

CREATING PROCEDURAL ANIMATION FOR THE TERRESTRIAL
LOCOMOTION OF TENTACLED DIGITAL CREATURES

A Thesis

by

SETH ABRAM SCHWARTZ

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,	Philip Galanter
	John Keyser
Head of Department,	Tim McLaughlin

May 2015

Major Subject: Visualization

Copyright 2015 Seth Schwartz

ABSTRACT

This thesis presents a prototype system to develop procedural animation for the goal-directed terrestrial locomotion of tentacled digital creatures. Creating locomotion for characters with multiple highly deformable limbs is time and labor intensive. This prototype system presents an interactive real-time physically-based solution to procedurally create tentacled creatures and simulate their goal-directed movement about an environment. Artistic control over both the motion path of the creature and the localized behavior of the tentacles is maintained. This system functions as a stand-alone simulation and a tool has been created to integrate it into production software. Applications include use in visual effects and animation where generalized behavior of tentacled creatures is required.

TABLE OF CONTENTS

	Page
ABSTRACT	ii
TABLE OF CONTENTS	iii
LIST OF FIGURES.....	v
LIST OF TABLES	vi
CHAPTER	
I INTRODUCTION.....	1
II RELATED WORK	3
II.A. Traditional Animation	3
II.B. Motion Capture	6
II.B.1. Early Motion Capture	6
II.B.2. Brilliance.....	9
II.B.3. Optical Motion Capture	9
II.B.4. Markerless Motion Capture	12
II.B.5. Capture from Video	13
II.B.6. Motion Capture for Octopodi.....	14
II.C. Procedural Motion	18
II.C.1. Articulated Rigid Bodies.....	18
II.C.2. Goal Directed Locomotion	19
II.C.3. Intuitive Control Systems	20
II.D. Case Studies	22
II.D.1. Spore	22
II.D.2. Pirates of the Caribbean 2: Dead Man’s Chest.....	24
II.D.3. Night at the Museum 2	26
III METHODOLOGY.....	28
III.A. Expressive Animation	28
III.B. Art Directability.....	29
III.C. Efficient Setup	31
III.D. Efficient Simulation	31
III.E. Integration.....	32

IV	IMPLEMENTATION	33
	IV.A. Phase One: Building a Creature	33
	IV.A.1. Scene Description File	34
	IV.A.2. Creation in Maya.....	35
	IV.B. Phase Two: Goal Directed Locomotion	37
	IV.C. Tools.....	40
V	FUTURE WORK	41
VI	RESULTS AND CONCLUSION	43
	REFERENCES.....	44

LIST OF FIGURES

FIGURE	Page
1 Sequence of pictures capturing the motion of horse and rider [15]	6
2 Motion photographed by Etienne-Jules Marey [8].....	7
3 Rotoscoping by Max Fleisher [15].....	8
4 A performance in an optical motion capture stage [12].....	10
5 Input versus reconstruction of a dancing girl wearing a skirt [4].....	12
6 Active contours overlay on a horse [19]	14
7 Anatomy of a typical octopus: (A) lateral view and (B) dorsal view [11]	15
8 Tentacle cross section showing major muscle and nerve components [11]	16
9 Animation system structure [9].....	21
10 Two level controller composition scheme [14].....	22
11 Different player-created leg configurations [6].....	23
12 Davey Jones [3].....	24
13 Rigid Tentacle Beard [3]	25
14 Museum Octopus [5].....	26
15 (A) Link capsule (B) Mantle cylinder.....	33
16 Creature creator GUI.....	35
17 Creature with rig in Maya	36
18 Simulated creature with three goals	38

LIST OF TABLES

TABLE		Page
1	Rig Controls	37

CHAPTER I

INTRODUCTION

Creating the locomotion of multi-limbed characters for visual effects, animation, and video games is time and labor intensive. The locomotion animation of bipeds, such as humans, can be generated via key-framing, motion capture, or procedural methods. Quadrupedal locomotion, while not as easily generated via motion capture, can be reasonably achieved via key-framing and procedural techniques. Beyond four legs, the problem becomes much more complex. Key-framing is significantly more time-consuming, motion capture suffers from an absence of suitable actors, and the use of procedural techniques remains relatively unexplored.

This thesis tackles the problem of generating the locomotion of multi-legged characters - specifically tentacled characters - through procedural techniques that guide both the action of the limbs and the direction of movement. A prototype system is developed that presents an interactive, real-time, physically based solution that can be integrated with production software. This solution will offer artistic control over both the motion of the creature and the localized behavior of tentacles.

This prototype system uses NVIDIA PhysX to produce simulations that run in real-time and tests have been concluded that show creatures with up to 8 tentacles exhibiting locomotion and writhing behavior similar to octopodi while moving across a substrate toward pre-determined and directable goals.

This technique provides an efficient method for generating procedural animation for the terrestrial locomotion of tentacled digital creatures for use in visual effects and animation where the generalized behavior of tentacled creatures is required.

CHAPTER II

RELATED WORK

II.A. Traditional Animation

Animation as an art form is primarily concerned with imbuing images with life. Through animation mops can dance (The Sorcerer's Apprentice), toys can cry (Toy Story), and dragons can stalk the earth (Harry Potter). The process of creating animation has changed over time from hand-drawn cell animation to a heavy reliance on computer animation in today's works, but the artistic goal of bringing the images to life has remained.

Early research in computer animation was focused on developing 2D animation techniques that were based on traditional cell animation. The fundamentals of character animation are based on the work of hand-drawn animated characters that was pioneered at Walt Disney Studios [10]. Disney and his early animators studied human figures and animals in motion, the analysis of which became important to the development of animation [7]. Some animators began to apply this knowledge to production animation, which became more sophisticated and realistic. Gradually these procedures were isolated, perfected, and named, and new artists would be taught these practices as rules of the trade. They became known as the 12 fundamental principles of animation [7].

These principles were:

1. Squash and Stretch – Giving a sense of weight and flexibility to an object by distorting its shape during an action.

2. Anticipation – The preparation for a major action the character is about to perform.
3. Staging – The presentation of an idea so that it is completely and unmistakably clear.
4. Straight Ahead and Pose to Pose – Drawing from the beginning of a scene through to the end as opposed to drawing the key frames first and inbetween frames later.
5. Follow Through and Overlapping Action – The termination of an action and establishing its relationship to the next action.
6. Slow In and Slow Out – Emphasizing extreme poses by spacing inbetween frames to achieve subtlety of timing and movement.
7. Arcs – The visual path of movement that gives animation a more natural action.
8. Secondary Action – An extra action that either results from the primary action or supports it.
9. Timing – The number of drawings or frames for an action which determines the amount of time it will take on screen.
10. Exaggeration – Accentuating the essence of an idea via the design and action.
11. Solid Drawing – Taking into account forms in 3D space, giving them weight and volume.
12. Appeal – Creating an easy to read design or action that will capture and involve the audience's interest.

Some of these principles translate easily from 2D, hand-drawn animation to 3D computer animation, while others do not. John Lasseter, currently Chief Creative Officer of both Walt Disney and Pixar Animation Studios, composed the first treatise on transitioning the 12 principles of traditional animation to computer animation in 1987 [10]. He described how 2D hand drawn animation dealt with a sequence of drawings that simulate motion. 3D computer animation is achieved by creating keyframe poses and allowing the computer to generate the inbetween frames. While timing, anticipation, staging, follow through and overlap, exaggeration, and secondary action can be applied independent of the medium; squash and stretch, slow in and out, arcs, straight ahead and pose to pose, and appeal are applied differently between 2D and 3D animation.

Although the animation principles allow the animator to create performances that are both art-directable and expressive, this animation can become increasingly complex and time consuming as characters become more and more complex. For example, the sea-witch Ursula in *The Little Mermaid* was drawn with only six tentacles instead of eight. This was due to the studios budget and the difficulty coordinating eight tentacles [16].

The system presented in this work incorporates several of these principles to develop expressive locomotion for tentacled creatures. Timing, arcs, anticipation, appeal, straight ahead and pose to pose, slow-in and slow-out, follow through and overlap and their uses will be further discussed in a later section.

