SKETCH-BASED ANIMATION TOOL FOR CHARACTER ANIMATION INTEGRATING INTO A PRODUCTION PIPELINE

A Thesis

by

SER EN LOW

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Tim McLaughlin
Committee Members,	Ergun Akleman
	Tracy Hammond
Head of Department,	Tim McLaughlin

May 2014

Major Subject: Visualization

ABSTRACT

Creating appealing character poses can be time-consuming in an animation production pipeline incorporating skeletal-based character rigs. Animators utilize point-and-click input devices such as a mouse and keyboard to manipulate the character pose, rather than interacting intuitively as they would in a hand-drawn medium. This paper describes a sketch-based animation tool integrated into Autodesk Maya, enabling non-destructive and spatially accurate control over the animation of the line of action of arms, legs and spines. The tool provides a faster and more natural method for animators to pose and animate CG characters compared to mouse and keyboard input.

DEDICATION

To my beloved family, my wife Vivian, and baby girl Elise.

ACKNOWLEDGEMENTS

I would like to thank to my excellent thesis chair, Tim McLaughlin, for his guidance and sharing of his expertise in Character Technical Direction and encouragement of me over the years we have known each other. I have been fortunate to have Ergun Akleman and Tracy Hammond as my committee members; I learned tremendously from them about the ray-traced method in solving 3D depth as well as the logic behind sketch recognition and human-computer interaction.

Special thanks to all the Visualization department faculty, staff and former students for making my graduate studies an enduring memory of Texas A&M University.

I would like to extend my gratitude to all my friends in the Vizlab for their help and tolerance. Thank you, Christine Li and Anahita Salimi, for working with me on the character model used for this thesis. Thank you Austin Hines, Chiang Leng, Cong Wang, Garrett Broussard, John Pettingill, Junze Zhou, Logan Kelly, Siran Liu,Wei Wang, and Will Kung, for your wonderful support and sharing of knowledge.

Most importantly, thank you so much to my parents for allowing me to pursue my studies and my wife, for her forever love and support of me. Last but not least, to my newborn baby girl, Elise; thank you for accompanying me with your cries when daddy is working on his thesis in the middle of the night.

NOMENCLATURE

CG	Computer graphics
3D	Three-dimensional
2D	Two-dimensional
GUI	Graphical user interface
API	Application programing interface

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Generating the poses for an expressive character animation is a challenging task in computer graphics (CG). Even with the sophisticated animation applications available today, without the understanding of fundamental principles of hand drawn character animation, animators are not able to bring an animated character to life that is convincing to the public audience [1]. Line-of-action (see Figure 1) is an important principle in a hand-drawn medium, enabling artists to achieve a lively figure drawing. It elaborates the natural flow within a character's body movement and avoids robotic posture.

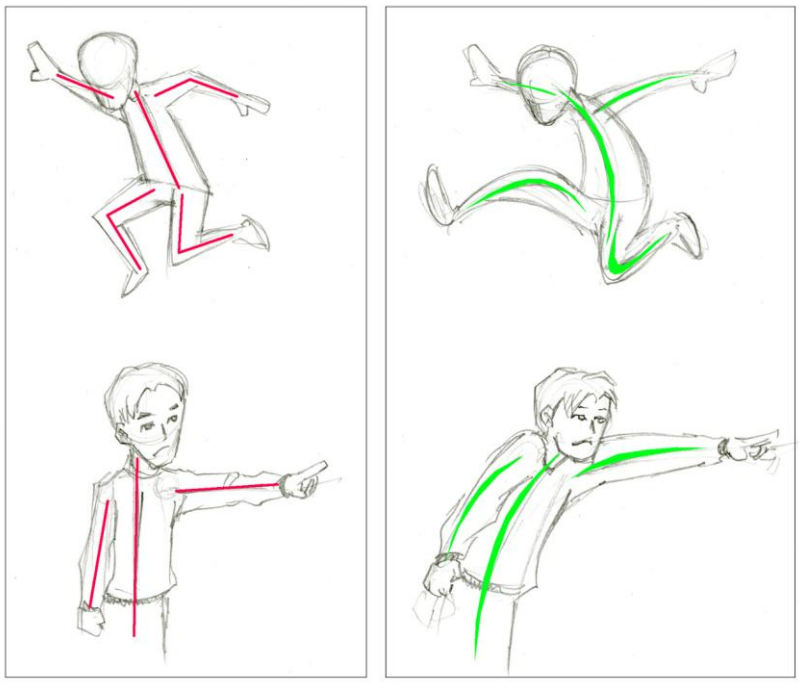

Figure 1. *(Left) Stiff line-of-action. (Right) Lively, fluid line-of-action.*

Therefore a hand-drawn medium has an advantage relative to CG in providing animators with natural input as they would in using pencil and paper. Current computer animation tools do not encourage animators' full artistic potential in delivering high quality character animation. Computer animators' natural hand-eye coordination is restricted by the point-and-click input devices, such as mouse and keyboard to animate a CG character. On top of that, current CG character's rigging system using skeletal joints to pose a character requires animators to attend to many parameters and settings in the working interface. A tool that is able to provide intuitive input and flexibility in handling

complex character rigs will be more effective in helping animators bringing CG characters to life.

