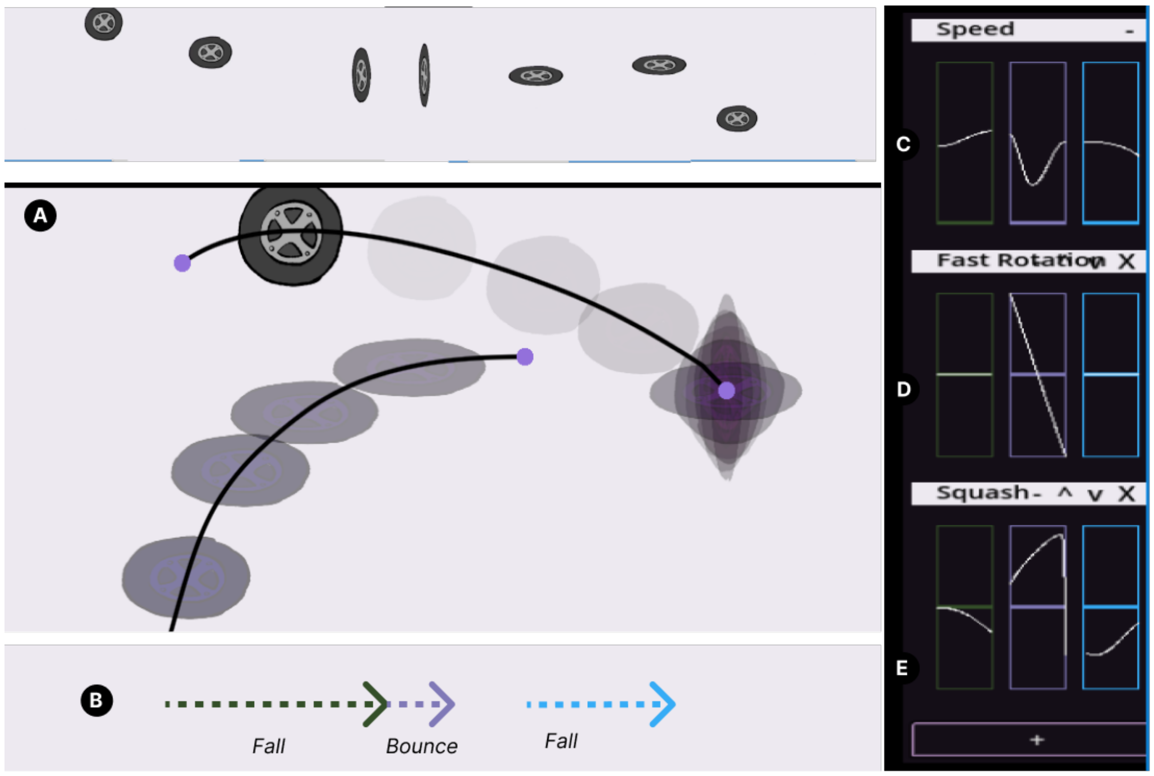


Figure 6.3: A) An animation of a tire bouncing off a wall; B) the trajectory is split into three; the last segment being its own trajectory; C)*Speed* is lower during the bouncing segment; D) the tire *Rotates* very fast during the bouncing segment E) the frog *Squashes and Stretches* as it reaches the wall.

key positions of the tire – its initial contact with the wall, the point of maximum compression upon impact, and the subsequent stages of its rebound. They would then manually draw the in-between frames, adjusting the tire's shape, rotation, and position to convey the dynamics of the bounce. This process is both time-consuming and requires a keen eye for detail to ensure the motion appears fluid and realistic.

With EaseMarks, the process of animating the bouncing tire becomes more streamlined and efficient. In our specific example of the tire bouncing off the wall, the animator sets EaseMarks for the crucial moments of the animation – the initial contact, the compression, and the rebound. To emphasize the tire's speed and impact against surfaces, it momentarily pauses at each key point before swiftly moving to the next. When the tire is in contact with the wall, it rapidly rotates and elongates vertically, as depicted in Figure 6.3. This effect was created by only modifying the middle segment. This example highlights the efficiency of our tool in fine-tuning such motions. Achieving the correct effect is challenging; it's crucial to ensure a seamless transition between the two arcs of the tire's movement – before and after hitting the wall – so the audience perceives it as a continuous motion.