II.B. Motion Capture

Motion capture (mocap) is the process of recording a live motion event and translating it into usable mathematical terms by tracking a number of key points in space over time and compiling them to obtain a single 3D representation of a performance [12]. Essentially, it is the technological process of translating a live performance into a digital performance. This data can then be used to study motion or give life directly to 3D computer models.

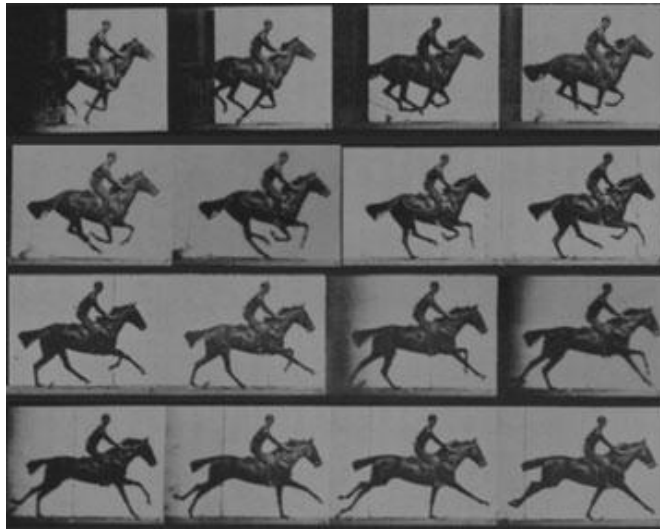


Fig. 1. Sequence of pictures capturing the motion of horse and rider [15].

II.B.1. Early Motion Capture

Early attempts at capturing motion date back to the 1800's when Eadweard Muybridge was hired to settle a bet on whether or not all four hooves of a horse leave the ground simultaneously [8]. To prove that this was in fact the case Muybridge used a sequence of photographs, shown in figure 1, taken with a dozen cameras to capture the

horse's movement. His sequential photographs are still used by artists as valuable reference material.

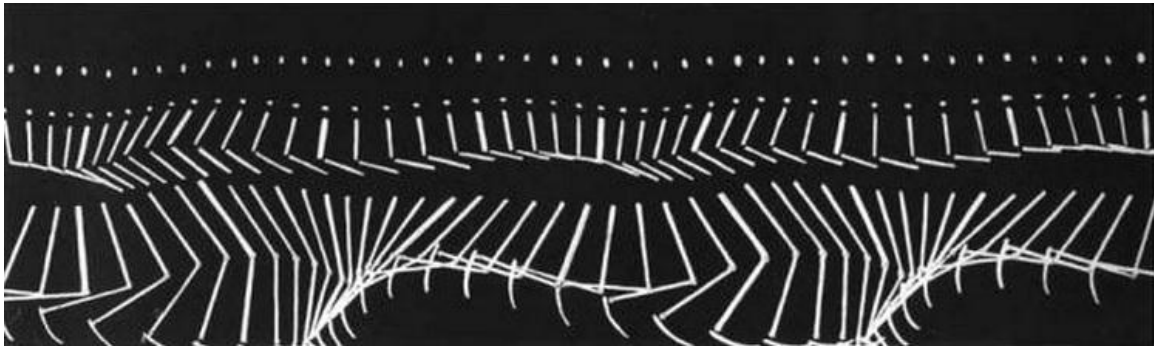


Fig. 2. Motion photographed by Etienne-Jules Marey [8].

In contrast to Muybridge's multiple camera technique, Etienne-Jules Marey used one camera to capture motion. He invented a chronophotographic fixed-plate camera with a timed shutter that allowed him to expose multiple images (sequential images of movement) on a single plate, as seen in figure 2[8]. His research subjects included the locomotion of humans, animals, birds, and insects.

Historically, motion capture is closely related to rotoscoping in which the photographed motion of a live actor is traced on an overlaid film, frame-by-frame, to create the motion of an animated character. The rotoscope device, shown in figure 3, was invented by cartoonist Max Fleisher in 1915, with the intent of automating the production of cartoon films [8].

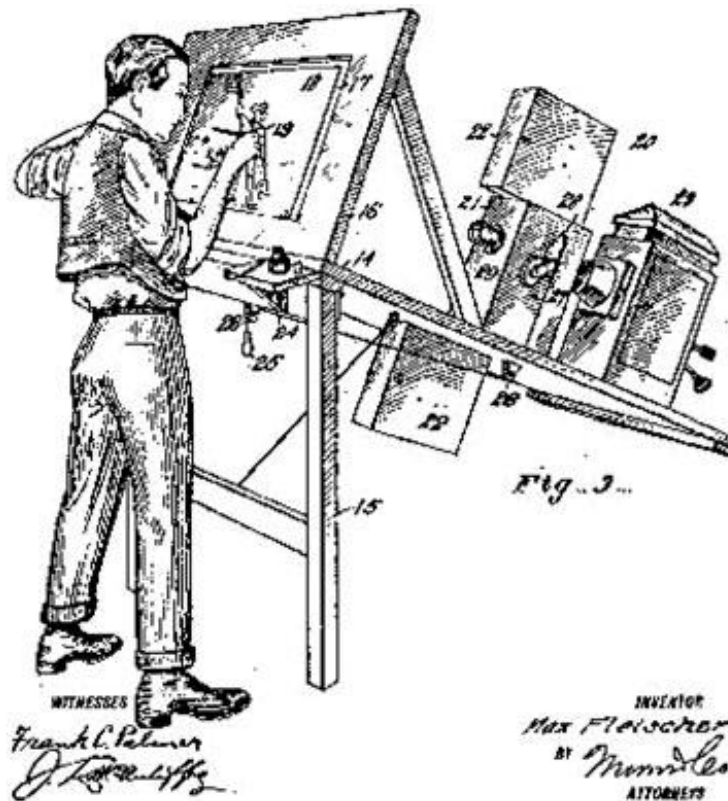


Fig. 3. Rotoscoping by Max Fleisher [15].

The first cartoon character ever rotoscoped was Koko the Clown. Fleisher filmed his brother, David, in a clown suit and they spent nearly a year making the first animation using rotoscoping. After patenting the device in 1917 Fleischer produced the *Out of the Inkwell* series, in which animation and live action were mixed. In 1937, Walt Disney Studios used rotoscoping to create the motion of some human characters in *Snow White* [7]. Snow white and the prince were both partially rotoscoped. Later Disney animation characters were highly stylized and rotoscoping was used as a method for studying human and animal motion.

II.B.2. Brilliance

Research and development of digital motion capture technology began with the pursuit of medical and military applications during the 1970's [8]. However, it was not until the 1980's that the CGI industry discovered the technology's potential for entertainment. The first successful application of mocap technology was featured in a commercial produced by Robert Abel and Associates for the National Canned Food Information Council and was aired during the 1985 Super Bowl [12]. This commercial, "Brilliance", featured a shiny female robot moving like a human. Abel and Associates painted black dots on 18 joints of a female model and photographed her on a swivel stool from multiple points of view. Images were then imported into Silicon Graphics Iris 1000 systems. They were then able to analyze the difference in measurement between pairs of joints for each point of view and develop a series of algorithms that would be used to animate the digital robot. The entire process was done frame-by-frame and took 4½ weeks.

II.B.3. Optical Motion Capture

Marker based optical motion capture systems are the preferred method of motion capture in movie production and game studios for measuring motion of real performers [12]. Typical optical systems are based on a single computer that controls the input of CCD (charge coupled device) cameras to triangulate 3D position data. CCD cameras are light-sensitive and use an array of photoelectric cells to capture light.



Fig. 4. A performance in an optical motion capture stage [12].