Prior works that are related to sketch-based applications attempted to present various solutions in providing intuitive methods for animators to interact. However they have limited usage when these applications are presented in their own customized working environment rather than standard animation production packages that are commonly known and used by most animators.

The tool developed as part of this thesis is integrated into Autodesk Maya, a CG industry standard tool, and assists animators in producing satisfying character posing efficiently by implementing the line-of-action concept through a sketch-based interface. Animators sketch a curve on a monitor screen, similar to how they draw in a hand-drawn medium, and the tool will generate a curve in perspective viewport without depth ambiguities. Then, this curve generates a chain of joints that influence the existing character rigs into a new character pose. With this tool, animators also have the ability to interpolate between any existing pose and the new sketch pose in a non-destructive manner. A graphical user interface (GUI) provides the animator with access to the features and controls of this tool.

The second chapter reviews the background of this thesis and related works to provide fundamental understanding for all readers. The third chapter describes the approach of this thesis in providing a solution, conceptually and technically. The fourth chapter explains about the result with visual examples and includes discuss of the current tool's limitations. Lastly, the fifth chapter concludes this thesis with future works.

CHAPTER II

BACKGROUND AND RELATED WORK

Line-of-action is an imaginary line extending through main body's movement of the character (see Figure2) [2]. It is essentially a C-shaped curve that represents an aesthetically pleasing flow of body, primarily on the torso and appendages such as arms, legs, head and tail. It is also the first component drawn when constructing a figure in a hand-drawn medium. Manipulating line-of-action influences the expressiveness character of motion. If we take the same directional force and apply more force to it, the more curved it becomes [3]. Glen Keane, supervising animator and executive producer of *Tangled*, Walt Disney Pictures' 50th animated movie, supervised the character poses in the movie by drawing and pushing the curves of the CG figures in enhancing their expressions [4]. Therefore, establishing a line-of-action in a character helps represent the character poses clearly and lively.

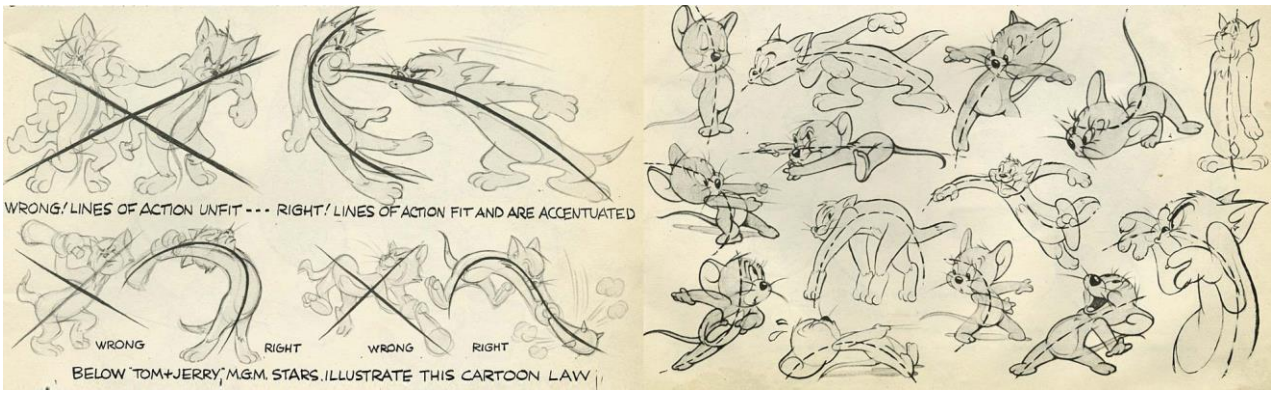

Figure 2. *Right and wrong in line-of-action.* [2]

Skeleton-based posing remains the most common posing and animation method in computer animation. There are three components: (a) the motion systems, which is a collection of joints, (b) the control system that provide an interface for animation input; (c) deformation system, which is responsible for manipulating the shape of the character model based upon the articulation of the associated motion and control systems [5]. There are numerous controllers on the character's motion system for each body part (see Figure 3) that required animators' attention to manipulate a CG character into the desired pose.

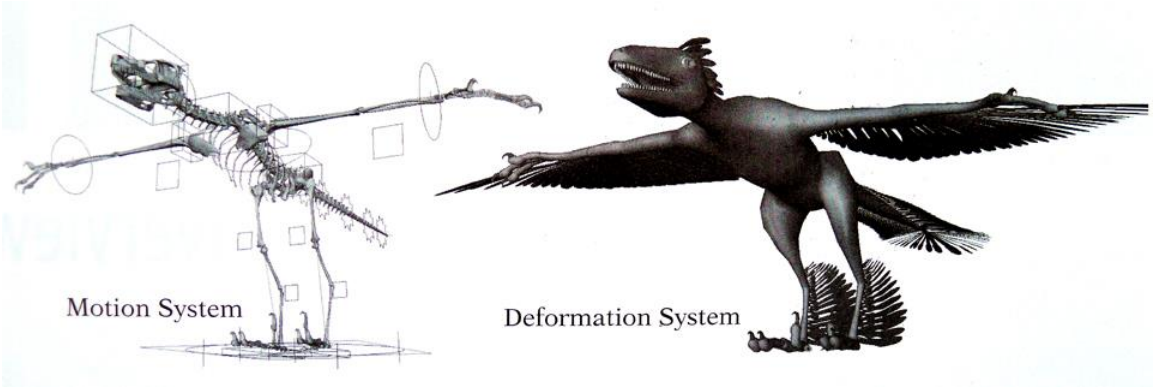

Figure 3. *Screen capture of a character motion and deformation system. Character by George Smaragdis and Rob O'Neill* [5].