In the user interface shown in Figure 6.3, limited animation is also applied to the EaseMarks. As the tire leaps from one position to another, its *Squash and Stretch* properties change abruptly, bypassing intermediate states. This technique enhances the clarity of the movement. The timeline's role is pivotal here, as it precisely tracks the tire's unconventional movement. Despite the tire following two trajectories, the animation is divided into three segments. This is because the tire remains stationary at the end of the first trajectory, adding a distinct pause in its motion. Skipping a segment allows for a bigger impact as the change will always be greater than smoothly transitioning between positions.

6.4 Easing

In traditional animation, the timing of motions is determined by the number of frames it is made of. An issue with this process if with tuning being a destructive process, as frames need to be mostly redrawn. Computer interpolation makes the use of timelines: if a motion is fast, its keys are spaced close together whereas a slow motion will have keys further apart. This requires thinking about time rather than distance.

In previous examples, *Speed* was employed for easing, but our current illustration demonstrates how easing can also be applied to modify EaseMarks. I depict a toy car advancing, colliding with an unseen barrier, reversing, and then resuming

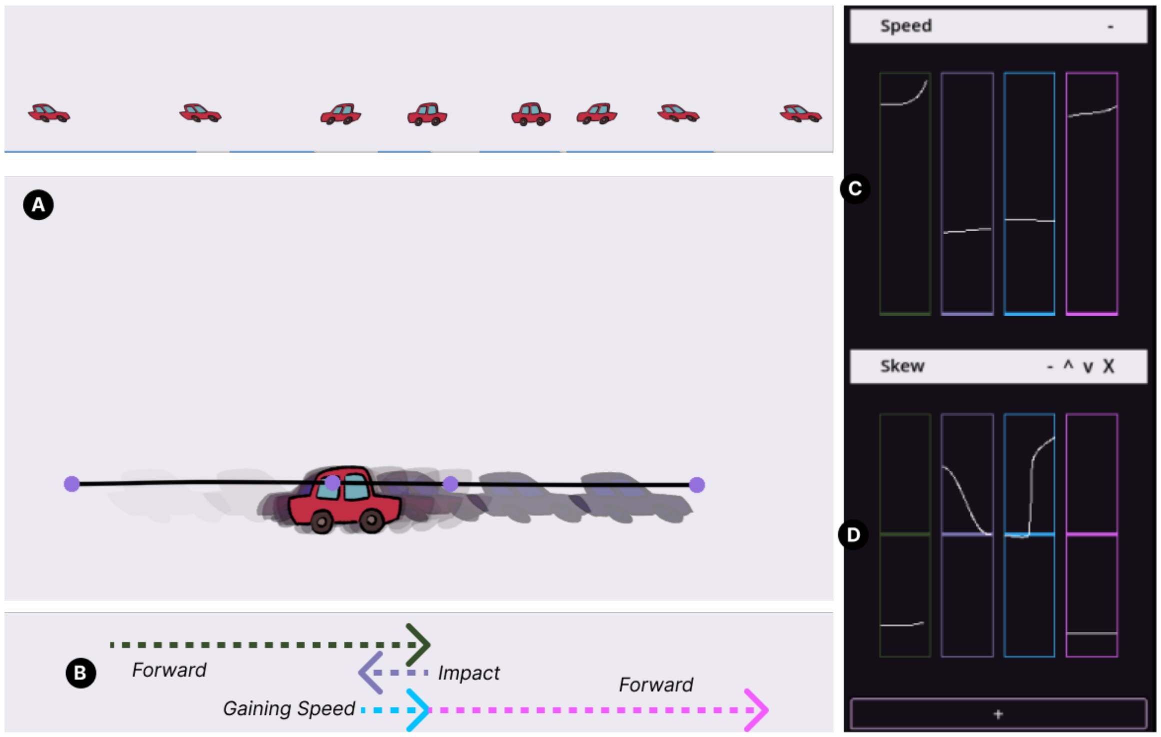


Figure 6.4: A) an animation of a toy car driving, hitting a wall, and driving again. B) the trajectory is divided into 4 segments. C) There is a strong difference in *Speed* between the middle segments and the outer ones as the car suddenly stops and takes time to gain up speed. D) represents the *Skewing* of the car that exaggerates the speed.

its movement. In Figure 6.4, the car's direction is conveyed through skewing. Upon halting, the car initially skews forward, then reverses its skew before moving again. This action is captured in the third segment of Figure 6.4. When considering *Speed*, there is an increase as the segment progresses from start to end, implying that the *Skew* begins slowly and then accelerates. Notably, even when the car is stationary, the *Speed* parameter continues to influence the motion dynamics.