By measuring the intensity of light for each cell a digital image can be created. Systems usually employ between 8 and 32 cameras that capture the marker positions at speeds ranging from 30 to 2000 samples per second. Motion capture actors wear either passive or active markers. Passive markers are made of reflective material and the shapes are usually spherical, semi-spherical, or circular. The shape and size of the markers will depend on both the camera resolution and the capture subject; smaller markers are used for faces and hands. These markers can be attached directly to skin or Velcroed to a mocap suit, which is a full-body unitard made of stretchy materials. An example of the optical setup can be seen in figure 4. Cameras in a passive marker

system are equipped with light-emitting diodes (LEDs) and the light emitted by the LEDs is reflected by the markers. Alternatively, markers in an active marker system are LEDs [8]. Modulating the amplitude or frequency of each LED allows systems to identify markers. Some of the latest active marker systems will work in natural lighting conditions. This allows for them to be used at locations outdoors. The advantages of using optical systems are that optical data can be extremely accurate and a large number of markers can be tracked simultaneously at a high sample rate. The disadvantages are that optical data requires extensive post processing, so operating costs are high. The hardware is also expensive, costing anywhere from \$50,000 to over a million for a high end system.

While these approaches work well for human actors, capturing animal motion presents some challenges. The first challenge is actually attaching equipment or markers to animals in order to track their movement [15]. Performances must then be constrained to an area covered by cameras. Treadmills can be used in certain cases, for example with horses or with dogs. However, this can produce uncharacteristic movement, walking on a treadmill is different from walking on grass, and this approach is unsuitable for wild animals. Data gathered this way will also be highly specific to the particular animal, making it difficult to change or incorporate with other motions. This method is inadequate for tracking movements of animals who resist the attachment of markers on their body, or who behave unnaturally with markers attached. Animals also make bad actors as they can be difficult to direct.

II.B.4. Markerless Motion Capture

De Aguiar et al. present a marker-less approach to capturing human performances in [4]. Applying motion captured data directly to a 3D model can sometimes look unrealistic. This is because it can be very difficult to get a direct correspondence between a 3D model and the captured human performer. Despite the high accuracy of marker-based systems, their very restrictive capturing conditions, tight fitting outfits and markers, make it infeasible to capture time varying body shape and fail to track people wearing loose apparel. Prior to video-recording human performances, de Aguiar et al. take a full-body laser scan of an actor while they are completely dressed. After scanning, motion is recorded using several cameras setup in a circular arrangement around the center of the scene. A color-based background subtraction is applied to footage to yield silhouette images of the captured actor.



Fig. 5. Input versus reconstruction of a dancing girl wearing a skirt [4].

The silhouette of the actor in the video is compared to the silhouette of the character model, this is then refined until there is a close match. Since their algorithm is purely mesh-based and makes as few possibly assumptions about the type of surface being tracked, it can capture performances of people wearing wide apparel, such as the dancer wearing a skirt in figure 5. This technique could be adapted for use with animals, although capturing the motion of hair or deep folds with self-collisions is not yet possible.

II.B.5. Capture from Video

Working with wild animals such as lions, tigers, cheetahs, and elephants present some obvious difficulties. Traditional motion capture methods are clearly unsuitable, so video processing is the most practical solution. Numerous wildlife documentaries feature such animals although it is difficult to take advantage of this footage. Since it is shot from a single viewpoint, standard 3D motion measurement techniques are not possible [19]. Wilhelms and Gelder use an innovative technique for extracting the motion of a horse from unrestricted monocular video and applying the data to 3D models. The video image is processed to identify features using active contours as seen in figure 6. Active contours are 2D curves that snap to and track image features. The horse model is then scaled and aligned to match the one in the video. The active contours of the video are then anchored to the 3D model of the horse. Playing the video

changes the shape of the contour lines, which in turn change the position of the horse's limbs.

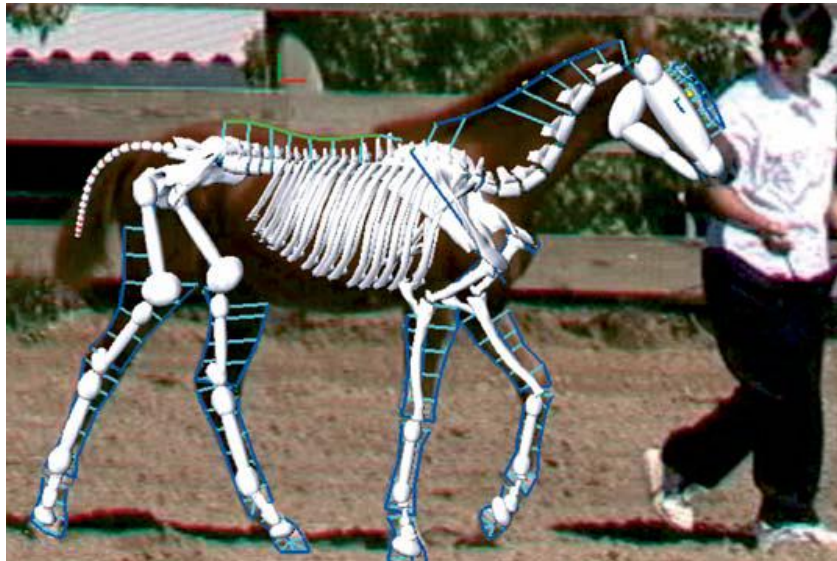


Fig. 6. Active contours overlay on a horse [19].

Although this method is very sensitive to noise, requiring the user to reinitialize active contours every few frames, complicated movements can be extracted regardless of background, camera movement and feature clarity. This technique was found to be most successful for moderately slow motions and where backgrounds were relatively simple.

II.B.6. Motion Capture for Octopodi

Octopodi motion capture raises many more difficult issues, mainly due to the deformable nature of their tentacles. The octopus is a cephalopod mollusk of the order

Octopoda. The octopus, shown in figure 7, has eight arms, and like other cephalopods, is bilaterally symmetric [11].

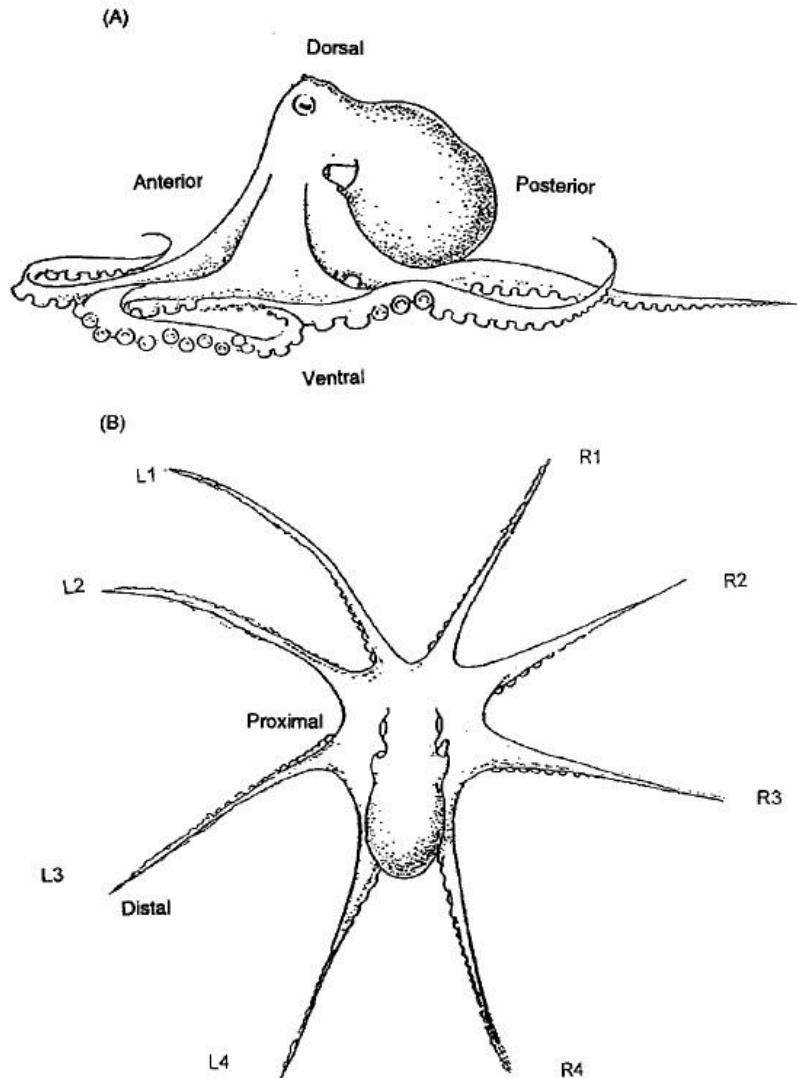


Fig. 7. Anatomy of a typical octopus: (A) lateral view and (B) dorsal view [11].