However, the skeletal posing method is not an ideal solution for animators to achieve an expressive CG character's pose. "On almost every shot, 20% of the effort went into tweaking the rig to more of a hand-drawn pose", said *Hotel Transylvania*'s director, Genndy Tartakovsky [6]. He further explained that he wanted the character animation feel more organic, as if he were drawing, however this was very challenging for animators and people who rigged these CG characters.

In the field of computer graphics, prior research has been conducted to improve and simplify the character posing and animation process through more natural methods of managing the control systems. Sketching methods have received a great deal of research attention because artists have traditionally been trained in pencil-and-paper form. Sketching is able to manifest artists' thinking, enabling quick cycles of generation and (re)interpretation of design [7].

In an effort to develop a more natural input method for CG animation, Davis et al. introduced a sketch-based tool that allowed animators to sketch several rough poses of a stick figure on top of the existing animation frames [8]. Based on these rough poses, the tool automatically extracts the location of joints in 2D. These joints are used to construct new 3D poses which serve as a good starting point for the animator to refine the final animation. The animator then selects a character pose from a series of poses generated by the tool and proceeds with the final animation. However, the tool requires the animator to annotate initial sketches with stick-figures to minimize ambiguities. Furthermore, the new 3D character poses that is synthesized by the tool limits the animator's creativity rather than nurtures them.

A sketch based animation synthesis tool proposed by Lin allows the animator to sketch a 2D representation of a 3D skeleton [9]. The animator starts by manually editing a skeleton's template to fit over a desired sketch. This 2D sketch is then mapped onto the 3D skeleton from a motion tracking database. The tool then searches for the closest poses in the database and synthesizes a new animation. The tool requires significant computation power in the workflow; it is also limited to characters with rigid body meshes. Characters with deformation, which are commonly used in animation production, are not supported by this tool.

Matthews et al. proposed another approach that allows the animator to animate a CG character by drawing stick figures to present the character's skeleton poses [10]. The animator then sketches a pose for each key frame instead of animating through the interface, as in common 3D animation applications. However, the animator must

indicate the different sets of the skeletons, such as spine, arms or legs, with predetermined associate colors. During the tool demonstration, only single mesh and low resolution mesh characters were provided as examples. Therefore the capacity of this tool for manipulating characters with complex articulation and geometry resolution that is standard for animation industry is unclear.

CHAPTER III

APPROACH

This chapter describes the components and features of this sketch-based animation tool:

III.A. Intuitive interaction

The basic sketch-based workflow is as follows: the animator begins by selecting a ‘picker’ to represent a body part that needs alteration - for example an arm, leg or spine (see Figure 4).

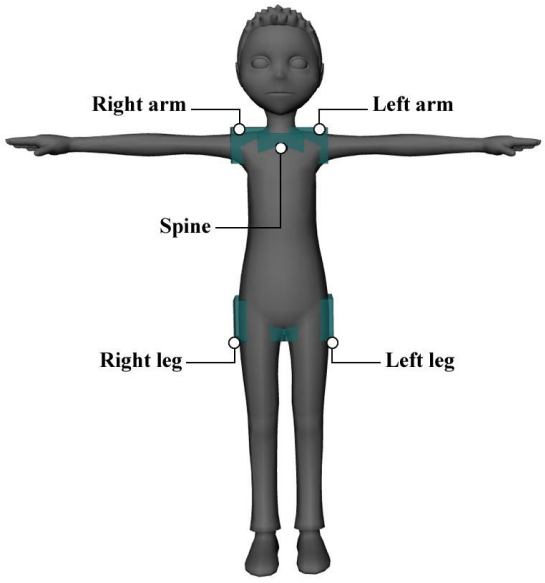

Figure 4. *Body part’s ‘picker’ for this sketch-based tool. Character modeled by John Doublestein. Available at http://www.creativecrash.com/maya/downloads/character-rigs/c/the-andy-rig.*

The animator draws a curve with the beginning point over the selected 'picker'. A new chain of joints is generated along the curves reflecting the curve's curvature. The number of joints generated in the new chain will match the existing chain of joints. Each of these new joints will then layer their transformation data onto the existing joints respectively (see Figure 5).