6.5 Oscillatory

Oscillatory motion is usually one of the most straightforward motions to create in both traditional animation and computer interpolation as copying previously drawn

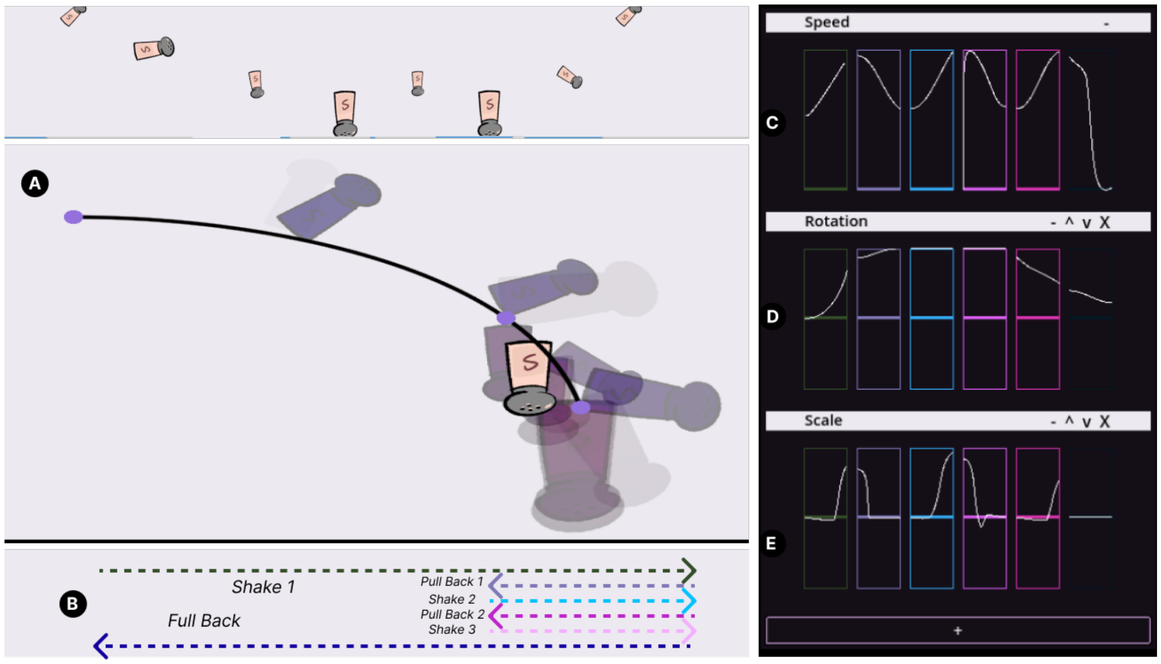


Figure 6.5: A) An animation of a salt shaker being shaken; B) the trajectory is split into six segments; going back and forth over the same points; C) *Speed* gets increased as the salt shaker comes down; D) the shaker *Rotates* to follow the direction of the motion. E) the shaker *Scales* up uniformly to show impact whenever it reaches down.

frames or keyed parameters can be as simple as copying and pasting. Traditional animation allows for tracing [4], which is using the transparency of celulloid sheets to draw on top of previous drawings.

However, one of the complexities in depicting oscillatory motions is the necessity to intersperse the cyclic pattern with alternating movements. This concept is exemplified by the action of shaking a salt shaker, which in this case is performed thrice. The sequence of these movements is essential to comprehend the dynamics of the motion. Figure 6.5 illustrates the various stages of the animation, where subtle variances are observed each time the shaker moves in opposing directions. Notably, as the shaker approaches the extremity of its swing, it increases in size, emphasizing the impact of the movement. In Figure 6.5, we observe that the motion's arc is divided into only three key points, yet the animation comprises six segments in total. While these key points are repeated, the organic quality of the motion arises from adjusting the EaseMarks. Each oscillation employs distinct combinations of *Scale* and *Rotation*, contributing to the animation's naturalistic appearance. Finally, a limitation of the tool is the clutter that appears on the timeline. Since the segments overlap, there is a lot of information density on a single area.