Unlike other cephalopods, the majority of octopi have almost entirely soft bodies with no internal skeleton. A beak, used for eating shelled mollusks and crabs, is the only

hard part of their body. Each of their eight tentacles has suckers along the ventral surface which act as suction cups. Along the length of the tentacle are longitudinal, circular, and radial muscles. This allows the tentacle to change both length and direction. Oblique muscles spiral along the periphery allowing the arm to twist. A cross section of the tentacle can be seen in figure 8.

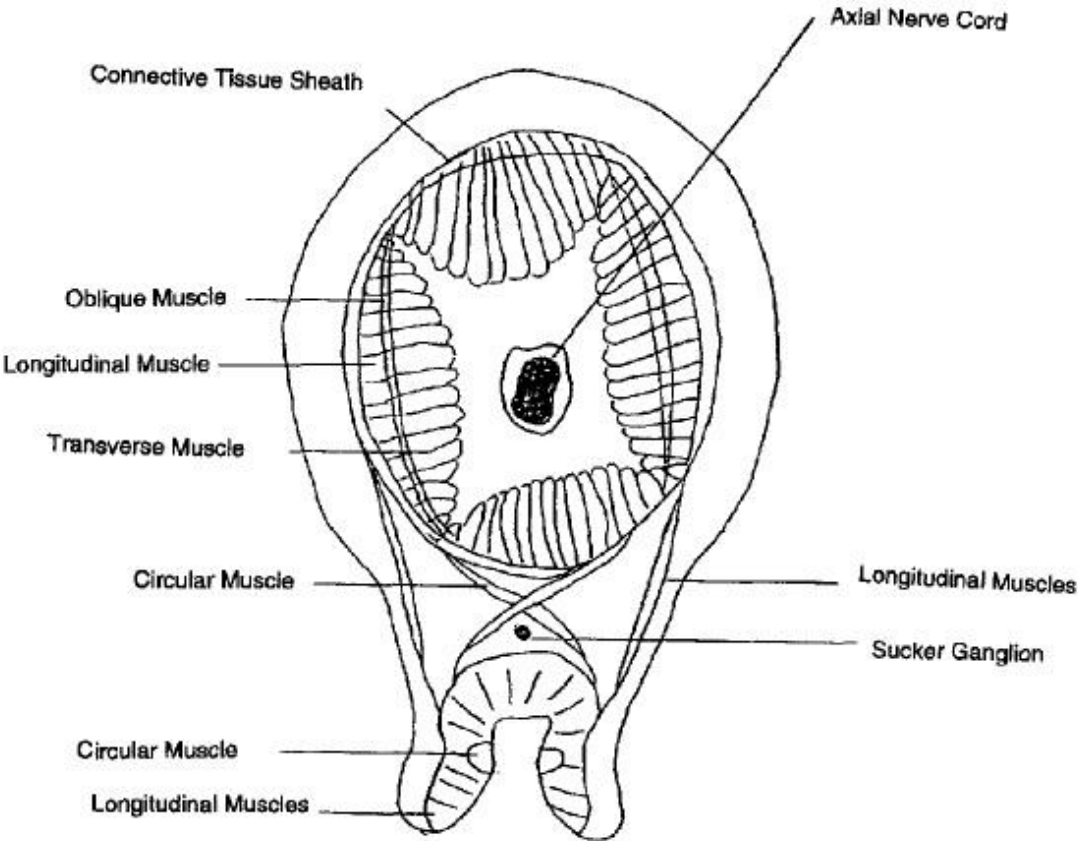


Fig. 8. Tentacle cross section showing major muscle and nerve components [11].

Arms are used for various tasks such as locomotion, food gathering, hunting, and sophisticated object manipulation. Tentacles have no fixed skeleton making them maximally extensible and flexible. They are able to bend in any plane at any point along its length. This bend point will propagate down the length of the tentacle during movements. Approaches to capturing their motion using markers are inadequate because an octopus will behave unnaturally while trying to remove objects from its skin. Further difficulties can arise due to the cluttered nature of their environment or reflections from water or glass when they are swimming in a tank [20]. Zelman et al. have been conducting a large-scale research investigation into octopus motor controls focusing on octopus arm actions and investigating kinematics, bio-mechanical and neural aspects of movement. Their system for recording and analyzing octopus behavior integrates segmentation, skeletal representation and 3D reconstruction methods. The input into the system is a pair of video sequences recorded by two video cameras in stereo configuration. The cameras need to be calibrated to later reconstruct the 3D movement. The system uses the following three main steps. First, 3D image segmentation is applied separately to each video sequence. This results in a pair of sequences in which the segmented arm is represented by silhouettes. Next, skeletal representation is extracted for each silhouette of the tentacle, resulting in a pair of sequences in which a virtual backbone of the tentacle is defined by 2D curves. Finally, each pair of 2D curves is used to reconstruct a 3D curve, resulting in a single spatio-temporal sequence describing the configuration of the tentacle in space as a function of time. Zelman et al. present a novel system for tracking octopus arm motion using

markerless motion capture techniques. Their system proves effective in the detection and extraction of tentacle data and could be adapted for use in visual effects and animation.

II.C. Procedural Motion

II.C.1. Articulated Rigid Bodies

A key aspect of animating digital characters is the animator's ability to achieve realistic motion with a minimal amount of effort. Armstrong and Green presented an approach to human figure animation that incorporated dynamics into the model of the figure [1]. Their model was able to achieve a wide range of motions simply by applying forces and torques to joints at key points in the animation. They believed a human figure model should have three characteristics:

1. "The model should produce realistic motion sequences when given realistic input data. In other cases the model should produce believable results."
2. "The amount of information the animator must provide should be minimal and proportional to the complexity of the motion."
3. "The model should be able to react to and act on its own environment... This obviously requires a model that accounts for such physical properties as mass, force, inertia, torque, and acceleration."

While their research was applied to the dynamics of animating human figures, we can apply these characteristics to the development of tentacled creatures.

Weinstein et al. proposed an impulse based approach for dynamically simulating articulated rigid bodies which undergo frequent and unpredictable contact and collision [18]. Impulses are applied one joint at a time to enforce the position and orientation based articulation constraints. The primary benefit of using impulses is their system can seamlessly integrate with impulse based contact and collision detection algorithms. Weinstein et al. expanded their body of work with an impulse-based approach to proportional derivative (PD) control for joints and muscles [17]. Inverse dynamics are used to alleviate complications from expanding PD control to multiple joints. By extending post-stabilization from a joint by joint approach to a global approach for an entire articulated rigid body, joints are allowed to move smoothly toward the target state. These approaches for articulated rigid body simulation were implemented in "Pirates of the Caribbean 2: Dead Man's Chest" in the creation of Davey Jones' tentacle beard. This work will be discussed in a later section.

II.C.2. Goal Directed Locomotion

The specification and control of motion in creature animation can be a challenge for animators. In goal-directed, procedural control systems, the amount of detail necessary to define a motion is greatly reduced. Bruderlin and Calvert discussed a hybrid approach to animating human locomotion which combined goal-directed and dynamic motion control [2]. Their approach specified locomotion parameters as tasks at the highest level of control. The forces and torques which generated a desired locomotion were calculated as a result of the stepwise decomposition of a few

locomotion parameters. Possible parameters the user could specify were velocity, step length, and step frequency. The forces and torques used to drive the dynamic simulation provided the low level control for the system. Equations of motion for the legs were tailored for locomotion instead of relying on a general dynamic model. These equations were restricted so only a specific range of movements were allowed. Goal directed locomotion for tentacled creatures can be achieved by following a similar procedure.