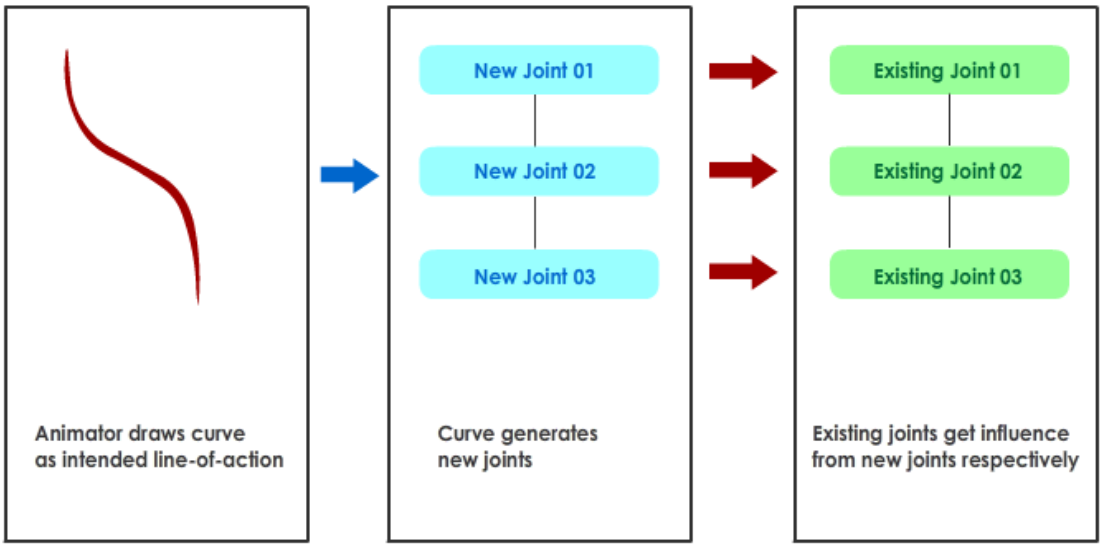

Figure 5. *Basic workflow of the sketch-based tool*

In order to implement the line-of-action into this tool, a parametric curve is constructed after the animator has sketched on a monitor screen. A parametric curve is used because it is easy to build with specific vertices determined by the tool while maintaining curvature. However, current curve construction tools in Autodesk Maya are

not sufficient for the purpose of this tool. Maya constructs curves with depth errors in perspective and camera viewport (see Figure 6).

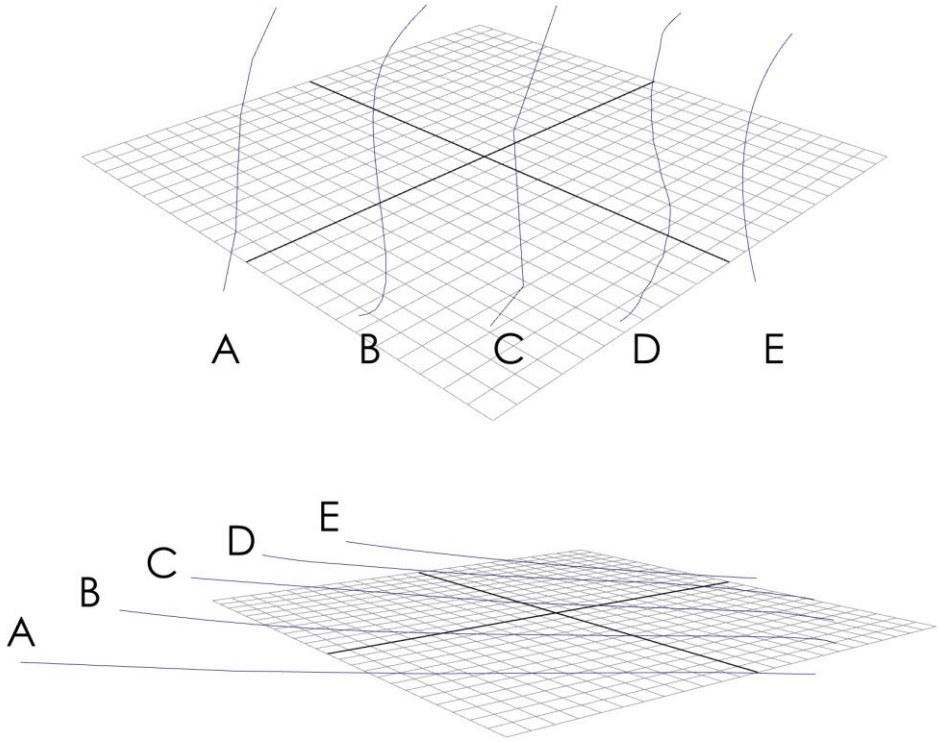

Figure 6. *(Top) Curves generated using Maya curve tools in perspective: (A) CV curve tool (B) EP curve tool (C) Bezier Curve Tool (D) Pencil Curve Tool (E) Arc Tool, directly in perspective viewport (Bottom) All curves (A to E) are generated with depth error, viewed from another camera angle.*

Therefore, a new curve construction plug-in has been developed using Maya API C++. C++ is used as programming language in accessing Maya API for its speed in calculation and because the relevant documentation is commonly available [11]. This plug-in identifies the active camera viewport from user interaction once an animator selects the 'picker' and captures the active camera's normal vector, *n*. When the animator starts to sketch directly on the viewport, the tool checks for the collision point of the first mouse click with the selected 'picker' and return this as *c*. With *c* and *n*, a temporary plane orthogonal to the camera can be constructed (see Figure 7). Any point on this temporary plane, *a,* can be calculated by using this formula [12]:

$$(c - a) \bullet n = 0$$

This plane serves as additional depth information for all vertices on the curve. With this information, the depth ambiguities of the curve in the camera view can be minimized.