CHAPTER 7

DISCUSSION

7.0.1 Crowdsourcing Sketch Mappings

I used crowdsourcing in order to elicitate a valid mapping between secondary sketch marks and motions. During this process, there was a need for an iterative approach as results were inconclusive, mostly due to the freedom of interpretation that comes with sketching. It is difficult to know why a mark was chosen over another and different participants might have different reasons to choose the same one. There is a possibility of using browser-based tools in order to let participants draw their own sketches. This would allow for a true elicitation process. However, such a task would require even more structure as results would need to be analyzed. The UI of the design tool presented in this paper does offer some of that structure, and could allow for small task that focus on elicitating one behavior through one segment only for example.

7.0.2 Sketching Languages

The EaseMark Sketching Language consists in great part of using visual cues to structure animation behavior. Having a direct connection to the way the behavior is spread out helps reducing the load needed to keep track of the different parameters of an animation. It also helps bring seemingly independent behaviors together, as they are all anchored at the same points. The sketching language itself is also fluid and can be used with behaviors outside of the ones presented in this thesis. Since the behaviors can be more complex than simple parameters, it is possible to programatically define behaviors that have specific rules or put together a range of parameters. The limit with the current architecture is that every behavior will have the same language, they are all one dimensional and going up and down. This greatly limits what is possible with sketch like annotations, hashes, etc. and is an avenue to explore.

7.0.3 EaseMarks in Practice

In practice, the current tool cannot yet be integrated in modern workflows as it requires a different architecture of the animation process. Its simplicity, however, could fit well with a design stage for animations. Because it is easy to quickly visualize a range of motion, it can be used like an animation sketchbook, a place to come up with short animations on the fly to get a better understanding of how a character should feel. As of now, it can also be integrated into games that rely on a small number of assets, as it can generate a breadth of animations from a simple image. This ability to create a diverse array of animations very quickly is an asset when it comes to a medium like video games where there are many deadlines and where animations need to be expressive but produced in a fast manner.

7.0.4 Failure Modes of EaseMarks

The first limit of EaseMark is in the choice of representing every EaseMark as horizontal curves going from left to right. Motions are not always left to right and there is a disconnect when the trajectory represented is very different than how it is presented on the timeline. This also limits how behaviors like rotation are represented: modifying rotation by sketching up and down goes against intuition. A more general approach requires an array of sketch interpretation and perhaps a dynamic way to display Easemarks depending on the motion provided. The idea of simplifying the communication of a motion seems to break down when implementing complex animation structures involving rigging and skeleton. Bone transforms get applied hierarchically: for example, let's say that a bone in the bottom of the hierarchy uses EaseMarks for its motion; if a bone higher in the hierarchy moves during the motion, the results are unpredictable, and the bone will not follow the trajectory set. This issue could be resolved by implementing a system similar to the Ephemeral Rigging [17] tool, where an animation could only focus on one bone and rearrange the hierarchy so that the bone that is affected by the EaseMark becomes the root of the skeleton for that specific motion.

Another limitation to the current approach is the fact that both the animation and the sketching aspect are 2-dimensional. This is both a side-effect and a feature of the medium. Screens are 2-Dimensional and this view of animation focuses on the end result, what will appear on screen. In recent stylized animation, 3D is used as an intermediary step between the concept art and the animation, but it is also an obstacle to overcome. A possible direction of our tool and process could be a reformulating of the animation pipeline, the input being 2D and the output being 2D as well. In this case, 3D would be used "under-the-hood," a way to connect a 3D physics engine to a 2D representation of a world. This "under-the-hood" approach to 3D can be seen in 2.5D Cartoon Models by Davis et al. [30]. Here, the spotlight is on "views," where each view is a 2D vector drawing, acting as keys from traditional animation. Each new view can then interpolate between the key views.