II.C.3. Intuitive Control Systems

Badler et al. presented an adaptive control scheme for an animation system which linked user specified kinematic requirements on primary motions to a dynamic simulation of the entire figure [9]. The user of the system provided kinematic trajectories as input for the degrees of freedom for the figure they wanted direct control over. The output motion was fully generated using forward dynamics. This system is shown in figure 9.

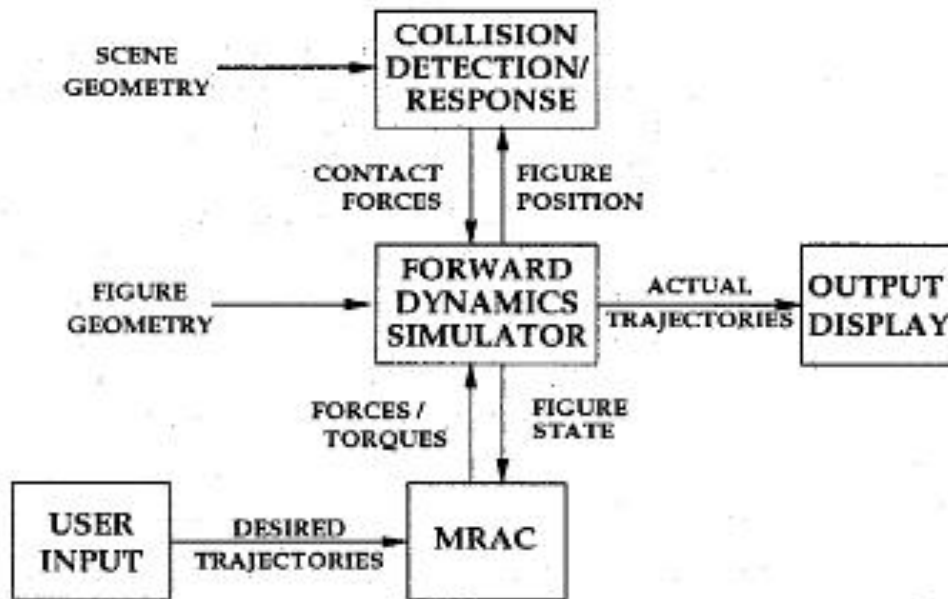


Fig. 9. Animation system structure [9].

“Hybrid Control for Interactive Character Animation” combined kinematic animation with physical simulation to animate interactive characters [14]. This system allowed for characters to be fully animated, fully simulated, or some combination of the two. By combining these techniques, the advantages of each can be seen. Individual controllers are seen as black boxes which encapsulate any animation technique. If an individual controller was capable of moving a dynamic character from its current state to a desired goal, it was added to a pool of controllers under the management of a supervisor controller. Controllers could be sequenced together in any order and the supervising controller would manage control transitions. This control scheme can be seen in figure 10.

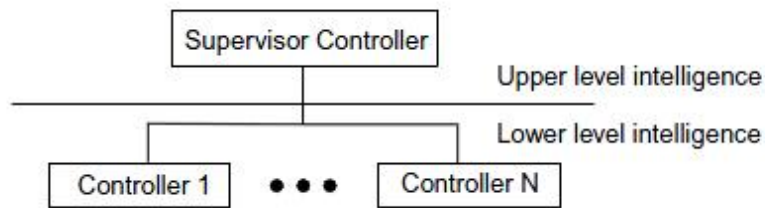


Fig. 10. Two level controller composition scheme [14].

Shapiro et al. introduced a tool kit for creating dynamic controllers for articulated characters under physical simulation [13]. The goal of their tool kit was to integrate dynamic controller methods into a usable interactive system intended to provide animators the means to quickly generate physically based motion. Dynamic controllers were developed through a combination of kinematics based control, reduced dimensionality physics, scripting controllers through the use of a controller language, and interactive control of dynamic characters.

II.D. Case Studies

II.D.1. Spore

Character animation in video games has traditionally relied on codifying skeletons early in a games development and creating animations rigidly tied to those skeletons [6]. An increasingly popular way to give players meaningful creative input into video games is in the form of user generated content. This was one of the primary

design goals for the game *Spore* (2008) which allowed players to create assets after the game had shipped [6].

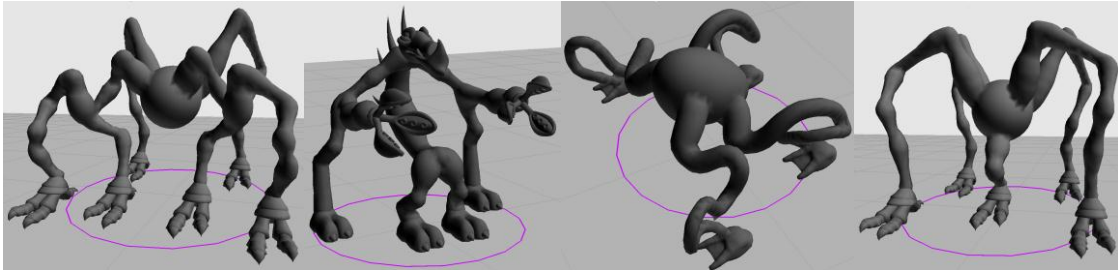


Fig. 11. Different player-created leg configurations [6].

Characters presented a particularly challenging problem. They required the synthesis of player created morphologies with authored animations for in-game actions. The style and quality of authored animations had to be preserved at runtime as they were retargeted to extremely varied player created morphologies. Hecker et al. introduced a novel system for animating characters whose morphologies are unknown at the time the animation is created [6]. Their authoring tool, Spasm, allowed animators to describe motion using familiar posing and key-framing methods. Data was recorded in a morphology-independent form that preserved the animation's structural relationships and stylistic information. At runtime, the generalized data was applied to specific characters to yield pose goals. Their system allowed them to animate characters with varying skeletal structures that did not exist when the animation was authored. Examples of unknown morphologies can be seen in figure 11.



Fig. 12 Davey Jones [3].

II.D.2. Pirates of the Caribbean 2: Dead Man's Chest

Pirates of the Caribbean 2: Dead Man's Chest (2006) introduced Davey Jones, shown in figure 12, a character with the dynamic rigged tentacle beard seen in figure 13 [3]. Each tentacle was a chain of rigid bodies with articulated point joints serving as connections between the bodies. Each tentacle had a controller to define parameters to achieve its desired dynamic behavior.



Fig. 13. Rigid Tentacle Beard [3].

To make the tentacles curl, the connecting point joints were activated using a sine wave controlled with attributes such as amplitude, frequency, and time. The control for each joint on a tentacle was accomplished using a force-based targeting system that calculated the difference between the target orientation and the current orientation. This system calculated torques between rigid objects constrained by a joint. The goal of the targeting system was to minimize the difference between the joint's target angle and its current angle. During the progression of the simulation the target angles were modified. The resulting difference, between the target angle and the current angle, produced an axis or rotation around which the connected rigid body could rotate. To simulate the effect of suction cups on a real tentacle, connecting springs between rigid bodies and contact objects were momentarily created and then destroyed. This allowed for the simulation of the suction cups grab and release.

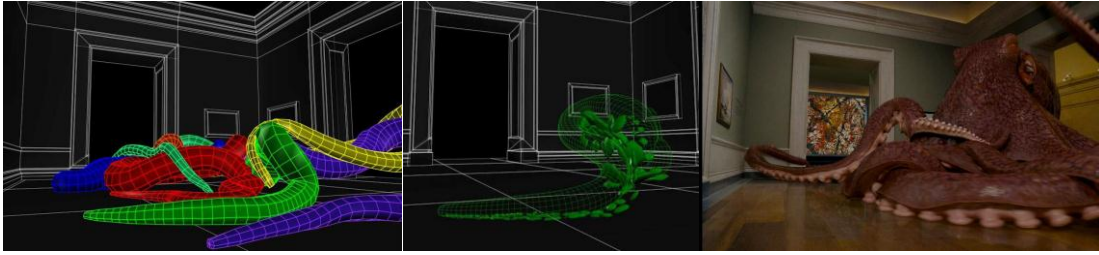


Fig. 14. Museum Octopus [5].