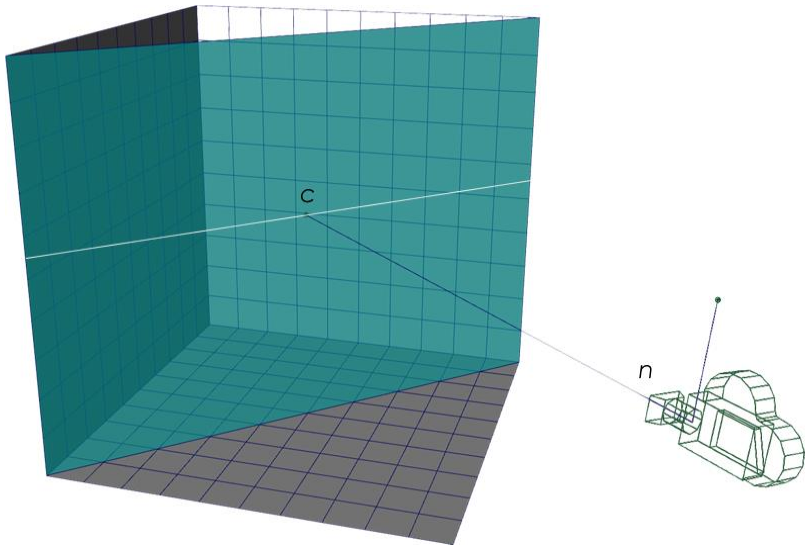

Figure 7. *Plane is orthogonal to active camera for additional depth information.*

The position of the 'picker' can be manipulated if an animator wishes to reposition the collision point to serve as the first point in the sketch curve. These 'pickers' are set to always reflect the position of each represented body part. This function provides another level of flexibility for animators in posing the CG characters.

III.B. Altering character kinematic system

In order to manipulate the character kinematic system in Maya, utility scripts with specific functionalities have been coded in Python. The modularity of Python's object-oriented structure has the advantage of allowing these functionalities to be implemented across a variety of CG characters with minimum script modification.

A Python script is used to capture the sketched curve's transformation data in 3D space. This curve is then optimized through uniform parameterization to match the number of curve's knots to the number of joints needed to maintain its curvature. The position of each vertex in the curve is calculated to generate a chain of 'sketch' joints. The 'sketch' joints transfer their transformation data to the existing joints of a CG character. Thereby the existing joints now will reflect the curvature of the curve sketched by the animator.

III.C. Interpolation between poses

Another function that is available in this tool is the ability to interpolate the transformation data between the existing joints and new sketched joints. This is done by applying a value of influence from the new sketch joints to the existing joints rather than replacing their existing transformation data entirely (see Figure 8). In this way, the position of the existing joints is preserved so that this tool can be non-destructive to animators' workflow. To further enhance this feature, a weight parameter from 0 to 10 is provided to enable animators to determine the percentage that the sketched joints will influence the existing joints. This is useful not only to preserved the existing joint data, but it also serves as a fine tuning tool for animators to achieve their desired result precisely.

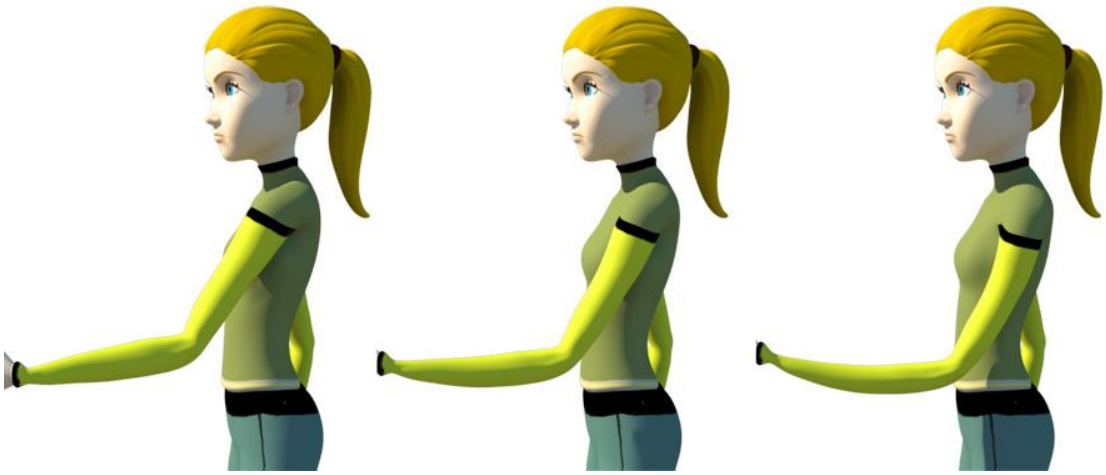

Figure 8. *(Left) Original 100% + Sketch weight 0 (Middle) Original 50% + Sketch weight 5 (Right) Original 0% + Sketch weight 10).*

III.D. Artistic control

To promote an artist-friendly interaction this tool supports multiple strokes input. This means an animator can sketch multiple times until the desired sketch is accomplished. This reflects the natural behavior of how many artists draw using pencil and paper in a hand-drawn medium. This is a useful feature promoting artistic workflow as well. If an animator is not satisfied with a curve drawn, more new curves can be drawn to minimize or even eliminate the influence from the unsatisfactory curve.