The origin of this project lies in the concept of animation blending. Given a set of animations set for specific context, we can come up with a new unique animation based on a new one. The ability to extract a hierarchy of motion spatially can allow us to create tools that are able to blend between very different animations by simply interpolating between different curves. This frees a lot of possibilities when it comes to current animation blending limits, which rely on discretized time-sensitive keys.

Arcs can be interpolated between one another, in the same way that EaseMarks can. The next step is to create a system that can modify the EaseMarks based on the context of a specific animation, for example, whether a character is punching up rather than down.

By using sketch to author complex animation, we were able to visualize relationships between different deformations and motion. Not only does it make it easier to animate, it also allows us to understand the important information that comes with these motions. This filtering of the information can be useful in data-driven generation of animation. If spatial information and deformations are given as training parameters, they could be used to tune animations generated by tools like the one presented by Qin et al. [16]. Similarly, this can be used to prune unnecessary frames from motion capture data or even exaggerate their expressions.

Our research in sketching language, both in our integration and our elicitation studies, can lead to a standardization of the language of motion. Such standard would allow for effective communication across platforms and authoring software. Most reuse of video game animation involves the mapping of skeletons with similar structures, but a standard language could allow for a higher-level understanding of these motion which would result on the possible re-appliance of the same motion type to different skeletons.

CHAPTER 8

CONCLUSION

The goal of this project was to restructure our the process of animating by looking at traditional animation as an inspiration. My focus was on translating visual input to visual output, which required me to understand more about giving instructions through sketch. The challenges that came with the project came from the need to create a intuitive language that would be clear and synthesize motion dynamics, but there is a balance to be had between a language that is as simple as possible and a language that allows for a diversity of cases.

Overall what the takeaway from this project is the idea of using a visual language to describe visual events. While my original objective was to come up with a way to design reusable animations, I learned a lot about the relationship between reusability and structure. For a visual language to work and be understood, I needed to structure it and give it context. The same goes for reusability, To be applied to different context, I first need to understand exactly how it is applied to the first one.

APPENDIX A

Code Repositories

The Godot application for EaseMark is available through Github: https://github.com/The-Hybrid-Atelier/EaseMark

Link to my GitHub: https://github.com/13hnguyen

Handout of the UI of the tool: https://drive.google.com/file/d/1RkWxOTpHNhDtO8CA7ThxJX
view?usp=sharing

REFERENCES

[1] A. M. Juan Linietsky, "Godot." [Online]. Available: https://www.godotengine.org/

[2] J. Ma, L.-Y. Wei, and R. H. Kazi, "A layered authoring tool for stylized 3d animations," in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, ser. CHI '22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/3491102.3501894

[3] J. Lasseter, "Principles of traditional animation applied to 3d computer animation," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '87. New York, NY, USA: Association for Computing Machinery, 1987, p. 35–44. [Online]. Available: https://doi.org/10.1145/37401.37407

[4] R. Williams, *The Animator's Survival Kit, Expanded Edition: A Manual of Methods, Principles and Formulas for Classical, Computer, Games, Stop Motion and Internet Animators*, expanded edition ed. Faber and Faber, 2009.

[5] J. Cooper, *Game Anim: Video Game Animation Explained*, 2nd ed. CRC Press, 2021.

[6] Arc System Works, "Guilty gear xrd," 2015.

[7] Insomniac Games, "Spider-man: Miles morales," 2020.

[8] Blender Foundation, "Blender." [Online]. Available: https://www.blender.org/

[9] J. Popović, S. Seitz, and M. Erdmann, "Motion sketching for control of rigid-body simulations," *ACM Trans. Graph.*, vol. 22, pp. 1034–1054, 2003.

[10] M.-W. Chao, C.-H. Lin, J. Assa, and T.-Y. Lee, “Human motion retrieval from hand-drawn sketch,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, pp. 729–740, 2012.

[11] M. Guay, R. Ronfard, M. Gleicher, and M.-P. Cani, “Space-time sketching of character animation,” *ACM Transactions on Graphics (TOG)*, vol. 34, pp. 1 – 10, 2015.

[12] Q. Li, W. dong Geng, T. Yu, X. Shen, N. Lau, and G. Yu, “Motionmaster: authoring and choreographing kung-fu motions by sketch drawings.” Eurographics Association, 2006.