II.D.3. Night at the Museum 2

For the film *Night at the Museum 2* (2009), Rhythm and Hues (R&H) developed a "smart tentacle" which was used to animate a giant octopus as seen in figure 14 [5]. They used a layered simulation approach with key motions animated by the animator. In order to clean up the details, they used multiple passes of simulations for skeleton, flesh, and tentacles. Each tentacle arm was simulated as a cloth strip and attached to the original animation as springs acting as soft constraints. This allowed the cloth simulation to roughly follow the animation from the animator. The cloth object follows the user guided rest shapes. In their in-house cloth solver, their bending energy term would constantly track these rest shapes in a bending angle. This allowed the cloth object to give the illusion of behaving actively within the environment. For example, if the animator supplied an abrupt bend in the tentacle the simulation would react to the bend as well as the environment.

The first pass of the cloth simulation was to represent the skeleton. Next they used envelope geometry in a soft body simulation to represent the flesh of the tentacle. Finally, to simulate the suckers, just like ILM, they dynamically attached and detached springs. The user would specify a release decision such as a buildup of spring energy or a distance from the original attachment point as the possible conditions for

sucker release. By using this method for the springs they could represent initial sticking as well as abrupt release.

CHAPTER III

METHODOLOGY

For this system to be successful in a production setting, I have defined guiding principles that must be adhered to:

- The capacity to create expressive animation as defined by the Principles of Animation.
- The capacity for art directability that allows specific performance goals such as position, direction of motion, and level of activity to be specified.
- An efficient method for setting up the character and defining its qualities.
- A simulation that can run in real-time or near real-time.
- Integration with standard animation software.

III.A. Expressive Animation

A talented animator can take a few abstract notions of a character and create a performance that captures the essence of that character. The expressiveness of an animation relies heavily on its success in conveying an emotion to an audience. For a procedural animation system to be successful it must be capable of creating motion that is both believable and physically plausible. A system such as this will be considered

believable if it displays appropriate animacy. In other words, it must convince the audience that it is sentient.

In this system, target goals give creatures a trajectory and once set, it is expected that creatures will determine on their own how best to reach that goal. Allowing creatures to move about in a straight ahead manner ensures a certain amount of spontaneity in the locomotion that is generated. To move around in an environment individual tentacles are identified as driving tentacles. A tentacle can drive locomotion in two ways. It either pushes, or pulls. The creature determines how a tentacle drives by following a simple check. If the creature's body is located between the target goal and the tentacle then a push is required. If the tentacle is positioned between the goal and the creature's body then a pull will be necessary. Once the tentacle has determined its driving motion it will add actions into a queue to be executed. Actions in the queue either directly drive locomotion with a push or a pull, or serve as a pre-action that prepares the tentacle for the driving action. For instance a tentacle will first compress before it pushes the creature toward the goal. Likewise the tentacle will extend to prepare for pulling. Each queued action has a variable amount of time to execute and the time factor of each action will directly influence the speed and power of the driving motion.

III.B. Art Directability

The user is given direct control of this system in both the creature creation phase and the creature locomotion phase. This allows for a wide variety of custom creatures to

be created that are both visually appealing as well as capable of expressive goal-directed locomotion.

In the creature creation phase, a tool was written for Autodesk Maya that allows a user to build tentacled creatures. This tool presents the user with a Graphical User Interface (GUI) for quickly setting attributes such as dimensions of the mantle, number of tentacles and uniform tentacle parameters which consist of link length and radius, number of links and a scale value for the last link in the chain. Once the uniform attributes have been set a creature will be constructed inside the Maya viewport along with a control system for making quick customizations to the creature. After all desired changes have been made, the creature will be written out to a parameter file that is read in by the simulation.

Inside the simulation the user is presented with a Heads Up Display (HUD) containing a robust set of controls used during the simulation. These controls provide the user with interactive control over specific performance goals such as direction of motion and level of activity. Direction is given to each creature individually by setting a goal or a series of goals that define a path. In addition to this goal directed locomotion the user can quickly adjust the performance of each creature by quickly tweaking local parameters governing the speed, target angle, and strength of actions that are used in tentacle locomotion. These parameters can either be set for all of a creature's tentacles all at once or individually for each tentacle.

III.C. Efficient Setup

Using parameter files with the simulation is a quick and efficient method for defining creatures in the program. When the program launches it will look for the parameter file that was exported from Maya. If the parameter file is found, the program will read in the file, parse it, and construct a creature identical to the one that was created in Maya.

Inside the simulation creatures are defined by a mantle (body) and a variable number of tentacles. The role of the mantle is to govern over the functions of each tentacle. Each tentacle is built following the ideas laid out by the creation of Davy Jones' tentacle beard. Each tentacle is a chain of rigid bodies connected with articulating point joints. The articulation joints are similar to those used by ILM for Davy Jones in that these joints serve as motors for driving tentacle motion. These motors find the difference between their current orientation and a target orientation, and use forces and torques to move the joint between the two. During the progression of the simulation the target orientations are determined by manipulating sine functions over time, creating a propagating wave that defines tentacle movement. Simulating the effect of suction cups is handled by momentarily creating and destroying connection springs between the tentacle and a target surface.

III.D. Efficient Simulation

In the prototype system physically based dynamic animation is used to procedurally generate the goal directed locomotion of tentacled creatures. For this

system to be successful it is paramount that simulations be capable of running in real-time or near real time. NVIDIA's PhysX engine is utilized for all physical based dynamic animation and is capable of handling the requirements of this solution efficiently.

III.E. Integration

A custom plugin has been written for Maya to integrate the data from the simulation. When the simulation runs it will continuously save position and orientation data pertaining to all creatures and tentacles in the scene. The data is saved out to a text file and read into Maya where it can be further manipulated and rendered.

CHAPTER IV

IMPLEMENTATION

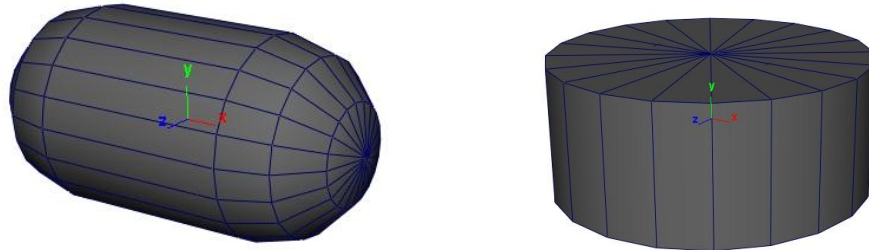


Fig. 15. (A) Link capsule (B) Mantle cylinder.

IV.A. Phase One: Building a Creature

Creatures and their tentacles are defined using a rule based algorithm. The creature's mantle is represented by a cylinder with height and radius oriented along the positive y-axis (figure 15B). Each tentacle is constructed as a chain of capsule links (figure 15A), where each link has a length, radius, and a scale factor. The link's local space transformation is centered at the origin and oriented along the positive x-axis. The first link in the chain is given an initial length and radius and is scaled at 100%. The length and radius of each consecutive child link is determined by multiplying the initial length and radius by its assigned scale factor. The scale factor is determined by three variables, the total number of links in the chain, the position of a link in the chain, and the end scale factor. For example, if a tentacle has n links, the scale factor for link i in the chain would be:

$$i \text{ scale factor} = 1 - (i / (n - 1)) * (1 - \text{end scale factor})$$

The global position of each child link relative to its parent link is equal to the parent link's global position + $\frac{1}{2}$ (parent length + child length) down the tentacles axis. In case of the initial link, its global position is equal to the creatures radius + $\frac{1}{2}$ * link length, positioning the base of the tentacle at the outside edge of the creatures base. Each tentacle is placed equidistant around the creatures base in counterclockwise order and its axis is equal to the x-axis vector rotated n degrees around the y-axis where n is equal to (360 degrees / (the number of tentacles))*(the tentacle number starting at 0). So if there were 6 tentacles, each spaced 60 degrees apart, the axis of tentacle 0 would be along the x-axis (1,0,0). Tentacle1 would be the vector (1,0,0) rotated 60 degrees around the y-axis. This ensures that each tentacle axis is aligned with the surface normal of the creature's base at the corresponding orientation. By defining creature and tentacle placement in this way, a variety of creatures can be determined quickly.