To calculate this operation, each vertex's position on a sketch curve is stored in X, Y and Z dimensions respectively. Subsequent new curves created are stored into new lists, accordingly. Each curve's vertices sum up according to their index number to compute the average value. These average values are returned to construct a new curve.

Technically, the multiple stroke's calculation is achieved by calling the Python's *zip* () function to take in multiple values and return slices of those iterate-able members until the end of each curve list. The values are then used to calculate the average of each member.

$$cvList0 = [[vertice1], [vertice2], [vertice3] \ldots]$$
$$average = float(sum(col))/len(col)\ for\ col\ in\ zip(*cvList0)$$

A Wacom stylus (see Figure 9) is the ideal input device for this tool because of its artist-friendly control and speedy stroke feedback [13]. In this thesis, the Wacom Cintiq (see Figure 10) is used. With the Cintiq, an animator can fixate on the monitor screen and have immediate visual feedback while drawing a curve, matching the experience of a hand-drawn medium.

Figure 9. *Wacom stylus range of products.*

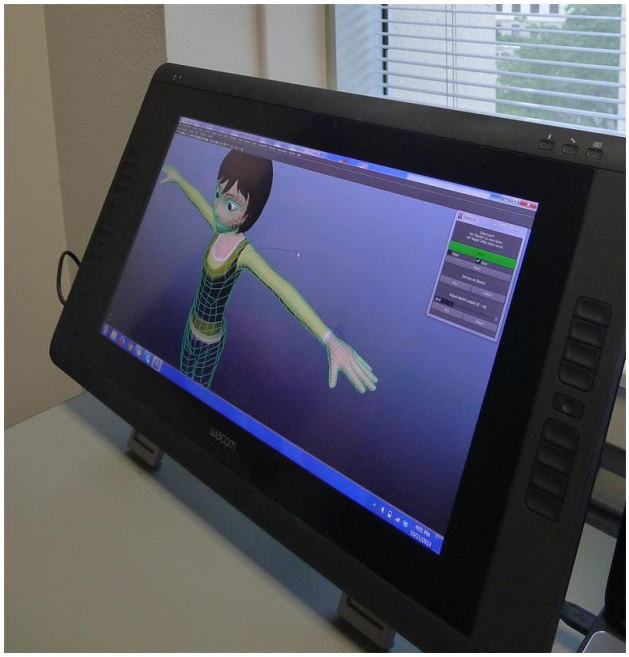

Figure 10. *Wacom Cintiq.*

III.E. Integration into a production environment

This tool is integrated into Autodesk Maya through a graphical user interface (GUI). Animator can access controls and options that are associates with this tool (see Figure 11).

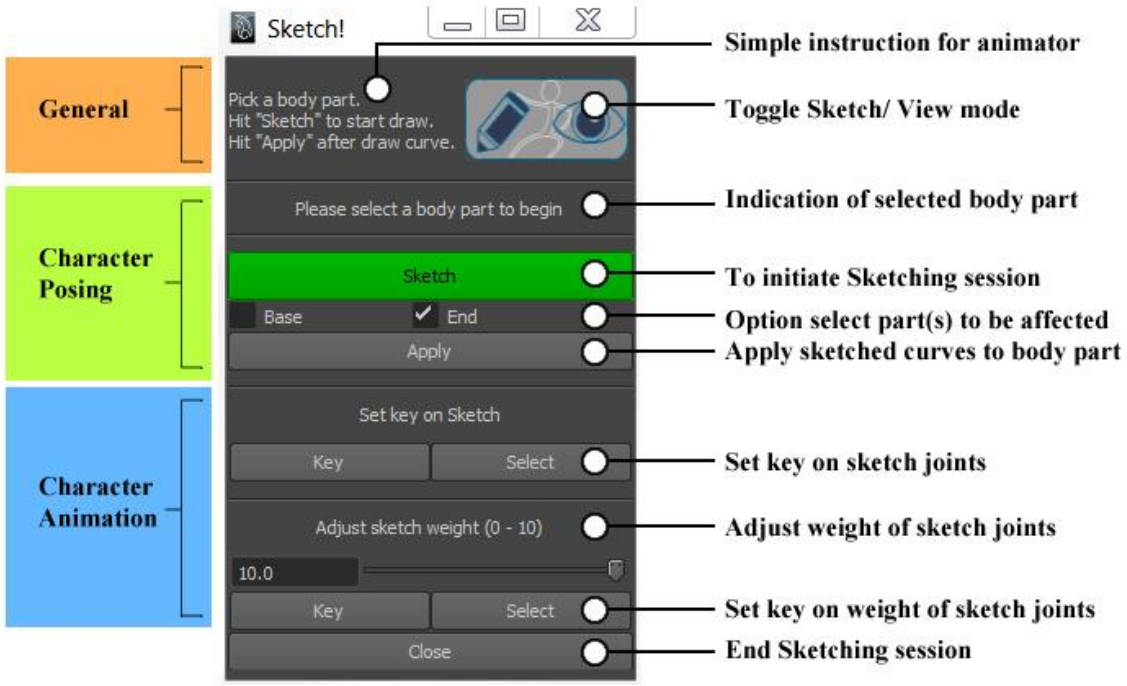

Figure 11. *Graphical user nterface layout*

Once the tool is initialized, Maya is switched from the existing 3D viewport to the sketch-based environment, that includes the custom curve plugin, displaying the

‘picker’s and temporarily hiding unnecessary character controls. This helps provide a spacious and clean working environment for animators to start work.