[13] Y. Xie, W. Xu, Y. Yu, and Y. Weng, “New technique: Sketch-based rotation editing,” *Journal of Zhejiang University SCIENCE C*, vol. 12, pp. 867–872, 2011.

[14] E. R. Moghaddam, J. Sadeghi, and F. Samavati, “Sketch-based dance choreography,” in *2014 International Conference on Cyberworlds*, 2014, pp. 253–260.

[15] S. Min-yong, “Sketch pose: A reusable motion representation for 2d cartoon animation,” *Journal of Engineering Graphics*, 2010.

[16] J. Qin, Y. Zheng, and K. Zhou, “Motion in-betweening via two-stage transformers,” *ACM Trans. Graph.*, vol. 41, no. 6, nov 2022. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/3550454.3555454

[17] R. Anzovin, “Fast, interpolationless character animation through ”ephemeral” rigging,” in *ACM SIGGRAPH 2019 Talks*, ser. SIGGRAPH ’19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/3306307.3328165

[18] R. C. Davis, B. Colwell, and J. A. Landay, “K-sketch: A ’kinetic’ sketch pad for novice animators,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI ’08. New York, NY, USA:

Association for Computing Machinery, 2008, p. 413–422. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/1357054.1357122

[19] R. H. Kazi, F. Chevalier, T. Grossman, S. Zhao, and G. Fitzmaurice, "Draco: Bringing life to illustrations with kinetic textures," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 351–360. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/2556288.2556987

[20] M. Thorne, D. Burke, and M. van de Panne, "Motion doodles: An interface for sketching character motion," *ACM Trans. Graph.*, vol. 23, no. 3, p. 424–431, aug 2004. [Online]. Available: https://doi.org/10.1145/1015706.1015740

[21] S. Hashim, T. Höllerer, and J. Jacobs, "Drawing transforms: A unifying interaction primitive to procedurally manipulate graphics across style, space, and time," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, ser. CHI '23. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/3544548.3580642

[22] L. Petikam, K. Anjyo, and T. Rhee, "Shading rig: Dynamic art-directable stylised shading for 3d characters," *ACM Trans. Graph.*, vol. 40, no. 5, sep 2021. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/3461696

[23] P. Grochola, F. Maccari, Y. J. Lee, and E. Boulet-Gilly, "Linework in spider-man across the spider-verse: An artistic driven approach to linework generation," in *ACM SIGGRAPH 2023 Talks*, ser. SIGGRAPH '23. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/3587421.3595456

[24] Z. H. Ibrahim, "The stretch-engine: A method for adjusting the exaggeration of bipedal characters through squash and stretch," in *Proceedings of the ACM*

SIGGRAPH / Eurographics Symposium on Computer Animation, ser. SCA '17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi-org.ezproxy.uta.edu/10.1145/3099564.3106639

[25] "Amazon mechanical turk." [Online]. Available: https://www.mturk.com/

[26] M. Pichlmair and M. Johansen, "Designing game feel: A survey," *IEEE Transactions on Games*, vol. 14, no. 2, pp. 138–152, 2022.

[27] P. J. Schneider, *Graphics Gems*, 1995, pp. 612–626.

[28] D. Ledo, S. Houben, J. Vermeulen, N. Marquardt, L. Oehlberg, and S. Greenberg, "Evaluation strategies for hci toolkit research," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–17. [Online]. Available: https://doi.org/10.1145/3173574.3173610

[29] Capcom, "Street fighter," 1987.

[30] A. Rivers, T. Igarashi, and F. Durand, "2.5d cartoon models," *ACM Trans. Graph.*, vol. 29, no. 4, jul 2010. [Online]. Available: https://doi.org/10.1145/1778765.1778796

BIOGRAPHICAL STATEMENT

Hadrien Nguyen was born in France in 2001. They came to the United States of America for High School in 2016 and entered the University of Texas at Arlington in 2018. There, they earned a Bachelor of Science in Computer Science and a Bachelor of Arts in Philosophy in 2022. They started taking graduate classes during their undergraduate degree through the Fast Track problem and quickly started on their Masters of Science in Computer Science soon after.