In between each tentacle link is an articulation joint. This joint is a spherical joint with an angular motor that is driven through joint acceleration springs. When given a target orientation, the motor will try and rotate the joint from its current orientation to the new target.

IV.A.1. Scene Description File

A scene description file has been developed to allow creatures to be placed in the scene quickly. When the program is launched it looks for a description file to read in which it will tell the program how many creatures are in the scene and the parameters for

each creature and its tentacles. Creature parameters include: the height and radius of the creature mantle, the global coordinates of the creature, and number of tentacles.

Parameters for each tentacle include the initial link's length and radius, the number of links in the tentacle and a scale factor for tapering.

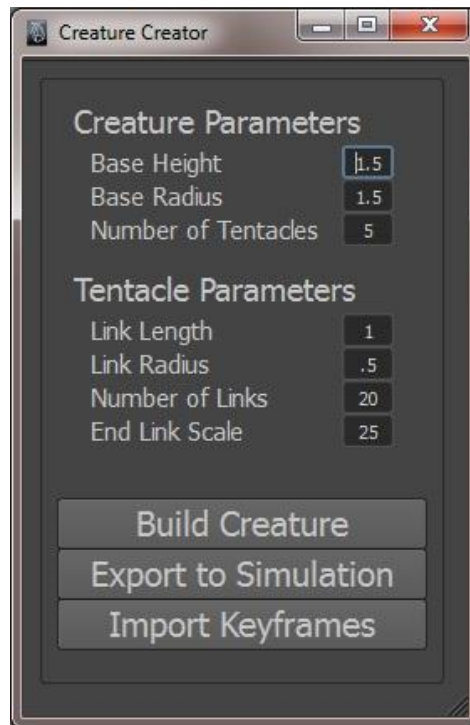


Fig. 16. Creature Creator GUI.

IV.A.2. Creation in Maya

Creating a creature in Maya takes place in two steps. First a basic GUI, as seen in figure 16, is provided with simple inputs for defining a creature's initial height, radius, and number of tentacles. The user is also able to set an initial length, radius, and end scale factor as well as the starting number of links for a tentacle. Clicking the *Build*

Creature button builds in the viewport a creature with uniform tentacles. Attached to this creature is a second rig that can be used to further customize the creature (figure 17). These controls, shown in detail in table 1, allow for quick and easy customization of each individual tentacle. When customization of the creature is completed, clicking the *Export to Simulation* button writes the creature parameters out to a scene description file which is read in by the simulation.

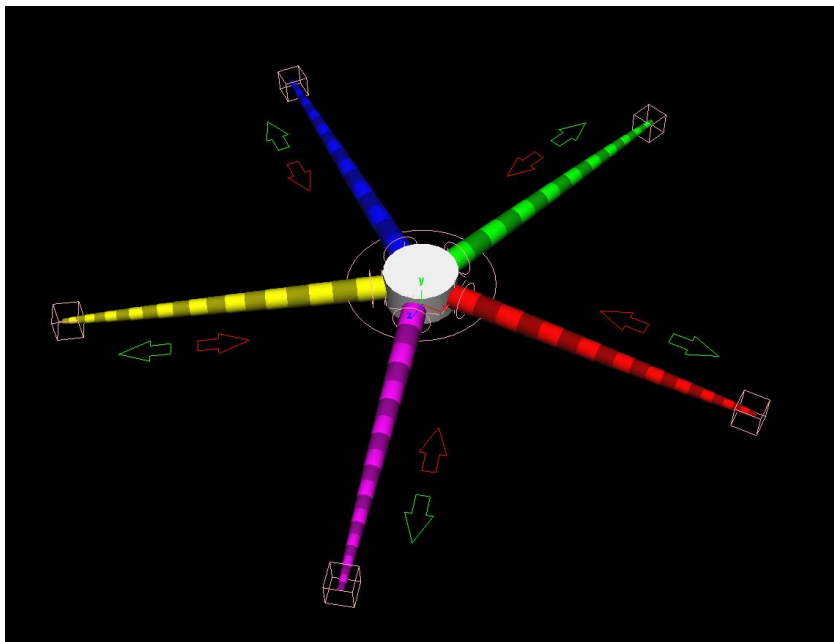


Fig. 17. Creature with rig in Maya


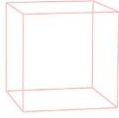
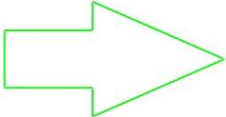
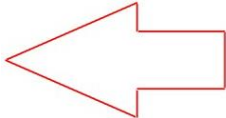
	<p>Translate to adjust length or height.</p> <p>Scale to adjust radius.</p>
	<p>Scale to adjust end scale factor.</p>
	<p>Select to add link to end of the chain.</p>
	<p>Select to remove link from the end of the chain.</p>

Table. 1. Rig Controls.

IV.B. Phase Two: Goal Directed Locomotion

Just like creature creation, the locomotion of each creature happens procedurally. Each creature is driven by its tentacles and will be driven in the same manner whether there are three tentacles or eight tentacles. Tentacle motion is achieved by modifying the joint angle between each tentacle link. All motions are based on sine waves designed to achieve a certain task. By modifying the frequency and period of each wave, different motions can be achieved. In addition, applying this wave to the motor strengths at each joint can further create organic writhing motion.

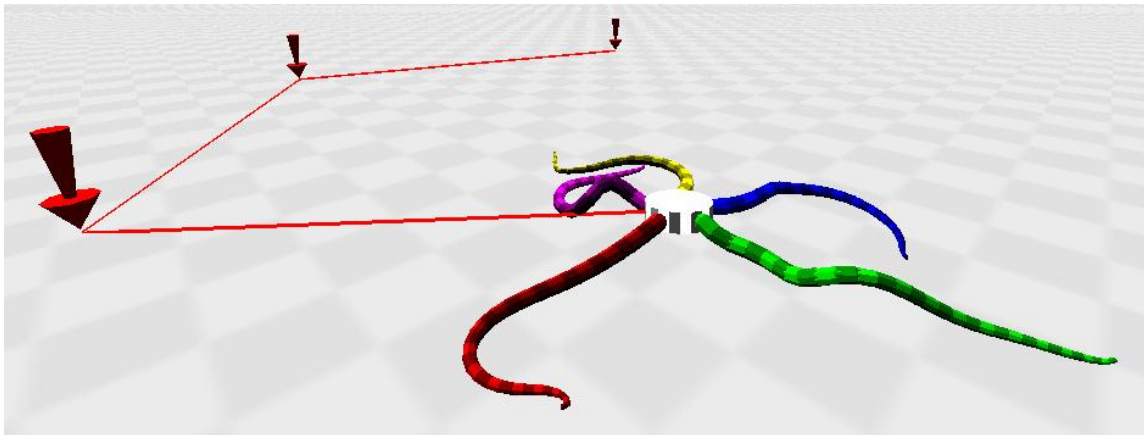


Fig. 18. Simulated creature with three goals.

Each creature in the simulation can have its own unique goal or set of goals. Goals can either exist singularly or in a list that defines a path. Figure 18 above shows a path with three goals. While each target is defined as a point in space, it will also have a radius to define a region around it. If the creature enters this defined region than the goal is considered met and will be removed from the list.

If a goal exists in the scene then the creature will try to reach it by driving tentacles one at a time. If there is no goal, all tentacles will be set to an idle model. When a tentacle is selected by the creature as the driving tentacle it must first determine where it is in the global coordinate system in relation to the creature's target goal. The purpose of this check is to determine if the tentacle will try to assume a pushing or pulling motion. To make this determination, two vectors are compared. The first vector is a normalized vector from the mantle to the target goal. The second vector is a normalized vector from the mantle to the base of the driving tentacle. By taking the dot product of these two vectors and examining the angle between them it is possible to

determine what side of the creature the tentacle is on. If the angle between the two vectors is between negative $\pi/2$ and positive $\pi/2$ then the tentacle is between the mantle and the target goal. The resulting motion should be a pull. Otherwise, if the angle is between $\pi/2$ and π or negative $\pi/2$ and negative π , then mantle is between the tentacle and the target goal; therefore the resulting motion of the tentacle should be a push.