The GUI is divided into 3 portions. The top portion (General) of the GUI is associated with general tool functions; the middle portion (Character Posing) has options for posing a CG character, while the bottom (Character Animation) is focused on setting key frames and extending the tool functionality to character animation. All the functionalities and controls are arranged from top-bottom, in sequence with an animator’s workflow.

In the ‘General’ section, there are simple instructions informing animators about the basic workflow of the tool. An image button can toggle between the sketching mode and view mode. This assists animators with visualizing the final result of the character posing.

‘Character Posing’ starts with the animator indicating the selected ‘picker’. The main “Sketch” button in green. It is the only colored button and visually leads animators while switching between the working viewport and this tool’s GUI. Underneath the “Sketch” button are two checkboxes with sub-components of the selected ‘picker’ that will be influenced by the sketch curve. This control is useful when the animator is satisfied with either end of the joint chains but needs to further adjust the curvature in between. After an animator has all the desired options selected, hitting the “Apply” button and will be display the results in the viewport.

The GUI is designed to be simple so that it will not clutter the animator’s working space. To achieve this, the indication of ‘picker’ automatically updates

depending on the user selection. Also, the option of affected body parts reflects automatically according to the animator's selection. For example, if arm is selected, the options will be 'shoulder' and 'wrist'; if spine is selected, the options will be 'chest' and 'hip'. Also, the multiple strokes feature is hidden from the interface until more than one curve is drawn by the animator are detected. It then automatically performs the average function described previously and returns the new value of a new sketch curve. With these features, the GUI layout is minimized and concise.

The bottom portion of the GUI, 'Character Animation' allows animators to set keys on the sketch joints created. An animator can hit the "Key" button and the selected body part will be keyed at the current frame in time line. By setting keys at different frames in time line this tool naturally extends its use from posing the CG character to animating the CG character through interpolation between keys. The 'Select' button allows an animator to visualize the position of each key frame set in the time line.

As discussed earlier, this tool also allows animators to achieve fine adjustment by interpolating between the position of the existing joints and the new sketch joints. The interface encourages intuitive input for this feature via a slider that transitions back and forth through possible results. Animators have the option to key in the slider's numerical values as well. At the same time, this slider's value can be keyed at different frames. This helps smooth the transition of the CG character's poses between the existing joint and sketched joints' position.

Lastly, hitting the 'Close' button will end the current session of the sketch-based tool and return the entire working environment back to standard Maya workspace. The

structure of the character rig remains unchanged. All temporary object and placeholders are deleted after each sketching session.

CHAPTER IV

FUTURE WORK

This thesis describes several possibilities in the area of computer animation field in expanding and extending natural functionalities.

IV.A. Pressure sensitivity

The current sketch-based animation tool has the advantage of intuitive input from the Wacom stylus but not to its full extent. The Wacom stylus is equipped with pressure sensitivity that can be used as another source of input data, such as controlling the depth value. One can configure the system that the harder pressure the vertices on the curve will be closer to the camera and vice versa. This again reflects the natural behavior of artists when they draw using pencil and paper.

IV.B. Facial animation

This sketch-based animation tool is easily extends the same processes to a CG character's facial animation where joint-based deformations are used. Currently, animators animate facial expression using the channel box's parameters or through a simple GUI. These methods are not intuitive when animators have to manage a variety of numerical data at once. With this tool, for example, an animator can now sketch a curve for the eyebrow to animate it. The sketch curve would generate a chain of joints that can be configured to influence the existing eye brow joints. The same weighting

features can be implemented to further fine tune the position without sacrificing the original animation data. Another benefit is the algorithm behind this tool can be simplified since we can use a 2D interface for animators to interact in order to eliminate the 3D depth ambiguities.

IV.C. Effects animation

Curves that generate from this sketch-based tool can be used to control the particle effects in 3D applications. This could be extremely useful for effect artists to control a particle's goal, flocking and curve flow. The pressure sensitivity's level could be incorporated into the tool to control other particle parameters, such as density, life, count, et cetera.

CHAPTER V

CONCLUSIONS

V.A. Results

For testing this sketch-based tool, a character was modeled and rigged in Autodesk Maya. Initially, the character was in a default T-pose (see Figure 12).