Once a desired motion has been identified, the tentacle will construct a list of actions that will help facilitate the desired motion. This is called the Action Queue. If there is already an active queue then actions will be added to the end of the queue. Each action in the queue has a timer, a target angle, and a strength value. The timer tells the tentacle how long it has to perform the designated action. The possible actions that can be queued are dependent on what motion the tentacle will perform. A tentacle can either push, or pull. A pushing motion is achieved by first compressing the tentacle, fixing the tip to the surface, and then extending the tentacle and pushing the creature. A pull is achieved in the opposite manner. First the tentacle is extended, the tip is fixed in place on the surface and then the tentacle is compressed and pulls the creature forward.

Distance constraints are used to allow tentacles to stick to a surface. When a tentacle is being used to locomote, constraints are created to lock a tentacle in place. For a pulling motion, this serves as an attachment point for the tentacle to pull from. In a pushing motion, this serves as an anchor that it is pushing against.

IV.C. Tools

The simulation portion of this thesis was written in C++ and uses NVIDIA PhysX and OpenGL. The PhysX engine provides efficient and reliable physics simulations across a broad spectrum of platforms, from smartphones, tablets and consoles to high-end CPU/GPU PC systems. Game developers have relied on PhysX technology to provide collision detection and simulation in over 300 game titles. OpenGL is a software interface to graphics hardware used to produce high-quality 2D and 3D computer-generated images and interactive applications. It is widely used in CAD, scientific visualization and video games.

Autodesk Maya 2012 was used in coordination with the stand alone simulation. Maya is a powerful, integrated 3D modeling, animation, visual effects, and rendering solution that has become an industry standard tool. Within Maya, Python and QT were used to develop tools for content creation. Python is a powerful object-oriented scripting language embedded in Maya. QT is a cross-platform application framework that is used for building Graphical User Interfaces (GUIs). Maya's C++ API was used to develop a keyframe import plugin. The API was used for this due to a need for a higher level of performance than could be provided with a Python script.

CHAPTER V

FUTURE WORK

Although the system presented in this thesis is capable of producing goal-directed locomotion for tentacled creatures, there are many ways in which it can be improved upon and optimized.

Creature movement should be more coordinated. The tentacle gate should be determined by what is in the best interest of creature movement as opposed to using a predetermined cyclical gate. This could be accomplished by prioritizing which tentacles are driving the creature based on the tentacles proximity to the goal and whether or not they are optimally suited for the task of locomotion. This would make locomotion more deliberate than chaotic. Tentacles should also be made more self-aware of where they are in the scene. When a tentacle is tasked to push or pull it should try and first orient itself on a vector that is better suited to closing the gap between creature and goal. Because an action queue is used for determining movements a tentacle will only check the relationship between the target goal, the creature base and itself when it is initially inserted into the queue. Instead of setting predetermined actions a tentacle's motions should be determined on the fly. Finally movement would have an altogether more organic feel if range of actions was increased. By developing more intermediary actions the resulting motion would appear more natural.

This system could be further improved for the user by developing an animation rig that would allow the artist to modify the simulation once it is completed and imported into the animation software.

CHAPTER VI

RESULTS AND CONCLUSION

This thesis has presented the initial steps in developing a system for procedurally generating goal-directed terrestrial locomotion for tentacled creatures. A pipeline was developed with tools that allow an artist to quickly generate a variety of creatures in Maya. Once created these creatures can be exported into the simulation program where goals can be created in an interactive workflow. After simulation the data can be exported back to Maya. Tests were run using a desktop computer equipped with a first generation Intel i7 CPU clocked at 2.67 GHz and an NVidia Geforce 280 GTX graphics card. These tests showed that the system was capable of handling four creatures each with six tentacles while running in real time at a frame rate of 15 frames per second or greater. Using physically based dynamics allowed for resulting animation to be expressive and physically plausible.

In its current iteration this system may not prove suitable as a replacement for traditional animation methods, such as keyframing or motion capture, needed for hero characters. However, this thesis does present a solid foundation for developing a system that can become the basis for background crowds where simple interactive controls would allow an artist to quickly place multiple creatures in a scene.

REFERENCES

1. Armstrong, W., and Green, M., "The dynamics of articulated rigid bodies for purposes of animation," *Graphics Interface '85*, p.407-415, May 1985.
2. Bruderlin, A., Calvert, T., "Goal-directed, Dynamic Animation of Human Walking," *ACM SIGGRAPH Computer Graphics*, v.23 n.3, p.233-242, July 1989.
3. Criswell, B., Derlich, K., Hatch, D., "Davy Jones' Beard: Rigid Tentacle Simulation," *ACM SIGGRAPH 2006 Sketches*, July 30-August 03, 2006, Boston, Massachusetts.
4. De Aguiar et al. "Performance Capture from Sparse Multi-view Video," *ACM SIGGRAPH Computer Graphics*, v. 27 n. 3, 2008.
5. Derksen, M., Kim, T., "Animation and Simulation of Octopus Arms in *The Night at the Museum 2*," *Proceedings of the 2009 ACM SIGGRAPH*, August 03-07, 2009, New Orleans, Louisiana.
6. Hecker et al. "Real-time Motion Retargeting to Highly Varied User-Created Morphologies," *ACM SIGGRAPH Computer Graphics*, v. 27 n. 3, 2008.
7. Johnson, O., Thomson, F., *The Illusion of Life: Disney Animation*. New York: Hyperion Books, 1995.
8. Kitagawa, M., Windsor, B., *Mocap for Artists: Workflow and Techniques for Motion Capture*. Burlington, MA: Focal Press, 2008.
9. Kokkevis, E., Metaxas, D., Badler, N., "User-Controlled Physics-Based Animation for Articulated Figures," *Proceedings of Computer Animation*, p.16, June 03-04, 1996.

10. Lasseter, J., "Principles of Traditional Animation Applied to 3D Computer Animation," in Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, pp. 35-44, 1987.
11. Mather, J., "How Do Octopuses Use Their Arms?" *Journal of Comparative Psychology*, v. 112 n. 3, p. 306-316, 1998.
12. Menache, A., *Understanding Motion Capture for Computer Animation*. Burlington, MA: Morgan Kaufmann Publishers, 2011.
13. Shapiro, A., Chu, D., Allen, B., Faloutsos, P., "A dynamic controller toolkit," Proceedings of the 2007 ACM SIGGRAPH symposium on Video games, August 04-05, 2007, San Diego, California.
14. Shapiro, A., Pighin, F., Faloutsos, P., "Hybrid Control for Interactive Character Animation," Proceedings of the 11th Pacific Conference on Computer Graphics and Applications, p.455, October 08-10, 2003.
15. Skrba et al. "Animating Quadrupeds: Methods and Applications," *Computer Graphics Forum*, v. 28 n. 6, p. 1541-1560, 2009.
16. "Ursula." *Disney Wiki*. Web. 21 Apr. 2015.
<<http://disney.wikia.com/wiki/Ursula>>.
17. Weinstein, R., Guendelman, E., Fedkiw, R., "Impulse-Based Control of Joints and Muscles," *IEEE Transactions on Visualization and Computer Graphics*, v.14 n.1, p.37-46, January 2008.
18. Weinstein, R., Teran, J., Fedkiw, R., "Dynamic Simulation of Articulated Rigid Bodies with Contact and Collision," *IEEE Transactions on Visualization and Computer Graphics*, v.12 n.3, p.365-374, May 2006.
19. Wilhelms, J., Van Gelder, A., "Combining Vision and Computer Graphics for Video Motion Capture," *The Visual Computer*, v. 19, p360-376, 2003.

20. Zelman et al. "Nearly Automatic Motion Capture System – Tracking Octopus Arm Movements," *Journal of Neuroscience Methods*, v. 182, p. 97-109, 2009.