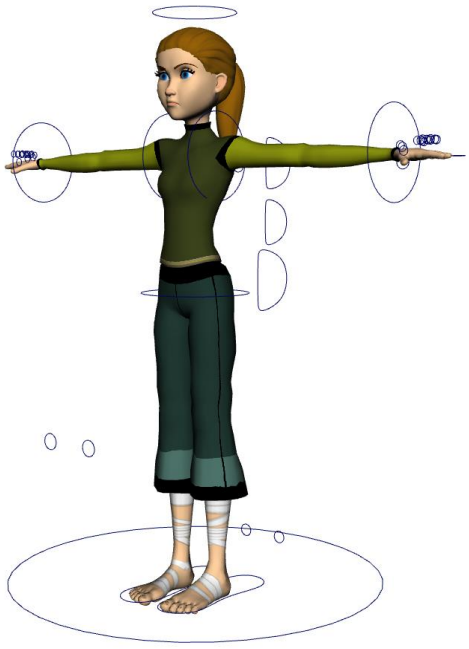

Figure 12. *Character default pose. Modeled by Christine Li.*

The character was then posed by manipulating the original character control system, which is the common method for most animators (see Figure 13).

Figure 13. *Character pose using original character rig and motion system.*

Next, the sketch-based tool was loaded into Autodesk Maya after matching the joint names within the tool's script. The naming conventions are listed at the beginning of the Python script to ease customization for each CG character's setup. Once the setup was complete, the sketch-based tool's window was loaded to start a new sketching session. Through it, the character's pose was modified creating the result seen in Figure 14.

Figure 14. *Character pose using sketch-based tool.*

For comparison, the character pose before using the sketch-based tool is shown in Figure 15. The pose in Figure 13 does not show the C-shaped curve that is desired under the principle of line-of-action. On contrary, the character pose after using the sketch-based tool as seen in Figure 14, portrays a more natural and lively line-of-action.

Figure 15. *Comparison before (Left) and after (Right) the sketch-based tool is used in posing the character.*

Manipulation of the character's pose using this sketch-based tool is easier and faster than manipulating individual controllers and parameters through a mouse and keyboard. Animators can simply select a desired body part, sketch a curve and view the new character pose instantly.

This sketch-based tool can be quickly implemented onto other characters (see Figure 16) to achieve the same level of flexibility because it only requires name-matching within a Python script.

Figure 16. *Expressive poses of various characters created with sketch-based tool. (Left) Modeled by John Doublestein (Middle) Modeled by Ser En Low (Right) Modeled by Anahita Salimi .*

This tool also can be customized to fit any body part or props. Below is an example of implementation on the ponytail of the female character (see Figure 17).

Figure 17. *Sketch-based tool on customized character's part. (Left) before sketch (Middle) sketched curve (Right) implemented new pose.*

V.B. Limitation

The current sketch-based tool is not able to deal with rotation parameters. This means animators are not able to rotate the body part directly using this tool. The twisting attributes of body parts remain constant as well. This is because the data retrieved from the sketch curve only specifies point position in world space. A temporary solution is to incorporate controls into the sketch-based tool that link back to the original motion system for manipulating rotation parameters (see Figure 18). In this example, a control has been attached to the CG character's wrist for an animator to rotate the hand quickly. The tool is also able to check the existing wrist controller for rotation values. If present, the values are copied over to the sketch wrist controller. With this functionality, animators are able to continue working in the sketch-based tool environment for the majority of posing and animation.

Figure 18. *Temporary solution to overcome limitation in rotation.*

REFERENCES

1. Lasseter, J. Principles of Traditional Animation Applied to 3D Computer Animation. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press (1987), New York, NY. 35 – 44.

2. Blair, P. *Cartoon Animation*. Laguna Hills, California: Walter Foster Publishing, 1994.

3. Mattesi, M.D. *Force: Character Design From Life Drawing*. Burlington, MA: Focal Press, 2008.

4. Gallagher, B. *Glen Keane Talks Tangled*. http://www.movieweb.com/news/glen-keane-talks-tangled. [Accessed April 2014]

5. O'neill, R. *Digital Character Development: Theory and Practice.* San Francisco, CA: Morgan Kaufmann, 2008.

6. Desowitz, B. *Hotel Transylvania Gets a 2D Transfusion*. http://www.awn.com/animationworld/hotel-transylvania-gets-2d-transfusion. [Accessed April 2014]

7. Craft, B., Caims, P. Sketching Sketching: Outlines of a Collaborative Design Method. *Proceedings of the 23rd British HCI Group Annual Conference on People and Computer: Celebrating People and Technology*. (2009), Cambridge, United Kingdom, 65-72.

8. Davis, J., Agrawala, M., Chuang, E., Poppovic, Z. and Salesin, D. A Sketching Interface for Articulated Figure Animation. *ACM SIGGRAPH* (2003), 1-8.

9. Lin, Y. 3D Character Animation Synthesis from 2D Sketches. *SESSION: 3D Computer Animation II* (2006), 93 - 96.

10. Matthews, T. A sketch-based articulated figure animation tool. *In Proceedings of the South African Institute of Computer Scientists and Information Technologists Conference on Knowledge, Innovation and Leadership in a Diverse, Multidisciplinary Environment.* (2011), VOGTS, D. Cape Town Waterfront, 151-160.

11. Autodesk. *Maya API*. http://download.autodesk.com/us/maya/2011help/API/index.html. [Accessed April 2014]

12. Shirley, P., Morley, R.K. *Realistic Ray Tracing*, Natick, MA: A.K. Peters, 2003.

13. Wacom. *Wacom creative products*. http://www.wacom.com/en/us/creative. [Accessed April 2014]